On the Combinatorial and Logical Complexities of Algebraic Structures

by

Michael Levet

B.S., Virginia Tech, 2015

B.A., Virginia Tech, 2015,

M.S., University of South Carolina- Columbia, 2018,

M.E., University of South Carolina- Columbia, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2023

Committee Members:

Joshua A. Grochow, Chair

Jeremy F. Alm

Peter Mayr

Sriram Sankaranarayanan

Ashutosh Trivedi

Levet, Michael (Ph.D., Computer Science)

On the Combinatorial and Logical Complexities of Algebraic Structures

Thesis directed by Prof. Joshua A. Grochow

In this thesis, we investigate the combinatorial and logical complexities of several algebraic structures, including groups, quasigroups, certain families of strongly regular graphs, and relation algebras. In Chapter 3, we leverage the Weisfeiler–Leman algorithm for groups (Brachter & Schweitzer, LICS 2020) to improve the parallel complexity of isomorphism testing for several families of groups including (i) coprime extensions $H \ltimes N$ where H is $O(1)$-generated and N is Abelian (c.f., Qiao, Sarma, & Tang, STACS 2011), (ii) direct product decompositions, and (iii) groups without Abelian normal subgroups (c.f., Babai, Codenotti, & Qiao, ICALP 2012). Furthermore, we show that the weaker *count-free* Weisfeiler–Leman algorithm is unable to even identify Abelian groups. As a consequence, we obtain that FO fails to capture all polynomial-time computable queries even on Abelian groups. Nonetheless, we leverage the count-free variant of Weisfeiler–Leman in tandem with bounded non-determinism and limited counting to obtain a new upper bound of $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$ for isomorphism testing of Abelian groups. This improves upon the previous $\mathsf{TC}^0(\mathsf{FOLL})$ upper bound due to Chattopadhyay, Torán, & Wagner (*ACM Trans. Comput. Theory*, 2013).

Weisfeiler–Leman is equivalent to the first in a hierarchy of Ehrenfeucht–Fraïssé pebble gammes (Hella, *Ann. Pur. Appl. Log.*, 1989). In Chapter 4, we explore the descriptive complexity theory of finite groups by examining the power of the second Ehrenfeucht-Fraïssé bijective pebble game in Hella's (*Ann. Pure Appl. Log.*, 1989) hierarchy. This is a Spoiler-Duplicator game in which Spoiler can place up to two pebbles each round. While it trivially solves graph isomorphism, it may be nontrivial for finite groups, and other ternary relational structures. We first provide a novel generalization of Weisfeiler-Leman (WL) coloring, which we call 2-ary WL. We then show that the 2-ary WL is equivalent to the second Ehrenfeucht-Fraïssé bijective pebble game in Hella's hierarchy.

Our main result is that, in the pebble game characterization, only $O(1)$ pebbles and $O(1)$ rounds are sufficient to identify all groups without Abelian normal subgroups. In particular, we show that within the first few rounds, Spoiler can force Duplicator to select an isomorphism between two such groups at each subsequent round. By Hella's results (*ibid.*), this is equivalent to saying that these groups are identified by formulas in first-order logic with generalized 2-ary quantifiers, using only $O(1)$ variables and $O(1)$ quantifier depth.

In Chapter 5, we show that Graph Isomorphism (GI) is not AC^0-reducible to several problems, including the Latin Square Isotopy problem and isomorphism testing of several families of Steiner designs. As a corollary, we obtain that GI is not AC^0-reducible to isomorphism testing of Latin square graphs and strongly regular graphs arising from special cases of Steiner 2-designs. We accomplish this by showing that the generator-enumeration technique for each of these problems can be implemented in $\beta_2\mathsf{FOLL}$, which cannot compute Parity (Chattopadhyay, Torán, & Wagner, *ibid.*).

Finally, in Chapter 6, we shed new light on the spectrum of the relation algebra we call A_n, which is obtained by splitting the non-flexible diversity atom of 6_7 into n symmetric atoms. Precisely, we show that the minimum value in $\text{Spec}(A_n)$ is at most $2n^{6+o(1)}$, which is the first polynomial bound and improves upon the previous bound due to Dodd & Hirsch (*J. Relat. Methods Comput. Sci.* 2013). We also improve the lower bound to $2n^2 + \Omega(n\sqrt{\log n})$. Prior to the work in this thesis, only the trivial bound of $n^2 + 2n + 3$ was known.

Dedication

I wish to dedicate this thesis to my aunts Eleanore and Edith, who are no longer with us; my grandmother Allegra who passed away of COVID at the onset of the pandemic; and my cat and best friend Drew, who rode with me in the car for 6 hours from the shelter in Sheridan, Wyoming and has been an ongoing source of support, such as by making me take breaks and requesting Greenies in lieu of coauthorship.

Acknowledgements

I owe a great deal to a number of people for their help, support, and encouragement during my time at CU. First and foremost, I am deeply grateful to my advisor, Josh Grochow. To say that I learned a great deal of mathematics and about research from Josh would be a British understatement. Beyond the research aspects, Josh went well above and beyond to support me as a person, including for instance helping to secure a hotel and pet supplies when I had to evacuate for Marshall fire. I am truly fortunate to have Josh as my advisor.

Next, I wish to thank Jeremy Alm for functionally serving as a second advisor. I have thoroughly enjoyed learning about relation algebras and collaborating with Jeremy, and I am grateful for his mentorship, friendship, and serving on my thesis committee. I wish to thank Peter Mayr, Sriram Sankaranarayanan, and Ashutosh Trivedi for serving on my thesis committee.

I owe a great deal to Rajshree Shrestha for all of her support. I am grateful to Éva Czabarka for her continued mentorship, guidance, and Hungarian witticisms. I wish to thank James Wilson for his mentorship regarding isomorphism testing, research, and navigating academia in general. I wish to thank Josh Cooper for helpful discussions regarding inverse Ramsey numbers, which led to Lem. 6.2.8. I wish to thank my lab mates Gabe Andrade, Charlie Carlson, Ezz El-Sai, Jessie Finocchiaro, Robby Green, Vishnu Murali, Maya Ornstein, Elise Tate, and Tzu-Chi Yen for their support. I also wish to thank Nate Collins for a fruitful and enjoyable collaboration on a number of results extending the work in Chapter 3 of this thesis.

Finally, I wish to thank my parents and brother Allen for their unwavering support and encouragement during my graduate studies.

Contents

Chapter

Figures

Figure

Chapter 1

Introduction

A central goal of mathematics is to classify given objects up to some notion of equivalence. Perhaps the most well-known example of this stems from group theory, in which the goal is to determine all groups of any given order n up to isomorphism. The classification problem for finite groups is introduced almost immediately in standard undergraduate courses in Modern Algebra. For instance, students begin by classifying (by hand) the groups of order n, where n is small. Standard theorems such as those of Lagrange, Cauchy, and Sylow provide more powerful and systematic means of determining the groups of order n. Still more powerful tools exist, such as those from representation theory and group cohomology. Nonetheless, the key barrier in obtaining classification results is one of combinatorial explosion: for certain values of n, there exist a rather significant number of groups. Take for instance the class of p-groups: for a prime p and $k \in \mathbb{N}$, there are $p^{2k^3/27+O(k^{8/3})}$ such groups of order p^k [112, 181, 44]. For groups of order 1024, there are $49,487,367,289$ such groups [42, 54], and $> 99\%$ of groups of order ≤ 1024 are of order exactly 1024. It is conjectured that the class of 2-groups (groups of order 2^k for some k) is dense within the class of finite groups.

We see this theme of classification in other areas of mathematics; we briefly highlight this in the cases of Latin squares, graphs, and relation algebras, which will be investigated in this thesis (a more thorough survey of the literature will be presented later). The study of Latin squares can be traced back to Islam circa 1200 [5]. We refer to [60, Page 12] for a more detailed history of Latin squares. Strongly regular graphs, which can be viewed as a generalization of Latin squares, actually admit a partial classification (see Section 1.1.3 for more discussion). For classification of relation algebras, the next object we study in this thesis, the following monographs serve as key references [153, 113, 88].

The powerful combination of modern computing and mathematical theory has led to advances in classification (e.g., [42, 54, 43]), as well as practical tools including computer algebra systems such as Magma [47], GAP [86], and Sage [187], and graph isomorphism packages such as `nauty` and `traces` [157], as well as `saucy` [59, 71]. Nonetheless, computation has not overcome the combinatorial barriers associated with classification problems. It is thus natural to inquire as to the *easy* instances that can be readily classified, as well as the *hard* instances that are resistant to classification. To this end, we require some precise notion as to the *complexity* of an object.

There are several notions of complexity from which to select. The role of computation in classification problems naturally suggests measures of computational complexity: algorithmic runtime, space, randomness, the parallel complexity (circuit complexity) of the corresponding isomorphism tests, compressibility (Kolmogorov complexity), and logical definability (descriptive complexity theory). This thesis investigates *combinatorial* measures of complexity for algebraic objects including those arising from Latin squares (e.g., (quasi)groups and certain families of strongly regular graphs) and relation algebras. In particular, we examine the deep relationship between these combinatorial measures of complexity and other measures of complexity, incuding the parallel complexity of their isomorphism tests and logical definability. These relationships in turn yield insights on problems such as graph isomorphism testing and the Flexible Atom Conjecture on relation algebras.

1.1 Graph Isomorphism

The Graph Isomorphism problem (GI) takes as input two graphs G and H, and asks if there is an isomorphism $\varphi : V(G) \to V(H)$. The best known algorithm to solve this problem in general is due to Babai [26], who exhibited an $n^{\Theta(\log^2 n)}$-runtime[1] procedure to test whether two graphs are the same. It is known that GI belongs to $\mathsf{NP} \cap \mathsf{coAM}$, and it is open as to whether GI belongs to coNP. Recent works [84, 94] have established that for any field $\mathbb{F}$, GI belongs to $\mathbb{F}$-Tensor Isomorphism ($\mathsf{TI}_{\mathbb{F}}$), which effectively captures the complexity of testing two 3-way arrays for equivalence up to simultaneous change of basis. That is, GI is no harder than multilinear algebra. When $\mathbb{F}$ is finite, $\mathsf{TI}_{\mathbb{F}} \subseteq \mathsf{NP} \cap \mathsf{coAM}$.[2] In contrast, the best known lower-bound for GI is DET– the class of problems that are NC^1-Turing reducible to computing the determinant of an $n \times n$ integer matrix. So in a precise sense, GI is sandwiched between linear algebra and multilinear algebra.

Key motivation for GI arises from the P vs. NP problem. One approach to settling P vs. NP is to exhibit an NP-intermediate language- that is, a language belonging to NP that is neither in P nor NP-complete. Ladner [140] established the converse: if $\mathsf{P} \neq \mathsf{NP}$, then there must exist a strict infinite hierarchy of NP-intermediate languages. Ladner's proof utilized a diagonalization technique, and so while he exhibited such languages under the assumption that $\mathsf{P} \neq \mathsf{NP}$, his construction does not yield insights as to natural candidates that might be NP-intermediate.

There has been considerable effort in identifying NP-intermediate candidates. Many of these candidates, such as Primality Testing and Linear Programming have been placed into P. In the case of Primality Testing, the polynomial-time algorithm is relatively recent [3]. For Linear Programming, the Simplex algorithm was the long-standing algorithmic tool in this area. Despite the fact that the Simplex algorithm performed well in practice, it still had a worst-case exponential runtime [135]. In 1979, Leonid Khachiyan introduced the Ellipsoid algorithm, which was the first polynomial-time algorithm for Linear Programming [131]. It is worth noting that neither the AKS procedure nor the Ellipsoid algorithm are used in practical implementations.

The remaining candidate NP-intermediate problems under (historical) consideration are either isomorphism problems such as GI or cryptographic problems, such as Integer Factorization or Discrete Logarithm. There is a precise sense in which Integer Factorization and

[1] Babai's analysis [26] suffices to provide a quasipolynomial bound, but does not make the polylogarithm in the exponent explicit. See [108] for a more explicit analysis, which yields the exponent of $\Theta(\log^2 n)$.

[2] We refer the reader to [94, Remark 3.4] for discussion on $\mathsf{TI}_{\mathbb{F}}$ when $\mathbb{F}$ is infinite.

Discrete Logarithm may in fact be easier than GI. For instance, both belong to $\mathsf{NP} \cap \mathsf{coNP}$, while GI is not known to belong to coNP. Furthermore, both Integer Factorization and Discrete Logarithm reduce to the Hidden Subgroup Problem over Abelian groups, which admits an efficient (BQP) quantum algorithm due to Shor [178]. In contrast, GI reduces to the Hidden Subgroup Problem over the symmetric group, which is not known to admit an efficient quantum algorithm. It is conjectured that no such efficient quantum algorithm exists, and it has been proven that most of the current techniques do not achieve this [162].

While GI has been resistant to polynomial-time algorithms in general, there is considerable evidence that it is unlikely to be NP-complete. For instance, as GI $\in \mathsf{coAM}$, we have that if GI were NP-complete, then $\mathsf{PH} = \Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}} = \mathsf{AM}$ (see for instance, [17]). As GI is solvable in quasipolynomial-time (QP) [26], we have that if GI were NP-complete, then $\mathsf{NP} \subseteq \mathsf{QP}$. This would violate the Exponential Time Hypothesis [122], as well as imply that $\mathsf{EXP} = \mathsf{NEXP}$ [53]. It is believed that $\mathsf{EXP} \neq \mathsf{NEXP}$.

Additionally, GI does not behave like any known NP-complete problem in several ways. For instance, GI is low for a number of complexity classes such as PP [136] and SPP [18]. Schöning [176] established that GI belongs to the second level $\mathsf{L}_2^{\mathsf{P}}$ of the Low Hierarchy, which is contained in NP. Thus, unless PH collapses to some level, GI is not NP-complete under several notions of reducibility that are weaker than many-one polynomial-time computable reductions. Finally, we note that Mathon [156] showed that the decision variant GI is polynomial-time equivalent to #GI, which asks for the number of isomorphisms between two graphs. No NP-complete problem is known to be polynomial-time equivalent to its counting version.

In light of Babai's [26] breakthrough result, further advances on the general Graph Isomorphism problem seem difficult at this time. It is thus natural to consider special cases that might be easier. We will consider special algebraic subproblems of GI that arise from Latin squares, including Group Isomorphism (GpI), Latin Square Isotopy, and Latin Square Graph Isomorphism. In the setting of GpI, we will investigate the Weisfeiler–Leman dimension and iteration number for several families of groups. To this end, we begin by introducing the Weisfeiler–Leman algorithm for graphs (see Section 1.1.1) prior to our discussions on GpI. By controlling both the Weisfeiler–Leman dimension and the iteration number, we are able to improve the parallel and descriptive complexities for isomorphism testing of several families of groups.

We will next consider the Latin Square Isotopy and Latin Square Graph Isomorphism problems. Precisely, we will show that these problems are strictly easier than GI under the ordering induced by many-one AC^0-computable reductions. This extends a result of Chattopadhyay, Torán, & Wagner [57] who established the analogous result for Quasigroup Isomorphism. This keeps with the secondary theme of parallel complexity. In light of the fact that $\mathsf{AC}^0 = \mathsf{FO}$ [161], our results also suggest that, in logics extending FO, quasigroups (up to isotopism) and Latin square graphs (up to isomorphism) might be definable via more succinct sentences in logics extending FO, than in the case of general graphs.

1.1.1 Weisfeiler–Leman

In the setting of GI, there is a natural measure of combinatorial complexity arising from the family of Weisfeiler–Leman (WL) algorithms (we will abuse notation and simply refer to this family as *the* Weisfeiler–Leman algorithm). For fixed $k \geq 1$, the k-dimensional Weisfeiler–Leman algorithm (k-WL) works by iteratively coloring k-tuples of vertices in an isomorphism invariant manner (see Section 2.7 for full details). To use k-WL as a non-isomorphism test, we run k-WL on the disjoint union of graphs G and H. If at the end of round r, the multiset of colors for G differs from that for H, then we can conclude that $G \not\cong H$. For fixed k, we have that k-WL runs in polynomial-time. The *Weisfeiler–Leman dimension* of a graph G is the minimum k such that k-WL distinguishes G from all non-isomorphic graphs H. The *iteration number* is the minimum number of rounds that suffice for WL to either distinguish two objects or for the coloring to stabilize – whichever comes first.

The Weisfeiler–Leman algorithm can be traced back to the works of Boris Weisfeiler and Andrei Leman [196, 197], who considered what is now known as 2-WL. This was generalized for $k > 2$ independently by Babai & Mathon [23] in the direction of coherent configurations, and

Immerman & Lander [120] (see as well [55]) in the direction of logics. On its own, Weisfeiler–Leman serves as an efficient polynomial-time isomorphism test for several families of graphs including trees [77, 120], planar graphs [134, 100], graphs of bounded treewidth [101, 104, 133, 144], graphs of bounded rank width [102], graphs of bounded genus [95, 99], and graphs for which a specified minor H is forbidden [96]. It is also worth noting that 1-WL identifies almost all graphs [173] and 2-WL identifies almost all regular graphs [45, 138]. In the case of graphs of bounded treewidth [104, 144] and planar graphs [104, 100], Weisfeiler–Leman serves even as an NC isomorphism test.

Despite the success of WL as an isomorphism test, Cai, Fürer, & Immerman [55] exhibited an infinite family of non-isomorphic pairs of graphs (G_k, H_k), for which $\Omega(|G_k|)$-dimensional WL was required to distinguish G_k from H_k. This yields a runtime of $n^{\Theta(n)}$, which is worse than simply enumerating over all possible bijections. The graphs in [55] have max-degree 4. Thus, the group-theoretic techniques of Luks [151] serve as a polynomial-time isomorphism test for this family. Recently, Neuen & Schweitzer [165] used the CFI construction in tandem with the multipede construction of Gurevich & Shelah [105] to give an exponential-time lower bound on the individualize-and-refine technique, including when k-WL is used for the refinement step instead of 1-WL. In light of the equivalence of WL with the logic $\mathsf{FO} + \mathsf{C}$ (first-order logic with counting quantifiers) [120, 55], we have a precise sense in which combinatorial techniques appear insufficient to place GI into P, while group theoretic techniques appear to be necessary.

Despite the fact that WL is not sufficiently powerful to place GI into P, it remains an active area of research. For instance, Babai's algorithm [26] combines $O(\log n)$-WL with group theoretic techniques. Weisfeiler–Leman also has close connections to linear programming [20, 103, 155] and machine learning [163, 164, 4, 106]. As an example, 1-WL has the same distinguishing power as a graph neural network (GNN), where the number of iterations of 1-WL corresponds to the depth of the GNN [164]. There is ongoing work to define generalized GNNs based on k-WL [164, 163]. We refer the reader to Sandra Kiefer's dissertation [132] for a comprehensive overview of the various connections of Weisfeiler–Leman to other areas.

1.1.2 Group Isomorphism

The Group Isomorphism problem takes as input two group G, H and asks whether there is an isomorphism $\varphi : G \to H$. The complexity of GpI depends on the manner in which the groups are represented. When the groups are given by their multiplication tables, GpI is known to be AC^0-reducible to GI [160] (see as well the reduction used in Weisfeiler–Leman Version III [48]). On the other hand, Chattopadhyay, Torán, & Wagner [57] ruled out several notions of parallel reductions from GI to GpI, when the groups are given by their multiplication tables. This includes, for instance, many-one AC^0-computable reductions.

On the other hand, when the groups are given succinctly, such as by generating sets of permutations [152] or matrices over finite fields (see for instance, [94]), GI *reduces to* GpI. Mekler's construction also provides a reduction from GI to GpI when the groups are given succinctly [159, 107]. When the groups are given by generating sequences of permutations or matrices, we have that GpI belongs to Σ_2^{P}. In the setting of black-box groups [35], GpI belongs to $\mathsf{Promise}\Sigma_2^{\mathsf{P}}$ – even verifying the group axioms is only known to be Π_2^{P}-computable. In the setting when the groups are given by abstract generators and relations, GpI is undecidable [2, 168].

In this thesis, attention will be restricted to GpI in the Cayley table model. Progress on this problem dates back to the mid-70's. We note that a group of order n admits a generating set of size at most $\lceil \log_p n \rceil$, where p is the smallest prime dividing n. Tarjan [160] and Lipton, Snyder, & Zalcstein [149] independently observed that this yields the generator-enumeration strategy, in which we find a generating set S for G and consider all the possible ways to map S into H. Tarjan [160] obtained an $n^{\log_p(n)+O(1)}$-time algorithm through this method. Lipton, Snyder, & Zalcstein [149] obtained a stronger bound of $\mathsf{DSPACE}(\log^2 n)$. In particular, $O(1)$-generated groups admit a polynomial-time isomorphism test. Lipton, Snyder & Zalcstein [149] also gave the first polynomial-time isomorphism test for Abelian groups.

In more than 40 years, the $n^{\log_p(n)+O(1)}$ bound has escaped largely unscathed: Rosenbaum [172] (see [142, Sec. 2.2]) improved this to $n^{(1/4)\log_p(n)+O(1)}$. And even the impressive body of work on practical algorithms for this problem, led by Eick, Holt, Leedham-Green and O'Brien (e. g., [41, 78, 40, 56]) still results in an $n^{\Theta(\log n)}$-time algorithm in the general case (see [199, Page 2]).

There has been considerable effort to improve the parallel complexity of the generator-enumeration strategy. Wolf [200] leveraged non-deterministic circuit complexity classes, with the key idea of using $O(\log^2 n)$ existentially-quantified non-deterministic bits to guess generating sequences, and then to test whether the induced map extends to an isomorphism (so called *marked-isomorphism testing*) in NC^2. We refer to this complexity class as $\beta_2\mathsf{NC}^2$, where the β_2 specifies the $O(\log^2 n)$ non-deterministic bits. A careful analysis of Wolf's result actually yields a bound of $\beta_2\mathsf{AC}^1$ – namely, we use the fact that the product of two group elements can be computed in AC^0 [38], while Wolf instead uses NC^1 circuits for this task. In his dissertation, Wagner [195] further improved this complexity to $\beta_2\mathsf{SAC}^1$. Finally, Tang [185] showed that marked isomorphism testing in general was L-computable, placing GpI into $\beta_2\mathsf{L}$. As a consequence of Tang's proof, $O(1)$-generated groups admit an L-isomorphism test. By leveraging cube-generating sequences, Chattopadhyay, Torán, & Wagner [57] independently showed that GpI belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$, and Tang [185] placed GpI into $\beta_2\mathsf{SC}^2$. As cube generating sequences must have size $\Theta(\log n)$, we do not obtain further improvements on $O(1)$-generated groups. In particular, the proof of Chattopadhyay, Torán, & Wagner [57] that GpI belongs to $\beta_2\mathsf{L}$ does not imply that $O(1)$-generated groups admit an L isomorphism test.

We now turn our discussion towards Abelian groups. Following [149], the runtime complexity of Abelian group isomorphism was subsequently improved due to Savage [174] who gave an $O(n^2)$ algorithm, Vikas [193] who gave an $O(n \log n)$ algorithm, and finally Kavitha [129] who showed that isomorphism testing of Abelian groups was $O(n)$-time computable. Iliopoulos [116] investigated the complexity of Abelian groups in succinct input models. In the direction of parallel complexity, Chattopadhyay, Torán, & Wagner [57] showed that isomorphism testing of Abelian groups was in $\mathsf{L} \cap \mathsf{TC}^0(\mathsf{FOLL})$. Prior to this the work in this thesis, only $O(1)$-generated and Abelian groups were known to admit NC isomorphism tests.

One strategy to obtain larger groups is to build them up from smaller groups. This leads us in the direction of group extensions. Independently, Wilson [198] and Kayal & Nezhmetdinov [130] exhibited polynomial-time algorithms to decompose a group into a direct product of indecomposable factors. Wilson's [198] result holds in the more succinct setting where the groups are given as quotients of permutation groups, while the algorithm of Kayal & Nezhmetdinov [130] is in terms of the Cayley table.

Finite nilpotent groups generalize Abelian groups in that they are precisely the groups that are direct products of their Sylow subgroups. (More generally, finite solvable groups are Zappa–Szép products of their Sylow subgroups.) The fact that we can compute direct product decompositions efficiently suggests that nilpotent groups – and in particular, p-groups – are natural to consider. Nilpotent groups, however, have been resistant to algorithmic progress. In particular, class 2 p-groups of exponent p are believed to be the hard cases of GpI, though there is little formal evidence. Recently, Dietrich & Wilson [74] gave a nearly-linear time algorithm for groups of almost all orders. The dense set of orders they considered, notably, did not include large prime powers. Grochow & Qiao [94] exhibited the first family of groups for which there exists a reduction to class 2 p-groups of exponent p. Precisely, they showed that for a class c p-group of exponent p (where $c < p$), isomorphism testing can be reduced to the case of class 2 p-groups. On the algorithmic side, few families of p-groups are known to admit a polynomial-time isomorphism test. Some notable families that admit efficient isomorphism tests include groups with bounded genus [51, 125], certain quotients of low-genus p-groups [145, 180], and the so-called *CFI groups* [48, 65, 64]. Garzon & Zalcstein [87] gave a polynomial-time algorithm for P_3 groups, which are class 2 nilpotent groups (though not necessarily p-groups).

In another direction, coprime extensions are natural to consider. Namely, the Schur–Zassenhaus theorem provides that in the case of coprime extensions, cohomology is trivial. When the normal Hall subgroup is Abelian, coprime extensions admit nice structure (see Section 3.3.1 for more details). Le Gall [85] leveraged this structure to obtain a polynomial-time isomorphism test for coprime extensions $H \ltimes N$, where H is a cyclic group acting on an Abelian group N. Building on Le Gall's [85] strategy, Qiao, Sarma, & Tang [167] generalized the result to obtain polynomial-time algorithms in the cases where H was either (i) $O(1)$-generated or (ii) Abelian (we still assume N to be Abelian in both cases). Babai & Qiao [34] further extended the work of Qiao, Sarma, & Tang [167] to the setting of groups with Abelian Sylow towers (iterated coprime extensions of Abelian p-groups). Grochow & Qiao [92] further generalized [34] to the setting of so-called *tame extensions*, where the extensions may not be coprime (in which case, the cohomology may be non-trivial), but the representation theory is tame. In this setting, testing whether the two actions are equivalent (the Action Compatibility problem) can be handled in polynomial-time.

Still in another direction, we consider for a finite group G the extension of its solvable radical $\text{Rad}(G)$ by the quotient $G/\operatorname{Rad}(G)$. Now $G/\operatorname{Rad}(G)$ has no Abelian normal subgroups. We refer to $G/\operatorname{Rad}(G)$ as *semisimple* (following [29]) or *Fitting-free.* In [29], Babai, Codenotti, Grochow, & Qiao set out to develop a polynomial-time algorithm for semisimple groups. Using novel Code Equivalence techniques, they obtained an $n^{O(\log \log n)}$ algorithm in the general case and polynomial-time algorithms in some special cases. In subsequent work [30], Babai, Codenotti, & Qiao obtained a polynomial-time isomorphism test for semisimple groups. Building on [29, 30], Grochow & Qiao [93] obtained $n^{O(\log \log n)}$ isomorphism tests for groups where (i) $\text{Rad}(G) = Z(G)$ or (ii) $Z(G) \lneq \text{Rad}(G)$ and $\text{Rad}(G)$ is elementary Abelian, as well as a number of polynomial-time algorithms in special cases.

In light of the fact that GpI is strictly easier than GI under the ordering induced by many-one AC^0-reductions, it is natural to ask as to whether techniques from GI can be fruitfully leveraged in the case of GpI. This motivates the study of Weisfeiler–Leman in the setting of groups. Previous works [146, 50] have attempted to use WL as a subroutine for GpI by reducing to some graph based on a group action over a vector space. Brachter & Schweitzer introduced three natural variants of WL in the setting of groups [48], and showed that they are equivalent up to a tradeoff of 2 in dimension. As demonstrations of the power of their model, Brachter & Schweitzer showed (amongst other results) that WL can distinguish the so called *CFI groups* (class 2 p-groups of exponent $p > 2$ arising from the CFI graphs [55] via Mekler's construction [159, 107]) in TC^1, as well as implicitly compute direct product decompositions in polynomial time [49].

In this thesis, we investigate the Group Isomorphism problem using both the Weisfeiler–Leman dimension and the iteration number as measures of combinatorial complexity. Showing that the Weisfeiler–Leman dimension is bounded for a class of groups implies that WL serves as a polynomial-time isomorphism test. If we can further show that it suffices to run WL for a polylogarithmic number of rounds, then we obtain NC upper bounds for isomorphism testing — see Section 2.10. The Weisfeiler–Leman dimension and iteration number can also be viewed as measures of logical complexity, in terms of an Ehrenfeucht–Fraïssé bijective pebble game — see Section 2.8; as well as in terms of formulas in the logic $\mathsf{FO} + \mathsf{C}$ (first-order logic with counting quantifiers) — see Section 2.9. Thus, the work in this thesis also serves to develop the descriptive complexity of finite groups.

In Chapter 3, we investigate the parallel complexity of Group Isomorphism using the Weisfeiler–Leman algorithms for groups introduced by Brachter & Schweitzer [48]. We consider the following threads.

(a) We first use the Weisfeiler–Leman Version II algorithm [48] to obtain NC bounds for isomoprhism testing in several families of groups, including most notably (i) coprime extensions $H \ltimes N$ where H is $O(1)$-generated and N is Abelian, and (ii) implicitly computing direct product decompositions. Our work on coprime extensions improves upon the upper bound of P from Qiao, Sarma, & Tang [167], and our work on direct products improves upon the previous result of Brachter & Schweitzer [49].

(b) Using the individualize-and-refine paradigm, we show that isomorphism testing of semisimple groups can be handled using $\mathsf{quasiSAC}^1$ circuits of size $n^{O(\log \log n)}$. While this does not improve upon the upper bound of P [30], it does improve the parallel complexity.

(c) We finally consider the weaker *count-free* variant of Weisfeiler–Leman, which compares the *set* rather than *multiset* of colors at each iteration. We show that count-free WL requires dimension $\Omega(\log n)$ to distinguish even Abelian groups. As a consequence, we obtain that FO fails to capture all polynomial-time computable queries for unordered groups. That is, intuitively, FO is not sufficiently powerful to capture GpI in polynomial-time.

Nonetheless, we show that count-free WL Versions I and III [48] can be used in tandem with bounded non-determinism and a single $\mathsf{Majority}$ gate to improve the parallel complexity of isomorphism testing for Abelian groups.

Remark 1.1.1. In follow-up work [65] with Nathaniel A. Collins (see also [64]), we extended and applied the techniques of Chapter 3 regarding count-free WL. Precisely, we investigated the role of counting in Group Isomorphism using the count-free Weisfeiler–Leman Version I algorithm. We first showed that $\text{poly} \log \log n$ rounds of count-free WL Version I in tandem with bounded non-determinism and a single $\mathsf{Majority}$ gate yield novel parallel isomorphism tests for certain families of coprime extensions, the CFI groups from [48], and direct products of non-Abelian finite simple groups. In the special case of the CFI groups, we improved the parallel complexity from TC^1 [48] to $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$. Furthermore, we show that for any fixed q, Spoiler requires $\Omega(\log n)$ pebbles to distinguish even Abelian groups in the q-ary count-free pebble game. The $q = 1$ case corresponds to count-free WL— see [120, 55] and Chapter 3.

Our work in Chapter 3 leaves open whether semisimple groups have bounded Weisfeiler–Leman dimension (or even WL-dimension $o(\log n)$). In Chapter 4, we further investigate the descriptive complexity of semisimple groups. Our work is motivated by the Ehrenfeucht–Fraïssé game for WL Version III. The key idea for WL Version III is that the multiplication table for a group is encoded as a graph; we then run the classical graph WL algorithm and pull back the coloring to the group elements. The multiplication relation in the group is encoded using graph gadgets. Thus, in the Ehrenfeucht–Fraïssé game for WL Version III, Spoiler and Duplicator can pebble non-group element vertices on these gadgets. Doing so fixes two elements at once. Brachter & Schweitzer [48] refer to this as *implicit pebbling.* While Duplicator must select bijections that preserve (setwise) the group element vertices, Duplicator does not need to select bijections where the multiplication gadgets are mapped consistently with the group element vertices [48].

This subtlety— and in particular, a suggestion from Pascal Schweitzer— led us to investigate the second Ehrenfeucht–Fraïssé game in Hella's hierachy [109, 110]. This game is played directly on the groups, similarly as in the Ehrenfeucht–Fraïssé games corresponding to WL Versions I and II. Duplicator selects a bijection on the group elements, and Spoiler may pebble up to two group elements in a given round. Note that in the setting of graphs, this 2-ary game immediately decides isomorphism. If two graphs G, H are non-isomorphic, then for any bijection $f : V(G) \to V(H)$ that Duplicator selects, there is an edge $\{u, v\} \in E(G)$ such that $\{f(u), f(v)\} \not\in E(H)$ (or vice-versa). Spoiler places pebbles on the vertices u, v and wins. As groups are ternary relational structures, this 2-ary game does not immediately resolve isomorphism in the same manner as for graphs.

Our main result in Chapter 4 is that if G is semisimple and H is arbitrary, then Spoiler has a winning strategy in this 2-ary game using $O(1)$ pebbles and $O(1)$ rounds. We also provide a novel higher-arity Weisfeiler–Leman procedure, corresponding to the 2-ary game. Hella [109, 110] previously established that the 2-ary game is equivalent to $\mathsf{FO}(Q)$, where Q is the set of all generalized binary quantifiers.

1.1.3 Strongly Regular Graphs

In one of the original papers on Graph Isomorphism (GI), by Corneil & Gotlieb, the authors observed that in practice, strongly regular graphs serve as difficult instances [69]. A common approach for producing such instances is to construct strongly regular graphs from combinatorial objects, such as Latin squares, nets (partial geometries), and combinatorial designs [69, 175]. While strongly-regular graphs have been perceived as hard instances, there is little evidence to suggest that they are GI-complete.

There is a beautiful classification of strongly-regular graphs (of valency $\rho < n/2$, which is without loss of generality by taking complements) due to Neumaier [166]. In using this classification in isomorphism testing, Spielman [184] and Babai–Wilmes [36] organize it as follows:

(a) Latin square graphs,

(b) Line graphs of Steiner 2-designs satisfying $\rho < f(n)$, for a certain function $f(n) \sim n^{3/4}$,

(c) Strongly-regular graphs of degree $\rho = (n-1)/2$ (a.k.a. conference graphs),

(d) Graphs satisfying a certain eigenvalue inequality referred to as *Neumaier's claw bound.*

In this thesis, we show (in particular) that (a) Latin square graphs, (b) line graphs arising from special cases of Steiner 2-designs, and (c) conference graphs are not GI-hard under AC^0-computable many-one reductions. As with [57], we show this by improving the parallel complexity of isomorphism testing in these classes of graphs to $\beta_2\mathsf{FOLL}$, which does not compute PARITY. As PARITY is AC^0-reducible to GI [189], this rules out AC^0-reductions (and more strongly, for any $i, c \geq 0$, rules out $\beta_i\mathsf{FO}((\log \log n)^c)$-reductions).

Prior to our work, there were only a few pieces of complexity-theoretic evidence to suggest that strongly-regular graphs are not GI-complete. Babai showed that there is no functorial reduction from GI to the isomorphism testing of strongly-regular graphs [25]. In the case of QUASIGROUP ISOMORPHISM, Chattopadhyay, Torán, and Wagner improved the upper bound to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. In particular, they showed that GI is not AC^0-reducible to QUASIGROUP ISOMORPHISM [57].

Let us briefly discuss why problems such as QUASIGROUP ISOMORPHISM, LATIN SQUARE ISOTOPY, and isomorphism testing of Steiner 2-designs are important from a complexity-theoretic perspective. Several families of strongly regular graphs (e.g., Latin square graphs, and the block-intersection graphs of Steiner 2-designs) arise naturally from the isomorphism problems for the underlying combinatorial objects. Precisely, many of these problems, including QUASIGROUP ISOMORPHISM, LATIN SQUARE ISOTOPY, and isomorphism testing of Steiner 2-designs, are polynomial-time (and in fact, AC^0) reducible to GI. Polynomial-time solutions for these problems have been elusive. In particular, each of these problems are candidate NP-intermediate problems.

There has been significant work on isomorphism testing of strongly regular graphs. In 1980, Babai used a simple combinatorial approach to test strongly regular graphs on n vertices and degree $\rho < n/2$ in time $\exp(O((n/\rho)\log^2 n))$ [37, 24]. We note that the complement of a strongly-regular graph is strongly regular, hence the assumptions on the degree. As $\rho \geq \sqrt{n-1}$, this gives an algorithm in moderately exponential time $\exp(O(\sqrt{n}\log^2 n))$ for all ρ. Spielman [184] subsequently improved this $\exp(O((n/\rho)\log^2 n))$ bound to $\exp(O(n^{1/3}\log^2 n))$. This improved upon what was at the time, the best known bound of $\exp(O(\sqrt{n\log n}))$ for isomorphism testing of general graphs [33, 21, 201] based on Luks' group theoretic method [151].

We note that Babai's [37] work already handled the case of conference graphs in time $n^{O(\log n)}$. Furthermore, the works of Babai and Spielman sufficed to handle isomorphism testing of strongly regular graphs that satisfy Neumaier's claw bound. Namely, Spielman [184] handled the case of $\rho \in o(n^{2/3})$ in time $\exp(O(n^{1/4}\log^2 n))$, and Babai [37] handled the case of $\rho \in \Omega(n^{2/3})$ in time $\exp(O(n^{1/3}\log^2 n))$.

In the case of Latin square graphs, Miller [160] resolved this in time $n^{O(\log n)}$. Wolf improved the complexity theoretic bound to $\beta_2\mathsf{AC}^1$ (uniform AC^1-circuits that accept $O(\log^2 n)$ existentially quantified non-deterministic bits).

The isomorphism problem for Steiner 2-designs also dates back to Miller [160], who considered the special cases of Steiner triple systems (Steiner $(2,3,n)$-designs), projective planes (Steiner $(2,q+1,q^2+q+1)$-designs), and affine planes (Steiner $(2,q,q^2)$-designs). In the case of Steiner triple systems, Miller leveraged a standard reduction to Quasigroup Isomorphism to obtain an $n^{O(\log n)}$ upper bound. M. Colbourn [61] subsequently extended this result to obtain an $v^{O(\log v)}$ canonization procedure for Steiner $(t,t+1)$-designs.

For the cases of projective and affine planes, Miller [160] showed that isomorphism testing can be done in time $n^{O(\log\log n)}$. In particular, Miller showed that the corresponding structures may be recovered from the block-intersection graphs in polynomial-time, providing the block-intersection graphs are of sufficiently high degree [160]. Thus, isomorphism testing of block-intersection graphs arising from Steiner triple systems can be done in time $n^{O(\log n)}$, and the block-intersection graphs arising from projective and affine planes can be identified in time $n^{O(\log\log n)}$.

In 1983, Babai & Luks showed that Steiner 2-designs with blocks of bounded size admit an $n^{O(\log n)}$ canonization procedure (and hence, isomorphism test). In particular, isomorphism testing of the corresponding block-intersection graphs is solvable in time $n^{O(\log n)}$ [33]. Here, Babai & Luks utilized Luks' group theoretic method [151]. Huber later extended this result to obtain an $n^{O(\log n)}$-

runtime isomorphism test for arbitrary t-designs, where both t and the block size are bounded, as well as the corresponding block-intersection graphs [115].

Spielman solved the general case of strongly-regular graphs of degree $f(n) \sim n^{3/4}$ arising from Steiner 2-designs in time $\exp(O(n^{1/4} \log^2 n))$ [184]. Independently, Babai & Wilmes [36] and Chen, Sun, & Teng [58] exhibited an $n^{O(\log n)}$-runtime isomorphism test for both Steiner 2-designs and the corresponding block-intersection graphs. Babai & Wilmes extended their result to Steiner t-designs. Here, each of these papers utilized the individualize and refine technique [184, 36, 58].

The isomorphism problem for combinatorial designs is another candidate NP-intermediate problem. In the general case, testing whether two combinatorial designs are isomorphic is GI-complete under polynomial-time Turing reductions [62]. Cyclic Steiner 2-designs of prime order are known to admit a polynomial-time isomorphism test [63]. To the best of our knowledge, the complexity of deciding whether two cyclic Steiner 2-designs are isomorphic remains open for arbitrary orders.

1.2 Relation Algebras

The 2-dimensional Weisfeiler–Leman algorithm yields a natural family of relations that are closed under composition. To see this, consider the following. Fix a graph G and consider the stable coloring of 2-WL applied to G. The color classes under the stable coloring form binary relations. Furthermore, as these relations are stable under color refinement, the composition of two such relations yields another (union of) color class(es). Now relational composition is associative, and so we have an algebraic structure. Abstracting the algebraic structure in turn motivates the study of relation algebras. In the setting of WL, it is interesting to ask as to which relation algebras arise in this way on n vertex graphs– see for instance [111, 196, 23]. In Chapter 6, we examine the converse question: given an abstract relation algebra, what is the smallest graph on which it is represented (without restricting as to whether the representation arises via WL). We note that certain families of relation algebras arising from so called *Ramsey schemes* [67, 137, 154, 14, 9] also have close connections with other combinatorial structures such as association schemes, coherent configurations, and permutation groups [67, 66].

We now more generally discuss the representation question. Given a class of finite algebraic structures, it is natural to ask which members can be *instantiated* or *represented* over a finite set S, where there exist natural operations on S corresponding to the operations of the algebraic structure. In the setting of finite groups, the representation question is answered by Cayley's theorem: every finite group can be instantiated as a finite permutation group. Similarly, Stone's representation theorem provides that every Boolean algebra is isomorphic to a Boolean algebra of sets. In this thesis, we consider the class of finite relation algebras, which are Boolean lattices that satisfy certain equational axioms that capture the notion of relational composition (see Section 2.11 for a more precise formulation). Here, we can ask not only about whether finite representations exist, but also as to the minimum sized representation. While there has been some work on the minimum size *faithful* permutation representation of a group (see, e.g., [79] and the references therein). Here, the minimum sized representation serves as a measure of combinatorial complexity of the relation algebras in question.

There exist finite relation algebras that do not admit representations even over infinite sets- see for instance [16] and the citations therein. It is essentially folklore that there are relation algebras that admit representations over infinite sets, but are not finitely representable. The so called *point algebra* is one such example and is essentially well-known amongst relation algebra specialists — see [12, Appendix A] for a proof.

Comer [66, Theorem 5.3] showed that every finite integral relation algebra with a flexible atom (i.e., an atom that does not participate in any forbidden diversity cycles) is representable over a countable set. It is natural to ask whether Comer's result can be strengthened to hold in the setting of finite sets. This is precisely the *Flexible Atom Conjecture*, which states that every finite integral relation algebra with a flexible atom is representable over a finite set. Jipsen, Maddux, & Tuza showed that the finite symmetric integral relation algebras in which every diversity atom is flexible (denoted $\mathfrak{E}_{n+1}(1, 2, 3)$), are finitely representable. In particular, the algebra with n flexible atoms is representable over a set of size $(2 + o(1))n^2$ [127]. We note that if all cycles are present, then all diversity atoms are flexible. Hence, the case considered in [127] is intuitively the *big end* of the Flexible Atom Conjecture.

The other extreme is when just enough cycles are present for one atom to be flexible. This case was handled by Alm, Maddux, & Manske [7], who exhibited a representation of the algebra A_n obtained from splitting the non-flexible diversity atom of the relation algebra 6_7 into n symmetric atoms. In particular, this construction yielded a representation of $A_2 = 32_{65}$ over a set of size $416,714,805,914$ (here, we use Maddux's [153] numbering for relation algebras such as 6_7 and 32_{65}). The combinatorial complexity of 32_{65} has received much attention. Dodd & Hirsch [75] subsequently improved the upper bound of the minimum representation size of 32_{65} to $63,432,274,896$. This was subsequently improved to 8192 by Alm & Sexton (unpublished [10]), and later 3432 by Alm & Andrews [11]. In Chapter 6, we will investigate the minimum-sized square representation of not only 32_{65}, but A_n for all $n \geq 2$. Let $f(n)$ denote the minimum-sized representation of A_n. We will show that:

$$2n^2 + \Omega(n\sqrt{\log n}) \leq f(n) \leq 2n^{6+o(1)}.$$

Prior to the work in this thesis, only the trivial lower bound of $f(n) \geq n^2 + 2n + 3$ was known.

Chapter 2

Preliminaries

In this chapter, we introduce key background from group theory, isomorphism testing, and computational complexity.

Notation 2.0.1. Let $n \in \mathbb{N}$. We use $[n]$ to denote $\{1, \ldots, n\}$, with the convention that $[0] := \emptyset$.

Notation 2.0.2. Let S be a set, and let $k \in \mathbb{N}$. We use $\binom{S}{k}$ to denote the set of k-element subsets of S.

Notation 2.0.3. Let Σ be a finite set. We refer to Σ as an *alphabet.* Denote Σ^* to be the set of all *finite* strings over Σ. Precisely:

$$\Sigma^* := \{\epsilon\} \cup \bigcup_{i \in \mathbb{Z}^+} \Sigma^i,$$

where we denote ϵ to be the empty string or the string of length 0.

2.1 Graph Theory

Here, we recall preliminary notions from graph theory. A *simple, undirected graph* $G(V, E)$ consists of a set of elements V, which we refer to as vertices, together with a set of edges $E \subseteq \binom{V}{2}$. For an edge $\{u, v\}$, we write uv. Unless otherwise stated, we will only consider simple, undirected graphs and will refer to them as *graphs.* To avoid ambiguity, the vertex set of the graph G will be frequently referred to as $V(G)$. Similarly, the edge set of the graph G will be referred to as $E(G)$.

For a vertex $v \in V(G)$, let $N(v) = \{u : uv \in E(G)\}$ be the *neighbors* of v. The *degree* of v is denoted $\deg(v) := |N(v)|$. We say that G is *regular* if $\deg(v) = \deg(u)$ for all $u, v \in V(G)$. A *strongly regular graph* with parameters (n, k, λ, μ) is a simple, undirected k-regular, n-vertex graph $G(V, E)$ where any two adjacent vertices share λ neighbors, and any two non-adjacent vertices share μ neighbors. The complement of a strongly regular graph is also strongly regular, with parameters $(n, n-k-1, n-2-2k+\mu, n-2k+\lambda)$.

Let G, H be graphs. We say that G and H are *isomorphic*, denoted $G \cong H$, if there exists a bijection $\varphi : V(G) \to V(H)$ such that:

$$uv \in E(G) \iff \varphi(u)\varphi(v) \in E(H).$$

Let G be a graph, and suppose we color the edges of G either red or blue. The classical Ramsey number $R(m, k)$ is the minimum number n, such that for any graph G on at least n vertices and any coloring $\varphi : E(G) \to \{\text{red}, \text{blue}\}$, there exists either a red clique of size m or a blue independent set of size k. The Ramsey number $R(m, k)$ is known to exist and be finite [169].

2.2 Group Theory

We assume familiarity with group theory at the level of a standard undergraduate Algebra course. Unless stated otherwise, all groups are assumed to be finite and represented by their multiplication (a.k.a. Cayley) tables. For a group of order n, the Cayley table has n^2 entries, each represented by a binary string of size $\lceil \log_2(n) \rceil$. For an element g in the group G, we denote the *order* of g as $|g|$. We use $d(G)$ to denote the minimum size of a generating set for the group G.

The *socle* of a group G, denoted $\text{Soc}(G)$, is the subgroup generated by the minimal normal subgroups of G. If G has no Abelian normal subgroups, then $\text{Soc}(G)$ decomposes as the direct product of non-Abelian simple factors. The *normal closure* of a subset $S \subseteq G$, denoted $\text{ncl}(S)$, is the smallest normal subgroup of G that contains S.

Semidirect Products. We say that a normal subgroup $N \trianglelefteq G$ *splits* in G if there exists a subgroup $H \leq G$ such that $H \cap N = \{1\}$ and $G = HN$. The conjugation action of H on N allows us to express multiplication of G in terms of pairs $(h, n) \in H \times N$. We note that the conjugation action of H on N induces a group homomorphism $\theta : H \to \text{Aut}(N)$ mapping $h \mapsto \theta_h$, where $\theta_h : N \to N$ sends $\theta_h(n) = hnh^{-1}$. So given (H, N, θ), we may define the group $H \ltimes_\theta N$ on the set $\{(h, n) : h \in H, n \in N\}$ with the product $(h_1, n_1)(h_2, n_2) = (h_1h_2, \theta_{h_2^{-1}}(n_1)n_2)$. We refer to the decomposition G into $H \ltimes_\theta N$ as a *semidirect product* demoposition. When the action θ is understood, we simply write $H \ltimes N$.

We are particularly interested in the case where $\gcd(|G/N|, |N|) = 1$; such an N is called a **normal Hall subgroup**. One of the key theorems for such groups is:

Theorem 2.2.1 (Schur–Zassenhaus (see, e. g., [171, (9.1.2)])). *Let G be a finite group of order n, and let N be a normal Hall subgroup. Then there exists a complement $H \leq G$, such that $gcd(|H|, |N|) = 1$ and $G = H \ltimes N$. Furthermore, if H and K are complements of N, then H and K are conjugate.*

We now recall some key notions regarding Cayley graphs.

Definition 2.2.2. Let Γ be a group, and let $S \subseteq \Gamma$ such that $1 \not\in S$ and $S = S^{-1}$. The *undirected Cayley graph* $\text{Cay}(\Gamma, S)$ has vertex set Γ and edge set $E = \{\{g, h\} : (\exists s \in S)[gs = h]\}$, where here, $gs = h$ denotes multiplication in the group.

We will use the following standard observation a few times:

Fact 2.2.3. *Let $G = \langle g_1, \ldots, g_d \rangle$. Then every element of G can be written as a word in the g_i of length at most $|G|$.*

Proof. Consider the Cayley graph of G with generating set $g_1, \ldots, g_d$. Words correspond to walks in this graph. We need only consider simple walks — those which never visit any vertex more than once — since if a walk visits a group element g more than once, then the part of that walk starting and ending at g is a word that equals the identity element, so it can be omitted. But the longest simple walk is at most the number of vertices, which is $|G|$. □

2.3 Quasigroups

A *quasigroup* consists of a set G and a binary operation $\star : G \times G \to G$ satisfying the following. For every $a, b \in G$, there exist unique x, y such that $a \star x = b$ and $y \star a = b$. When the multiplication operation is understood, we simply write ax for $a \star x$.

Unless otherwise stated, all quasigroups are assumed to be finite and represented using their Cayley (multiplication) tables. For a quasigroup of order n, thet Cayley table has n^2 entries, each represented by a string of size $\lceil \log_2(n) \rceil$.

A *Latin square* of order n is an $n \times n$ matrix L where each cell $L_{ij} \in [n]$, and each element of $[n]$ appears exactly once in each row or each column. Latin squares are precisely the Cayley tables corresponding to quasigroups. An isotopy of Latin squares L_1 and L_2 is most easily defined in terms of an isotopy of the corresponding quasigroups Q_1 and Q_2: an *isotopy* is an ordered triple (α, β, γ), where $\alpha, \beta, \gamma : L_1 \to L_2$ are bijections satisfying the following: whenever $ab = c$ in L_1, we have that $\alpha(a)\beta(b) = \gamma(c)$ in L_2. Alternatively, we may view α as a permutation of the rows of L_1, β as a permutation of the rows of L_2, and γ as a permutation of the values in the table. Here, L_1 and L_2 are isotopic precisely if (i) the (i, j) entry of L_1 is the $(\alpha(i), \beta(j))$ entry of L_2, and (ii) x is the (i, j) entry of L_1 if and only if $\gamma(x)$ is the $(\alpha(i), \beta(j))$ entry of L_2. We will frequently abuse notation by treating the Latin square and its corresponding quasigroup as interchangeable.

As quasigroups are non-associative (which as usual in this corner of mathematics, indicates that quasigroups are not necessarily associative rather than necessarily not associative), the parenthesization of a given expression may impact the resulting value. We restrict attention to balanced parenthesizations, which ensure that words of the form $g_0 g_1 \cdots g_k$ are evaluated using a balanced binary tree with $k+1$ leves. As every quasigroup is generated by a set of size at most $\log_2 n$ [160], this tree has depth $O(\log \log n)$.

For a sequence $S := (s_0, s_1, \ldots, s_k)$ from a quasigroup, define:

$$\text{Cube}(S) = \{s_0 s_1^{e_1} \cdots s_k^{e_k} : e_1, \ldots, e_k \in \{0, 1\}\}.$$

We say that S is a *cube generating sequence* if Cube(S) contains every element of the quasigroup. Every quasigroup is known to admit a cube generating sequence of size $O(\log n)$ [57].

For a given Latin square L of order n, we associate a *Latin square graph* $G(L)$ that has n^2 vertices; one for each triple (a, b, c) that satisfies $ab = c$. Two vertices (a, b, c) and (x, y, z) are adjacent in $G(L)$ precisely if $a = x$, $b = y$, or $c = z$. Miller showed that two Latin squares L_1 and L_2 are main-class isotopic if and only if $G(L_1) \cong G(L_2)$ [160]. A Latin square graph on n^2 vertices is a strongly regular graph with parameters $(n^2, 3(n-1), n, 6)$. Conversely, a strongly regular graph with these same parameters $(n^2, 3(n-1), n, 6)$ is called a *pseudo-Latin square graph.* Bruck showed that for $n > 23$, a pseudo-Latin square graph is a Latin square graph [52]. While we are not aware as to whether this bound can be improved, it is worth pointing out that pseudo-Latin square graphs which are not also Latin square graphs exist- take for instance, the Shrikhande graph [179].

Albert showed that a quasigroup Q is isotopic to a group G if and only if Q is isomorphic to G. In general, isotopic quasigroups need not be isomorphic [6].

2.4 Combinatorial Designs

Let $t \leq k \leq v$ and λ be positive integers. A (t, k, λ, v) design is an incidence structure $\mathcal{D} = (X, \mathcal{B}, I)$. Here, X denotes our set of v points, $\mathcal{B}$ is a subset of $\binom{X}{k}$, and each t-element subset of X belongs to exactly λ elements of $\mathcal{B}$. The elements of $\mathcal{B}$ are called *blocks.* Now I is the point-block incidence relation, where $(x, B) \in X \times B$ belongs to I precisely if $x \in B$. If $t < k < v$, we say that the design is *non-trivial.* If $\lambda = 1$, the design is referred to as a *Steiner design.* We denote Steiner designs as (t, k, v)-designs when we want to specify v the number of points, or Steiner (t, k)-designs when referring to a family of designs. We note that Steiner $(2, 3)$-designs are known as *Steiner triple systems.* We assume that designs are given by the point-block incidence matrix.

Let $\mathcal{D} = (X, \mathcal{B}, I)$ be a design, and let $A \subseteq X$. The *derived design* at A, denoted $\mathcal{D}(A)$, has the set of points $X \setminus A$ and blocks $\{B \setminus A : B \in \mathcal{B}, A \subsetneq B\}$. If $\mathcal{D}$ is a Steiner (t, k, v)-design, then $\mathcal{D}(A)$ is a Steiner $(t - |A|, k - |A|, v - |A|)$-design.

For a design $\mathcal{D} = (X, \mathcal{B}, I)$, we may define a *block intersection graph* (also known as a *line graph*) $G(V, E)$, where $V(G) = \mathcal{B}$ and two blocks B_1, B_2 are adjacent in G if and only if $B_1 \cap B_2 \neq \emptyset$. In the case of a Steiner 2-design, the block-intersection graph is strongly regular. For Steiner triple systems, the block-intersection graphs are strongly regular with parameters $(n(n-1)/6, 3(n-3)/2, (n+3)/2, 9)$. Conversely, strongly regular graphs with the parameters $(n(n-1)/6, 3(n-$

$3)/2, (n+3)/2, 9)$ are referred to as *pseudo-STS graphs.* Bose showed that pseudo-STS graphs graphs with strictly more than 67 vertices are in fact STS graphs [46]. We are not aware as to whether this bound is tight.

A *net* or *partial geometry* of order $n \geq 1$ and degree $k \geq 1$, denoted $\mathcal{N}(n, k)$, consists of a set P of n^2 points and a set L of kn lines satisfying the following.

(a) Each line in L has exactly n points.

(b) Each point in P lies on exactly k distinct lines.

(c) The kn lines of L fall into k parallel classes. No two lines in the same parallel class intersect. If $\ell_1, \ell_2 \in L$ are in different parallel classes, then ℓ_1 and ℓ_2 share exactly one common point.

Necessarily, $k \leq n+1$. If $k = n+1$, then the net is called an *affine plane* of order n. A *projective plane* is the extension of an affine plane $\mathcal{N}$ by adding $n+1$ new points $p_1, \ldots, p_{n+1}$ (one per parallel class). The point p_i is then added to each line of the ith parallel class. We finally add a new line containing $\{p_1, \ldots, p_{n+1}\}$. We may recover an affine plane from a projective plane by removing a line and the associated points from the projective plane. Note that projective planes are Steiner $(2, q+1, q^2+q+1)$-designs, and affine planes are Steiner $(2, q, q^2)$-designs. When $k = 3$, the net corresponds to a Latin square (quasigroup) in the following manner: one parallel class specifies the rows, a second specifies the columns, and the third specifies the values. We assume that nets are given by the point-block incidence matrix.

Given a net $\mathcal{N}(n, k)$ where $k \leq n$, we may construct a *net graph* of order n and degree k $G(V, E)$ on n^2 vertices corresponding to the points in $\mathcal{N}$. Two vertices are adjacent in G if their corresponding points in $\mathcal{N}$ determine a line. We note that a net graph uniquely determines the net $\mathcal{N}(n, k)$ [52]. Miller showed that, provided $n > (k-1)^2$, this equivalence holds under polynomial-time reductions [160]. A net graph is strongly-regular with parameters $(n^2, k(n-1), n-2+(k-1)(k-2), k(k-1))$. Conversely, a strongly regular graph with parameters $(n^2, k(n-1), n-2+(k-1)(k-2), k(k-1))$ is referred to as a *pseudo-net graph.* Bruck showed that for n sufficiently large, a pseudo-net graph of order n and degree k is a net graph [52]. We are not aware as to whether this bound is tight.

2.5 Computational Complexity

For a comprehensive overview, we refer the reader to standard references [202, 17, 194]. We assume familiarity with standard notions of Turing machines at the level of an undergraduate Theory of Computation course, including multitape and non-deterministic Turing machines. We begin by recalling the complexity classes $\mathsf{P}, \mathsf{NP}, \mathsf{L}$, and NL.

Time-Based Complexity Classes. Let $T(n) : \mathbb{N} \to \mathbb{N}$. Define $\mathsf{DTIME}(T(n))$ to be the set of languages L, such that there exists a deterministic, multitape Turing machine M such that M decides L, and for all $x \in \{0,1\}^*$, $M(x)$ halts in at most $O(T(|x|))$ steps. The complexity class $\mathsf{NTIME}(T(n))$ is defined analogously, where we instead use a non-deterministic, multitape Turing machine. Define:

$$\mathsf{P} := \bigcup_{k \in \mathbb{N}} \mathsf{DTIME}(n^k).$$

Equivalently, a language L belongs to the complexity class P if there exists a polynomial $p(n)$ depending only on L and a deterministic, multitape Turing machine M such that M decides L and runs in time $O(p(n))$.

We analogously define NP:

$$\mathsf{NP} := \bigcup_{k \in \mathbb{N}} \mathsf{NTIME}(n^k).$$

Equivalently, a language L belongs to the complexity class NP if there exists a polynomial $p(n)$ depending only on L and a multitape deterministic Turing machine M such that for every $x \in L$, there exists a certificate $C \in \{0,1\}^*$ such that $M(x, C) = 1$. Furthermore, for every string $y \notin L$ and every $C' \in \{0,1\}^*$, $M(y, C') = 0$.

Space Complexity. A *logspace transducer* is a 3-tape Turing machine with a read-only input tape, a write-only output tape, and a work tape where only $O(\log n)$ tape cells may be used. Here, n is the length of the input string. The complexity class L is the set of languages decidable by deterministic logspace transducers, and NL is the set of languages decidable by non-deterministic logspace transducers.

Remark 2.5.1. It is well-known, though not obvious, that the composition of two logspace transducers is again a logspace transducer. We refer the reader to [182] for a proof.

Reductions. Given two computational problems, we often wish to determine which is more difficult. To this end, we introduce several notions of reduction. We restrict attention to decision problems (languages).

Definition 2.5.2. Let L_1, L_2 be languages. We say that L_1 is *many-one* reducible to L_2 if there exists a computable function $\varphi : \{0,1\}^* \to \{0,1\}^*$ such that the following conditions hold:

- φ preserves *yes* instances; that is, $x \in L_1 \implies \varphi(x) \in L_2$; and
- φ preserves *no* instances; that is, $x \not\in L_1 \implies \varphi(x) \not\in L_2$.

Here, we write $L_1 \leq_m L_2$. Now let $\mathcal{C}$ be a complexity class. If φ is $\mathcal{C}$-computable, we write $L_1 \leq_m^{\mathcal{C}} L_2$ to indicate this.

We now turn towards recalling the notion of a Turing reduction. We first recall the notion of an oracle Turing machine. An *oracle Turing machine* is a Turing machine with a separate oracle tape and oracle query state. When enters the query state, it transitions to one of two specified states based on whether the string on the oracle tape belongs to the oracle. When the oracle is unspecified, we write $M^{\square}$. When the oracle $\mathcal{O}$ is specified, we write $M^{\mathcal{O}}$.

We now formalize the notion of a Turing reduction. Let L_1, L_2 be languages. Intuitively, we say that L_1 is Turing reducible to L_2 if given an algorithm to decide L_2, then we can design an algorithm to decide L_1. This notion makes precise key barriers to improving the complexity of solving certain problems. For instance, suppose that L_1 is not known to admit an efficient solution. If L_1 can be decided efficiently given an oracle for L_2, then solving L_1 efficiently is an obstacle to solving L_2 efficiently. This is made precise with our next definition.

Definition 2.5.3. Let L_1, L_2 be languages. A *Turing reduction* from L_1 to L_2 is an oracle Turing machine $M^{\square}$ such that M^{L_2} decides L_1. If there exists a Turing reduction from L_1 to L_2, we write $L_1 \leq_T L_2$.

We will be particularly interested in Turing reductions from L_1 to L_2, where M^{L_2} runs in polynomial time. In such instances we write $L_1 \leq_T^{\mathsf{P}} L_2$.

The last notion of reduction that will be used here is a *truth-table* reduction. Rather than providing the original definition, we will instead use the following characterization due to Ladner, Lynch, & Selman [139].

Proposition 2.5.4 ([139, Proposition 3.4])**.** *Let L_1, L_2 be languages. We have that L_1 is polynomial-time truth-table reducible to L_2, denoted $L_1 \leq_{tt}^{\mathsf{P}} L_2$, if and only if there exists an oracle Turing machine M^{L_2} and a polynomial-time computable function f such that M^{L_2} decides L_1 in polynomial-time, and on each input x, M^{L_2} only asks questions from the list $f(x)$.*

We now turn towards formalizing the notion of completeness and hardness for a given complexity class.

Definition 2.5.5. Let $\mathcal{C}$ be a complexity class, and let $\leq_r$ be the order induced by r-reductions. We say that a function f is $\mathcal{C}$-hard under r-reductions if for every $g \in \mathcal{C}$, $g \leq_r f$. If furthermore, $f \in \mathcal{C}$, we say that f is $\mathcal{C}$-complete under $\leq_r$.

Remark 2.5.6. The usual notion of NP-completeness is defined in terms of many-one polynomial-time computable reductions ($\leq_m^{\mathsf{P}}$).

We will on occasion reference the following complexity classes of functions, which contain more than languages (decision problems). Let FP denote the set of functions $f : \{0,1\}^* \to \{0,1\}^*$ computable by deterministic Turing machines that halt in polynomial time, and let FL denote the set of functions $f : \{0,1\}^* \to \{0,1\}^*$ computable by deterministic logspace transducers.

Circuit Complexity. We now turn towards introducing notions from circuit complexity. Here, we consider Boolean circuits over the gates $\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$, and $\mathsf{Majority}$, where $\mathsf{Majority}(x_1, \ldots, x_n) = 1$ if and only if $\geq n/2$ of the inputs are 1. Otherwise, $\mathsf{Majority}(x_1, \ldots, x_n) = 0$.

Definition 2.5.7. Fix $k \geq 0$. We say that a language L belongs to uniform NC^k if there exist a family of circuits $(C_n)_{n \in \mathbb{N}}$ over the $\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$ gates such that the following hold:

- The $\mathsf{AND}, \mathsf{OR}$ 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 $C_{|x|}(x) = 1$.

The complexity class AC^k is defined analogously as NC^k, except that the $\mathsf{AND}, \mathsf{OR}$ gates are permitted to have unbounded fan-in. That is, a single AND gate can compute an arbitrary conjunction, and a single OR gate can compute an arbitrary disjunction. The complexity class TC^k is defined analogously as AC^k, except that our circuits are now permitted $\mathsf{Majority}$ gates of unbounded fan-in. The complexity class SAC^k is defined analogously to AC^k, except that the AND gates have bounded fan-in (while the OR gates may still have unbounded fan-in). For every k, the following containments are well-known:

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

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

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

The complexity class FOLL is the set of languages decidable by uniform circuit families with AND, OR, and NOT gates of depth $O(\log \log n)$, polynomial size, and unbounded fan-in. It is known that $\mathsf{AC}^0 \subsetneq \mathsf{FOLL} \subsetneq \mathsf{AC}^1$, and it is open as to whether FOLL is contained in NL [38].

The complexity class MAC^0 is the set of languages decidable by constant-depth uniform circuit familes with a polynomial number of AND, OR, and NOT gates, and at most one $\mathsf{Majority}$ gate. The class MAC^0 was introduced (but not so named) in [19], where it was shown that $\mathsf{MAC}^0 \subsetneq \mathsf{TC}^0$. This class was subsequently given the name MAC^0 in [126].

For a complexity class $\mathcal{C}$, we define $\beta_i\mathcal{C}$ 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 $(x, y) \in L'$. Chattopadhyay, Torán, and Wagner [57] established the following.

Theorem 2.5.8 ([57, Theorem 4.1]). *For any $i, c \geq 0$, $\beta_i\mathsf{FO}((\log\log n)^c)$ cannot compute* Parity.

2.6 Colored Graphs

Let $k \in \mathbb{N}$, and let Γ be a graph. A *k-coloring*[1] over Γ is a map $\gamma : V(\Gamma)^k \to \mathcal{K}$, where $\mathcal{K}$ is our finite set of colors. A k-coloring partitions $V(\Gamma)^k$ into color classes. When $k = 1$, we refer to the coloring as an *element coloring.* For another natural number $m < k$, a k-coloring $\gamma^{(k)} : V(G)^k \to \mathcal{K}$ induces an m-coloring $\gamma^{(m)} : V(G)^k \to \mathcal{K}$ via:

$$\gamma^{(m)}((g_1, \ldots, g_m)) := \gamma^{(k)}((g_1, \ldots, g_m, g_m, \ldots, g_m)).$$

Definition 2.6.1. A *colored graph* is a pair (Γ, γ), where Γ is a graph and $\gamma : V(G) \to \mathcal{K}$ is an element-coloring. We say that two colored graphs $(\Gamma_1, \gamma_1), (\Gamma_2, \gamma_2)$ are *isomorphic* if there is a graph isomorphism $\varphi : V(\Gamma_1) \to V(\Gamma_2)$ that respects the colorings; that is, $\gamma_2 \circ \varphi = \gamma_1$. We define $\text{Aut}_\gamma(\Gamma) = \{\varphi \in \text{Aut}(\Gamma) : \gamma \circ \varphi = \gamma\}$.

2.7 Weisfeiler–Leman

In this section, we introduce the Weisfeiler–Leman algorithm in both the settings of graphs and groups. We refer the reader to standard references on Weisfeiler–Leman and Descriptive Complexity Theory [147, 97, 76, 119]. We begin by recalling the Weisfeiler–Leman algorithm for (colored) graphs, which computes an isomorphism-invariant coloring. Let (Γ, χ) be a colored graph, and let $k \geq 2$ be an integer. The k-dimensional Weisfeiler–Leman, or k-WL, algorithm begins by constructing an initial coloring $\chi_0 : V(\Gamma)^k \to \mathcal{K}$, where $\mathcal{K}$ is our set of colors, by assigning each

[1] Not to be confused with the usual "proper k-coloring" of a graph, that is, an assignment of one out of k colors to each vertex such that no two adjacent vertices receive the same color. Despite this terminological overloading, we stick with this terminology for consistency with [49].

k-tuple a color based on its isomorphism type. That is, two k-tuples $(v_1, \ldots, v_k)$ and $(u_1, \ldots, u_k)$ receive the same color under χ_0 if and only if the following conditions hold:

- The map $v_i \mapsto u_i$ (for all $i \in [k]$) is an isomorphism of the induced subgraphs $\Gamma[\{v_1, \ldots, v_k\}]$ and $\Gamma[\{u_1, \ldots, u_k\}]$,
- For all i, j, $v_i = v_j \Leftrightarrow u_i = u_j$, and
- For all i, $\chi(v_i) = \chi(u_i)$.

For $r \geq 0$, the coloring computed at the rth iteration of Weisfeiler–Leman is refined as follows. For a k-tuple $\overline{v} = (v_1, \ldots, v_k)$ and a vertex $x \in V(\Gamma)$, define

$$\overline{v}(v_i/x) = (v_1, \ldots, v_{i-1}, x, v_{i+1}, \ldots, v_k).$$

The coloring computed at the $(r+1)$st iteration, denoted χ_{r+1}, stores the color of the given k-tuple $\overline{v}$ at the rth iteration, as well as the colors under χ_r of the k-tuples obtained by substituting a single vertex in $\overline{v}$ for another vertex x. We examine this multiset of colors over all such vertices x. This is formalized as follows:

$$\chi_{r+1}(\overline{v}) =(\chi_r(\overline{v}), \{\!\!\{(\chi_r(\overline{v}(v_1/x)), \ldots, \chi_r(\overline{v}(v_k/x)) \big| x \in V(\Gamma)\}\!\!\}),$$

where $\{\!\!\{\cdot\}\!\!\}$ denotes a multiset. Note that the coloring χ_r computed at iteration r induces a partition of $V(\Gamma)^k$ into color classes.

The *Color Refinement* (or 1-dimensional Weisfeiler–Leman) algorithm works similarly by iteratively coloring the vertices of G in an isomorphism invariant manner. The algorithm begins by assigning each vertex an initial color based on their degree; that is, $\chi_0(v) = \deg(v)$. Now for $r \geq 0$, we refine the coloring in the following manner:

$$\chi_{r+1}(u) = (\chi_r(u), \{\{\chi_r(v) : v \in N(u)\}\}),$$

where $\{\{\cdot\}\}$ denotes a multiset. The *count-free* variant of Color Refinement works analogously, except the refinement step considers the set rather than multiset of colors at each round:

$$\chi_{r+1}(u) = (\chi_r(u), \{\chi_r(v) : v \in N(u)\}).$$

The Weisfeiler–Leman algorithm terminates when this partition is not refined, that is, when the partition induced by χ_{r+1} is identical to that induced by χ_r. The final coloring is referred to as the *stable coloring*, which we denote $\chi_\infty := \chi_r$.

Weisfeiler–Leman is employed as an isomorphism test on colored graphs (Γ_1, χ_1) and (Γ_2, χ_2) by running the algorithm on the disjoint union $\Gamma_1 \dot{\cup} \Gamma_2$. We compare the multiset of colors for $V(\Gamma_1)^k$ and $V(\Gamma_2)^k$. If they differ at the end of a given round r, then we may conclude that $\Gamma_1 \not\cong \Gamma_2$.

For fixed k, k-WL has runtime $O(n^{k+1} \log(n))$, where n is the number of vertices [120, 121]. Grohe & Verbitsky also observed that each round of k-WL can be implemented using a TC^0 circuit [104] (see Section 2.10 for more discussion). In particular, controlling the number of rounds may yield improved complexity-theoretic upper bounds for classes contained within P.

Remark 2.7.1. The Weisfeiler–Leman algorithm is classically defined on ordinary graphs. We may view these input graphs as colored with all vertices having the same color (prior to the start of the algorithm).

Brachter & Schweitzer [48] introduced three variants of WL for groups, as well as notions of colored groups. We first introduce the notions of Weisfeiler–Leman for groups. We state these versions for uncolored groups. However, if initial colorings are provided, we may take them into account in the analogous manner as for graphs. WL Versions I and II are both executed directly on the groups, where k-tuples of group elements are initially colored. For WL Version I, two k-tuples $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ receive the same initial color iff (a) for all $i, j, \ell \in [k]$, $g_i g_j = g_\ell \iff h_i h_j = h_\ell$, and (b) for all $i, j \in [k]$, $g_i = g_j \iff h_i = h_j$. Here, we say that $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ are *partially isomorphic*.

For WL Version II, $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ receive the same initial color iff the map $g_i \mapsto h_i$ for all $i \in [k]$ extends to an isomorphism of the generated subgroups $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1, \ldots, h_k \rangle$. Here, we say that $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ are *marked isomorphic*.

For both WL Versions I and II, refinement is performed in the classical manner as for graphs.

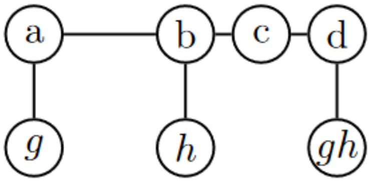

Figure 2.1: The multiplication gadget that encodes the group multiplication $g \cdot h$ [48].

Namely, for a given k-tuple $\overline{g}$ of group elements,

$$\chi_{r+1}(\overline{g}) = (\chi_r(\overline{g}), \{\!\{(\chi_r(\overline{g}(g_1/x)), \ldots, \chi_r(\overline{g}(g_k/x)) \big| x \in G\}\!\}).$$

WL Version III works as follows. Given the Cayley table for a group G, we first apply a reduction from GpI to GI in the setting of simple, undirected graphs. We then apply the standard k-WL for graphs and pull back to a coloring on G^k.

Given the Cayley table for a group G, we construct the graph Γ_G as follows. We begin with a set of isolated vertices, corresponding to the elements of G. For each pair of elements $(g, h) \in G^2$, we add a multiplication gadget $M(g, h)$, which is constructed as follows (see Figure 2.1).

- We add vertices $a_{gh}, b_{gh}, c_{gh}, d_{gh}$.
- We add the edges:

$$\{g, a_{gh}\}, \{h, b_{gh}\}, \{b_{gh}, c_{gh}\}, \{c_{gh}, d_{gh}\}, \{gh, d_{gh}\}.$$

Observe that Γ_G has $\Theta(|G|^2)$ vertices. By consideration of vertex degrees, we also note that G forms a canonical subset of $V(\Gamma_G)$, in that any isomorphism between Γ_G and Γ_H must map (setwise) $G \mapsto H$.

Remark 2.7.2. We note that the construction of Γ_G can be done in AC^0. We may store Γ_G as an adjacency matrix. Adding a single edge can be done using an AC^0 circuit. Each edge can be added independently and in parallel; and thus, without increasing the depth of the circuit. As $|\Gamma_G| \in \Theta(|G|^2)$, we need only a polynomial number of gates to add the appropriate edges. So Γ_G can be constructed using an AC^0 circuit.

In the setting of groups, Brachter & Schweitzer [49] defined notions of colored groups in analogue of Weisfeiler–Leman Versions I and II [48]. Namely, we color the group elements directly as we do vertices in the setting of graphs.

It is also possible to define a notion of colored groups in the case of Weisfeiler–Leman Version III. Let G be a group, and let $\gamma : G \to \mathcal{K}$ be a coloring. Let Γ_G be the graph produced from G, using the reduction in the classical WL Version III algorithm. We obtain a colored graph Γ_G by constructing a coloring $\gamma' : V(\Gamma_G) \to \mathcal{K}$, where $\gamma'(g) = \gamma(g)$ for all $g \in G$.

To the best of our knowledge, the notion of colored group isomorphism was first introduced by Le Gall & Rosenbaum [142]. Brachter & Schweitzer [49] subsequently introduced a notion of a colored group in close analogue to that of a colored graph.

2.8 Pebbling Game

We recall the Hella style pebble game [109, 110] for WL on graphs. This game is often used to show that two colored graphs (X, χ_X) and (Y, χ_Y) cannot be distinguished by k-WL. The game is an Ehrenfeucht–Fraïssé game, with two players: Spoiler and Duplicator. We begin with $k+1$ pairs of pebbles, which are placed beside the graph. Each round proceeds as follows.

(1) Spoiler picks up a pair of pebbles (p_i, p_i'). These could be pebbles on the graphs or unused pebbles to the side of the board.

(2) We check the winning condition, which will be formalized later.

(3) Duplicator chooses a bijection $f : V(X) \to V(Y)$.

(4) Spoiler places p_i on some vertex $v \in V(X)$. Then p_i' is placed on $f(v)$.

Let $v_1, \ldots, v_m$ be the vertices of X pebbled at the end of step 1 of the given round, and let $v_1', \ldots, v_m'$ be the corresponding pebbled vertices of Y. Spoiler wins precisely if either (i) the map $v_\ell \mapsto v_\ell'$ does not extend to an isomorphism of the induced subgraphs $X[\{v_1, \ldots, v_m\}]$ and $Y[\{v_1', \ldots, v_m'\}]$, or (ii) $\chi_X(v_\ell) \neq \chi_Y(v_\ell')$ for some $\ell \in [m]$. Duplicator wins at the end of the given round otherwise. Spoiler wins, by definition, at round 0 if X and Y do not have the same number of vertices. We note that X and Y are not distinguished by the first r rounds of k-WL if and only if Duplicator wins the first r rounds of the $(k+1)$-pebble game [109, 110, 55].

For groups instead of graphs, Versions I and II of the pebble game are defined analogously, where Spoiler pebbles group elements. Precisely, for colored groups (G, χ_G) and (H, χ_H), each round proceeds as follows.

(1) Spoiler picks up a pair of pebbles (p_i, p_i').

(2) We check the winning condition, which will be formalized later.

(3) Duplicator chooses a bijection $f : G \to H$.

(4) Spoiler places p_i on some vertex $g \in G$. Then p_i' is placed on $f(g)$.

Suppose that $(g_1, \ldots, g_\ell) \mapsto (h_1, \ldots, h_\ell)$ have been pebbled. In Version I, Duplicator wins at the given round if this map satisfies the initial coloring condition of WL Version I: (a) for all $i, j, m \in [\ell]$, $g_i g_j = g_m \iff h_i h_j = h_m$, (b) for all $i, j \in [\ell]$, $g_i = g_j \iff h_i = h_j$, and (c) for all $i \in [\ell]$, $\chi_G(g_i) = \chi_H(h_i)$. In Version II, Duplicator wins at the given round if (i) the map $(g_1, \ldots, g_\ell) \mapsto (h_1, \ldots, h_\ell)$ extends to an isomorphism of the generated subgroups $\langle g_1, \ldots, g_\ell \rangle$ and $\langle h_1, \ldots, h_\ell \rangle$, and (ii) for all $i \in [\ell]$, $\chi_G(g_i) = \chi_H(h_i)$. Brachter & Schweitzer established that for $J \in \{I, II\}$, (k, r)-WL Version J is equivalent to version J of the $(k+1)$-pebble, r-round pebble game [48].

Remark 2.8.1. In our work, we explicitly control for both pebbles and rounds. In our theorem statements, we state explicitly the number of pebbles on the board at the end of the given round. So if Spoiler can win with k pebble pairs on the board, then we are playing in the $(k+1)$-pebble game. Note that k-WL corresponds to k-pebbles on the board.

The pebble game for graphs is the same pebble game used to analyze the k-WL Version III algorithm for groups. Note that placing a pebble pair on vertices corresponding to the multiplication gadgets $M(g_1, g_2)$ and $M(h_1, h_2)$ (but not on the vertices corresponding to the group elements) induces a pairing of group elements. Here, we adopt the convention from Brachter & Schweitzer [48] in saying that $(g_1, g_2) \mapsto (h_1, h_2)$ are *implicitly pebbled* if a pebble is placed on a non-group-element vertex of the multiplication gadget $M(g_1, g_2)$. Pebbling such a vertex is nearly as strong as pebbling two pairs of group elements simultaneously, though there is some subtlety. Namely, unless a non-group element vertex on $M(g_1, g_2)$ is pebbled, Duplicator need not select bijections $f : G \to H$ that map $M(g_1, g_2) \mapsto M(f(g_1), f(g_2))$. This distinction led us to our results in Chapter 4.

Brachter & Schweitzer showed that Duplicator must respect the multiplication structure of the implicitly pebbled elements, as well as the subgroups induced by the (implicitly) pebbled group elements [48]:

Lemma 2.8.2 (Brachter & Schweitzer [48, Lemma 3.10])**.** *Consider the k-pebble game on graphs Γ_G and Γ_H. If $k \geq 4$ and one of the following happens:*

(a) Duplicator chooses a bijection on $f : V(\Gamma_G) \to V(\Gamma_H)$ with $f(G) \neq H$,

(b) after choosing a bijection $f : V(\Gamma_G) \to V(\Gamma_H)$, there is a pebble pair (p, p') for which pebble p is on some vertex of $M(g_1, g_2)$ that is not a group element and p' is on some vertex of $M(h_1, h_2)$ that is not a group element, but

$$(f(g_1), f(g_2), f(g_1g_2)) \neq (h_1, h_2, h_1h_2),$$

or

(c) the map induced by the group elements pebbled or implicitly pebbled by $k - 2$ pebbles does not extend to an isomorphism between the corresponding generated subgroups,

then Spoiler can win with at most two additional pebbles

Remark 2.8.3. The original statement of [48, Lemma 3.10] did not specify the number of additional rounds. However, in Section 3.2, based on joint work with J.A. Grochow [90], we are able to modify the proof of [48, Lemma 3.10(c)] to show that only $\log_2(n) + O(1)$ additional rounds suffice.

In light of Lemma 2.8.2 (a), we may – even in Version III – simply consider bijections that Duplicator selects on the group elements. That is, we may without loss of generality consider bijections $f : G \to H$. We stress that condition (b) applies only when there is already a pebble on some element of a multiplication gadget. A priori, Duplicator need not select bijections that respect multiplication gadgets. That is, if there is no pebble on a multiplication gadget $M(g_1, g_2)$, then Duplicator need not map $f(M(g_1, g_2)) = M(f(g_1), f(g_2))$. This is the same subtlety regarding implicit pebbling as discussed on the previous page.

We also note that Lemma 2.8.2 (c) provides that if Duplicator does not respect the subgroup structure of the (implicitly) pebbled group elements, then Spoiler can quickly win.

Brachter & Schweitzer [48, Theorem 3.9] also previously showed that WL Version I, II, and III are equivalent up to a factor of 2 in the dimension.

Theorem 2.8.4 ([48, Theorem 3.9]). *Let G and H be groups of order n. Let $k \geq 2$. We have the following.*

(a) If k-WL Version I distinguishes G and H, then k-WL Version II distinguishes G and H.

(b) If k-WL Version II distinguishes G and H, then $(\lceil k/2 \rceil + 2)$-WL Version III distinguishes G and H.

(c) If k-WL Version III distinguishes G and H, then $(2k + 1)$-WL Version I distinguishes G and H.

In Theorem 3.2.1 of Section 3.2, based on joint work with J.A. Grochow [90], we strengthen their analysis to explicitly control for rounds.

2.9 Logics

We recall the central aspects of first-order logic. While we considered Weisfeiler–Leman and the pebble games for colored groups and colored graphs, we consider the logics for only the uncolored setting. We have a countable set of variables $\{x_1, x_2, \ldots, \}$. Formulas are defined inductively. As our basis, $x_i = x_j$ is a formula for all pairs of variables. Now if φ, ψ are formulas, then so are the following: $\varphi \wedge \psi, \varphi \vee \psi, \neg\varphi, \exists x_i\, \varphi$, and $\forall x_i\, \varphi$. Variables are permitted to be reused within nested quantifiers.

The *quantifier depth* of a first-order formula φ, denoted $\text{qd}(\varphi)$, is the depth of the quantifier nesting. More formally, we have:

- If φ has no quantifiers, then $\text{qd}(\varphi) = 0$.
- $\text{qd}(\varphi) = \text{qd}(\neg\varphi)$.
- Let φ_1, φ_2 be first-order formulas. $\text{qd}(\varphi_1 \vee \varphi_2) = \text{qd}(\varphi_1 \wedge \varphi_2) = \max\{\text{qd}(\varphi_1), \text{qd}(\varphi_2)\}$.
- Let Q be a quantifier. We have that $\text{qd}(Qx\, \varphi) = \text{qd}(\varphi) + 1$.

In order to define logics on groups, it is necessary to define a relation that relates the group multiplication. We recall the two different logics introduced by Brachter & Schweitzer [48].

- **Version I:** We add a ternary relation R where $R(x_i, x_j, x_\ell) = 1$ if and only if $x_i x_j = x_\ell$ in the group. In keeping with the conventions of [55], we refer to the first-order logic with relation R as $\mathcal{L}^I$ and its k-variable fragment as $\mathcal{L}^I_k$. We refer to the logic $\mathcal{C}^I$ as the logic obtained by adding counting quantifiers $\exists^{\geq n} x_i\, \varphi$ (there exist at least n distinct x_i that satisfy φ) and $\exists! n\, \varphi$ (there exist exactly n distinct x_i that satisfy φ) and its k-variable fragment as $\mathcal{C}^I_k$. If furthermore we restrict the formulas to have quantifier depth at most r, we denote this fragment as $\mathcal{C}^I_{k,r}$.
- **Version II:** We add a relation R, defined as follows. Let $w \in (\{x_{i_1}, \ldots, x_{i_t}\} \cup \{x_{i_1}^{-1}, \ldots, x_{i_t}^{-1}\})^*$. We have that $R(x_{i_1}, \ldots, x_{i_t}; w) = 1$ if and only if multiplying the group elements according to w yields the identity. For instance, $R(a, b; [a, b])$ holds precisely if a, b commute. Again, in keeping with the conventions of [55], we refer to the first-order logic with relation R as $\mathcal{L}^{II}$ and its k-variable fragment as $\mathcal{L}^{II}_k$. We refer to the logic $\mathcal{C}^{II}$ as the logic obtained by adding counting quantifiers $\exists^{\geq n} x_i\, \varphi$ and $\exists! n\, \varphi$ and its k-variable fragment as $\mathcal{C}^{II}_k$. If

furthermore we restrict the formulas to have quantifier depth at most r, we denote this fragment as $\mathcal{C}_{k,r}^{II}$.

Remark 2.9.1. Brachter & Schweitzer [48] refer to the logics with counting quantifiers as $\mathcal{L}_I$ and $\mathcal{L}_{II}$. We instead adhere to the conventions in [55].

Let $J \in \{I, II\}$. Brachter & Schweitzer [48] established that two groups G, H are distinguished by (k, r)-WL Version J if and only if there exists a formula $\varphi \in \mathcal{C}_{k+1,r}^{J}$ such that $G \models \varphi$ and $H \not\models \varphi$. In the setting of graphs, the equivalence between Weisfeiler–Leman and first-order logic with counting quantifiers is well known [120, 55].

2.10 Weisfeiler–Leman as a Parallel Algorithm

Grohe & Verbitsky [104] previously showed that for fixed k, the classical k-dimensional Weisfeiler–Leman algorithm for graphs can be effectively parallelized. More precisely, each iteration (including the initial coloring) can be implemented using a logspace uniform TC^0 circuit. As they mention [104, Remark 3.4], their implementation works for any first-order structure, including groups. However, because here we have three different versions of WL, we explicitly list out the resulting parallel complexities, which differ slightly between the versions.

- **WL Version I:** Let $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ be two k-tuples of group elements. We may test in AC^0 whether (a) for all $i, j, m \in [k]$, $g_i g_j = g_m \iff h_i h_j = h_m$, and (b) $g_i = g_j \iff h_i = h_j$. So we may decide if two k-tuples receive the same initial color in AC^0. Comparing the multiset of colors at the end of each iteration (including after the initial coloring), as well as the refinement steps, proceed identically as in [104]. Thus, for fixed k, each iteration of k-WL Version I can be implemented using a logspace uniform TC^0 circuit.

- **WL Version II:** Let $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ be two k-tuples of group elements. We may use a result of Tang [185] to test in L whether the map sending $g_i \mapsto h_i$ for all $i \in [k]$

extends to an isomorphism of the generated subgroups $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1, \ldots, h_k \rangle$ (if this map extends to an isomorphism, we say that $(g_1, \ldots, g_k)$ and $(h_1, \ldots, h_k)$ are *marked isomorphic*). So we may decide whether two k-tuples receive the same initial color in L. Comparing the multiset of colors at the end of each iteration (including after the initial coloring), as well as the refinement steps, proceed identically as in [104]. Thus, for fixed k, the initial coloring of k-WL Version II is L-computable, and each refinement step is TC^0-computable.

- **WL Version III:** From Remark 2.7.2, we have that constructing the graph from the Cayley table is AC^0-computable. We now appeal directly to the parallel WL implementation for graphs due to Grohe & Verbitsky [104]. Thus, each iteration of WL Version III can be implemented with a logspace uniform TC^0-circuit, and constructing the graph does not further increase the asymptotic complexity.

2.11 Relation Algebras

We recall preliminary notions from lattice theory and relation algebras. We refer the reader to [158] for background on lattice theory and [153, 88, 113] for background on relation algebras.

Definition 2.11.1. A *lattice* $\langle A, \leq, \vee, \wedge \rangle$ is a poset $(A, \leq)$ together with binary operations $\vee, \wedge : A \times A \to A$ that are both commutative and associative. Furthermore, $\vee, \wedge$ satisfy the following for every $a, b \in A$:

$$a \vee (a \wedge b) = a$$

$$a \wedge (a \vee b) = a.$$

$$a \leq b \iff a = a \wedge b \iff b = a \vee b.$$

We refer to $\vee$ as *join* and $\wedge$ as *meet*. When the lattice is understood, we simply write A. We say that A is *distributive* if it satisfies one (and therefore both) of the distributive laws, that for every

$x, y, z \in A$:

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$$

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z).$$

Definition 2.11.2. A *Boolean lattice* $\langle A, \leq, \vee, \wedge, \neg, 0, 1\rangle$ is a distributive lattice $\langle A, \leq, \vee, \wedge\rangle$ and a unary complementation operation $\neg : A \to A$ satisfying the following, for all $x, y \in A$:

$$x \wedge \neg x = 0$$

$$x \vee \neg x = 1$$

$$x \vee 0 = x$$

$$x \wedge 1 = x$$

$$\neg(\neg x) = x$$

$$\neg(x \vee y) = \neg x \wedge \neg y$$

$$\neg(x \wedge y) = \neg x \vee \neg y.$$

Definition 2.11.3. A *relation algebra* is an algebra $\langle A, \leq, \wedge, \vee, \neg, 0, 1, \circ, \breve{}, 1'\rangle$ that satisfies the following.

- $\langle A, \leq, \wedge, \vee, \neg, 0, 1\rangle$ is a Boolean lattice, with $\neg$ the unary negation operator, 0 the identity for $\vee$, and 1 the identity for $\wedge$.

- $\langle A, \circ, 1'\rangle$ is a monoid, with $1'$ the identity with respect to $\circ$. We refer to $\circ$ as *(relational) composition.* So we have that relational composition is associative, and there is an identity $1'$ with respect to $\circ$.

- $\breve{}$ is the unary converse operation, which is an anti-involution with respect to composition. Namely, $\breve{\breve{a}} = a$ for all $a \in A$, and $\overparen{a \circ b} = \breve{b} \circ \breve{a}$ for all $a, b \in A$.

- Converse and composition both distribute over disjunction. Precisely, for all $a, b, c \in A$, we have that:

$$\widetilde{(a \vee b)} = \breve{a} \vee \breve{b}, \text{ and}$$

$$(a \vee b) \circ c = (a \circ c) \vee (b \circ c).$$

- (Triangle Symmetry) For all $a, b, c \in A$, we have that:

$$(a \circ b) \wedge c = 0 \iff (\breve{c} \circ a) \wedge \breve{b} = 0 \iff (b \circ \breve{c}) \wedge \breve{a} = 0.$$

When the relation algebra is understood, we simply write A rather than $\langle A, \leq, \wedge, \vee, \neg, 0, 1, \circ, \breve{}, 1' \rangle$.

Definition 2.11.4. Let A be a relation algebra. We say that A is *integral* if whenever $x \circ y = 0$, we have that $x = 0$ or $y = 0$.

Definition 2.11.5. Let A be a relation algebra. We say that $a \in A$ is an *atom* if $a \neq 0$ and $b < a \implies b = 0$. Furthermore, we say that a is a *diversity atom* if a also satisfies $a \wedge 1' = 0$. We say that a diversity atom is *symmetric* if $a = \breve{a}$.

In this thesis, attention will be restricted to finite relation algebras. This ensures that every element of a relation algebra can be written as the join of finitely many atoms.

Remark 2.11.6. In the special case when a, b, c are diversity atoms, the Triangle Symmetry axiom defines an equivalence relation on such triples that corresponds to the symmetries of the triangle.

Definition 2.11.7. For atoms a, b, c, the triple (a, b, c) – usually denoted abc – is called a *cycle.* We say that the cycle abc is *forbidden* if $(a \circ b) \wedge c = 0$ and *mandatory* if $a \circ b \geq c$. If a, b, c are diversity atoms, then abc is called a *diversity cycle.*

Any cycle that is not forbidden is mandatory [153, Chapter 6]. We now have the following observation.

Observation 2.11.8. Let A be an integral relation algebra. The composition operation $\circ$ is determined by the mandatory diversity cycles.

Proof. Let $a, b \in A$ be diversity atoms. We note that if $c \in A$ is an atom such that abc forms a diversity cycle, then $a \circ b \geq c$. Now we note that if $b = \breve{a}$, then $a \circ b \geq 1'$. Define:

$$C := \{c \in A : c \text{ is an atom s.t. } a \circ b \geq c\}.$$

By the definition of a mandatory cycle, C contains the diversity atoms c such that abc is a mandatory diversity cycle. Now if $b = \breve{a}$, then $ab1'$ is a mandatory cycle, in which case $1'$ also belongs to C. As any cycle is either mandatory or forbidden, there are no other elements in C. Now as A is finite and a, b are atoms, we have that:

$$a \circ b = \bigvee_{c \in C} c.$$

Now for any $R_1, R_2 \in A$, we have that:

$$R_1 \circ R_2 = \bigvee \{a \circ b : a \leq R_1, b \leq R_2, \text{ and } a, b \text{ are atoms}\}.$$

The result now follows. □

Definition 2.11.9. Let f be a diversity atom. We say that f is *flexible* if for all diversity atoms a, b, we have that abf is mandatory.

Definition 2.11.10. We say that a relation algebra A is *representable* if there exists a set U and an equivalence relation $E \subseteq U \times U$ such that A embeds into

$$\langle 2^E, \subseteq, \cup, \cap, ^c, \circ, ^{-1}, \emptyset, E, \text{Id}_U \rangle.$$

Here, ^{c} is set complementation, and $^{-1}$ is the relational inverse.

We will only be concerned with *simple* relation algebras, in which there exists a set U such that $E = U \times U$. We call such a representation *square*.

Definition 2.11.11. Let A be a finite relation algebra. Denote:

$$\text{Spec}(A) := \{\alpha \leq \omega : A \text{ has a square representation over a set of cardinality } \alpha\}.$$

If Spec(A) contains a natural number, then we say that A admits a finite representation. The minimum element in Spec(A) serves as a measure of combinatorial complexity for the relation algebra.

Definition 2.11.12. Let G be a group. Define the *complex algebra*:

$$\text{Cm}(G) = \langle 2^G, \subseteq, \cup, \cap, ^c, \cdot, ^{-1}, \emptyset, G, \{e\} \rangle,$$

where ^{c} is the setwise complementation, $\cdot : 2^G \times 2^G \to 2^G$ is the map sending :

$$X \cdot Y = \{xy : x \in X, y \in Y, \text{ and } xy \text{ is considered in the group}\},$$

$^{-1} : 2^G \to 2^G$ maps $X \mapsto X^{-1} := \{x^{-1} : x \in X\}$, and e is the identity in the group. A *group representation* of the relation algebra A over the group G is an injective homomorphism $\varphi : A \to \text{Cm}(G)$. Given a group representation φ, we may construct a representation A as follows. Take $U = G$. Now for each $X \subseteq G$ in Im(φ), map X to the relation $\{(u, v) : uv^{-1} \in X\} \subseteq G \times G$.

Remark 2.11.13. In light of Observation 2.11.8, deciding whether an integral relation algebra A admits a finite representation is equivalent to finding a finite complete graph with an appropriate edge coloring. Note that if A is symmetric, we consider the undirected complete graph. If A is not symmetric, then we instead consider the directed complete graph. In this thesis, attention will be restricted to the symmetric case.

Given a complete graph on m vertices, we seek to color the edges using the diversity atoms as colors. Now suppose $a \in A$ is a diversity atom, and uv is an edge colored a (we abuse notation by using the diversity atom as the label for the edge color). If abc is a mandatory diversity cycle, then there must exist a vertex w such that vw is colored b and uw is colored c. If instead abc is a forbidden diversity cycle, then for any vertex w, we have that either vw is not colored b or uw is not colored c. Now if there exists a finite m, then A is finitely representable. In analyzing the combinatorial complexity of a relation algebra, we will be interested in finding the smallest such m.

Chapter 3

Parallel Complexity of Group Isomorphism and Canonization via Weisfeiler–Leman

3.1 Overview

In this chapter, we show that Weisfeiler–Leman serves as a key subroutine in developing efficient parallel isomorphism tests for several families of groups. This is based upon joint work with J.A. Grochow [90].

Brachter & Schweitzer [48] introduced three different versions of WL for groups. While they are equivalent in terms of pebble complexity up to constant factors, their round complexity may differ by up to an additive $O(\log n)$ (see Theorem 2.8.4), and their parallel complexities differ (see Section 2.10). Because of these differences we are careful to specify which version of WL for groups each result uses.

We first examine coprime extensions of the form $H \ltimes N$, where N is Abelian and H is $O(1)$-generated. Qiao, Sarma, & Tang [167] previously established an upper bound of P for isomorphism testing of this family of groups. We improve this bound as follows.

Theorem 3.1.1. *Groups of the form $H \ltimes N$, where N is Abelian, H is $O(1)$-generated, and $|H|$ and $|N|$ are coprime are identified by $(O(1), O(1))$-WL Version II. Consequently, isomorphism between a group of the above form and arbitrary groups can be decided in L.*

We next parallelize a result of Brachter & Schweitzer [49], who showed that Weisfeiler–Leman can identify direct products in polynomial-time provided it can also identify the indecomposable direct factors in polynomial-time. Specifically, we show:

Theorem 3.1.2. *For all $G = G_1 \times \cdots \times G_d$ with the G_i directly indecomposable, and all $k \geq 5$, if $(k, O(\log^c n))$-WL Version II identifies each G_i for some $c \geq 1$, then $(k+1, O(\log^c n))$-WL identifies G.*

More specifically, we show that for $k \geq 5$ and $r(n) \in \Omega(\log n)$, if a direct product G is not distinguished from some group H by (k, r)-WL Version II, then H is a direct product, and there is some direct factor of H that is not distinguished from some direct factor of G by $(k-1, r)$-WL.

Prior to Theorem 3.1.2, the best-known upper bound on computing direct product decompositions was P [198, 130] (we note that [198] works in polynomial-time even in the more succinct and practical model of groups given by generating sets of permutations). While Weisfeiler–Leman does not return explicit direct factors, it can implicitly compute a direct product decomposition in $O(\log n)$ rounds, which is sufficient for parallel isomorphism testing. In light of the parallel WL implementation due to Grohe & Verbitsky, our result effectively provides that WL can decompose direct products in TC^1.

We next consider groups without Abelian normal subgroups. Using the individualize-and-refine paradigm, we obtain a new upper bound of $\mathsf{quasiSAC}^1$ for not only deciding isomorphisms, but also listing isomorphisms. While this does not improve upon the upper bound of P for isomorphism testing [30], this does parallelize the previous bound of $n^{\Theta(\log \log n)}$ runtime for listing isomorphisms [29].

Theorem 3.1.3. *Let G be a group without Abelian normal subgroups, and let H be arbitrary. We can test isomorphism between G and H using an SAC circuit of depth $O(\log n)$ and size $n^{\Theta(\log \log n)}$. Furthermore, all such isomorphisms can be listed in this bound.*

In the case of serial complexity, if we restrict the number of simple direct factors of $\text{Soc}(G)$ to be just slightly less than maximal, even listing isomorphisms can be done in FP [29]. Under the same restriction, we improve this to FL:

Corollary 3.1.4 (Cf. [29, Corollary 4.4])**.** *Let G be a group without Abelian normal subgroups, and let H be arbitrary. Suppose that the number of non-Abelian simple direct factors of $Soc(G)$ is $O(\log n / \log \log n)$. Then we can decide isomorphism between G and H, as well as list all such isomorphisms, in* FL.

It remains open as to whether isomorphism testing of groups without Abelian normal subgroups is even in NC.

Given the lack of lower bounds on GpI, and Grohe & Verbitsky's parallel WL algorithm, it is natural to wonder whether our parallel bounds could be improved. One natural approach to this is via the *count-free* WL algorithm, which compares the set rather than the multiset of colors at each iteration. We show unconditionally that this algorithm fails to serve as a polynomial-time isomorphism test for even Abelian groups.

Theorem 3.1.5. *There exists an infinite family $(G_n, H_n)_{n \geq 1}$ where $G_n \not\cong H_n$ are Abelian groups of the same order and count-free WL requires dimension $\geq \Omega(\log |G_n|)$ to distinguish G_n from H_n.*

Remark 3.1.6. Even prior to [55], it was well-known that the count-free variant of Weisfeiler–Leman failed to place GI into P [120]. In fact, count-free WL fails to distinguish almost all graphs [80, 117], while two iterations of the standard counting 1-WL almost surely assign a unique label to each vertex [32, 31]. In light of the equivalence between count-free WL and the logic FO (first-order logic *without* counting quantifiers), this rules out FO as a viable logic to capture P on unordered graphs. Finding such a logic is a central open problem in Descriptive Complexity Theory. On ordered structures such a logic was given by Immerman [118] and Vardi [190].

Theorem 3.1.5 establishes the analogous result, ruling out FO as a candidate logic to capture P on unordered groups. This suggests that some counting may indeed be necessary to place GpI into P. As DET is the best known lower bound for GI [189], counting is indeed necessary to place GI into P. There are no such lower bound known for GpI. Furthermore, the work of [57] shows that GpI is not hard (under AC^0-reductions) for any complexity class that can compute Parity, such as DET. Determining which families of groups can(not) be identified by count-free WL remains an intriguing open question.

While count-free WL is not sufficiently powerful to compare the multiset of colors, it turns out that $O(\log\log n)$-rounds of count-free $O(1)$-WL Version III will distinguish two elements of different orders. Thus, the multiset of colors computed by the count-free $(O(1), O(\log\log n))$-WL Version III for non-isomorphic Abelian groups G and H will be different. We may use $O(\log n)$ non-deterministic bits to guess the color class where G and H have different multiplicities, and then an MAC^0 circuit to compare said color class. This yields the following.

Theorem 3.1.7. *Abelian Group Isomorphism is in $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$.*

We note that this and Theorem 3.1.3 illustrate uses of WL for groups as a *subroutine* in isomorphism testing, which is how it is so frequently used in the case of graphs. To the best of our knowledge, the only previous uses of WL as a subroutine for GpI were in [146, 50].

Remark 3.1.8. The previous best upper bounds for isomorphism testing of Abelian groups are linear time [129, 193, 174] and $\mathsf{L} \cap \mathsf{TC}^0(\mathsf{FOLL})$ [57]. As $\beta_1\mathsf{MAC}^0(\mathsf{FOLL}) \subseteq \mathsf{TC}^0(\mathsf{FOLL})$, Theorem 3.1.7 improves the upper bound for isomorphism testing of Abelian groups.

Methods. We find the comparison of methods at least as interesting as the comparison of complexity. Here we discuss at a high level the methods we use for each of our three main theorems above, and compare them to the methods of their predecessor results.

For Theorem 3.1.1, its predecessor in Qiao–Sarma–Tang [167] leveraged a result of Le Gall [141] on testing conjugacy of elements in the automorphism group of an Abelian group. (By further delving into the representation theory of Abelian groups, they were also able to solve the case where H and N are coprime and both are Abelian without any restriction on the number of generators; we leave that as an open question in the setting of WL.) Here, we use the pebbling game. Our approach is to first pebble generators for the complement H, which fixes an isomorphism of H. For groups that decompose as a coprime extension of H and N, the isomorphism type is completely determined by the multiplicities of the indecomposable H-module direct summands (Lemma 3.3.3). So far, this is the same group-theoretic structure leveraged by Qiao, Sarma, and Tang [167]. However, we then use the representation-theoretic fact that, since $|N|$ and $|H|$ are coprime, each indecomposable H-module is generated by a single element (Lemma 3.3.4); this is crucial in our setting, as it allows Spoiler to pebble that one element in the WL pebbling game. Then, as the isomorphism of H is fixed, we show that any subsequent bijection that Duplicator selects must restrict to H-module isomorphisms on each indecomposable H-submodule of N that is a direct summand.

For Theorem 3.1.3, solving isomorphism of semisimple groups took a series of two papers [29, 30]. Our result is really only a parallel improvement on the first of these (we leave the second as an open question). In Babai *et al.* [29], they used Code Equivalence techniques to identify semisimple groups where the minimal normal subgroups have a bounded number of non-Abelian simple direct factors, and to identify general semisimple groups in time $n^{O(\log \log n)}$. In contrast, WL—along with individualize-and-refine in the second case—provides a single, combinatorial algorithm that is able to detect the same group-theoretic structures leveraged in previous works to solve isomorphism in these families.

In parallelizing Brachter & Schweitzer's direct product result in Theorem 3.1.2, we use two techniques. The first is simply carefully analyzing the number of rounds used in many of the proofs. In several cases, a careful analysis of the rounds used was not sufficient to get a strong parallel result. In those cases, we use the notion of *rank*, which may be of independent interest and have further uses.

Given a subset C of group elements, the C-rank of $g \in G$ is the minimal word-length over C required to generate g. If C is easily identified by Weisfeiler–Leman, then WL can identify $\langle C \rangle$ in $O(\log n)$ rounds. This is made precise (and slightly stronger) with our Rank Lemma:

Lemma 3.1.9 (Rank lemma). *If $C \subseteq G$ is distinguished by (k, r)-WL, then any bijection f chosen by Duplicator must respect C-rank, in the sense that $\text{rk}_C(g) = \text{rk}_{f(C)}(f(g))$ for all $g \in G$, or Spoiler can win with $k + 1$ pebbles and $\max\{r, \log d + O(1)\}$ rounds, where $d = diam(Cay(\langle C \rangle, C)) \leq |\langle C \rangle| \leq |G|$.*

One application of our Rank Lemma is that WL identifies verbal subgroups where the words are easily identified. Given a set of words $w_1(x_1, \ldots, x_n), \ldots, w_m(\vec{x})$, the corresponding *verbal subgroup* is the subgroup generated by $\{w_i(g_1, \ldots, g_n) : i = 1, \ldots, m, g_j \in G\}$. One example that we use in our results is the commutator subgroup. If Duplicator chooses a bijection $f : G \to H$ such that $f([x, y])$ is not a commutator in H, then Spoiler pebbles $[x, y] \mapsto f([x, y])$ and wins in two additional rounds. Thus, by our Rank Lemma, if Duplicator does not map the commutator subgroup $[G, G]$ to the commutator subgroup $[H, H]$, then Spoiler wins with 1 additional pebble and $O(\log n)$ additional rounds.

Brachter & Schweitzer [49] obtained a similar result about verbal subgroups using different techniques. Namely, they showed that if WL assigns a distinct coloring to certain subsets $S_1, \ldots, S_t$, then WL assigns a unique coloring to the set of group elements satisfying systems of equations over $S_1, \ldots, S_t$. They analyzed the WL colorings directly. As a result, it is not clear how to compose their result with the pebble game. For instance, while their result implies that if Duplicator does not map $f([G, G]) = [H, H]$ then Spoiler wins, it is not clear how Spoiler wins nor how quickly Spoiler can win. Our result addresses these latter two points more directly. In particular, recall that the number of rounds impacts the parallel complexity (see Section 2.10).

3.2 Parallel Equivalence Between WL Versions

In this section, we show that WL Version I-III are equivalent up to a tradeoff of $O(\log n)$ rounds.

Theorem 3.2.1. *Let G and H be groups of order n. Let $k \geq 2, r \geq 1$. We have the following.*

(a) If (k, r)-WL Version I distinguishes G and H, then (k, r)-WL Version II distinguishes G and H.

(b) If (k, r)-WL Version II distinguishes G and H, then $(\lceil k/2 \rceil + 2, 3r + O(\log n))$-WL Version III distinguishes G and H.

(c) If (k, r)-WL Version III distinguishes G and H, then $(2k+1, 2r)$-WL Version I distinguishes G and H.

We begin by strengthening Lemma 2.8.2 (c).

Lemma 3.2.2 (Lemma 2.8.2 (c))**.** *Let G, H be groups of order n. Let $k \geq 4$. Consider the k-pebble game on the graphs Γ_G and Γ_H. Suppose that at most $k - 2$ pebbles on the board. If the map induced by the group elements pebbled or implicitly pebbled does not extend to an isomorphism between the corresponding generated subgroups, then Spoiler can win with 2 additional pebbles and $O(\log n)$ additional rounds.*

Proof of Lemma 2.8.2 (c). By assumption, there are at most $m := 2(k-2)$ implicitly pebbled pairs of group elements corresponding to at most $k - 2$ pebbles currently on the board. Without loss of generality, we may assume there are exactly m such pairs of group elements $(g_1, \ldots, g_m) \mapsto (h_1, \ldots, h_m)$ pebbled. Let $f : G \to H$ by the bijection Duplicator selects. By Lemma 2.8.2 (b), Duplicator must select bijections respecting the pairing induced by implicitly pebbled group elements. By assumption, this correspondence does not extend to an isomorphism of $\langle g_1, \ldots, g_m \rangle$ and $\langle h_1, \ldots, h_m \rangle$. Let $w := g_{i_1} \cdots g_{i_t}$ be a minimal word such that

$$f(w) \neq f(g_{i_1}) \cdots f(g_{i_t}).$$

Spoiler pebbles $w \mapsto f(w)$. Let $f' : G \to H$ be the bijection Duplicator selects on the next round. Suppose that $f'(g_{i_2} \cdots g_{i_t}) = f(g_{i_2} \cdots g_{i_t})$. By the minimality of w, we have that:

$$f(g_{i_2} \cdots g_{i_t}) = f(g_{i_2}) \cdots f(g_{i_t}).$$

So in this case, Spoiler pebbles $g_{i_2} \cdots g_{i_t} \mapsto f'(g_{i_2} \cdots g_{i_t}) = f(g_{i_2} \cdots g_{i_t})$ and wins with 2 additional pebbles and $O(1)$ additional rounds by Lemma 2.8.2 (b).

Suppose instead that $f'(g_{i_2} \cdots g_{i_t}) \neq f(g_{i_2} \cdots g_{i_t})$. Then at least one of the following must hold:

$$f'(g_{i_2} \cdots g_{i_t}) \neq f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \cdot f'(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}),$$

$$f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \neq f'(g_{i_2}) \cdots f'(g_{i_{\lceil t/2 \rceil}}), \text{or}$$

$$f'(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}) \neq f'(g_{i_{\lceil t/2 \rceil+1}}) \cdots f'(g_{i_t}).$$

We now consider the following cases.

- **Case 1:** Suppose that either

  $$f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \neq f'(g_{i_2}) \cdots f'(g_{i_{\lceil t/2 \rceil}}), or$$

  $$f'(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}) \neq f'(g_{i_{\lceil t/2 \rceil+1}}) \cdots f'(g_{i_t}).$$

  Suppose first that:

  $$f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \neq f'(g_{i_2}) \cdots f'(g_{i_{\lceil t/2 \rceil}})$$

  Spoiler pebbles $g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}} \mapsto f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}})$. The alternative case when

  $$f'(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}) \neq f'(g_{i_{\lceil t/2 \rceil+1}}) \cdots f'(g_{i_t})$$

  is handled identically.

- **Case 2:** Suppose that Case 1 does not occur. Then we necessarily have that:

  $$f'(g_{i_2} \cdots g_{i_t}) \neq f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \cdot f'(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}).$$

  In this case, Spoiler begins by pebbling $g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}} \mapsto f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}})$. At the next round, Duplicator must select a bijection $f'' : G \to H$ mapping $f''(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t})$ so that

  $$f'(g_{i_2} \cdots g_{i_t}) = f'(g_{i_2} \cdots g_{i_{\lceil t/2 \rceil}}) \cdot f''(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}).$$

  Otherwise, by Lemma 2.8.2 (b), Spoiler wins with one additional pebble and $O(1)$ additional rounds. But then

  $$f''(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t}) \neq f''(g_{i_{\lceil t/2 \rceil+1}}) \cdots f''(g_{i_t}).$$

  Spoiler pebbles $g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t} \mapsto f''(g_{i_{\lceil t/2 \rceil+1}} \cdots g_{i_t})$.

After at most two rounds, we have pebbled a word w' such that $|w'| \leq |w|/2$. Thus, at most $2\log_2(t) + 1$ rounds are required until we can reduce to Lemma 2.8.2 (b). By Fact 2.2.3, we have $t \leq n$. So only $O(\log n)$ rounds are required. To see that only two additional pebbles are required, at the round after Spoiler pebbles w', Spoiler picks up the pebble pair on $w \mapsto f(w)$. The result now follows. □

To prove Theorem 3.2.1, we follow the strategy of [48, Section 3.5].

Proof of Theorem 3.2.1 .

(a) Suppose that Spoiler wins with k pebbles on the board at round r of Version I of the pebble game. Suppose that $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ have been pebbled. As Spoiler wins, there exist $i, j, m \in [k]$ such that WLOG, $g_i g_j = g_m$ but $h_i h_j \neq h_m$. So the map $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ does not extend to an isomorphism of $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1, \ldots, h_k \rangle$. So Spoiler may use the same strategy in Version II of the pebble game and win with k pebbles and r rounds.

(b) Suppose that Spoiler wins with k pebbles on the board at round r of Version II of the pebble game. Suppose that $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ have been pebbled. As Spoiler wins, the map $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ does not extend to an isomorphism of $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1, \ldots, h_k \rangle$.

Brachter & Schweitzer [48, Lemma 3.11] previously established that, using $\lceil k/2 \rceil$ pebbles and $3r$ rounds in the Version III pebble game, that Spoiler can obtain a configuration $(g_1', \ldots, g_k') \mapsto (h_1', \ldots, h_k')$ that does not extend to an isomorphism of $\langle g_1', \ldots, g_k' \rangle$ and $\langle h_1', \ldots, h_k' \rangle$.

Now by Lemma 2.8.2 (a), we may assume in the Version III pebble game that Duplicator selects bijections $f : V(\Gamma_G) \to V(\Gamma_H)$ that restrict to bijections $f|_G : G \to H$; otherwise, Spoiler may win with 2 pebbles and 2 rounds. Thus, without loss of generality, we may assume that if Duplicator selects $f : V(\Gamma_G) \to V(\Gamma_H)$ at round $1 \leq i \leq r$ of the Version III pebble game that Duplicator selects $f|_G$ at round i of the Version II pebble game. So without loss of generality, we may assume that

$$(g_1', \ldots, g_k') = (g_1, \ldots, g_k), \text{ and}$$
$$(h_1', \ldots, h_k') = (h_1, \ldots, h_k).$$

Now recall that as the map $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ does not extend to an isomorphism of $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1, \ldots, h_k \rangle$, Spoiler wins at round r of the Version II pebble game. By Lemma 2.8.2 (c), Spoiler can win in the Version III pebble game using 2 additional pebbles and $O(\log n)$ additional rounds. The result now follows.

(c) Brachter & Schweitzer [48, Lemma 3.12] showed that (k', r')-WL Version III can be simulated by $(2k' + 1, 3r')$-WL Version I. Now take $k' = \lfloor k/2 \rfloor + 2$ and $r' = 3r + O(\log n)$. The result follows.

□

3.3 Weisfeiler–Leman for Coprime Extensions

In this section, we consider groups that admit a Schur–Zassenhaus decomposition of the form $G = H \ltimes N$, where N is Abelian, and H is $O(1)$-generated and $|H|$ and $|N|$ are coprime. Qiao, Sarma, and Tang [167] previously exhibited a polynomial-time isomorphism test for this family of groups, as well as the family where H and N are arbitrary Abelian groups. This was extended by Babai & Qiao [34] to groups where $|H|$ and $|N|$ are coprime, N is Abelian, and H is an arbitrary group given by generators in any class of groups for which isomorphism can be solved efficiently. Among the class of such groups where H is $O(1)$-generated, we are able to improve the parallel complexity to L via WL Version II.

3.3.1 Additional preliminaries for groups with Abelian normal Hall subgroup

Here we recall additional preliminaries needed for our algorithm in the next section. None of the results in this section are new, though in some cases we have rephrased the known results in a form more useful for our analysis.

Coprime extensions (see Section 2.2) are determined entirely by their actions:

Lemma 3.3.1 (Taunt [186]). *Let $G = H \ltimes_\theta N$ and $\hat{G} = \hat{H} \ltimes_{\hat{\theta}} \hat{N}$. If $\alpha : H \to \hat{H}$ and $\beta : N \to \hat{N}$ are isomorphisms such that for all $h \in H$ and all $n \in N$,*

$$\hat{\theta}_{\alpha(h)}(n) = (\beta \circ \theta_h \circ \beta^{-1})(n),$$

then the map $(h, n) \mapsto (\alpha(h), \beta(n))$ is an isomorphism of $G \cong \hat{G}$. Conversely, if G and $\hat{G}$ are isomorphic and $|H|$ and $|N|$ are coprime, then there exists an isomorphism of this form.

Remark 3.3.2. Lemma 3.3.1 can be significantly generalized to arbitrary extensions where the normal subgroup is characteristic. When the characteristic subgroup is Abelian, this is standard in group theory, and has been useful in practical isomorphism testing (see, e.g., [114]). In general, the equivalence of group extensions deals with both Action Compatibility and Cohomology Class Isomorphism. Generalizations of cohomology to non-Abelian coefficient groups was done by Dedecker in the 1960s (e.g. [73]) and Inassaridze at the turn of the 21st century [123]. Unaware of this prior work on non-Abelian cohomology at the time, Grochow & Qiao re-derived some of it in the special case of H^2—the cohomology most immediately relevant to group extensions and the isomorphism problem—and showed how it could be applied to isomorphism testing [93, Lemma 2.3], generalizing Taunt's Lemma. In the setting of coprime extensions, the Schur–Zassenhaus Theorem provides that the cohomology is trivial. Thus, in our setting we need only consider Action Compatibility.

A $\mathbb{Z}H$-module is an abelian group N together with an action of H on N, given by a group homomorphism $\theta \colon H \to \text{Aut}(N)$. Sometimes we colloquially refer to these as "H-modules." A *submodule* of an H-module N is a subgroup $N' \leq N$ such that the action of H on N' sends N'

into itself, and thus the restriction of the action of H to N' gives N' the structure of an H-module compatible with that on N. Given a subset $S \subseteq N$, the smallest H-submodule containing S is denoted $\langle S \rangle_H$, and is referred to as the H-submodule *generated by* S. An H-module generated by a single element is called *cyclic*. Note that a cyclic H-module N need not be a cyclic Abelian group. As an example, we note that if $|H|$ is coprime to p, then any irreducible representation of H of dimension > 1 over $\mathbb{F}_p$ will be a cyclic H-module, but not a cyclic Abelian group.

Two H-modules N, N' are isomorphic (as H-modules), denoted $N \cong_H N'$, if there is a group isomorphism $\varphi\colon N \to N'$ that is H-equivariant, in the sense that $\varphi(\theta(h)(n)) = \theta'(h)(\varphi(n))$ for all $h \in H, n \in N$. An H-module N is *decomposable* if $N \cong_H N_1 \oplus N_2$ where N_1, N_2 are nonzero H-modules (and the direct sum can be thought of as a direct sum of Abelian groups); otherwise N is *indecomposable*. An equivalent characterization of N being decomposable is that there are nonzero H-submodules N_1, N_2 such that $N = N_1 \oplus N_2$ as Abelian groups (that is, N is generated as a group by N_1 and N_2, and $N_1 \cap N_2 = 0$). The Remak–Krull–Schmidt Theorem says that every H-module decomposes as a direct sum of indecomposable modules, and that the multiset of H-module isomorphism types of the indecomposable modules appearing is independent of the choice of decomposition, that is, it depends only on the H-module isomorphism type of N. We may thus write

$$N \cong_H N_1^{\oplus m_1} \oplus N_2^{\oplus m_2} \oplus \cdots \oplus N_k^{\oplus m_k}$$

unambiguously, where the N_i are pairwise non-isomorphic indecomposable H-modules. When we refer to the multiplicity of an indecomposable H-module as a direct summand in N, we mean the corresponding m_i.[1]

The version of Taunt's Lemma that will be most directly useful for us is:

Lemma 3.3.3. *Suppose $G_i = H \ltimes_{\theta_i} N$ ($i = 1, 2$) are two semi-direct products with $|H|$ coprime to $|N|$. Then $G_1 \cong G_2$ if and only if there is an automorphism $\alpha \in \mathrm{Aut}(H)$ such that each*

[1] For readers familiar with (semisimple) representations over fields, we note that the multiplicity is often equivalently defined as $\dim_{\mathbb{F}} \mathrm{Hom}_{\mathbb{F}H}(N_i, N)$. However, when we allow N to be an Abelian group that is not elementary Abelian, we are working with $(\mathbb{Z}/p^k\mathbb{Z})[H]$-modules, and the characterization in terms of hom sets is more complicated, because one indecomposable module can be a submodule of another, which does not happen with semisimple representations.

indecomposable $\mathbb{Z}H$-module appears as a direct summand in (N, θ_1) and in $(N, \theta_2 \circ \alpha)$ with the same multiplicity.

The lemma and its proof are standard but we include it for completeness.

Proof. If there is an automorphism $\alpha \in \operatorname{Aut}(H)$ such that the multiplicity of each indecomposable $\mathbb{Z}H$-module as a direct summand of (N, θ_1) and $(N, \theta_2\alpha)$ are the same, then there is a $\mathbb{Z}H$-module isomorphism $\beta\colon (N, \theta_1) \to (N, \theta_2 \circ \alpha)$ (in particular, β is an automorphism of N as a group). Then it is readily verified that the map $(h, n) \mapsto (\alpha(h), \beta(n))$ is an isomorphism of the two groups.

Conversely, suppose that $\varphi\colon G_1 \to G_2$ is an isomorphism. Since $|H|$ and $|N|$ are coprime, N is characteristic in G_i, so we have $\varphi(N) = N$. And by order considerations $\varphi(H)$ is a complement to N in G_2. We have $\theta_1(h)(n) = hnh^{-1}$. Since φ is an isomorphism, we have $\varphi(\theta_1(h)(n)) = \varphi(hnh^{-1}) = \varphi(h)\varphi(n)\varphi(h)^{-1} = \theta_2(\varphi(h))(\varphi(n))$. Thus $\theta_1(h)(n) = \varphi^{-1}(\theta_2(\varphi(h))(\varphi(n)))$. So we may let $\alpha = \varphi|_H$, and then we have that (N, θ_1) is isomorphic to $(N, \theta_2 \circ \varphi|_H)$, where the isomorphism of H-modules is given by $\varphi|_N$. The Remak–Krull–Schmidt Theorem then gives the desired equality of multiplicities. □

The following lemma is needed for when N is Abelian, but not elementary Abelian. A $(\mathbb{Z}/p^k\mathbb{Z})[H]$-module is a $\mathbb{Z}H$-module N where the exponent of N (the LCM of the orders of the elements of N) divides p^k.

Lemma 3.3.4 (see, e. g., Thevénaz [188])**.** *Let H be a finite group. If p is coprime to $|H|$, then any indecomposable $(\mathbb{Z}/p^k\mathbb{Z})[H]$-module is generated (as an H-module) by a single element.*

Proof. Thevénaz [188, Cor. 1.2] shows that there are cyclic $(\mathbb{Z}/p^k\mathbb{Z})[H]$-modules $M_1, \ldots, M_n$, each with underlying group of the form $(\mathbb{Z}/p^k\mathbb{Z})^{d_i}$ for some d_i, such that each indecomposable $(\mathbb{Z}/p^k\mathbb{Z})[H]$-module is of the form M_i/p^jM_i for some i, j, and for distinct pairs (i, j) we get non-isomorphic modules. □

3.3.2 Coprime Extensions with an $O(1)$-Generated Complement

Our approach is to first pebble generators for the complement H, which fixes an isomorphism of H. As the isomorphism of H is then fixed, we show that any subsequent bijection that Duplicator selects must restrict to H-module isomorphisms on each indecomposable H-submodule of N that is a direct summand. For groups that decompose as a coprime extension of H and N, the isomorphism type is completely determined by the multiplicities of the indecomposable H-module direct summands (Lemma 3.3.3). We then leverage the fact that, in the coprime case, indecomposable H-modules are generated by single elements (Lemma 3.3.4), making it easy for Spoiler to pebble.

Lemma 3.3.5. *Let $G = H \ltimes N$, where N is Abelian, H is $O(1)$-generated, and gcd$(|H|, |N|) = 1$. Let K be an arbitrary group of order $|G|$. If K does not decompose as $H \ltimes N$ (for any action), then $(O(1), O(1))$-WL Version II will distinguish G and K.*

Proof. Let $f : G \to K$ be the bijection that Duplicator selects. As $N \leq G$, as a subset, is uniquely determined by its orders—it is precisely the set of all elements in G whose orders divide $|N|$—we may assume that K has a normal Hall subgroup of size $|N|$. For first, if for some $n \in N$, $|n| \neq |f(n)|$, Spoiler can pebble $n \mapsto f(n)$ and win immediately. By reversing the roles of K and G, it follows that K must have precisely $|N|$ elements whose orders divide $|N|$. Second, suppose that those $|N|$ elements do not form a subgroup. Then there are two elements $x, y \in f(N)$ such that $xy \notin f(N)$. At the first round, Spoiler pebbles $a := f^{-1}(x) \mapsto x$. Let $f' : G \to K$ be the bijection Duplicator selects at the next round. As K has precisely $|N|$ elements of order dividing $|N|$, we may assume that $f'(N) = f(N)$ (setwise). Let $b \in N$ s.t. $f'(b) = y$. Spoiler pebbles $b \mapsto y$. Now as N is a group, $ab \in N$. However, as $f(a)f'(b) \notin f(N)$, $|ab| \neq |f(a)f'(b)|$. So the map $(a, b) \mapsto (x, y)$ does not extend to an isomorphism. Spoiler now wins.

Now we have that $f(N)$ is a subgroup of K, and because it is the set of all elements of these orders, it is characteristic and thus normal. Suppose that $f(N) \ncong N$. We have two cases: either $f(N)$ is not Abelian, or $f(N)$ is Abelian but $N \ncong f(N)$. Suppose first that $f(N)$ is not Abelian. Let $x \in f(N)$ such that $x \notin Z(f(N))$, and let $g := f^{-1}(x) \in N$. Spoiler pebbles $g \mapsto x$. Let

$f' : G \to K$ be the bijection that Duplicator selects at the next round. We may again assume that $f'(N) = f(N)$ (setwise), or Spoiler wins with two additional pebbles and two additional rounds. Now let $y \in f(N)$ such that $[x, y] \neq 1$. Let $h \in G$ such that $f'(h) = y$. Spoiler pebbles $h \mapsto y$. Now the map $(g, h) \mapsto (x, y)$ does not extend to an isomorphism, so Spoiler wins. Suppose instead that $f(N)$ is Abelian. As Abelian groups are determined by their orders, we have by the discussion in the first paragraph that Spoiler wins with 2 pebbles and 2 rounds.

So now suppose that $N \cong f(N) \leq K$ is a normal Abelian Hall subgroup, but that $f(N)$ does not have a complement isomorphic to H. We note that if K contains a subgroup H' that is isomorphic to H, then by order considerations, H' and $f(N)$ would intersect trivially in K and we would have that $K = H' \cdot f(N)$. That is, K would decompose as $K = H \ltimes f(N)$. So as $f(N)$ does not have a complement in K that is isomorphic to H, by assumption we have that K does not contain any subgroup isomorphic to H. In this case, Spoiler pebbles the $O(1)$ generators of H in G. As K has no subgroup isomorphic to H, Spoiler immediately wins after the generators for $H \leq G$ have been pebbled. The result follows. □

Theorem 3.3.6. *Let $G = H \ltimes N$, where N is Abelian, H is $O(1)$-generated, and gcd$(|H|, |N|) = 1$. We have that $(O(1), O(1))$-WL Version II identifies G.*

Proof. Let $K \not\cong G$ be an arbitrary group of order $|G|$. By Lemma 3.3.5, we may assume that $K = H \ltimes N$; otherwise, Spoiler wins in at most 2 rounds. Furthermore, from the proof of Lemma 3.3.5, we may assume that Duplicator selects bijections $f : G \to K$ mapping $N \cong f(N)$ (though $f|_N$ need not be an isomorphism), or Spoiler wins with a single round.

Spoiler uses the first $k := d(H)$ rounds to pebble generators $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ for H. As $K = H \ltimes N$, we may assume that the map $(g_1, \ldots, g_k) \mapsto (h_1, \ldots, h_k)$ induces an isomorphism with a copy of $H \leq K$; otherwise, Spoiler immediately wins. Let $f : G \to K$ be the bijection that Duplicator selects. As G, K are non-isomorphic groups of the form $H \ltimes N$, they differ only in their actions. Now the actions are determined by the multiset of indecomposable H-modules in N. As $|H|, |N|$ are coprime, we have by Lemma 3.3.4 that the indecomposable H-modules are cyclic. As

$G \not\cong K$, we have by Lemma 3.3.1 that there exists $n \in N \leq G$ such that $\langle n \rangle_H$ is indecomposable, and $\langle n \rangle_H$ and $\langle f(n) \rangle_{f(H)}$ are inequivalent H-modules. Spoiler now pebbles $n \mapsto f(n)$. Thus, the following map

$$(g_1, \ldots, g_k, n) \mapsto (h_1, \ldots, h_k, f(n))$$

does not extend to an isomorphism. So Spoiler wins. $\square$

Remark 3.3.7. We see that the main places we used coprimality were: (1) that N was characteristic, and (2) that all indecomposable H-modules (in particular, those appearing in N) were cyclic.

3.4 A "rank" lemma

Definition 3.4.1. Let $C \subseteq G$ be a subset of a group G that is closed under taking inverses. We define the *C-rank* of $g \in G$, denoted $\mathrm{rk}_C(g)$, as the minimum m such that g can be written as a word of length m in the elements of C. If g cannot be so written, we define $\mathrm{rk}_C(g) = \infty$.

Our definition and results actually extend to subsets that aren't closed under taking inverses, but we won't have any need for that case, and it would only serve to make the wording less clear.

Remark 3.4.2. Our terminology is closely related to the usage of "X-rank" in algebra and geometry, which generalizes the notions of matrix rank and tensor rank: if $X \subseteq V$ is a subset of an $\mathbb{F}$-vector space, then the X-rank of a point $v \in V$ is the smallest number of elements of $x \in X$ such that v lies in their linear span. If we replace X by the union $\mathbb{F}^*X$ of its nonzero scaled versions (which is unnecessary in the most common case, in which X is the cone over a projective variety), then the X-rank in the sense of algebraic geometry would be the $\mathbb{F}X$-rank in our terminology above. For example, matrix rank is X-rank inside the space of $n \times m$ matrices under addition, where X is the set of rank-1 matrices (which is already closed under nonzero scaling).

We first introduce the notion of what it means for WL to distinguish a set.

Definition 3.4.3. Let G be a group, and let $C \subseteq G$. Let $k \geq 2, r \geq 1$, and $J \in \{I, II, III\}$. We say that (k, r)-WL Version J *distinguishes* C if whenever $g \in C$ and $g' \notin C$, then the coloring χ_r computed by (k, r)-WL satisfies $\chi_r((g, \ldots, g)) \neq \chi_r((g', \ldots, g'))$.

Lemma 3.4.4 (Rank lemma). *If $C \subseteq G$ is distinguished by (k, r)-WL Version II, then any bijection f chosen by Duplicator must respect C-rank, in the sense that $\text{rk}_C(g) = \text{rk}_{f(C)}(f(g))$ for all $g \in G$, or Spoiler can win with $k + 1$ pebbles and $\max\{r, \log d + O(1)\}$ rounds, where $d = diam(Cay(\langle C \rangle, C)) \leq |\langle C \rangle| \leq |G|$.*

Our primary uses of this lemma in this paper are to show that if C is distinguished by (k, r)-WL, then $\langle C \rangle$ is distinguished by $(k + O(1), r + \log n)$-WL. However, the preservation of C-rank itself, rather than merely the subgroup generated by C, has found use in further applications- see Chapter 4 and [65]. In particular, Lemma 3.4.4 shows that WL can identify verbal subgroups in $O(\log n)$ rounds, provided WL can readily identify each word.

Proof. Note that since C is detectable by (k, r)-WL, there is a set $C' \subseteq H$ such that for any bijection f chosen by Duplicator, $f(C) = C'$, otherwise Spoiler can win with k pebbles in r rounds. We will thus use $\text{rk}(x)$ to denote $\text{rk}_C(x)$ if $x \in G$, and $\text{rk}_{C'}(x)$ if $x \in H$. We proceed by induction on the rank.

By assumption, rank-1 elements must be sent to rank-1 elements, since $C = \{g \in G : \text{rk}(g) = 1\}$, or Spoiler can win with k pebbles in r rounds.

Let $r \geq 1$, and suppose for all $1 \leq j \leq r$, if $\text{rk}(x) = j$, then $\text{rk}(f(x)) = \text{rk}(x)$. Suppose $x \in G$ is such that $\text{rk}(x) = r + 1$ but $\text{rk}(f(x)) \neq \text{rk}(x)$. Since f is a bijection on elements of smaller rank, the only possibility is $\text{rk}(f(x)) > r + 1$. Spoiler begins by pebbling $x \mapsto f(x)$.

Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Write $x = x_1 \cdots x_{r+1}$, where for each i, $x_i \in C$. For $1 \leq i \leq j \leq r + 1$, write $x[i, \ldots, j] := x_i \cdots x_j$. We consider the following cases.

- **Case 1:** Suppose first that $\text{rk}(y) = \text{rk}(f'(y))$ for all $y \in G$ with $\text{rk}(y) \leq r$. In this case, Spoiler pebbles $x[2, \ldots, r + 1] \mapsto f'(x[2, \ldots, r + 1])$. Let $f'' : G \to H$ be the bijection that

Duplicator selects at the next round. If $\text{rk}(x_1) = \text{rk}(f''(x_1)) = 1$, then $f''(x_1) \cdot f'(x[2, \ldots, r + 1]) \neq f(x)$, since $\text{rk}(f''(x_1)) = 1$ and $\text{rk}(f'(x[2, \ldots, r+1])) = r$, so their product has rank at most $r + 1 < \text{rk}(f(x))$. In this case, Spoiler pebbles x_1 and wins immediately since f'' does not extend to a bijection on the pebbled elements $x_1, x[2, \ldots, r + 1]$.

If instead, $1 = \text{rk}(x_1) < \text{rk}(f''(x_1))$, Spoiler pebbles x_1 and wins with $k - 1$ additional pebbles and r additional rounds by assumption. Note that once $x_1 \mapsto f''(x_1)$ has been pebbled, Spoiler can reuse the pebble on x. So we only need $k - 1$ additional pebbles rather than k pebbles.

- **Case 2:** Suppose instead that the hypothesis of Case 1 is not satisfied. Then $\text{rk}(y) \neq \text{rk}(f'(y))$ for some $y \in \langle C \rangle$ with $\text{rk}(y) \leq r$. In the next two rounds, Spoiler pebbles $x[1, \ldots, \lceil (r+1)/2 \rceil]$ and $x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]$. Let $f'' : G \to H$ be the next bijection that Duplicator selects. If

$$f(x) \neq f''(x[1, \ldots, \lceil (r+1)/2 \rceil]) \cdot f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]),$$

then Spoiler immediately wins. If

$$f(x) = f''(x[1, \ldots, \lceil (r+1)/2 \rceil]) \cdot f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]),$$

then either

$$\begin{aligned}&\text{rk}(x[1, \ldots, \lceil (r+1)/2 \rceil]) < \text{rk}(f''(x[1, \ldots, \lceil (r+1)/2 \rceil])) \text{ or}\\ &\text{rk}(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]) < \text{rk}(f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1])),\end{aligned}$$

since $\text{rk}(f(x)) > \text{rk}(x) = \text{rk}(x[1, \ldots, \lceil (r+1)/2 \rceil]) + \text{rk}(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1])$.

Without loss of generality, suppose that $\text{rk}(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]) < \text{rk}(f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]))$. As Spoiler has already pebbled $x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1] \mapsto f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1])$, we may iterate on this argument starting from $x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1] \mapsto f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1])$. Note that at most $\log(r+1) + 1$

iterations are necessary until we hit the case when $\text{rk}(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]) = 1 < \text{rk}(f''(x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]))$. By assumption, every element of $\langle C \rangle$ can be written as a word of length at most d in the elements of C, so we have $r \leq d$, and thus only $O(\log d)$ rounds are required. (That $d \leq |\langle C \rangle|$ follows from Fact 2.2.3.)

We claim that the preceding procedure can be implemented with at most $k+2$ pebbles. After Spoiler pebbles $x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]$, they may reuse the pebbles on x and $x[1, \ldots, \lceil (r+1)/2 \rceil]$. We eventually reach a case in which there exists $g \in C$ such that Duplicator maps g to some element outside of C'. In this case, Spoiler pebbles g, using the pebble on x. Now Spoiler uses the pebbles on $x[1, \ldots, \lceil (r+1)/2 \rceil]$, $x[\lceil (r+1)/2 \rceil + 1, \ldots, r+1]$, and $k-2$ additional pebbles to win. In total, Spoiler has used $k+1$ pebbles. In a similar manner as Case 1, at most $k-1$ additional pebbles are required to identify C.

Finally, we must handle the case of infinite rank. By symmetry, it suffices to show that Spoiler can win in the case when $\text{rk}(x) = r+1$, but $\text{rk}(f(x)) = \infty$. In this case, the same argument starting from the third paragraph works *mutatis mutandis*, as $\text{rk}(f(x)) = \infty > r+1$. □

3.5 Direct Products

Brachter & Schweitzer previously showed that Weisfeiler–Leman Version II can detect direct product decompositions in polynomial-time. Precisely, they showed that the WL dimension of a group G is at most one more than the WL dimensions of the direct factors of G. We strengthen the result to control for rounds, effectively showing that WL Version II can compute direct product decompositions using $O(\log n)$ rounds.

In this section, we establish the following.

Theorem 3.5.1. *Let $G = G_1 \times \cdots \times G_d$ be a decomposition into indecomposable direct factors, let $k \geq 5$, and let $r := r(n) \in \Omega(\log n)$. If G and H are not distinguished by (k,r)-WL Version II, then there exist direct factors $H_i \leq H$ such that $H = H_1 \times \cdots \times H_d$ such that for all $i \in [d]$, G_i and H_i are not distinguished by $(k-1,r)$-WL Version II.*

The structure and definitions in this section closely follow those of [49, Sec. 6] for ease of comparison.

3.5.1 Preliminaries

We begin by introducing some additional preliminaries.

Definition 3.5.2. Let G_1, G_2 be groups, and let $Z_i \leq Z(G_i)$ be central subgroups. Given an isomorphism $\varphi : Z_1 \to Z_2$, the *central product* of G_1 and G_2 with respect to φ is:

$$G_1 \times_\varphi G_2 = (G_1 \times G_2)/\{(g, \varphi(g^{-1})) : g \in Z_1\}.$$

A group G is the *(internal) central product* of subgroups $G_1, G_2 \leq G$, provided that $G = G_1G_2$ and $[G_1, G_2] = \{1\}$.

Remark 3.5.3. In general, a group may have several inherently different central decompositions. On the other hand, indecomposable direct decompositions are unique in the following sense.

Lemma 3.5.4 (See, e.g., [171, 3.8.3])**.** *Let $G = G_1 \times \ldots \times G_m = H_1 \times \ldots \times H_n$ be two direct decompositions of G into directly indecomposable factors. Then $n = m$, and there exists a permutation $\sigma \in Sym(n)$ such that for all i, $G_i \cong H_{\sigma(i)}$ and $G_iZ(G) \cong H_{\sigma(i)}Z(G)$.*

By the preceding lemma, the multiset of subgroups $\{\{G_iZ(G)\}\}$ is invariant under automorphism.

Definition 3.5.5 ([49, Def. 6.3])**.** We say that a central decomposition $\{H_1, H_2\}$ of $G = H_1H_2$ is *directly induced* if there exist subgroups $K_i \leq H_i$ ($i = 1, 2$) such that $G = K_1 \times K_2$ and $H_i = K_iZ(G)$.

Lemma 3.5.6 ([49, Lemma 6.4])**.** *Let $k \geq 4, r \geq 1$. Let G_1, G_2, H_1, H_2 be groups such that G_i and H_i (for all $i \in [2]$) are not distinguished by (k, r)-WL. Then $G_1 \times G_2$ and $H_1 \times H_2$ are not distinguished by (k, r)-WL.*

Remark 3.5.7. Lemma 3.5.6 states that when the multiset of direct factors for G is indistinguishable from that of H (under (k, r)-WL), then (k, r)-WL will not distinguish G and H. However, Lemma 3.5.6 does not address conditions under which G and H are distinguishable, with regards to their direct factors. In particular, a group may admit multiple direct product decompositions, even when the direct factors are indecomposable- see [128] for an example.

Remark 3.5.8. The statement of [49, Lemma 6.4] does not mention rounds; however, the proof holds when considering rounds.

3.5.2 Abelian and Semi-Abelian Case

Definition 3.5.9 ([49, Def. 6.5])**.** Let G be a group. We say that $x \in G$ *splits* from G if there exists a complement $H \leq G$ such that $G = \langle x \rangle \times H$.

We recall the following technical lemma [49, Lemma 6.6] that characterizes the elements that split from an Abelian p-group.

Lemma 3.5.10 ([49, Lemma 6.6])**.** *Let A be a finite Abelian p-group, and let $A = A_1 \times \cdots \times A_m$ be an arbitrary cyclic decomposition. Then $a = (a_1, \ldots, a_m) \in A$ splits from A if and only if there exists some $i \in [m]$ such that $|a| = |a_i|$ and $a_i \in A_i \setminus (A_i)^p$. In particular, x splits from A if and only if there does not exist a $y \in A$ such that $|xy^p| < |x|$.*

We utilize this lemma to show that WL can detect elements that split from A.

Lemma 3.5.11. *Let A, B be Abelian p-groups of order n, and let $f : A \to B$ be the bijection Duplicator selects. If $x \in A$ splits from A, but $f(x)$ does not split from B, then Spoiler can win with 2 pebbles and 2 rounds.*

Proof. Spoiler begins by pebbling $x \mapsto f(x)$. Let $f' : A \to B$ be the bijection that Duplicator selects at the next round. As $f(x)$ does not split from B, there exists $z \in B$ such that $|f(x) \cdot z^p| < |f(x)|$. Let $y = (f')^{-1}(z) \in A$. Spoiler pebbles $y \mapsto f'(y) = z$. Now $|xy^p| \neq |f(x) \cdot z^p|$. So Spoiler immediately wins. □

Remark 3.5.12. To characterize when an element splits in a general Abelian group A, we begin by considering the decomposition of A into its Sylow subgroups: $A = P_1 \times \cdots \times P_m$. Now $x = (x_1, \ldots, x_m) \in A$ splits from A if and only if for each $i \in [m]$, x_i is either trivial or splits from P_i. See, e.g., [49, Lemma 6.8].

Lemma 3.5.13. *Let A, B be Abelian groups. Let $A = P_1 \times \cdots \times P_m$ and $B = Q_1 \times \cdots \times Q_m$, where the P_i are the Sylow subgroups of A and the Q_i are the Sylow subgroups of B (for each i, P_i and Q_i are p_i-subgroups for the same prime p_i). Let $f : A \to B$ be the bijection that Duplicator selects. Let $x = (x_1, \ldots, x_m)$ be the decomposition of x, where $x_i \in P_i$, and let $f(x) = (y_1, \ldots, y_m)$, where $y_i \in Q_i$. Suppose that Spoiler pebbles $x \mapsto f(x)$. Let $f' : A \to B$ be the bijection that Duplicator selects at the next round.*

(a) If $f'(x_i) \neq y_i$, then Spoiler can win with 1 additional pebbles and 1 additional round.

(b) If $x \in A$ splits from A, but $f(x)$ does not split from B, then Spoiler can win with 2 pebbles and 2 rounds.

Proof. We proceed as follows.

(a) Suppose there exists an $i \in [m]$ such that $f'(x_i) \neq y_i$. Spoiler pebbles $x_i \mapsto f'(x_i)$. Suppose that P_i, Q_i are Sylow p-subgroups of A, B respectively. As $x_i \in P_i$, we have that $\langle x \cdot x_i^{-1} \rangle$ has order coprime to p. However, as $f(x_i) \neq y_i$, $\langle f(x) \cdot f(x_i)^{-1} \rangle$ has order divisible by p. So $|x \cdot x_i^{-1}| \neq |f(x) \cdot f(x_i)^{-1}|$. Thus, Spoiler wins at the end of this round.

(b) We recall that nilpotent groups are direct products of their Sylow subgroups. Furthermore, for a given prime divisor p, the Sylow p-subgroup of a nilpotent group is unique and contains all the elements whose order is a power of p. Thus, each Sylow subgroup of a nilpotent group is characteristic as a set. So now by (a), we may assume that $f'(x_i) = y_i$. Let $i \in [m]$ such that x_i splits from P_i, but $f'(x_i) = y_i$ does not split from Q_i. Spoiler pebbles $x_i \mapsto f'(x_i) = y_i$. Now by Lemma 3.5.11, applied to $x_i \mapsto f'(x_i)$, Spoiler wins with 1 additional pebble (reusing the pebble on $x \mapsto y$) and 2 additional rounds.

□

We now show that Duplicator must select bijections that preserve both the center and commutator subgroups setwise. Here is our first application of the Rank Lemma 3.4.4, which was not present in [49]. We begin with the following standard definition.

Definition 3.5.14. For a group G and $g \in G$, the *commutator width* of g, denoted $\text{cw}(g)$, is the $\{[a, b] : a, b \in G\}$-rank (see Definition 3.4.1). The commutator width of G, denoted $\text{cw}(G)$, is the maximum commutator width of any element of $[G, G]$.

Lemma 3.5.15. *Let G, H be finite groups of order n. Let $f : G \to H$ be the bijection that Duplicator selects.*

(a) If $f(Z(G)) \neq Z(H)$, then Spoiler can win with 2 pebbles and 2 rounds.

(b) If there exist $x, y \in G$ such that $f([x, y])$ is not a commutator $[h, h']$ for any $h, h' \in H$, then Spoiler can win with 3 pebbles and 3 rounds.

(c) If there exists $g \in G$ such that $\text{cw}(g) \neq \text{cw}(f(g))$, then Spoiler can win with 4 pebbles and $O(\log \text{cw}(G)) \leq O(\log n)$ rounds.

Brachter & Schweitzer previously showed that 2-WL Version II identifies $Z(G)$, and 3-WL Version II identifies the commutator $[G, G]$ [49]. Here, using our Rank Lemma 3.4.4 for commutator width, we obtain that 4-WL identifies the commutator in $O(\log n)$ rounds.

Proof of Lemma 3.5.15 .

(a) Let $x \in Z(G)$ such that $f(x) \notin Z(H)$. Spoiler begins by pebbling $x \mapsto f(x)$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Let $y \in H$ such that $f'(x)$ and y do not commute. Let $a := (f')^{-1}(y) \in G$. Spoiler pebbles $a \mapsto f'(a) = y$ and wins.

(b) Spoiler pebbles $[x, y] \mapsto f([x, y])$. At the next two rounds, Spoiler pebbles x, y. Regardless of Duplicator's choices, Spoiler wins.

(c) Apply the Rank Lemma 3.4.4 to the set of commutators. By part (b), this set is identified by $(4, O(1))$-WL, so the Rank Lemma gives part (c).

□

By Lemma 3.5.15, Duplicator must select bijections that preserve the center and commutator subgroups setwise (or Spoiler can win). A priori, these bijections need not restrict to isomorphisms on the center or commutator. We note, however, that we may easily decide whether two groups have isomorphic centers, as the center is Abelian. Precisely, by [49, Corollary 5.3], $(2, 1)$-WL identifies Abelian groups. Note that we need an extra round to handle the case in which Duplicator maps an element of $Z(G)$ to some element not in $Z(H)$. So $(2, 2)$-WL identifies both the set of elements in $Z(G)$ and its isomorphism type.

We now turn to detecting elements that split from arbitrary groups. To this end, we recall the following lemma from [49].

Lemma 3.5.16 ([49, Lemma 6.9]). *Let G be a finite group and $z \in Z(G)$. Then z splits from G if and only if $z[G, G]$ splits from $G/[G, G]$ and $z \notin [G, G]$.*

We apply this lemma to show that WL can detect the set of elements that split from an arbitrary finite group, improving [49, Corollary 6.10] to control for rounds:

Lemma 3.5.17 (Compare rounds cf. [49, Corollary 6.10]). *Let G, H be finite groups. Let $f : G \to H$ be the bijection that Duplicator selects. Suppose that x splits from G, but $f(x)$ does not split from H. Then Spoiler can win with 4 pebbles and $O(\log n)$ rounds.*

Proof. By Lemma 3.5.15, we have that if $x \notin [G, G]$ but $f(x) \in [H, H]$, then Spoiler can win with 4 pebbles and $O(\log n)$ rounds. So suppose that $x \notin [G, G]$ and $f(x) \notin [H, H]$. It suffices to check whether $x[G, G]$ splits from $G/[G, G]$, but $f(x)[H, H]$ does not split from $H/[H, H]$. By [49, Lemma 4.11], it suffices to consider the pebble game on $(G/[G, G], H/[H, H])$. To this end, we apply Lemma 3.5.13 to $G/[G, G]$ and $H/[H, H]$.

As $f([G,G]) = [H,H]$, f induces a bijection $\overline{f} : G/[G,G] \to H/[H,H]$ on the cosets. As f was an arbitrary bijection that preserves $[G,G]$, we may assume WLOG that Duplicator selected $\overline{f}$ in the quotient game on $(G/[G,G], H/[H,H])$. The result now follows. □

We now consider splitting in two special cases.

Lemma 3.5.18 ([49, Lemma 6.11])**.** *Let $U \leq G$ and $x \in Z(G) \cap U$. If x splits from G, then x splits from U.*

Lemma 3.5.19 ([49, Lemma 6.12])**.** *Let $G = G_1 \times G_2$, and let $z := (z_1, z_2) \in Z(G)$ be an element of order p^k for some prime p. Then z splits from G if and only if there exists an $i \in \{1,2\}$ such that z_i splits from G_i and $|z_i| = |z|$.*

We now consider the semi-Abelian case. Here our groups are of the form $H \times A$, where H has no Abelian direct factors and A is Abelian.

Theorem 3.5.20 (Compare rounds cf. [49, Lemma 6.13])**.** *Let $G_1 = H \times A$, with a maximal Abelian direct factor A. Then the isomorphism class of A is identified by $(4, O(1))$-WL Version II. That is, if $(4, O(1))$-WL fails to distinguish G and $\widetilde{G}$, then $\widetilde{G}$ has a maximal Abelian direct factor isomorphic to A.*

Proof. We adapt the proof of [49, Lemma 6.13] to control for rounds. Let $\widetilde{G}$ be a group such that $(4, O(1))$-WL fails to distinguish G and $\widetilde{G}$. By Lemma 3.5.15 and the subsequent discussion, we may assume that $Z(G) \cong Z(\widetilde{G})$ using $(2,2)$-WL Version II. As Abelian groups are direct products of their Sylow subgroups, it follows that $Z(G)$ and $Z(\widetilde{G})$ have isomorphic Sylow subgroups. Write $\widetilde{G} = \widetilde{H} \times \widetilde{A}$, where $\widetilde{A}$ is the maximal Abelian direct factor. As $Z(G) \cong Z(\widetilde{G})$, we write Z for the Sylow p-subgroup of $Z(G) \cong Z(\widetilde{G})$. Consider the primary decomposition of Z:

$$Z := Z_1 \times \ldots \times Z_m,$$

where $Z_i \cong (\mathbb{Z}/p^i\mathbb{Z})^{e_i}$, for $e_i \geq 0$. For each $i \in [m]$, there exist subgroups $H_i \leq Z(H)$ and $A_i \leq A$ such that $Z_i \cong H_i \times A_i$. Similarly, there exist $\widetilde{H_i} \leq \widetilde{H}$ and $\widetilde{A_i} \leq \widetilde{A}$ such that $Z_i \cong \widetilde{H_i} \times \widetilde{A_i}$. As

$Z(G) \cong Z(\widetilde{G})$, we have that $H_i \times A_i \cong \widetilde{H_i} \times \widetilde{A_i}$. It suffices to show that each $A_i \cong \widetilde{A_i}$. As H does not have any Abelian direct factors, we have by [49, Lemma 6.12] (reproduced as Lemma 3.5.19 above) that a central element x of order p^i splits from G if and only if the projection of x onto A_i, denoted $A_i(x)$, has order p^i. The same holds for $\widetilde{G}$ and the $\widetilde{A_i}$ factors. By Lemma 3.5.13, we may assume that Duplicator selects bijections $f : G \to H$ such that if $x \in Z(G)$ splits from Z_i, then $f(x)$ splits from $f(Z_i)$. The result follows. □

We now recall the definition of a component-wise filtration, introduced by Brachter & Schweitzer [49] to control the non-Abelian part of a direct product.

Definition 3.5.21 ([49, Def. 6.14])**.** Let $G = L \times R$, and let π_L (resp. π_R) be the natural projection map onto L with kernel R (resp., the natural projection onto R with kernel L). A *component-wise filtration* of $U \leq G$ with respect to L and R is a chain of subgroups $\{1\} = U_0 \leq \cdots \leq U_r = U$, such that for all $i \in [r]$, we have that $U_{i+1} \leq \pi_L(U_i \times \{1\})$ or $U_{i+1} \leq \pi_R(\{1\} \times U_i)$. For $J \in \{I, II, III\}$, the filtration is k-WL$_J$-detectable, provided all subgroups in the chain are k-WL$_J$-detectable.

Brachter & Schweitzer previously showed [49, Lemma 6.15] that there exists a component-wise filtration of $Z(G)$ with respect to H and A that is 4-WL$_I$-detectable. We extend this result to control for rounds. The proof that such a filtration exists is identical to that of [49, Lemma 6.15]; we get a bound on the rounds using our Lemma 3.5.17, which is a round-controlled version of their Corollary 6.10. For completeness, we indicated the needed changes here.

Lemma 3.5.22. *Let $G = H \times A$, with maximal Abelian direct factor A. The component-wise filtration of $Z(G)$ with respect to H and A from [49, Lemma 6.15] (reproduced above) is $(4, O(\log n))$-WL$_{II}$-detectable.*

Proof. Their proof that the filtration is 4-WL detectable uses only two parts: the fact that central e-th powers are detectable, and their Corollary 6.10. Using our Lemma 3.5.17 in place of their Corollary 6.10, we get 4 pebbles and $O(\log n)$ rounds, so all that is left to handle is central e-th powers. Suppose Duplicator selects a bijection $f : G \to H$ where $g = x^e$ for some $x \in Z(G)$ and

$f(g)$ is not a central eth power. We have already seen that Duplicator must map the center to the center, so we need only handle the condition of being an e-th power. At the first round, Spoiler pebbles $g \mapsto f(g)$. At the next round, Spoiler pebbles x and wins. Thus Spoiler can win with 2 pebbles in 2 rounds. □

We now show that in the semi-Abelian case $G = H \times A$, with maximal Abelian direct factor A, the WL dimension of G depends on the WL dimension of H.

Lemma 3.5.23 (Compare rounds to [49, Lemma 6.16])**.** *Let $G = H \times A$ and $\widetilde{G} = \widetilde{H} \times \widetilde{A}$, where A and $\widetilde{A}$ are maximal Abelian direct factors. Let $k \geq 5$ and $r \in \Omega(\log n)$. If $(k-1, r)$-WL fails to distinguish G and $\widetilde{G}$, then (k, r)-WL fails to distinguish H and $\widetilde{H}$.*

Proof. By Theorem 3.5.20, we may assume that $A \cong \widetilde{A}$. Consider the component-wise filtrations from the proof of [49, Lemma 6.15], $\{1\} = U_0 \leq \cdots \leq U_r = Z(G)$ with respect to the decomposition $G = H \times A$ and $\{1\} = \widetilde{U_0} \leq \cdots \leq \widetilde{U_r} = Z(\widetilde{G})$ with respect to the decomposition $\widetilde{G} = \widetilde{H} \times \widetilde{A}$.

Let $V_i, W_i, \widetilde{V_i}, \widetilde{W_i}$ be as defined in the proof of [49, Lemma 6.15] and recalled above. We showed in the proof of Lemma 3.5.22 that for any bijection $f : G \to \widetilde{G}$ Duplicator selects, $f(V_i) = \widetilde{V_i}$ and $f(W_i) = \widetilde{W_i}$, or Spoiler may win with 4 pebbles and $O(\log n)$ rounds.

In the proof of [49, Lemma 6.16], Brachter & Schweitzer established that for all $1 \neq x \in Z(H) \times \{1\}$ and all $1 \neq y \in \{1\} \times A$, $\min\{i : x \in U_i\} \neq \min\{i : y \in U_i\}$. Furthermore, by [49, Lemma 4.14], we may assume that Duplicator selects bijections at each round that respect the subgroup chains and their respective cosets, without altering the number of rounds (their proof is round-by-round). It follows that whenever $g_1 g_2^{-1} \in Z(H) \times \{1\}$, we have that $f(g_1)f(g_2)^{-1} \notin \{1\} \times A$.

Furthermore, Brachter & Schweitzer also showed in the proof of [49, Lemma 6.16] that Duplicator must map $H \times \{1\}$ to a system of representatives modulo $\{1\} \times \widetilde{A}$. Thus, Spoiler can restrict the game to $H \times \{1\}$. Now if (k, r)-WL Version II distinguishes H and $\widetilde{H}$, then Spoiler can ultimately reach a configuration $((h_1, 1), \ldots, (h_{k-1}, 1)) \mapsto (x_1, a_1), \ldots, (x_{k-1}, a_{k-1})$ such that the induced configuration over $(G/(\{1\} \times A), \widetilde{G}/(\{1\} \times \widetilde{A}))$ fulfills the winning condition for Spoiler. That is, considered as elements of $G/(\{1\} \times A)$ (resp., $\widetilde{G}/(\{1\} \times \widetilde{A})$), the map $(h_1, \ldots, h_{k-1}) \mapsto (x_1, \ldots, x_{k-1})$ does not extend to an isomorphism. This implies that the pebbled map $((h_1, 1), \ldots, (h_{k-1}, 1)) \mapsto (x_1, a_1), \ldots, (x_{k-1}, a_{k-1})$ in the original groups (rather than their quotients) does not extend to an isomorphism. For suppose f is any bijection extending the pebbled map. By the above, without loss of generality, f maps $H \times \{1\}$ to a system of coset representatives of $\{1\} \times \widetilde{A}$, that is, if Duplicator can win, Duplicator can win with such a map. Let $\overline{f}$ be the induced bijection on the quotients $G/(\{1\} \times A) \to \tilde{G}/(\{1\} \times \tilde{A})$. Since the pebbled map on the quotients does not extend to an isomorphism, there is a word w such that $\overline{f}(w(h_1, \ldots, h_{k-1})) \neq w(x_1, \ldots, x_{k-1})$. But then when we consider f restricted to $H \times \{1\}$, we find that $f(w((h_1, 1), \ldots, (h_{k-1}, 1))) = f((w(h_1, \ldots, h_{k-1}), 1)) \neq (w(x_1, \ldots, x_{k-1}), w(a_1, \ldots, a_{k-1}))$, because their H coordinates are different. □

Lemma 3.5.23 yields the following corollary.

Corollary 3.5.24. *Let $G = H \times A$, where H is identified by $(O(1), O(\log n))$-WL and does not have an Abelian direct factor, and A is Abelian. Then $(O(1), O(\log n))$-WL identifies G. In particular, isomorphism testing of G and an arbitrary group $\widetilde{G}$ is in TC^1.*

Proof. By Lemma 3.5.23, as H is identified by $(O(1), O(\log n))$-WL, we have that G is identified by $(O(1), O(\log n))$-WL. As only $O(\log n)$ rounds are required, we apply the parallel WL implementation due to Grohe & Verbitsky [104] to obtain the bound of TC^1 for isomorphism testing. □

Remark 3.5.25. Das & Sharma [72] previously exhibited a nearly-linear time algorithm for groups of the form $H \times A$, where H has size $O(1)$ and A is Abelian. Corollary 3.5.24 generalizes this to the setting where H is $O(1)$-generated. While Corollary 3.5.24 does not improve the runtime, it does establish a new parallel upper bound for isomorphism testing.

3.5.3 General Case

Following the strategy in [49], we reduce the general case to the semi-Abelian case. Consider a direct decomposition $G = G_1 \times \ldots \times G_d$, where each G_i is directly indecomposable. The multiset of subgroups $\{\{G_iZ(G)\}\}$ is independent of the choice of decomposition. We first show that Weisfeiler–Leman detects $\bigcup_i G_iZ(G)$. Next, we utilize the fact that the connected components of the non-commuting graph on G, restricted to $\bigcup_i G_iZ(G)$, correspond to the subgroups $G_iZ(G)$.

Definition 3.5.26. For a group G, the *non-commuting graph* X_G has vertex set G, and an edge $\{g, h\}$ precisely when $[g, h] \neq 1$.

Proposition 3.5.27 ([1, Proposition 2.1])**.** *If G is non-Abelian, then $X_G[G \setminus Z(G)]$ is connected.*

Our goal now is to first construct a canonical central decomposition of G that is detectable by WL. This decomposition will serve to approximate $\bigcup_i G_iZ(G)$ from below.

Definition 3.5.28 ([49, Definition 6.19])**.** Let G be a finite, non-Abelian group. Let M_1 be the set of non-central elements g whose centralizers $C_G(g)$ have maximal order among all non-central elements. For $i \geq 1$, define M_{i+1} to be the union of M_i and the set of elements $g \in G \setminus \langle M_i \rangle$ that have maximal centralizer order $|C_G(g)|$ amongst the elements in $G \setminus \langle M_i \rangle$. Let $M := M_\infty$ be the stable set resulting from this procedure.

Consider the subgraph $X_G[M]$, and let $X_1, \ldots, X_m$ be the connected components. Set $N_i := \langle X_i \rangle$. We refer to $N_1, \ldots, N_m$ as the *non-Abelian components* of G.

Brachter & Schweitzer previously established the following [49].

Lemma 3.5.29 ([49, Lemma 6.20])**.** *In the notation of Definition 3.5.28, we have the following.*

(a) M is 3-WL_{II}-detectable.

(b) $G = N_1 \cdots N_m$ is a central decomposition of G. For all i, $Z(G) \leq N_i$ and N_i is non-Abelian. In particular, M generates G.

(c) If $G = G_1 \times \ldots \times G_d$ is an arbitrary direct decomposition, then for each $i \in [m]$, there exists a unique $j \in [d]$ such that $N_i \subseteq G_jZ(G)$. Collect all such i for one fixed j in an index set I_j. Then

$$N_{j_1}N_{j_2} \cdots N_{j_\ell} = G_jZ(G),$$

where $I_j = \{j_1, \ldots, j_\ell\}$.

We note that Lemma 3.5.29 (b)-(c) are purely group theoretic statements. For our purposes, it is necessary, however, to adapt Lemma 3.5.29 (a) to control for rounds. This is our second use of our Rank Lemma 3.4.4, this time applied to the set M_i from Definition 3.5.28.

Lemma 3.5.30. *Let G and H be finite non-Abelian groups, let $M_{i,G}$ (resp., $M_{i,H}$) denote the sets from Definition 3.5.28 for G (resp., H). Let $f : G \to H$ be a bijection that Duplicator selects. If for some i, $\text{rk}_{M_{i,G}}(g) \neq \text{rk}_{M_{i,H}}(f(g))$, then Spoiler can win with 3 pebbles and $O(\log n)$ rounds.*

Proof. Let $M_{i,G}, M_G$ be the sets in G as in Definition 3.5.28, and let $M_{i,H}, M_H$ be the corresponding sets in H. We show that $f(M_{i,G}) = M_{i,H}$ and $f(\langle M_{i,G}\rangle) = \langle M_{i,H}\rangle$. These statements imply that $f(M_G) = M_H$. The proof proceeds by induction over i, and within each i, we use the Rank Lemma 3.4.4 applied to M_i-rank. Note that each M_i is closed under taking inverses, since $C_G(g) = C_G(g^{-1})$.

We first note that if $|C_G(g)| \neq |C_H(f(g))|$, then Duplicator may win with 2 pebbles and 2 rounds. Without loss of generality, suppose that $|C_G(g)| > |C_H(f(g))|$. Spoiler pebbles $g \mapsto f(g)$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Now there exists $x \in C_G(g)$ such that $f'(x) \notin C_H(f(g))$. Spoiler pebbles $x \mapsto f'(x)$ and wins immediately.

Thus $M_{1,G}$ is identified by $(2,2)$-WL. By the Rank Lemma 3.4.4 applied to M_1-rank, we get that $f(\langle M_{1,G}\rangle) = \langle M_{1,H}\rangle$ or Spoiler can win with 3 pebbles in $O(\log n)$ rounds.

As Duplicator must select bijections $f : G \to H$ where $f(\langle M_{1,G}\rangle) = \langle M_{1,H}\rangle$, we may iterate on the above argument replacing 1 with i, to obtain that $f(M_{i,G}) = M_{i,H}$ and $f(\langle M_{i,G}\rangle) = \langle M_{i,H}\rangle$. The result now follows by induction. □

Definition 3.5.31. Let $G = N_1 \cdots N_m$ be the decomposition into non-Abelian components, and let $G = G_1 \times \cdots \times G_d$ be an arbitrary direct decomposition. We say that $x \in G$ is *full* for $(G_{j_1}, \ldots, G_{j_r})$, if

$$\{i \in [m] : [x, N_i] \neq 1\} = \bigcup_{\ell=1}^{r} I_{j_\ell},$$

where the I_{j_ℓ} are as in Lemma 3.5.29 (c). For all $x \in G$, define $C_x := \prod_{[x,N_i]=\{1\}} N_i$ and $N_x = \prod_{[x,N_i]\neq\{1\}} N_i$.

We now recall some technical lemmas from [49].

Remark 3.5.32 ([49, Observation 6.22])**.** For an arbitrary collection of indices $J \subseteq [m]$, the group elements $x \in G$ that have $C_x = \prod_{i\in J} N_i$ are exactly those elements of the form $x = z\prod_{i\in J} n_i$ with $z \in Z(G)$ and $n_i \in N_i \setminus Z(G)$. In particular, full elements exist for every collection of non-Abelian direct factors and any direct decomposition, and they are exactly given by products over non-central elements from the corresponding non-Abelian components.

Lemma 3.5.33 ([49, Lemma 6.23])**.** *Let G be non-Abelian, and let $G = G_1 \times \cdots \times G_d$ be an indecompsable direct decomposition. For all $x \in G$, we have a central decomposition $G = C_xN_x$, with $Z(G) \leq C_x \cap N_x$. The decomposition is directly induced if and only if x is full for a collection of direct factors of G.*

Lemma 3.5.34 (Compare rounds cf. [49, Lemma 6.24])**.** *Let $G = G_1\times G_2$. For $k \geq 4, r \in \Omega(\log n)$, assume that (k,r)-WL Version II detects $G_1Z(G)$ and $G_2Z(G)$. Let H be a group such that (k,r)-WL Version II does not distinguish G and H. Then for $i \in \{1,2\}$, there exist subgroups $H_i \leq H$ such that $H = H_1 \times H_2$ and (k,r)-WL does not distinguish $G_iZ(G)$ and $H_iZ(H)$.*

Proof. The proof is largely identical to that of [49, Lemma 6.24]. We adapt their proof to control for rounds.

As (k,r)-WL detects $G_1Z(G)$ and $G_2Z(G)$, we have that for any two bijections $f, f' : G \to H$ that $f(G_iZ(G)) = f'(G_iZ(G))$ for $i \in \{1,2\}$. It follows that there exist subgroups of $\widetilde{H_i} \leq H$ such that $f(G_iZ(G)) = \widetilde{H_i}$. As $Z(G) \leq G_iZ(G)$, we have necessarily that $Z(H) \leq \widetilde{H_i}$. Consider the decompositions $Z(G) = Z(G_1) \times Z(G_2)$ and $G_iZ(G) = G_i \times Z(G_{i+1 \mod 2})$. By Lemma 3.5.19, we have that if x splits from $Z(G)$, then x also splits from $G_1Z(G)$ or $G_2Z(G)$.

Write $\widetilde{H_i} = R_i \times B_i$, where B_i is a maximal Abelian direct factor of $\widetilde{H_i}$.

> **Claim 1:** For all choices of R_i, B_i, it holds that $R_1 \cap R_2 = \{1\}$. Otherwise, Spoiler can win with 2 additional pebbles and 2 additional rounds.

Proof of Claim 1. By assumption, $\widetilde{H_1} \cap \widetilde{H_2} = Z(H)$. So $R_1 \cap R_2 \leq Z(H)$. Suppose to the contrary that there exists $z \in R_1 \cap Z_2$ such that $|z| = p$ for some prime p. Then there also exists a central p-element w that splits from $Z(H)$ and where $z \in \langle w \rangle$ (for instance, we may take w to be a root of z^{p^N}, where N is the largest p-power order in the Abelian group $Z(H)$). Write $w = (r_i, b_i)$ with respect to the chosen direct decomposition for $\widetilde{H_i}$. As $z \in \langle w \rangle$, we have that $w^m = z \in R_1 \cap R_2$. So $w^m \neq 1$. Furthermore, we may write $w^m = (r_1^m, 1) = (r_2^m, 1)$. As w has p-power order, we have as well that $|b_i| < |r_i|$ for each $i \in \{1, 2\}$. Now w does not split from $\widetilde{H_i}$; otherwise, by Lemma 3.5.19, we would have that r_i splits from R_i. However, neither R_1 nor R_2 admit Abelian direct factors.

It follows that w splits from $Z(H)$, but not from $\widetilde{H_1}$ or $\widetilde{H_2}$. Such elements do not exist in $G_1Z(G)$ or $G_2Z(G)$. Thus, in this case, we have by Lemma 3.5.17 that Spoiler can win with 2 additional pebbles and 2 additional rounds. □

We next consider maximal Abelian direct factors $A \leq G$ and $B \leq H$. Write $H = R \times B$. By Theorem 3.5.20, we may assume that $A \cong B$. We now argue that R_1 and R_2 can be chosen such that $R_1R_2 \cap B = \{1\}$. For $i \in \{1, 2\}$, we may write:

$$\widetilde{H_i} = \langle (r_1, b_1), \ldots, (r_t, b_t) \rangle \leq R \times B.$$

As $B \leq \widetilde{H_i}$, we have that:

$$\widetilde{H_i} = \langle (r_1, 1), (1, b_1), \ldots, (r_t, 1), (b_t, 1) \rangle = \langle (r_1, 1), \ldots, (r_t, 1) \rangle \times B.$$

It follows that we may choose $R_1R_2 \leq R$. By Claim 1, we have that $R_1 \cap R_2 = \{1\}$. So $R_1R_2B = R_1 \times R_2 \times B \leq H$. As (k, r)-WL fails to distinguish G and H, we have necessarily that $|R_1| \cdot |R_2| \cdot |B| = |H|$. So in fact, $H = R_1 \times R_2 \times B$, which we may write as $(R_1 \times B_1) \times (R_2 \times B_2)$, where $B_i \leq H_i$ are chosen such that $B = B_1 \times B_2$ and B_i is isomorphic to a maximal Abelian direct factor of G_i. Furthermore, we have that $R_iZ(H) = \widetilde{H_i}$, by construction. The result follows. □

Lemma 3.5.35. *Let $G = N_1 \cdots N_m$ and $H = Q_1 \cdots Q_m$ be the decompositions of G and H into non-Abelian components. Let $G = G_1 \times \ldots \times G_d$ be a decomposition into indecomposable direct factors. Let $f : G \to H$ be the bijection that Duplicator selects. Let $k \geq 4, r \in \Omega(\log n)$. If $x \in G$ is full for $(G_{j_1}, \ldots, G_{j_r})$, but $f(x)$ is not full for a collection $(H_{j_1}, \ldots, H_{j_r})$ of indecomposable direct factors of H, then Spoiler may win with 5 pebbles and $O(\log n)$ rounds.*

Proof. Spoiler begins by pebbling $x \mapsto f(x)$. Let $f' : G \to H$ be the bijection Duplicator selects at the next round. By Lemma 3.5.30, we may assume that $f'(N_x) = N_{f(x)}$ and $f'(C_x) = C_{f(x)}$, or Spoiler wins with 3 pebbles and $O(\log n)$ rounds. So suppose that x and $f(x)$ are not distinguished by (k, r)-WL. Then as the central decomposition $G = C_x N_x$ is directly induced, we have that by Lemma 3.5.34, the central decomposition $H = C_{f(x)} N_{f(x)}$ has to be directly induced or Spoiler can win with 4 pebbles and $O(\log n)$ rounds. So by Lemma 3.5.33, we have that $f(x)$ is full.

As Duplicator preserves C_x and N_x, we obtain that if x is full for a collection of r direct factors, then so is $f(x)$. □

Corollary 3.5.36. *Let $G = G_1 \times \ldots \times G_d$ be a decomposition of G into directly indecomposable factors. Let H be arbitrary. Let $\mathcal{F}_G$ be the set of full elements for G, and define $\mathcal{F}_H$ analogously. If Duplicator does not select a bijection $f : G \to H$ satisfying:*

$$f\left(\bigcup_{g \in \mathcal{F}_G} N_g\right) = \bigcup_{h \in \mathcal{F}_H} N_h,$$

then Spoiler can win using 5 pebbles and $O(\log n)$ rounds.

Proof. By Lemma 3.5.35, we may assume that $f(\mathcal{F}_G) = \mathcal{F}_H$ (or Spoiler wins with 4 pebbles and $O(\log n)$ rounds). Now suppose that for some $g \in G$ that there exists an $x \in N_g$ such that $f(x) \not\in N_h$ for any $h \in \mathcal{F}_H$. Spoiler pebbles $x \mapsto f(x)$. Let $f' : G \to H$ be the bijection Duplicator selects at the next round. Again, we may assume that $f(\mathcal{F}_G) = \mathcal{F}_H$ (or Spoiler wins). Spoiler now pebbles $g \mapsto f'(g)$. Now on any subsequent bijection, Duplicator cannot map $N_g \mapsto N_{f'(g)}$. So by Lemma 3.5.30, Spoiler wins with 4 additional pebbles and $O(\log n)$ rounds. □

Theorem 3.5.37. *Let $k \geq 5, r \in \Omega(\log n)$. Let G be a non-Abelian group, and let $G = G_1 \times \ldots \times G_d$ be a decomposition into indecomposable direct factors. If (k, r)-WL Version II fails to distinguish G and H, then there exist indecomposable direct factors $H_i \leq H$ such that $H = H_1 \times \ldots \times H_d$ and $(k-1, r)$-WL Version II fails to distinguish G_i and H_i for all $i \in [d]$. Furthermore, G and H have isomorphic maximal Abelian direct factors and (k, r)-WL fails to distinguish $G_i Z(G)$ from $H_i Z(H)$.*

Proof. Without loss of generality, we may assume that H is non-Abelian as well. Let $f : G \to H$ be the bijection that Duplicator selects. By Corollary 3.5.36, we may assume that Duplicator preserves the full elements (or Spoiler wins with 4 pebbles and $O(\log n)$ rounds); that is, $f(\mathcal{F}_G) = \mathcal{F}_H$. It follows that H must admit a decomposition $H = H_1 \times \ldots \times H_\ell$, where the H_j factors are directly indecomposable and $\mathcal{F}_H = \bigcup_j H_j Z(H) \subseteq H$, which we again note is indistinguishable from $\mathcal{F}_G$. Let X_G be the non-commmuting graph of G, and let X_H be the non-commuting graph of H. Recall from [1, Proposition 2.1] that as G, H are non-Abelian, X_G and X_H are connected.

As different direct factors centralize each other, we obtain that for each non-singleton connected component in K of $X_G[\mathcal{F}_G]$ that there exists a unique indecomposable direct factor G_i such that $K = G_i Z(G) \setminus Z(G)$. Thus, $G_i Z(G) = \langle K \rangle$. Again by [1, Proposition 2.1], all such non-Abelian direct factors appear in this way.

We note that the claims in the preceeding paragraph applies to H as well. So if (k, r)-WL Version II does not distinguish G and H, there must exist a bijection between the connected components of $X_G[\mathcal{F}_G]$ and $X_H[\mathcal{F}_H]$. Namely, we may assume that G and H admit a decomposition into $\ell = d$ directly indecomposable factors, and that these subgroups are indistinguishable by (k, r)-WL. In particular, we have a correspondence (after an appropriate reordering of the factors) between $G_i Z(H)$ and $H_i Z(H)$, where $G_i Z(H)$ and $H_i Z(H)$ are not distinguished by (k, r)-WL. By Lemma 3.5.23, we have that (k, r)-WL Version II does not distinguish G_i from H_i. By Theorem 3.5.20, G and H must have isomorphic maximal Abelian direct factors. So when G_i, H_i are Abelian, we even have $G_i \cong H_i$. □

3.6 Weisfeiler–Leman for Semisimple Groups

In this section, we show that Weisfeiler–Leman can be fruitfully used as a tool to improve the parallel complexity of isomorphism testing of groups with no Abelian normal subgroups, also known as semisimple or Fitting-free groups. The main result of this section is:

Theorem 3.6.1. *Let G be a semisimple group, and let H be arbitrary. We can test isomorphism* *between G and H using an* SAC *circuit of depth $O(\log n)$ and size $n^{\Theta(\log \log n)}$. Furthermore, all such isomorphisms can be listed in this bound.*

The previous best complexity upper bounds were P for testing isomorphism [30], and $\mathsf{DTIME}(n^{O(\log \log n)})$ for listing isomorphisms [29].

We start with what we can observe from known results about direct products of simple groups. Brachter & Schweitzer previously showed that 3-WL Version II identifies direct products of finite simple groups. A closer analysis of their proofs [49, Lemmas 5.20 & 5.21] show that only $O(1)$ rounds are required. Thus, we obtain the following.

Corollary 3.6.2 (cf. Brachter & Schweitzer [49, Lemmas 5.20 & 5.21])**.** *Isomorphism between a direct product of non-Abelian simple groups and an arbitrary group can be decided in* L.

Our parallel machinery also immediately lets us extend a similar result to direct products of *almost* simple groups (a group G is almost simple if there is a non-Abelian simple group S such that $\text{Inn}(S) \leq G \leq \text{Aut}(S)$; equivalently, if $\text{Soc}(G)$ is non-Abelian simple).

Corollary 3.6.3. *Isomorphism between a direct product of almost simple groups and an arbitrary group can be decided in* TC^1.

Proof. Because almost simple groups are 3-generated [70], they are identified by $(O(1), O(1))$-WL. By Theorem 3.5.1, direct products of almost simple groups are thus identified by $(O(1), O(\log n))$-WL. □

3.6.1 Preliminaries

We recall some facts about semisimple groups from [29]. As a semisimple group G has no Abelian normal subgroups, we have that $\text{Soc}(G)$ is the direct product of non-Abelian simple groups. The conjugation action of G on $\text{Soc}(G)$ permutes the direct factors of $\text{Soc}(G)$. So there exists a faithful permutation representation $\alpha : G \to G^* \leq \text{Aut}(\text{Soc}(G))$. G is determined by $\text{Soc}(G)$ and the action α. Let H be a semisimple group with the associated action $\beta : H \to \text{Aut}(\text{Soc}(H))$. We have that $G \cong H$ precisely if $\text{Soc}(G) \cong \text{Soc}(H)$ via an isomorphism that makes α equivalent to β.

We now introduce the notion of permutational isomorphism, which is our notion of equivalence for α and β. Let A and B be finite sets, and let $\pi : A \to B$ be a bijection. For $\sigma \in \text{Sym}(A)$, let $\sigma^{\pi} \in \text{Sym}(B)$ be defined by $\sigma^{\pi} := \pi^{-1}\sigma\pi$. For a set $\Sigma \subseteq \text{Sym}(A)$, denote $\Sigma^{\pi} := \{\sigma^{\pi} : \sigma \in \Sigma\}$. Let $K \leq \text{Sym}(A)$ and $L \leq \text{Sym}(B)$ be permutation groups. A bijection $\pi : A \to B$ is a *permutational isomorphism* $K \to L$ if $K^{\pi} = L$.

The following lemma, applied with $R = \text{Soc}(G)$ and $S = \text{Soc}(H)$, precisely characterizes semisimple groups [29].

Lemma 3.6.4 ([29, Lemma 3.1]). *Let G and H be groups, with $R \triangleleft G$ and $S \triangleleft H$ groups with trivial centralizers. Let $\alpha : G \to G^* \leq \text{Aut}(R)$ and $\beta : H \to H^* \leq \text{Aut}(S)$ be faithful permutation representations of G and H via the conjugation action on R and S, respectively. Let $f : R \to S$ be an isomorphism. Then f extends to an isomorphism $\hat{f} : G \to H$ if and only if f is a permutational isomorphism between G^* and H^*; and if so, $\hat{f} = \alpha f^* \beta^{-1}$, where $f^* : G^* \to H^*$ is the isomorphism induced by f.*

We also need the following standard group-theoretic lemmas. The first provides a key condition for identifying whether a non-Abelian simple group belongs in the socle. Namely, if $S_1 \cong S_2$ are non-Abelian simple groups where S_1 is in the socle and S_2 is not in the socle, then the normal closures of S_1 and S_2 are non-isomorphic. In particular, the normal closure of S_1 is a direct product of non-Abelian simple groups, while the normal closure of S_2 is not a direct product of non-Abelian simple groups. We will apply this condition later when S_1 is a simple direct factor of $\text{Soc}(G)$; in which case, the normal closure of S_1 is of the form S_1^k. We include the proofs of these two lemmas for completeness.

Lemma 3.6.5. *Let G be a finite semisimple group. A subgroup $S \leq G$ is contained in Soc(G) if and only if the normal closure of S is a direct product of nonabelian simple groups.*

Proof. Let N be the normal closure of S. Since the socle is normal in G and N is the smallest normal subgroup containing S, we have that S is contained in $\text{Soc}(G)$ if and only if N is.

Suppose first that S is contained in the socle. Since $\mathrm{Soc}(G)$ is normal and contains S, by the definition of N we have that $N \leq \mathrm{Soc}(G)$. As N is a normal subgroup of G, contained in $\mathrm{Soc}(G)$, it is a direct product of minimal normal subgroups of G, each of which is a direct product of non-Abelian simple groups.

Conversely, suppose N is a direct product of nonabelian simple groups. We proceed by induction on the size of N. If N is minimal normal in G, then N is contained in the socle by definition. If N is not minimal normal, then it contains a proper subgroup $M \lneq N$ such that M is normal in G, hence also $M \trianglelefteq N$. However, as N is a direct product of nonabelian simple groups $T_1, \ldots, T_k$, the only subgroups of N that are normal in N are direct products of subsets of $\{T_1, \ldots, T_k\}$, and all such normal subgroups have direct complements. Thus we may write $N = L \times M$ where both L, M are nontrivial, hence strictly smaller than N, and both L and M are direct product of nonabelian simple groups.

We now argue that L must also be normal in G. Since conjugating N by $g \in G$ is an automorphism of N, we have that $N = gLg^{-1} \times gMg^{-1}$. Since M is normal in G, the second factor here is just M, so we have $N = gLg^{-1} \times M$. But since the direct complement of M in N is unique (since N is a direct product of *non-Abelian* simple groups), we must have $gLg^{-1} = L$. Thus L is normal in G.

By induction, both L and M are contained in $\mathrm{Soc}(G)$, and thus so is N. We conclude since $S \leq N$. □

Corollary 3.6.6. *Let G be a finite semisimple group. A nonabelian simple subgroup $S \leq G$ is a direct factor of Soc(G) if and only if its normal closure $N = ncl_G(S)$ is isomorphic to S^k for some $k \geq 1$ and $S \trianglelefteq N$.*

Proof. Let S be a nonabelian simple subgroup of G. If S is a direct factor of $\mathrm{Soc}(G)$, then $\mathrm{Soc}(G) = S^k \times T$ for some $k \geq 1$ and some T; choose T such that k is maximal. Then the normal closure of S is a minimal normal subgroup of $\mathrm{Soc}(G)$ which contains S as a normal subgroup. Since the normal subgroups of a direct product of nonabelian simple groups are precisely direct products of subsets of the factors, the normal closure of S is some $S^{k'}$ for $1 \leq k' \leq k$.

Conversely, suppose the normal closure N of S is isomorphic to S^k for some $k \geq 1$ and $S \trianglelefteq N$. By Lemma 3.6.5, S is in $\text{Soc}(G)$, and thus so is N (being the normal closure of a subgroup of the socle). Furthermore, as a normal subgroup of G contained in $\text{Soc}(G)$, N is a direct product of minimal normal subgroups and a direct factor of $\text{Soc}(G)$ (in fact it is minimal normal itself, but we haven't established that yet, nor will we need to). Since S is a normal subgroup of N, and N is a direct product of non-Abelian simple groups, S is a direct factor of N. Since N is a direct factor of $\text{Soc}(G)$, and S is a direct factor of N, S is a direct factor of $\text{Soc}(G)$. This completes the proof. □

Lemma 3.6.7. *Let $S_1, \ldots, S_k \leq G$ be nonabelian simple subgroups such that for all distinct $i, j \in [k]$ we have $[S_i, S_j] = 1$. Then $\langle S_1, \ldots, S_k \rangle = S_1 S_2 \cdots S_k = S_1 \times \cdots \times S_k$.*

Proof. By induction on k. The base case $k = 1$ is vacuously true. Suppose $k \geq 2$ and that the result holds for $k - 1$. Then $T := S_1 S_2 \cdots S_{k-1} = S_1 \times \cdots \times S_{k-1}$. Now, since S_k commutes with each S_i, and they generate T, we have that $[S_k, T] = 1$. Hence T is contained in the normalizer (or even the centralizer) of S_k, so $TS_k = S_k T = \langle T, S_k \rangle$, and S_k and T are normal subgroups of TS_k. As $TS_k = \langle T, S_k \rangle$ and T, S_k are both normal subgroups of TS_k with $[T, S_k] = 1$, we have that TS_k is a central product of T and S_k. As $Z(T) = Z(S_k) = 1$, it is their direct product. □

3.6.2 Groups without Abelian Normal Subgroups in Parallel

Here we establish Theorem 3.6.1. We begin with the following.

Proposition 3.6.8. *Let G be a semisimple group of order n, and let H be an arbitrary group of order n. If H is not semisimple, then 3-WL will distinguish G and H in at most 4 rounds.*

Proof. Recall that a group is semisimple if and only if it contains no Abelian normal subgroups. As H is not semisimple, $\text{Soc}(H) = A \times T$, where A is the non-trivial direct product of elementary Abelian groups and T is a direct product of non-Abelian simple groups. We show that Spoiler can win using at most 3 pebbles on the board and at most 4 rounds. Let $f : G \to H$ be the bijection that Duplicator selects. Let $a \in A$ such that $a \neq 1$. So $\text{ncl}_H(a) \leq A$. Let $b := f^{-1}(a) \in G$ (note that as $a \neq 1$, we have that $b \neq 1$; otherwise, Spoiler immediately wins by pebbling $b \mapsto a$), and let $B := \text{ncl}_G(b)$. As G is semisimple, we have that B is not Abelian. Spoiler begins by pebbling $b \mapsto a$.

So there exist $g_1, g_2 \in G$ such that $g_1bg_1^{-1}$ and $g_2bg_2^{-1}$ do not commute (for B is generated by $\{gbg^{-1} : g \in G\}$, and if they all commuted then B would be Abelian). Let $f', f'' : G \to H$ be the bijections that Duplicator selects at the next two rounds. Spoiler pebbles $g_1 \mapsto f'(g_1)$ and $g_2 \mapsto f''(g_2)$ at the next two rounds. As $\text{ncl}(a) \leq A$ is Abelian, $f'(g_1)f(b)f'(g_1)^{-1}$ and $f''(g_2)f(b)f''(g_2)^{-1}$ commute. Spoiler now wins. □

We now apply Lemma 3.6.5 to show that Duplicator must map the direct factors of $\text{Soc}(G)$ to isomorphic direct factors of $\text{Soc}(H)$.

Lemma 3.6.9. *Let G, H be finite semisimple groups of order n. Let Fac(Soc(G)) denote the set of simple direct factors of Soc(G). Let $S \in$ Fac(Soc(G)) be a non-Abelian simple group. Let $a \in S$, and let $f : G \to H$ be the bijection that Duplicator selects.*

(a) If $f(a)$ does not belong to some element of Fac(Soc(H)), or

(b) If there exists some $T \in$ Fac(Soc(H)) such that $f(a) \in T$, but $S \not\cong T$,

then Spoiler wins with at most 4 pebbles and 5 rounds.

Proof. Spoiler begins by pebbling $a \mapsto f(a)$. At the next two rounds, Spoiler pebbles generators x, y for S. Let $f' : G \to H$ be the bijection Duplicator selects at the next round. Denote $T := \langle f'(x), f'(y) \rangle$. We note that if $T \not\cong S$ or $f(a) \notin T$, then Spoiler wins.

So suppose that $f(a) \in T$ and $T \cong S$. We have two cases.

- **Case 1:** Suppose first that T does not belong to $\text{Soc}(H)$. As $S \triangleleft \text{Soc}(G)$, the normal closure $\text{ncl}(S)$ is minimal normal in G [124, Exercise 2.A.7]. As T is not even contained in $\text{Soc}(H)$, we have by Lemma 3.6.5 that $\text{ncl}(T)$ is not a direct product of non-Abelian simple groups, so $\text{ncl}(S) \not\cong \text{ncl}(T)$. We note that $\text{ncl}(S) = \langle \{gSg^{-1} : g \in G\} \rangle$.

As $\text{ncl}(T)$ is not isomorphic to a direct power of S, there is some conjugate $gSg^{-1} \neq S$ such that $f'(g)Tf'(g)^{-1}$ does not commute with T, by Lemma 3.6.7. Yet since $S \trianglelefteq \text{Soc}(G)$, gSg^{-1} and S do commute. Spoiler moves the pebble pair from $a \mapsto f(a)$ and pebbles g with $f'(g)$. Since Spoiler has now pebbled x, y, g which generate $\langle S, gSg^{-1} \rangle = S \times gSg^{-1} \cong S \times S$ but the image is not isomorphic to $S \times S$, the map $(x, y, g) \mapsto (f'(x), f'(y), f'(g))$ does not extend to an isomorphism of $S \times T$. Spoiler now wins. In total, Spoiler used 3 pebbles and 4 rounds.

- **Case 2:** Suppose instead that $T \leq \text{Soc}(H)$, but that T is not normal in $\text{Soc}(H)$. As T is not normal in $\text{Soc}(H)$, there exists $Q = \langle q_1, q_2 \rangle \in \text{Fac}(\text{Soc}(H))$ such that Q does not normalize T. At the next two rounds, Spoiler pebbles q_1, q_2, and their respective preimages, which we label r_1, r_2. When pebbling $r_1 \mapsto q_1$, we may assume that Spoiler moves the pebble placed on $a \mapsto f(a)$. By Case 1, we may assume that $r_1, r_2 \in \text{Soc}(G)$, or Spoiler wins with an additional 1 pebble and 1 round. Now as $S \trianglelefteq \text{Soc}(G)$, $\langle r_1, r_2 \rangle$ normalizes S. However, Q does not normalize T. So the pebbled map $(x, y, r_1, r_2) \mapsto (f'(x), f'(y), q_1, q_2)$ does not extend to an isomorphism. Thus, Spoiler used 4 pebbles and 5 rounds.

□

Lemma 3.6.10. *Let G be a semisimple group. There is a logspace algorithm that decides, given $g_1, g_2 \in G$, whether $\langle g_1, g_2 \rangle \in$ Fac(Soc(G)).*

Proof. Using a membership test [39, 185], we may enumerate the elements of $S := \langle g_1, g_2 \rangle$ by a logspace transducer. We first check whether S is simple. For each $g \in S$, we check whether $\text{ncl}_S(g) = S$. This check is L-computable [192, Thm. 7.3.3].

It remains to check whether $S \in \text{Fac}(\text{Soc}(G))$. By Corollary 3.6.6, $S \in \text{Fac}(\text{Soc}(G))$ if and only if $N := \text{ncl}_G(S) = S^k$ for some k and $S \trianglelefteq N$. As S is simple, it suffices to check that each conjugate of S is either (1) equal to S or (2) intersects trivially with S and commutes with S. For a given $g \in G$ and each $h \in S$, we may check whether $h \in gSg^{-1}$. If there exist non-trivial $h_1, h_2 \in S$

such that $h_1 \in gSg^{-1}$ and $h_2 \notin gSg^{-1}$, we return that $S \notin \text{Fac}(\text{Soc}(G))$. Otherwise, we know that all conjugates of S are either equal to S or intersect S trivially. Next we check that those conjugates that intersect S trivially commute with S. For each $g \in G, h_1, h_2 \in S$ we check whether $gh_1g^{-1} \in S$; if not, we check that $[gh_1g^{-1}, h_2] = 1$. If not, then we return that $S \notin \text{Fac}(\text{Soc}(G))$. If all these tests pass, then S is a direct factor of the socle. For both of these procedures, we only need to iterate over 3- and 4-tuples of elements of G or S, so this entire procedure is L-computable. The result follows. □

Lemma 3.6.11. *Let G be a semisimple group. We can compute the direct factors of Soc(G) using a logspace transducer.*

Proof. Using Lemma 3.6.10, we may identify in L the ordered pairs that generate direct factors of $\text{Soc}(G)$. Now for $x \in G$ and a pair (g_1, g_2) that generates a direct factor of $\text{Soc}(G)$, define an indicator $Y(x, g_1, g_2) = 1$ if and only if $x \in \langle g_1, g_2 \rangle$. We may use a membership test [39, 185] to decide in L whether $x \in \langle g_1, g_2 \rangle$. Thus, we are able to write down the direct factors of $\text{Soc}(G)$ and their elements in L. □

We now prove Theorem 3.6.1

Proof of Theorem 3.6.1 . We first note that, by Lemma 3.6.9, if $\text{Soc}(G) \ncong \text{Soc}(H)$, then $(4, O(1))$-WL Version II will distinguish G from H. For in this case, there is some simple normal factor $S \in \text{Fac}(\text{Soc}(G))$ such that there are more copies of S in $\text{Fac}(\text{Soc}(G))$ than in $\text{Fac}(\text{Soc}(H))$. Thus under any bijection Duplicator selects, some element of S must get mapped into a simple direct factor of $\text{Soc}(H)$ that is not isomorphic to S, and thus by Lemma 3.6.9, Spoiler can win with 4 pebbles and 5 rounds.

So suppose $\text{Soc}(G) \cong \text{Soc}(H)$. By Lemma 3.6.11, in L we may enumerate the non-Abelian simple direct factors of $\text{Soc}(G)$ and $\text{Soc}(H)$. Furthermore, we may decide in L—and therefore, SAC^1—with a membership test [39, 185] whether two non-Abelian simple direct factors of the socle are conjugate. Thus, in SAC^1, we may compute a decomposition $\text{Soc}(G)$ and $\text{Soc}(H)$ of the form

$T_1^{t_1} \times \cdots \times T_k^{t_k}$, where each T_i is non-Abelian simple and each $T_i^{t_i}$ is minimal normal.

There are $O(|S|^2)$ automorphisms of each simple factor $|S|$, and so there are at most

$$n^2 k! \prod_{i=1}^{k} t_i!$$

isomorphisms between $\text{Soc}(G)$ and $\text{Soc}(H)$. From [29], we note that this quantity is bounded by $n^{O(\log \log n)}$. (This is bound is tight, as in the case of the groups A_5^k.) Given a bijection $\psi : \text{Fac}(\text{Soc}(G)) \to \text{Fac}(\text{Soc}(H))$, we may in L enumerate the $O(n^2)$ isomorphisms between $\text{Soc}(G)$ and $\text{Soc}(H)$ respecting ψ by fixing generators of each element $S \in \text{Fac}(\text{Soc}(G))$ and enumerating their possible images in $\psi(S)$.

We now turn to testing isomorphism of G and H. To do so, we use the individualize and refine strategy. We individualize in G arbitrary generators for each element of $\text{Fac}(\text{Soc}(G))$ (2 for each factor). Then for each configuration of generators for the elements of $\text{Fac}(\text{Soc}(H))$, we individualize those in such a way that respects ψ. Precisely, if $\psi(S) = T$ and (g_1, g_2) are individualized in S, then for the desired generators (h_1, h_2) of T, we individualize h_i to receive the same color as g_i. Observe that in two more rounds, no two elements of $\text{Soc}(G)$ have the same color. Similarly, in two more rounds, no two elements of $\text{Soc}(H)$ have the same color. However, an element of $\text{Soc}(G)$ and an element of $\text{Soc}(H)$ may share the same color.

Suppose now that $G \not\cong H$. Let $f : G \to H$ be the bijection that Duplicator selects. As $G \not\cong H$, there exists $g \in G$ and $s \in \text{Soc}(G)$ such that $f(gsg^{-1}) \neq f(g)f(s)f(g^{-1})$. Spoiler pebbles g. Let $f' : G \to H$ be the bijection Duplicator selects at the next round. As no two elements of $\text{Soc}(G)$ have the same color and no two elements of $\text{Soc}(H)$ have the same color, we have that $f'(s) = f(s)$. Spoiler pebbles s and wins. So after the individualization step, $(2, 4)$-WL Version II will decide whether the given map extends to an isomorphism of $G \cong H$. Now $(2, 4)$-WL Version II is L-computable, and so SAC^1 computable. As we have to test at most $n^{O(\log \log n)}$ isomorphisms of $\text{Soc}(G) \cong \text{Soc}(H)$, our circuit has size $n^{O(\log \log n)}$. The result now follows. □

Remark 3.6.12. Here, we use Weisfeiler–Leman to decide in L whether a given isomorphism of $\text{Soc}(G) \cong \text{Soc}(H)$ extends to an isomorphism of $G \cong H$. This procedure was known to be polynomial-time computable via membership checking in the setting of permutation groups (given by their generators); the proof of [29, Proposition 3.1] cites [83]. Furthermore, membership checking in the setting of permutation groups is known to be NC-computable [22]. The proof from [22] is quite involved; it effectively computes Schreier generators using $O(\log n)$ iterations. Furthermore, multiplying a product of permutations is FL-complete [68]. Thus, it does not appear that membership testing in the permutation group is known to be even AC^1-computable. So already, our $\mathsf{quasiSAC}^1$ bound is new. Furthermore, Weisfeiler–Leman provides a much simpler algorithm. We note, however, that Weisfeiler–Leman requires access to the multiplication table for the underlying group. Thus, this technique cannot be leveraged for more general membership testing in permutation groups.

We also obtain the following corollary, which improves upon [29, Corollary 4.4] in the direction of parallel complexity.

Corollary 3.6.13. *Let G and H be semisimple with $Soc(G) \cong Soc(H)$. If $Soc(G) \cong Soc(H)$ have $O(\log n/\log\log n)$ non-Abelian simple direct factors, then we can decide isomorphism between G and H, and list all the isomorphisms between G and H in L.*

3.7 Count-Free Weisfeiler–Leman

In this section, we examine the consequences for parallel complexity of the *count-free* WL algorithm. Our first main result here is to show a $\Omega(\log |G|)$ lower bound (optimal and maximal, up to the constant factor) on count-free WL-dimension for identifying Abelian groups (Theorem 3.7.9). Despite this result showing that count-free WL on its own is not useful for testing isomorphism of Abelian groups, we nonetheless use count-free WL for Abelian groups, in combination with a few other ideas, to get improved upper bounds on the parallel complexity of testing isomorphism (Theorem 3.7.15) of Abelian groups.

We begin by defining analogous pebble games and logics for the three count-free WL versions. Furthermore, we establish the equivalence of the three count-free WL versions up to $O(\log n)$ rounds.

These results extend [48, Section 3] to the count-free setting.

3.7.1 Equivalence Between Count-Free WL, Pebble Games, and Logics

We define analogous pebble games for count-free WL Versions I-III. The count-free $(k+1)$-pebble game consists of two players: Spoiler and Duplicator, as well as $(k+1)$ pebble pairs (p, p'). In Versions I and II, Spoiler wishes to show that the two groups G and H are not isomorphic; and in Version III, Spoiler wishes to show that the corresponding graphs Γ_G, Γ_H are not isomorphic. Duplicator wishes to show that the two groups (Versions I and II) or two graphs (Version III) are isomorphic. Each round of the game proceeds as follows.

(1) Spoiler picks up a pebble pair (p_i, p_i').

(2) The winning condition is checked. This will be formalized later.

(3) In Versions I and II, Spoiler places one of the pebbles on some group element (either p_i on some element of G or p_i' on some element of H). In Version III, Spoiler places one of the pebbles on some vertex of one of the graphs (either p_i on some vertex of Γ_G or p_i' on some element of Γ_H).

(4) Duplicator places the other pebble on some element of the other group (Versions I and II) or some vertex of the other graph (Version III).

Let $v_1, \ldots, v_m$ be the pebbled elements of G (resp., Γ_G) at the end of step 1, and let $v_1', \ldots, v_m'$ be the corresponding pebbled vertices of H (resp., Γ_H). Spoiler wins precisely if the map $v_\ell \mapsto v_\ell'$ does not extend to a marked equivalence in the appropriate version of WL. Duplicator wins otherwise. Spoiler wins, by definition, at round 0 if G and H do not have the same number of elements. We note that G and H (resp., Γ_G, Γ_H) are not distinguished by the first r rounds of k-WL if and only if Duplicator wins the first r rounds of the $(k+1)$-pebble game.

The count-free r-round, k-WL algorithm for graphs is equivalent to the r-round, $(k+1)$-pebble count-free pebble game [55]. Thus, the count-free r-round, k-WL Version III algorithm for groups introduced in Brachter & Schweitzer [48] is equivalent to the r-round, $(k+1)$-pebble count-free pebble game on the graphs Γ_G, Γ_H associated to the groups G, H. We establish the same equivalence for the count-free WL Versions I and II.

Lemma 3.7.1. *Let $\overline{g} := (g_1, \ldots, g_k) \in G^k$ and $\overline{h} := (h_1, \ldots, h_k) \in H^k$. If the count-free (k, r)-WL distinguishes $\overline{g}$ and $\overline{h}$, then Spoiler can win in the count-free $(k+1)$-pebble game within r moves on the initial configuration $(\overline{g}, \overline{h})$. (We use the same version of WL and the pebble game).*

Proof.

- **Version I:** For $r = 0$, then $\overline{g}$ and $\overline{h}$ differ with respect to the Version I marked equivalence type. Fix $r > 0$. Suppose that $\chi_r(\overline{g}) \neq \chi_r(\overline{h})$. We have two cases. Suppose first that $\chi_{r-1}(\overline{g}) \neq \chi_{r-1}(\overline{h})$. Then by the inductive hypothesis, Spoiler can win in the $(k+1)$-pebble game using at most $r-1$ moves.

  Suppose instead that $\chi_{r-1}(\overline{g}) = \chi_{r-1}(\overline{h})$. So without loss of generality, there exists an $x \in G$ such that the color configuration $(\chi_{r-1}(\overline{g}(g_1/x)), \ldots, \chi_{r-1}(\overline{g}(g_k/x))$ does not appear amongst the colored k-tuples of H. Thus, for some $j \in [k]$ and all $y \in H$, $\chi_{r-1}(\overline{g}(g_j/x)) \neq \chi_{r-1}(\overline{h}(h_j/y))$. Spoiler moves pebble p_j to x. By the inductive hypothesis, Spoiler wins with $r-1$ additional moves.

- **Version II:** We modify the Version I argument above to use the Version II marked equivalence type. Otherwise, the argument is identical.

□

We now prove the converse.

Lemma 3.7.2. *Let $\overline{g} := (g_1, \ldots, g_k) \in G^k$ and $\overline{h} := (h_1, \ldots, h_k) \in H^k$. Suppose that Spoiler can win in the count-free $(k+1)$-pebble game within r moves on the initial configuration $(\overline{g}, \overline{h})$. Then the count-free (k, r)-WL distinguishes $\overline{g}$ and $\overline{h}$. (We use the same version of WL and the pebble game).*

Proof.

- **Version I:** If $r = 0$, then the initial configuration is already a winning one for Spoiler. By definition, $\overline{g}, \overline{h}$ receive different colorings at the initial round of WL. Let $r > 0$, and suppose that Spoiler wins at round $r > 1$ of the pebble game. Suppose that at round r, Spoiler moved the jth pebble from g_j to x. Suppose Duplicator responded by moving the corresponding pebble from h_j to y. Then the map $(g_1, \ldots, g_{j-1}, x, g_{j+1}, \ldots, g_k) \mapsto (h_1, \ldots, h_{j-1}, y, h_{j+1}, \ldots, h_k)$ is not a marked equivalence. By the inductive hypothesis, $\overline{g}(g_j/x)$ and $\overline{h}(h_j/x)$ receive different colors at round $r-1$ of k-WL. As Spoiler had a winning strategy by moving the jth pebble from $g_j \mapsto x$, we have that for any $y \in H$, $\chi_{r-1}(\overline{g}(g_j/x)) \neq \chi_{r-1}(\overline{h}(h_j/))$. By the definition of the WL refinement, it follows that $\chi_r(\overline{g}) \neq \chi_r(\overline{h})$. The result follows.

- **Version II:** We modify the Version I argument above to use the Version II marked equivalence type. Otherwise, the argument is identical.

□

Lemma 3.7.3. *Let G and H be groups of order n. Consider the count-free k-pebble game on the graphs Γ_G and Γ_H. If $k \geq 6$ and one of the following happens:*

(a) Spoiler places a pebble p on a vertex corresponding to a group element $g \in G$ and Duplicator places the corresponding pebble p' on a vertex v that does not correspond to a group element of H, then Spoiler can win with at most 4 additional pebbles and 4 additional rounds.

(b) Suppose that there is a pebble pair (p, p') for which pebble p is on some vertex of $M(g_1, g_2)$ that is not a group element and p' is on some vertex of $M(h_1, h_2)$ that is not a group element. Write $g_3 := g_1 g_2$ and $h_3 := h_1 h_2$. If Spoiler places a pebble on g_i $(i = 1, 2, 3)$ and Duplicator does not respond by pebbling h_i (or vice-versa), then Spoiler can win with 4 pebbles and $O(1)$ additional rounds.

(c) *the map induced by the group elements pebbled or implicitly pebbled by $k-2$ pebbles does not extend to an isomorphism between the corresponding generated subgroups, then Spoiler can win with* 2 *additional pebbles and $O(\log n)$ additional rounds.*

Proof. We have the following.

(a) The vertices that do not correspond to group elements have degree at most 3. So Spoiler can win with 4 additional pebbles by pebbling the neighbors of the vertex corresponding to the group element.

(b) We note that if pebble p is not on the same type of vertex (i.e., type a, b, c, or d, as in Figure 2.1) as pebble p', then Spoiler wins in $O(1)$ more rounds with at most 4 more pebbles and 4 more rounds, as either the vertices or their neighbors have different degrees.

So suppose now that p and p' are on the same type of vertex. Now without loss of generality, suppose that pebble q is placed on g_i for some $i = 1, 2, 3$ and the corresponding pebble q' is not placed on h_i. Observe that for each non-group-element vertex in a gadget $M(g_1, g_2)$, the distances to the three group-element vertices of that gadget are all distinct, and any path from a vertex not in $M(g_1, g_2)$ to a non-group-element vertex in $M(g_1, g_2)$ must go through one of the group element vertices g_1, g_2, g_1g_2. Thus, there is some k such that the vertex pebbled by p is connected to g_i by a path made up of exactly k non-group element vertices, but the same is not true for any path from the vertex pebbled by p' to that pebbled by q'. Using a third pebble pair, Spoiler can explore the path from p to g_i and win, using at most 7 additional rounds (as a multiplication gadget has 7 vertices).

(c) Suppose that the map $f : g_i \mapsto h_i$ for all $i \in [k-2]$ does not extend to an isomorphism of $\langle g_1, \ldots, g_{k-2} \rangle$ and $\langle h_1, \ldots, h_{k-2} \rangle$. Let $\hat{f} : \langle g_1, \ldots, g_{k-2} \rangle \to \langle h_1, \ldots, h_{k-2} \rangle$ be an extension of f. As $\hat{f}$ is not an isomorphism, there exists a smallest word $\omega = g_{i_1} \cdots g_{i_j}$ over $g_1, \ldots, g_{k-2}$ such that $\hat{f}(\omega) \neq \hat{f}(g_{i_1}) \cdots \hat{f}(g_{i_j})$. By minimality, we have that

$$\hat{f}(g_{i_1}) \cdots \hat{f}(g_{i_j}) = \hat{f}(g_{i_1}) \hat{f}(g_{i_2} \cdots g_{i_j}) \neq \hat{f}(\omega).$$

Spoiler pebbles ω in Γ_G, and Duplicator responds by pebbling ω' in Γ_H. Denote $\omega[x, \ldots, y] := g_{i_x} \cdots g_{i_y}$. At the next round, Spoiler implicitly pebbles $(\omega[1, \ldots, \lfloor j/2 \rfloor], \omega[\lfloor j/2 \rfloor + 1, \ldots, j])$. Duplicator responds by pebbling a pair (c, d). By part (b), we may assume that $cd = \omega'$; otherwise, Spoiler can win by reusing the pebble pair on ω, ω' and exploring the multiplication gadget $M(\omega[1, \ldots, \lfloor j/2 \rfloor], \omega[\lfloor j/2 \rfloor + 1, \ldots, j])$. So now either:

$$c \neq h_{i_1} \cdots h_{i_{\lfloor j/2 \rfloor}}, \text{ or}$$
$$d \neq h_{i_{\lfloor j/2 \rfloor + 1}} \cdots h_{i_j}.$$

Without loss of generality, suppose that:

$$c \neq h_{i_1} \cdots h_{i_{\lfloor j/2 \rfloor}}.$$

Spoiler iterates on the above strategy, starting from c rather than ω. We eventually reach the case of part (b), for a total of $\log_2 j + O(1) \leq \log_2 n + O(1)$ rounds. To see that two additional pebbles are required, after implicitly pebbling the multiplication gadget, Spoiler may move reuse the pebble from the previous round. The result now follows.

□

3.7.2 Logics

Brachter & Schweitzer [48, Lemma 3.6] showed that for $J \in \{I, II\}$ two k-tuples $\overline{g}, \overline{h}$ receive a different initial color under k-WL Version J if and only if there is a quantifier-free formula in $\mathcal{C}_J$ that distinguishes $\overline{g}, \overline{h}$. As such formulas do not use any quantifiers, $\overline{g}, \overline{h}$ receive a different initial color under k-WL Version J if and only if there is a quantifier-free formula in $\mathcal{L}_J$ that distinguishes $\overline{g}, \overline{h}$. Now the equivalence between the $(k + 1)$-pebble, r-round Version J count-free pebble game and the $(k+1)$-varible, quantifier-depth r fragment of $\mathcal{L}_J$ follows identically from the argument as in the case of graphs [55]. We record this with the following theorem.

Theorem 3.7.4. *Let G and H be groups of order n, and let $J \in \{I, II\}$. We have that the count-free (k, r)-WL Version J distinguishes G from H if and only if there exists a sentence $\varphi \in \mathcal{L}_J$ that* *uses at most $k+1$ variables and quantifier depth r, such that φ holds on one group but not the other.*

3.7.3 Equivalence of Count-Free WL Versions

We show that the three count-free WL Versions are equivalent, up to a factor of 2 in the dimension and up to a tradeoff of $O(\log n)$ additional rounds.

Definition 3.7.5. Let $k, k' \geq 2, r, r' \geq 1$, and $J, J' \in \{I, II, III\}$. We say that (k, r)-WL Version $J \preceq (k', r')$-WL Version J' if whenever (k, r)-WL Version J distinguishes groups G and H, then (k', r')-WL Version J' also distinguishes G and H.

Theorem 3.7.6. *Fix $k \geq 2$ and $r \geq 1$. In the count-free setting, we have the following:*

(a) (k, r)-WL Version $I \preceq (k, r)$-WL Version II,

(b) (k, r)-WL Version $II \preceq (\lceil k/2 \rceil + 2, 3r + O(\log n))$-WL Version III,

(c) $(\lceil k/2 \rceil + 2, 3r + O(\log n))$-WL Version $III \preceq (k+5, 6r + O(\log n))$-WL Version I.

We first note that count-free (k, r)-WL Version II can simulate each step of (k, r)-WL Version I. Thus, it remains to prove Theorem 3.7.6 (b)-(c). We do so with a series of lemmas.

Lemma 3.7.7. *Let G and H be groups of order n. Suppose that the count-free (k, r)-WL Version II distinguishes G and H. Then the count-free $(\lceil k/2 \rceil + 2, 3r + O(\log n))$-WL Version III algorithm distinguishes G and H.*

Proof. We adapt the strategy of [48, Lemma 3.11] to the count-free setting and control for rounds. Suppose that Spoiler has a winning strategy in the r-round Version II $(k+1)$-pebble game. Let $g_1, \ldots, g_r$ be the sequence of group elements that Spoiler pebbles. Suppose that at round $1 \leq i \leq r$ of the Version II game that Spoiler introduces a new pebble. In the Version III game: if there are an even number of group elements pebbled, then Spoiler pebbles the group element vertex

g_i; if instead there are an odd number of group elements pebbled, then Spoiler implicitly pebbles (g_i, g_{i+1}) and reuses the pebble on g_i at the next round.

Suppose that Spoiler instead moves a pebble at round i of the Version II game. If the corresponding pebble in the Version III game is on a group element vertex, then this is treated identically as in the case when a new pebble is introduced. Suppose instead the corresponding pebble in the Version III game is on a multiplication gadget vertex $M(a, b)$ and in the Version II game, Spoiler moves the pebble from b. In this case in the Version III game, Spoiler introduces a new pebble onto g_i, and then moves the pebble from $M(a, b)$ to a non-group element vertex of $M(a, g_i)$. At the next round, Spoiler reuses the pebble on g_i.

We now argue that, without loss of generality, we may assume that the configuration of pebbled group elements at the end of at most $3r$ rounds in the Version III game is the same configuration at the end of round r of the Version II game. Suppose at the end of round r of the Version II game that Duplicator has pebbled $(h_1, \ldots, h_k)$, and suppose that at the end of round $3r$ of the count-free pebble game that Duplicator has (implicitly) pebbled $(h_1', \ldots, h_k')$. By Lemma 3.7.3 (a), if a pebble p_i belongs to a group element (respectively, multiplication gadget) vertex and p_i' does not belong to a group element (respectively, multiplication gadget) vertex, then Spoiler can win with 2 pebbles and $O(1)$ rounds. So we may assume at the end of round $3r$ of the Version III game that Duplicator has (implicitly) pebbled k group elements.

Now suppose for a contradiction that Duplicator wins with this strategy in the Version III game, even with 2 additional pebbles and $O(\log n)$ additional rounds. Then by Lemma 2.8.2 (c), the map $(g_1, \ldots, g_k) \mapsto (h_1', \ldots, h_k')$ extends to an isomorphism of the subgroups $\langle g_1, \ldots, g_k \rangle$ and $\langle h_1', \ldots, h_k' \rangle$. So in the Version II game, Duplicator could have won by pebbling $(h_1', \ldots, h_k')$ rather than $(h_1, \ldots, h_k)$, contradicting the assumption that Spoiler wins at round r of the Version II pebble game. Thus, we may assume at the end of round r of the Version III game that $(h_1, \ldots, h_k) = (h_1', \ldots, h_k')$.

As Spoiler wins at the end of round r at the Version II game, we have that the induced map on the configurations does not extend to an isomorphism. So by Lemma 3.7.3 (c), Spoiler wins in the Version III game with 2 additional pebbles and $O(\log n)$ additional rounds, as desired. □

Lemma 3.7.8. *Let G and H be groups of order n. Suppose that the count-free (k, r)-WL Version III distinguishes G and H. Then the count-free $2k+1$-WL Version I algorithm distinguishes G and H in at most $2r$ rounds.*

Proof. We adapt the strategy of [48, Lemma 3.12] to the count-free setting and control for rounds. Suppose that at round $0 \leq i \leq r$ of the Version III pebble game, that Spoiler pebbles the group element vertex g_i. Then in the Version I game, Spoiler may pebble g_i. Suppose instead in the Version III game that Spoiler implicitly pebbles $M(x_1, x_2)$, and Duplicator responds by implicitly pebbling $M(y_1, y_2)$. We simulate this step in two rounds of the Version I game. At the first stage, Spoiler pebbles x_1. Duplicator responds by pebbling some group element y_1'. At the next stage, Spoiler pebbles x_2, and Duplicator responds by pebbling y_2'. Observe that at most 2 rounds of the Version I game are required to simulate 1 round of the Version III game.

Now suppose at round r of the Version III game that Duplicator has pebbled $(h_1, \ldots, h_d)$, where due to implicit pebbling, $d \leq 2k$. Suppose that at round $2r$ of the Version I game that Duplicator has pebbled $(h_1', \ldots, h_d')$. Now suppose for a contradiction that Duplicator wins at round $2r$ of the Version I game. Let Γ_G' be the induced subgraph $\Gamma_G[\{g_1, \ldots, g_d\}]$ together with the multiplication gadgets $M(g_i, g_j)$ for all $i, j \in [d]$ where $g_i g_j \in \{g_1, \ldots, g_d\}$. Define Γ_H' analogously for $\{h_1', \ldots, h_d'\}$. Consider the map $\varphi : V(\Gamma_G') \to V(\Gamma_H')$ induced by $(g_1, \ldots, g_d) \mapsto (h_1', \ldots, h_d')$. By the definition of the Version I winning condition, the map $(g_1, \ldots, g_d) \mapsto (h_1', \ldots, h_d')$ respects multiplication. Thus, φ will be a graph isomorphism, as multiplicativity can be expressed equivalently in terms of mapping the multiplication gadgets accordingly. It follows that Duplicator could have won in the Version III pebble game by (implicitly) pebbling $(h_1', \ldots, h_d')$ instead of $(h_1, \ldots, h_d)$, contradicting the assumption that Spoiler wins in the count-free $(k + 1)$-pebble, r-round Version III game. The result now follows. □

3.7.4 Count-Free WL and Abelian Groups

We now turn to showing that the count-free WL Version II algorithm fails to yield a polynomial-time isomorphism test for even Abelian groups.

Theorem 3.7.9. *For $n \geq 5$, let $G_n := (\mathbb{Z}/2\mathbb{Z})^n \times (\mathbb{Z}/4\mathbb{Z})^n$ and $H_n := (\mathbb{Z}/2\mathbb{Z})^{n-2} \times (\mathbb{Z}/4\mathbb{Z})^{n+1}$. The $\lfloor n/4 \rfloor$-dimensional count-free WL Version II algorithm does not distinguish G_n from H_n.*

Proof. The proof is by induction on the number of pebbles. For our first pebble, Spoiler may pebble one element in G_n, which generate one of the following subgroups: $\{1\}, \mathbb{Z}/2\mathbb{Z}$, or $\mathbb{Z}/4\mathbb{Z}$. For each of these options, Duplicator may respond in kind.

Now fix $1 \leq k \leq n/4$. Suppose that Duplicator has a winning strategy with k pebbles. In particular, we suppose that pebble pairs $(p_1, p_1'), \ldots, (p_k, p_k')$ have been placed on the board, and that the map $p_i \mapsto p_i'$ for all $i \in [k]$ extends to a marked isomorphism on a subgroup of the form $(\mathbb{Z}/2\mathbb{Z})^a \times (\mathbb{Z}/4\mathbb{Z})^b$. Furthermore, suppose that $0 \leq a_1 \leq a$ of the $\mathbb{Z}/2\mathbb{Z}$ direct factors of $\langle p_1, \ldots, p_k \rangle$ are contained in copies of $\mathbb{Z}/4\mathbb{Z}$ in G_n. Duplicator will maintain the invariant that the same number of copies of the $\mathbb{Z}/2\mathbb{Z}$ direct factors of $\langle p_1', \ldots, p_k' \rangle$ are contained in copies of $\mathbb{Z}/4\mathbb{Z}$ in H_n.

As Duplicator had a winning strategy in the k-pebble game, it is not to Spoiler's advantage to move pebbles $p_1, \ldots, p_k$. Thus, Spoiler picks up a new pebble p_{k+1}. Spoiler may pebble one additional element G_n. Now if the element Spoiler pebbles belongs to $\langle p_1, \ldots, p_k \rangle$; then as the map $p_i \mapsto p_i'$ for all $i \in [k]$ extends to a marked isomorphism, Duplicator may respond by pebbling the corresponding element in $\langle p_1', \ldots, p_k' \rangle$.

We may now assume that Spoiler does not pebble any element in $\langle p_1, \ldots, p_k \rangle$. Spoiler may pebble one additional element. We have the following cases.

- **Case 1:** As $k \leq n/4$, if Spoiler pebbles an element g_i generating $\mathbb{Z}/2\mathbb{Z}$, then Duplicator may respond in kind. We note that any copy of $\mathbb{Z}/2\mathbb{Z}$ that is pebbled and does not belong to $\langle p_1, \ldots, p_k \rangle$ is a direct complement to $\langle p_1, \ldots, p_k \rangle$ (in the sense that they commute and intersect trivially—they will not generate the whole group). Similarly, any copy of $\mathbb{Z}/2\mathbb{Z}$ that is pebbled and does not belong to $\langle p_1', \ldots, p_k' \rangle$ is a direct complement to $\langle p_1', \ldots, p_k' \rangle$.

Furthermore, as $k \leq n/4$, we may assume that Duplicator pebbles a copy of $\mathbb{Z}/2\mathbb{Z}$ that is contained within a copy of $\mathbb{Z}/4\mathbb{Z}$ in H_n if and only if Spoiler pebbles a copy of $\mathbb{Z}/2\mathbb{Z}$ that is contained within a copy of $\mathbb{Z}/4\mathbb{Z}$ in G_n.

- **Case 2:** We now consider the case in which Spoiler pebbles an element g_1 generating a copy of $\mathbb{Z}/4\mathbb{Z}$. We consider the following subcases.

  ∗ **Subcase 2(a):** As $k \leq n/4$, there exist copies of $\mathbb{Z}/4\mathbb{Z}$ in G_n that intersect trivially (and thus, are direct complements) with $\langle p_1, \ldots, p_k \rangle$; and similarly, there exist copies of $\mathbb{Z}/4\mathbb{Z}$ that intersect trivially (and thus, are direct complements) with $\langle p_1', \ldots, p_k' \rangle$. So if $\langle g_1 \rangle$ forms a direct complement with $\langle p_1, \ldots, p_k \rangle$, then Duplicator may respond by pebbling an element h_1 that generates a copy of $\mathbb{Z}/4\mathbb{Z}$ which is a direct complement with $\langle p_1', \ldots, p_k' \rangle$.

  ∗ **Subcase 2(b):** Suppose instead that $\langle g_1 \rangle \cong \mathbb{Z}/4\mathbb{Z}$ intersects properly and non-trivially with $\langle p_1, \ldots, p_k \rangle$. In this case, $\langle g_1 \rangle$ shares a copy of $\mathbb{Z}/2\mathbb{Z}$ with $\langle p_1, \ldots, p_k \rangle$. This yields two additional subcases.

    – **Subcase 2(b).i:** Suppose first that this copy of $\mathbb{Z}/2\mathbb{Z}$ is contained within a copy of $\mathbb{Z}/4\mathbb{Z} \leq \langle p_1, \ldots, p_k \rangle$. By the inductive hypothesis, both $\langle p_1, \ldots, p_k \rangle$ and $\langle p_1', \ldots, p_k' \rangle$ have a_1 copies of $\mathbb{Z}/2\mathbb{Z}$ that are contained in copies of $\mathbb{Z}/4\mathbb{Z}$ within G_n and H_n respectively. Using this fact, together with the fact that $k \leq n/4$, we have that Duplicator may respond by pebbling an element h_1 generating a copy of $\mathbb{Z}/4\mathbb{Z}$, where $\langle h_1 \rangle \cap \langle p_1', \ldots, p_k' \rangle$ is a copy of $\mathbb{Z}/2\mathbb{Z}$ that is contained within a copy of $\mathbb{Z}/4\mathbb{Z} \leq \langle p_1', \ldots, p_k' \rangle$.

    – **Subcase 2(b).ii:** Suppose instead that this copy of $\mathbb{Z}/2\mathbb{Z}$ is contained within a copy of $\mathbb{Z}/4\mathbb{Z} \not\leq \langle p_1, \ldots, p_k \rangle$. By the inductive hypothesis, both $\langle p_1, \ldots, p_k \rangle$ and $\langle p_1', \ldots, p_k' \rangle$ have a_1 copies of $\mathbb{Z}/2\mathbb{Z}$ that are contained in copies of $\mathbb{Z}/4\mathbb{Z}$ within G_n and H_n respectively. Using this fact, together with the fact that $k \leq n/4$, we have that Duplicator may respond by pebbling an element h_1 generating a copy of $\mathbb{Z}/4\mathbb{Z}$, where $\langle h_1 \rangle \cap \langle p_1', \ldots, p_k' \rangle$ is a copy of $\mathbb{Z}/2\mathbb{Z}$ that is not contained within a copy of $\mathbb{Z}/4\mathbb{Z} \leq \langle p_1', \ldots, p_k' \rangle$.

Thus, in all cases, Duplicator has a winning strategy at round $k+1$. The result now follows by induction. □

Remark 3.7.10. As the count-free k-WL runs in time $O(n^{k+1} \log n)$, Theorem 3.7.9 shows that count-free WL fails to serve as a polynomial-time (or even $|G|^{o(\log |G|)}$) isomorphism test for Abelian groups. In particular, by Theorem 3.7.6, our lower-bound holds (up to a constant factor in the number of pebbles) for all three versions of WL. As the $n/4$-dimensional count-free WL algorithm fails to distinguish G_n and H_n, we also obtain an $\Omega(\log(|G_n|))$ lower bound on the quantifier rank of any FO formula identifying G_n. In particular, this suggests that GpI is not in $\mathsf{FO}(\text{poly} \log \log n)$, even for Abelian groups. As $\mathsf{FO}(\text{poly} \log \log n)$ cannot compute Parity [183], this suggests that counting is necessary to solve GpI. This is particularly interesting, as Parity is not AC^0-reducible to GpI [57].

Remark 3.7.11. In subsequent work with N. Collins, we considered a higher arity version of the count-free pebble game, where Spoiler is able to place at most q pebbles in a single round. When q is fixed, we showed [65, Theorem 7.1] that Spoiler still requires $\Omega(\log n)$ pebbles to distinguish even Abelian groups.

While count-free WL is unable to distinguish Abelian groups, the multiset of colors computed actually provides enough information to do so. Barrington, Kadau, Lange, & McKenzie [38] previously showed that order-finding is FOLL-computable. Our next result (Proposition 3.7.13) shows that the count-free Weisfeiler–Leman effectively implements this strategy.

Lemma 3.7.12. *Let G, H be groups of order n. Suppose in the count-free WL-III game, pebbles have already been placed on $g \mapsto h$ and $g^i \mapsto x$ with $x \neq h^i$. Then Spoiler can win with $O(1)$ additional pebbles in $O(\log \log i)$ rounds.*

Proof. By induction on i. If $i = 0$ the result follows from the fact that the identity is the unique element such that the gadget $M(1, 1)$ has all three of its group element vertices the same and Lemma 3.7.3 (b). If $i = 1$, the result follows immediately from the winning condition of the game. So we now suppose $i > 1$, and that the result is true for all smaller exponents, say in $\leq c \log \log i'$ rounds for all $i' < i$.

The structure of the argument is as follows. If i is not a power of 2, we show how to cut the number of 1s in the binary expansion of i by half using $O(1)$ rounds and only $O(1)$ pebbles that may be reused. Since the number of 1s in the binary expansion of i is at most $\log_2 i$, and we cut this number in half each time, this takes only $O(\log \log i)$ rounds (and $O(1)$ pebbles) before i has just one 1 in its binary expansion, that is, i is a power of 2. Once i is a power of 2, we will show how to cut $\log_2 i$ in half using $O(1)$ rounds and $O(1)$ pebbles that may be reused. This takes only $O(\log \log i)$ rounds (and $O(1)$ pebbles) before getting down to the base case above. Concatenating these two strategies uses only $O(1)$ pebbles and $O(\log \log i)$ rounds. Now to the details.

If i is not a power of 2, we will show how cut the number of 1s in the binary expansion of i in half. Write $i = j + k$ where j, k each have at most half as many 1s in their binary expansion as i does (rounded up). (Examine the binary expansion $i_\ell i_{\ell-1} \cdots i_0$ and finding an index z such that half the ones are on either side of z. Then let j have binary expansion $i_\ell i_{\ell-1} \cdots i_z 00 \dots 0$ and let k have binary expansion $i_{z-1} i_{z-2} \cdots i_0$.) Spoiler implicitly pebbles (g^j, g^k). Duplicator responds by implicitly pebbling a pair (a, b). If $ab \neq x$, then we have a pebble on a non-group-element vertex of $M(g^j, g^k)$ as well as on its group element vertex $g^{j+k} = g^i$. But the corresponding pebbles are on $M(a, b)$ and x which differs from ab, so Spoiler wins by Lemma 3.7.3 (b). Thus we may now assume $ab = x$.

Since $x \neq h^i$, we necessarily have $\{a, b\} \neq \{h^j, h^k\}$. Without loss of generality, suppose $a \notin \{h^j, h^k\}$. Spoiler now picks up the pebble on g^i and places it on g^j instead. Because of the implicit pebble mapping $M(g^j, g^k) \mapsto M(a, b)$, Duplicator must respond by placing the pebble on a or Spoiler can win by Lemma 3.7.3 (b). At this point, Spoiler can reuse the implicit pebble on $M(g^j, g^k)$ and the pebble on g^i, and we are now in a situation where $g \mapsto h$ and $g^j \mapsto a \neq h^j$ are pebbled, and j has at most half as many 1s in its binary expansion as i did. (So there are only two pebbles that can't be re-used, which is precisely the number we started with.) The cost to get here was $O(1)$ rounds and no non-reusable pebbles.

After that has been iterated $\log \log i$ times, we come to the case where i is a power of 2. We will show how to reduce to a case where $\log_2 i$ has been cut in half. Write $i = jk$ with jk powers of 2 such that $\log_2 j, \log_2 k \leq \lceil \frac{\log_2 i}{2} \rceil$ (if $i = 2^z$, let $j = 2^{\lceil z/2 \rceil}, k = 2^{z - \lceil z/2 \rceil}$). Note that we have $g^i = (g^j)^k$. Spoiler now pebbles g^j, and Duplicator responds by pebbling some a. If $a \neq h^j$, then Spoiler can re-use the pebble from g^i, and we now have $g \mapsto h, g^j \mapsto a \neq h^j$ pebbled with $\log_2 j \leq \lceil (1/2) \log_2 i \rceil$. This took $O(1)$ rounds and no non-reusable pebbles. On the other hand, if $a = h^j$, then we have $a^k = h^{jk} = h^i \neq x$. Spoiler may now reuse the pebble on $g \mapsto h$, and we are now in a situation where $g^j \mapsto a$ and $(g^j)^k \mapsto x \neq a^k$, just as we started, and with $\log_2 k \leq \lceil (1/2) \log_2 i \rceil$. As in the other case, this took $O(1)$ rounds and no non-reusable pebbles. This completes the proof. □

Proposition 3.7.13 (Order finding in WL-III)**.** *Let G be a group. Let $g, h \in G$ such that $|g| \neq |h|$. The count-free $(O(1), O(\log \log n))$-WL Version III distinguishes g and h.*

Proof. We use the pebble game characterization, starting from the initial configuration $((g), (h))$. We first note that if $g = 1$ and $h \neq 1$, that Spoiler implicitly pebbles the multiplication gadget $M(1, 1)$. This is the unique multiplication gadget where all three group element vertices are the same. Regardless of what Duplicator pebbles, we have by Lemma 3.7.3 that Spoiler can win with $O(1)$ additional pebbles and $O(1)$ additional rounds.So now suppose that $g \neq 1$ and $h \neq 1$.

Without loss of generality suppose $|g| < |h|$. Note that $g \mapsto h$ has already been pebbled by assumption. Spoiler now pebbles 1. By the same argument as above, Duplicator must respond by pebbling 1. But we have $g^i = 1$ and by assumption $h^i \neq 1$. Thus, by Lemma 3.7.12, Spoiler can now win with $O(1)$ pebbles in $O(\log \log |g|) \leq O(\log \log n)$ rounds. □

As finite simple groups are uniquely identified amongst all groups by their order and the set of orders of their elements [191], we obtain the following immediate corollary.

Corollary 3.7.14. *If G is a finite simple group, then G is identified by the count-free $(O(1), O(\log \log n))$-WL. Consequently, isomorphism testing between a finite simple group G and an arbitrary group H is in FOLL.*

We also obtain an improved upper bound on the parallel complexity of Abelian Group Isomorphism:

Theorem 3.7.15. GpI *for Abelian groups is in $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$.*

Here, $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$ denotes the class of languages decidable by a (uniform) family of circuits that have $O(\log n)$ nondeterministic input bits, are of depth $O(\log \log n)$, have gates of unbounded fan-in, and the only gate that is not an AND, OR, or NOT gate is the output gate, which is a $\mathsf{Majority}$ gate of unbounded fan-in. Note that, by simulating the poly(n) possibilities for the nondeterministic bits, $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$ is contained in $\mathsf{TC}^0(\mathsf{FOLL})$, at the expense of using poly(n) $\mathsf{Majority}$ gates. Thus, our result improves on the prior upper bound of $\mathsf{TC}^0(\mathsf{FOLL})$ [57].

Theorem 3.7.15 is an example of the strategy of using *count-free* WL, followed by a limited amount of counting afterwards. (We contrast this with the parallel implementation of the classical (counting) WL algorithm, which—for fixed k—uses a polynomial number of $\mathsf{Majority}$ gates at each iteration [104].) After the fact, we realized this same bound could be achieved by existing techniques; we include both proofs to highlight an example of how WL was used in the discovery process.

Proof using Weisfeiler–Leman. Let G be Abelian, and let H be an arbitrary group such that $G \not\cong H$. Suppose first that H is not Abelian. We show that count-free $(O(1), O(1))$-WL can distinguish G from H. Suppose first that H is not Abelian. Spoiler implicitly pebbles a pair of elements (x, y) in H that do not commute. H responds by pebbling $(u, v) \in G$. At the next round, Spoiler implicitly pebbles (v, u). Regardless of what Duplicator pebbles, we have by Lemma 3.7.3 (b) that Spoiler wins with $O(1)$ additional pebbles and $O(1)$ additional rounds.

Suppose now that H is Abelian. We run the count-free $(O(1), O(\log \log n))$-WL using the parallel WL implementation due to Grohe & Verbitsky. As G and H are non-isomorphic Abelian groups, they have different order multisets. In particular, there exists a color class of greater multiplicity in G than in H. By Proposition 3.7.13, two elements with different orders receive different colors. We use a $\beta_1\mathsf{MAC}^0$ circuit to distinguish G from H. Using $O(\log n)$ non-deterministic bits, we guess the color class C where the multiplicity differs. At each iteration, the parallel WL implementation due to Grohe & Verbitsky records indicators as to whether two k-tuples receive the same color. As we have already run the count-free WL algorithm, we may in AC^0 decide whether two k-tuples have the same color. For each k-tuple of $V(\Gamma_G)^k$ having color class C, we feed a 1 to the Majority gate. For each k-tuple of $V(\Gamma_H)^k$ having color class C, we feed a 0 to the Majority gate. The Majority gate outputs a 1 if and only if there are strictly more 1's than 0's. The result now follows. □

Alternative proof using prior techniques, that we only realized after discovering the WL proof. This proof follows the strategy of Chattopadhyay, Torán, & Wagner [57], realizing that their use of many threshold gates can be replaced by $O(\log n)$ nondeterministic bits and a single threshold gate.

Compute the multiset of orders in FOLL [38, Prop. 3.1], guess the order k such that G has more elements of order k than H does. Use a single Majority gate to compare those counts. □

3.8 Canonizing Groups in Parallel via Weisfeiler–Leman

One approach to isomorphism testing is to canonize the input structures. Precisely, the goal is to compute a standard representation of the input structure that depends only on the isomorphism type and not on the representation of the object. In the setting of graphs, we may define a canonical form as follows.

Definition 3.8.1. A *graph canonization* for a graph class $\mathcal{C}$ is a function $\kappa : \mathcal{C} \to \mathcal{C}$ such that:

(a) $\kappa(G) \cong G$ for all $G \in \mathcal{C}$, and

(b) $\kappa(G) = \kappa(H)$ whenever $G \cong H$.

A *group canonization* may be defined similarly. The isomorphism problem for a class $\mathcal{C}$ of either graphs or groups reduces to computing a corresponding canonization. It is open both in the settings of graphs and groups, as to whether a reduction exists from canonization to isomorphism testing. However, combinatorial approaches to isomorphism testing, such as Weisfeiler–Leman, can easily be adapted to canonization procedures. We refer to Fortnow & Grochow [82] for a general study on the power of canonization, complete invariants, and polynomial-time algorithms for equivalence relations.

Theorem 3.8.2 (Folklore)**.** *Let $\mathcal{C}$ be a class of graphs, and suppose that k-WL identifies all colored graphs in $\mathcal{C}$. Then there exists a graph canonization for $\mathcal{C}$ that can be computed in time $O(n^{k+3} \log n)$.*

Remark 3.8.3. While Theorem 3.8.2 is well-known to those who work on Weisfeiler–Leman, an originating reference appears to be unknown. We defer to Grohe & Neuen for a proof of Theorem 3.8.2 [102, Appendix A].

3.8.1 Canonizing in Parallel via Weisfeiler–Leman

In this section, we show how to use Weisfeiler–Leman to obtain a parallel canonization procedure for groups. The key idea in establishing Theorem 3.8.2 is to individualize a vertex and then invoke WL on the colored graph. We iterate for each vertex, making n calls to k-WL. Thus, in the setting of graphs, the parallel WL implementation due to Grohe & Verbitsky [104] is necessary but not sufficient for canonization. However, groups have more structure. The key observation behind the generator-enumeration idea is that for $g_1, \ldots, g_k \in G$ and $g_{k+1} \not\in \langle g_1, \ldots, g_k \rangle$, $\langle g_1, \ldots, g_{k+1} \rangle$ has at least twice as many elements as $\langle g_1, \ldots, g_k \rangle$.

This suggests the following strategy to compute canonical forms for groups via WL. Suppose that (k, r)-WL identifies a group G. We first run $(k + 1, r)$-WL and select a group element g that corresponds to the lexicographically least color label. We individualize g_1, and then run $(k + 1, r)$-WL on this colored graph. Now as g_1 has been individualized and (k, r)-WL identifies G, we have that $(k + 1, r)$-WL assigns a unique color to each element in $\langle g_1 \rangle$. More generally, suppose that at iteration $i \geq 1$ that we individualize the group element g_i in WL Versions I and II, or the vertex corresponding to the group element g_i in WL Version III. At the next iteration when (k, r)-WL is applied, it assigns a unique color $\langle g_1, \ldots, g_n \rangle$. We invoke WL a total of $O(\log n)$ times.

The complexity of each iteration depends on the version of WL for groups that we use. For WL Versions I and III, each iteration of our canonization procedure can be computed with a logspace uniform TC circuit of depth r. Thus, if G is identified by (k, r)-WL, then a canonical form for G can be computed using a uniform TC circuit of depth $O(r \log n)$. For WL Version II, we consider separately the cases of $r \in O(1)$ and $r \in \omega(1)$. If $r \in O(1)$, then $(k + 1, r)$-WL Version II can be implemented in L. As canonization requires $O(\log n)$ calls to $(k + 1, r)$-WL Version II and $\mathsf{L} \subseteq \mathsf{SAC}^1$, this yields an upper bound of SAC^2 for canonization. Now if $r \in \omega(1)$, then $(k+1, r)$-WL Version II can be implemented using a TC-circuit of depth $O(\log n + r(n))$. As canonization requires $O(\log n)$ calls to $(k + 1, r)$-WL Version II, we may compute canonical forms using a TC circuit of depth $O(\log^2 n + r(n) \log n)$.

We first recall the notion of determining orbits. We say that (k, r)-WL *determines the orbits* of group (or graph G) if for every group (resp., graph) H, every element $g \in G$, and every $w \in H$ such that $\chi^G_{(k,r)}((v, \ldots, v)) = \chi^H_{(k,r)}((w, \ldots, w))$, there is an isomorphism $\varphi : G \cong H$ such that $\varphi(v) = w$.

We formalize this procedure with Algorithm 1.

We observe that once $g_1, \ldots, g_i$ have been individualized, WL will assign a unique color to each $g \in \langle g_1, \ldots, g_i \rangle$. For WL Version II, this happens at the first round, while in WL Version I, we require $r \in \Omega(\log n)$ rounds. At Line 11, we record the elements that receive a unique color. Thus, at Line 12, considering color classes of size greater than 1 ensures that we are considering elements outside of $\langle \text{Gens} \rangle$. So argmin picks an arbitrary element that minimizes $\chi_{G,i}(g)$.

Before establishing the correctness of Algorithm 1, we recall the following theorem from Grohe & Neuen [102].

Theorem 3.8.4 ([102, Theorem A.1]). *Let $\mathcal{C}$ be a class of graphs such that k-WL identifies all (colored) graphs $G \in \mathcal{C}$. Then $(k + 1)$-WL determines orbits for all graphs $G \in \mathcal{C}$.*

Algorithm 1 Canonization Algorithm for group class $\mathcal{C}$

```
Require: $G \in \mathcal{C}$ is identified by $(k, r)$-WL
Ensure: $\kappa(G)$
$n := |G|$
$G_0 = G$
Gens $:= \emptyset$
$\psi := \emptyset$
$i := 0$

while $\langle \text{Gens} \rangle \neq G$ do
    Let $\chi_i$ be the coloring computed by $(k+1, r)$-WL applied to $G_i$
    Define $\chi_{G,i+1}(v) = \chi_i^{G_i,k+1}(v, \ldots, v)$ for all $v \in G$
    For each $g \in G$ belonging to a color class of size 1, set $\psi(g) = \chi_{G,i+1}(g)$.
    Let $g_{i+1} \in \text{argmin}\{\chi_{G,i+1}(g) : g \in G \backslash \langle \text{Gens} \rangle, \text{The color class } \chi(g) \text{ has more than one element}\}$
    Gens := Gens $\cup \{g_{i+1}\}$.
    $\psi(g_{i+1}) = \chi_{G,i+1}(g_{i+1})$.

    Set $G_{i+1}$ to be the colored group arising from $G$, where the group elements are individualized according to $\psi$.
    Set $i := i + 1$
end while

return $\kappa(G) := ([n], \{(g, h, gh) : g, h \in G\}, g \mapsto \psi(g))$.
```

Remark 3.8.5. The original statement of [102, Theorem A.1] did not control for rounds, but the proof holds when we consider rounds. The proof also holds when we consider the count-free WL algorithm, as well as when we consider colored groups rather than graphs. For completeness, we provide a proof below.

Theorem 3.8.6. *We have the following.*

(a) Let $J \in \{I, II\}$, and suppose that $\mathcal{C}$ be a class of groups such that (k, r)-WL Version J identifies all colored groups $G \in \mathcal{C}$. Then $(k + 1, r)$ WL Version J determines orbits for all groups $G \in \mathcal{C}$.

(b) Let $J \in \{I, II\}$, and suppose that $\mathcal{C}$ be a class of groups such that the count-free (k, r)-WL Version J identifies all colored groups $G \in \mathcal{C}$. Then $(k + 1, r)$ WL Version J determines orbits for all groups $G \in \mathcal{C}$.

(c) Let $\mathcal{C}$ be a class of (colored) graphs such that the classical counting (k, r)-WL algorithm for graphs identifies all colored graphs $G \in \mathcal{C}$. Then $(k + 1, r)$-WL determines orbits for all graphs $G \in \mathcal{C}$.

(d) Let $\mathcal{C}$ be a class of (colored) graphs such that the count-free (k, r)-WL algorithm for graphs identifies all (colored) graphs $G \in \mathcal{C}$. Then count-free $(k + 1, r)$-WL determines orbits for all graphs $G \in \mathcal{C}$.

Proof. We proceed as follows.

(a) Let $G \in \mathcal{G}$, and let $v, w \in G$ such that the coloring $\chi_r^{G,k+1}(v, \ldots, v) = \chi_r^{G,k+1}(w, \ldots, w)$. Then (k, r)-WL fails to distinguish the colored group $(G, \chi_r^{(v)})$ from $(G, \chi_r^{(w)})$, where $(G, \chi_r^{(v)})$ is the colored group where we have individualized v to receive the color $\chi_r^{G,k+1}(v, \ldots, v)$. As (k, r)-WL identifies all groups in $\mathcal{G}$, we have that $(G, \chi_r^{(v)}) \cong (G, \chi^{(w)})$. So there is an automorphism $\varphi \in \text{Aut}(G)$ such that $\varphi(v) = w$.

(b) We modify the proof of (a) to use count-free (k, r)-WL Version J rather than the standard counting (k, r)-WL. The proof now goes through *mutatis mutandis.*

(c) Let $G \in \mathcal{G}$, and let $v, w \in V(G)$ such that the coloring $\chi_r^{G,k+1}(v, \ldots, v) = \chi_r^{G,k+1}(w, \ldots, w)$. Then (k, r)-WL fails to distinguish the colored graph $(G, \chi_r^{(v)})$ from $(G, \chi_r^{(w)})$, where $(G, \chi_r^{(v)})$ is the colored graph where we have individualized v to receive the color $\chi_r^{G,k+1}(v, \ldots, v)$. As (k, r)-WL identifies all graphs in $\mathcal{G}$, we have that $(G, \chi_r^{(v)}) \cong (G, \chi^{(w)})$. So there is an automorphism $\varphi \in \text{Aut}(G)$ such that $\varphi(v) = w$.

(d) We modify the proof of (c) to use count-free (k, r)-WL rather than the standard counting (k, r)-WL. The proof now goes through *mutatis mutandis*.

□

We now establish the correctness of Algorithm 1.

Theorem 3.8.7. *Let $k \geq 2$ be a constant, and let $r := r(n)$ be a function, where n denotes the order of the input groups. Let $J \in \{I, II\}$. Let $\mathcal{C}$ be a class of groups such that (k, r)-WL Version J identifies all colored groups in $\mathcal{C}$. For any $G \in \mathcal{C}$, Algorithm 1 correctly returns a canonical form for G.*

Proof. Let $G \in \mathcal{C}$ be our input group, and let $\kappa(G)$ be the result of Algorithm 1. We show that κ canonizes $\mathcal{C}$. By construction, the map $i \mapsto \psi(i)$ is an isomorphism of $G \cong \kappa(G)$.

Now let $H \in \mathcal{C}$ be a second group such that $G \cong H$. Let $g_1, \ldots, g_k \in G$ be the sequence of group elements added to Gens by Algorithm 1, and let $h_1, \ldots, h_k$ be the corresponding sequence for H. We show by induction on $0 \leq i \leq k$ that there is an isomorphism $\varphi : G \cong H$ that restricts to an isomorphism of $\langle g_1, \ldots, g_i \rangle \cong \langle h_1, \ldots, h_i \rangle$. The base case when $i = 0$ is precisely the assumption that $G \cong H$. Now fix $i \geq 0$ and let $\varphi : G \cong H$ such that $\varphi(g_j) = h_j$ for all $j \leq i - 1$. As φ is an isomorphism mapping $\varphi(g_j) = h_j$ for all $j \leq i - 1$, it follows that φ restricts to the isomorphism of $\langle g_1, \ldots, g_{i-1} \rangle \cong \langle h_1, \ldots, h_{i-1} \rangle$ induced by the map $\varphi(g_j) = h_j$ for all $j \leq i - 1$. So we have that $(G, \chi_{G,i}) \cong (H, \chi_{H,i})$.

As g_i, h_i were selected by the algorithm at line 12, we have that $\chi_{G,i}(g) = \chi_{H,i}(\varphi(g))$. As $(k + 1, r)$-WL determines orbits for all colored groups $G \in \mathcal{C}$, it follows that there is a color-preserving isomorphism $\varphi : (G, \chi_{G,i}) \to (H, \chi_{H,i})$. As g_j, h_j belong to their own color class for $j \leq i$, φ also has to map $\varphi(g_j) = h_j$. Necessarily, φ must also restrict to the isomorphism of $\langle g_1, \ldots, g_i \rangle \cong \langle h_1, \ldots, h_i \rangle$ induced by the map $g_j \mapsto h_j$ for $1 \leq j \leq i$. The result now follows by induction. □

Theorem 3.8.8. *Let $k \geq 2$ be a constant, and let $r := r(n)$ be a function, where n denotes the order of the input groups.*

(a) Let $\mathcal{C}$ be a family of groups. Suppose that $r \geq 2$ is a constant. If (k, r)-WL Version II (counting or count-free) identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of SAC *circuits of depth $O(\log^2 n)$ and size $O(r \cdot n^{O(k)})$.*

(b) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If $(k, r(n))$-WL Version II identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of TC *circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.*

(c) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If the count-free $(k, r(n))$-WL Version II identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of AC *circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.*

Proof. By Theorem 3.8.7, we have that Algorithm 1 correctly computes a canonical form for any group $G \in \mathcal{C}$. It remains to establish the complexity bounds. The key work lies in the `while` loop. At line 9, we compute $(k + 1, r)$-WL using the parallel WL implementation due to Grohe & Verbitsky [104], adapted for WL Version II. We first observe that the initial coloring of WL Version II is L-computable. Deciding whether two k-tuples of group elements are marked isomorphic is L-computable [185]. Using a logspace transducer, we can write down for all $\binom{2n^k}{2}$ pairs $\{\overline{u}, \overline{v}\}$ of k-tuples whether $\overline{u}, \overline{v}$ are marked isomorphic.

Each refinement step in the counting WL Version II is TC^0-computable, and each refinement step in the count-free WL Version II is AC^0-computable. We thus have the following.

- For (a), both the counting and count-free variants of $(k+1, O(1))$-WL Version II are L-computable.
- For (b), $(k+1, r)$-WL Version II can be implemented with a logspace uniform TC circuit of depth $O((r(n) + \log(n)) \log n)$ and size $O(r \cdot n^{O(k)})$.
- For (c), the count-free $(k+1, r)$-WL Version II can be implemented with a logspace uniform AC circuit of depth $O((r(n) + \log(n)) \log n)$ and size $O(r \cdot n^{O(k)})$.

We now observe that the remaining steps of the `while` loop are AC^0-computable. To see that Line 12 is AC^0-computable, we appeal to the characterization that $\mathsf{AC}^0 = \mathsf{FO}$ [161]. We may write down a first-order formula for the minimum element, and so finding the minimum color class is AC^0-computable. Furthermore, identifying the members of a given color class is AC^0-computable. Thus, computing the argmin is AC^0-computable.

Finally, we note that at line 12, we select $g_i \in G \setminus \langle \text{Gens} \rangle$. Thus, the size of $\langle \text{Gens} \rangle$ is at least doubled at each iteration. At the start of line 9 of the iteration of the `while` loop after g_{i+1} is added to Gens, each element $g \in \langle \text{Gens} \rangle$ has a unique color class under $(k+1, r)$-WL (in particular, this is handled by a single refinement step of WL Version II). So the number of iterations k of the `while` loop is at most $\log n + 1$. Thus, we have the following.

- For (a), we have a logspace uniform family of SAC circuits with depth $O(\log^2 n)$ and size $O(r \cdot n^{O(k)})$.
- For (b), we have a logspace uniform family of TC circuits with depth $O((r(n) + \log(n)) \log n)$ and size $O(r \cdot n^{O(k)})$.
- For (c), we have a logspace uniform family of AC circuits with depth $O((r(n) + \log(n)) \log n)$ and size $O(r \cdot n^{O(k)})$.

The result follows. □

For groups that are $O(1)$-generated, using a different strategy, it is possible to canonize such groups using only one call to Weisfeiler–Leman.

Proposition 3.8.9. *For groups that are $O(1)$-generated, we may compute canonical forms in L.*

Proof. Let $d := d(G)$. We run the count-free $(d, 1)$-WL Version II. Now for each color class $K \in \text{Im}(\chi)$ where there exists a d-tuple $(g_1, \ldots, g_d)$ such that $g_1, \ldots, g_d$ are all distinct (and in such case, the elements of any d-tuple in K are all distinct), we may use Tang's marked isomorphism procedure [185] to test whether the given k-tuple generates the group. To obtain a canonical form, we take the coloring $\overline{\chi_{(g_1,\ldots,g_d)}}$ obtained by individualizing $(g_1, \ldots, g_d)$ from the smallest color class under χ, where $G = \langle g_1, \ldots, g_d \rangle$. □

Remark 3.8.10. The strategy here is different, in that we don't need all colored d-generated groups to be identified by WL. A d-generated subgroup using more than d colors may be harder to identify.

The complexity results in Theorem 3.8.8 also hold if we use WL Version I for canonization. However, the analysis is slightly different. The key idea is that the logspace computations get pushed to computing $\langle \text{Gens} \rangle$ at line 11, rather than at the initial coloring as in WL Version II.

Theorem 3.8.11. *Let $k \geq 2$ be a constant, and let $r := r(n)$ be a function, where n denotes the order of the input groups.*

(a) Let $\mathcal{C}$ be a family of groups. Suppose that $r \geq 2$ is a constant. If (k, r)-WL Version I (counting or count-free) identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of SAC circuits of depth $O(\log^2 n)$ and size $O(r \cdot n^{O(k)})$.

(b) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If $(k, r(n))$-WL Version I identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of TC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.

(c) *Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If the count-free $(k, r(n))$-WL Version I identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of AC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.*

Proof. By Theorem 3.8.7, we have that Algorithm 1 correctly computes a canonical form for any group $G \in \mathcal{C}$. It remains to establish the complexity bounds. The key work lies in the `while` loop. At line 9, we compute $(k + 1, r)$-WL using the parallel WL implementation due to Grohe & Verbitsky [104] (which immediately applies to WL Version I; see [104, Remark 3.4]). In the case of the standard counting variant of WL Version I, each iteration can be implemented using a logspace uniform TC^0 circuit. In the case of count-free WL Version I, each iteration can be implemented using a logspace uniform AC^0 circuit. The remaining steps of the `while` loop, except for Line 11, are all AC^0-computable.

At line 11, we compute $\langle \text{Gens} \rangle$, which is L-computable using a membership test [38, 81] or constructive generation procedure [185]. From the proof of Theorem 3.8.8, selecting the minimum color class is AC^0-computable.

We thus have the following.

(1) Let $\mathcal{C}$ be a family of groups. If $(k, O(1))$-WL Version I (counting or count-free) identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of SAC circuits of depth $O(\log^2 n)$ and size $O(r \cdot n^{O(k)})$.

(2) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If $(k, r(n))$-WL Version I identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of TC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.

(3) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If the count-free $(k, r(n))$-WL Version I identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of AC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.

□

We now modify Algorithm 1 to use WL Version III- see Algorithm 2, as well as establish its correctness.

Theorem 3.8.12. *Let $k \geq 2$ be a constant, and let $r := r(n)$ be a function, where n denotes the order of the input groups. Let $\mathcal{C}$ be a class of groups such that (k, r)-WL Version III identifies all colored groups in $\mathcal{C}$. For any $G \in \mathcal{C}$, Algorithm 2 correctly returns a canonical form for G.*

Proof. Let $G \in \mathcal{C}$ be our input group, and let $\kappa(G)$ be the result of Algorithm 1. We show that κ canonizes $\mathcal{C}$. By construction, the map $i \mapsto \psi(i)$ is an isomorphism of $G \cong \kappa(G)$.

Now let $H \in \mathcal{C}$ be a second group such that $G \cong H$. Let $g_1, \ldots, g_k \in G$ be the sequence of vertices added to Gens by Algorithm 1, and let $h_1, \ldots, h_k$ be the corresponding sequence for H (we will show later that $k \leq \log n + 1$). We show by induction on $0 \leq i \leq k$ that there is an isomorphism $\varphi : G \cong H$ that restricts to an isomorphism of $\langle g_1, \ldots, g_i \rangle \cong \langle h_1, \ldots, h_i \rangle$. The base case when $i = 0$ is precisely the assumption that $G \cong H$. Now fix $i \geq 0$ and let $\varphi : G \cong H$ such that $\varphi(g_j) = h_j$ for all $j \leq i - 1$. As φ is an isomorphism mapping $\varphi(g_j) = h_j$ for all $j \leq i - 1$, it follows that φ restricts to the isomorphism of $\langle g_1, \ldots, g_{i-1} \rangle \cong \langle h_1, \ldots, h_{i-1} \rangle$ induced by the map $\varphi(g_j) = h_j$ for all $j \leq i - 1$. Then, using the fact that the map $G \mapsto \Gamma_G$ is a many-one reduction from GPI to GI, we have that $(\Gamma_G, \chi_{G,i}) \cong (\Gamma_H, \chi_{H,i})$. As g_i, h_i were selected by the algorithm at line 12, we have that $\chi_{G,i}(g) = \chi_{H,i}(\varphi(g))$. As $(k+1, r)$-WL determines orbits for all graphs Γ_G arising from groups $G \in \mathcal{C}$, it follows that there is a color-preserving isomorphism $\varphi : (\Gamma_G, \chi_{G,i}) \cong (\Gamma_H, \chi_{H,i})$ such that $\varphi(g_i) = h_i$. As g_j, h_j belong to their own color class for $j \leq i$, φ also has to map $\varphi(g_j) = h_j$. Necessarily, φ must also restrict to the isomorphism of $\langle g_1, \ldots, g_i \rangle \cong \langle h_1, \ldots, h_i \rangle$ induced by the map $g_j \mapsto h_j$ for $1 \leq j \leq i$. The result now follows by induction. □

Theorem 3.8.13. *Let $k \geq 2$ be a constant, and let $r := r(n)$ be a function, where n denotes the order of the input groups.*

(a) Let $\mathcal{C}$ be a family of groups. If $(k, O(1))$-WL Version III (counting or count-free) identifies

Algorithm 2 Canonization Algorithm for group class $\mathcal{C}$

```
Require: $G \in \mathcal{C}$ is identified by $(k, r)$-WL III
Ensure: $\kappa(G)$
$n := |G|$
$\Gamma_G := G(V, E)$                                        ▷ Graph created by WL III
$\Gamma_0 := \Gamma_G$
Gens $:= \emptyset$
$\psi := \emptyset$
$i := 0$

while $\langle \text{Gens} \rangle \neq G$ do
    Let $\chi_i$ be the coloring computed by $(k+1, r)$-WL III applied to $\Gamma_i$
    Define $\chi_{G,i+1}(v) = \chi_i^{\Gamma_i,k+1}(v, \ldots, v)$ for all $v \in G$
    For each $g \in G$ belonging to a color class of size 1, set $\psi(g) = \chi_{G,i+1}(g)$.
    Let $g_{i+1} \in \text{argmin}\{\chi_{G,i+1}(g) : g \in G \setminus \langle \text{Gens} \rangle, \text{The color class } \chi(g) \text{ has more than one element}\}$
    Gens := Gens $\cup \{g_{i+1}\}$.
    $\psi(g_{i+1}) = \chi_{G,i+1}(g_{i+1})$.

    Set $\Gamma_{i+1}$ to be the graph $\Gamma_G$, where the vertices are individualized according to $\psi$.
    Set $i := i + 1$
end while

return $\kappa(G) := ([n], \{(g, h, gh) : g, h \in G\}, g \mapsto \psi(g))$.
```

all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of SAC circuits of depth $O(\log^2 n)$ and size $O(r \cdot n^{O(k)})$.

(b) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If $(k, r(n))$-WL Version III identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of TC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.

(c) Let $\mathcal{C}$ be a family of groups, and suppose that $r(n) \in \omega(1)$. If the count-free $(k, r(n))$-WL Version III identifies all colored groups belonging to $\mathcal{C}$, then we may compute canonical forms using a logspace uniform family of AC circuits of depth $O((r(n) + \log n) \log n)$ and size $O(r \cdot n^{O(k)})$.

Proof. The proof is identical to Theorem 3.8.11, replacing WL Version I with WL Version III. We also note that constructing the graph Γ_G at line 3 is AC^0-computable; see- Remark 2.7.2. □

As only $\log n + 1$ calls to $(k+1)$-WL are required in the setting of groups, we obtain the following improvement to the serial runtime, compared to Theorem 3.8.2.

Corollary 3.8.14. *Let $\mathcal{C}$ be a class of groups.*

(a) If k-WL Versions I or II identify all colored groups in $\mathcal{C}$, then there exists a group canonization for $\mathcal{C}$ that can be computed in time $O(n^{k+2} \log^2 n)$.

(b) If k-WL Version III identifies all colored graphs in $\Gamma_{\mathcal{C}} = \{\Gamma_G : G \in \mathcal{C}\}$, then there exists a group canonization for $\mathcal{C}$ that can be computed in time $O(|G|^{2k+3} \log^2 |G|)$.

Remark 3.8.15. We note that for WL Version III, Theorem 3.8.2 yields a runtime of $O(n^{2k+4} \log n)$.

Chapter 4

Descriptive Complexity of Groups without Abelian Normal Subgroups

The work in this chapter is joint with Joshua A. Grochow, resulting in [91].

In this chapter, we investigate the descriptive complexity theory of groups without Abelian normal subgroups, which are a rather large and non-trivial class of groups. Motivation for groups without Abelian normal subgroups arises as follows. Every group G can be written as an extension of its solvable radical $\text{Rad}(G)$ by the quotient $G/\text{Rad}(G)$, which does not have Abelian normal subgroups. As such, the latter class of groups is quite natural, both group-theoretically and computationally. Computationally, it has been used in algorithms for general finite groups both in theory (e.g., [27, 28]) and in practice (e.g., [56]). Isomorphism testing in this family of groups can be solved efficiently in practice [56], and is known to be in P through a series of two papers [29, 30].

The extension structure is captured by the following characteristic filtration:

$$1 \leq \text{Rad}(G) \leq \text{Soc}^*(G) \leq \text{PKer}(G) \leq G,$$

which arises in the computational complexity community where it is known as the Babai–Beals filtration [27], as well as in the development of practical algorithms for computer algebra systems (c.f., [56]). We now explain the terms of this chain. Here, $\text{Rad}(G)$ is the *solvable radical*, which is the unique maximal solvable normal subgroup of G. The socle of a group, denoted $\text{Soc}(G)$, is the subgroup generated by all the minimal normal subgroups of G. $\text{Soc}^*(G)$ is the preimage of the socle $\text{Soc}(G/\text{Rad}(G))$ under the natural projection map $\pi : G \to G/\text{Rad}(G)$. To define PKer, we start by examining the action on $\text{Soc}(G/\text{Rad}(G)) \cong \text{Soc}^*(G)/\text{Rad}(G)$ that is induced by the action of G on $\text{Soc}^*(G)$ by conjugation. As $\text{Soc}^*(G)/\text{Rad}(G) \cong \text{Soc}(G/\text{Rad}(G))$ is the direct product of finite, non-Abelian simple groups $T_1, \ldots, T_k$, this action permutes the k simple factors, yielding a homomorphism $\varphi : G \to S_k$. The kernel of this action is denoted $\text{PKer}(G)$.

While the family of groups without Abelian normal subgroups is known to admit a polynomial-time isomorphism test, we have been unable to establish that even the classical $o(\log n)$-WL algorithm identifies this family. This, combined with the notion of *implicit pebbling* in WL Version III, led us to explore the power of the second Ehrenfeucht–Fraïssé bijective pebble game in Hella's [109] heirarchy. This is a Spoiler–Duplicator game in which Spoiler can place up to two pebbles each round. While it trivially solves graph isomorphism, it may be nontrivial for finite groups, and other ternary relational structures.

We note that in the pebble game corresponding to WL Version III, Spoiler may pebble two group elements at a time by placing a pebble on a multiplication gadget. However, while Duplicator must select bijections respecting the group elements, these bijections need not map group element vertices and non-group element vertices on the multiplication gadgets consistently. We contrast this with the second Ehrenfeucht–Fraïssé bijective pebble game in Hella's heirarchy, where Duplicator selects bijections on the group elements and Spoiler then pebbles at most 2 group elements in the given round. We do not have to worry about the inconsistencies that Duplicator can create in the pebble game corresponding to WL Version III.

We first provide a novel generalization of Weisfeiler–Leman (WL) coloring, which we call *2-ary* WL. We then show that the 2-ary WL is equivalent to the second Ehrenfeucht–Fraïssé bijective pebble game in Hella's heirarchy. Our main result is that, in the pebble game characterization, $O(1)$ pebbles and $O(1)$ rounds are sufficient to identify all groups without Abelian normal subgroups. In particular, we show that within the first few rounds, Spoiler can force Duplicator to select an isomorphism between two such groups at each subsequent round. By Hella's results [109, 110], this is equivalent to saying that these groups are identified by formulas in first-order logic with generalized 2-ary quantifiers, using only $O(1)$ variables and $O(1)$ quantifier depth.

For preliminaries on groups without Abelian normal subgroups, we refer the reader to Section 3.6.1.

4.1 Main Results

We initiate the study of Hella's 2-ary Ehrenfeucht–Fraïssé-style pebble game, in the setting of groups. Our main result is that this pebble game efficiently characterizes isomorphism in a class of groups for which isomorphism testing is known to be in P, but only by quite a nontrivial algorithm [30].

Theorem 4.1.1. *Let G be a group with no Abelian normal subgroups (a.k.a. Fitting-free or semisimple), and let H be arbitrary. If $G \not\cong H$, then Spoiler has a winning strategy the Ehrenfeucht–Fraïssé game at the second level of Hella's hierarchy, using 9 pebbles and $O(1)$ rounds.*

In proving Theorem 4.1.1, we show that with the use of only a few pebbles, Spoiler can effectively force Duplicator to select an isomorphism of G and H.

In Section 4.3 we also complete the picture by giving a Weisfeiler–Leman-style coloring procedure and showing that it corresponds precisely to Hella's q-ary pebble games and q-ary generalized Lindstrom quantifiers [148].

4.2 Pebbling Game

Hella [109, 110] exhibited a heirarchy of pebble games where, for $q \geq 1$, Spoiler could pebble a sequence of $1 \leq j \leq q$ elements $(v_1, \ldots, v_j) \mapsto (f(v_1), \ldots, f(v_j))$ in a single round. The case of $q = 1$ corresponds to the case of Weisfeiler–Leman. As remarked by Hella [110, p. 6, just before §4], the q-ary game immediately identifies all relational structures of arity $\leq q$. For example, the $q = 2$ game on graphs solves GI: for if two graphs X and Y are non-isomorphic, then any bijection $f : V(X) \to V(Y)$ that Duplicator selects must map an adjacent pair of vertices u, v in X to a non-adjacent pair $f(u), f(v)$ in Y or vice-versa. Spoiler immediately wins by pebbling $(u, v) \mapsto (f(u), f(v))$. However, as groups are ternary relational structures (the relation being

$\{(a, b, c) : a, b, c \in G, ab = c\}$), the $q = 2$ case can, at least in principle, be non-trivial on groups.

Brachter & Schweitzer [48] adapted Hella's [109, 110] pebble games in the $q = 1$ case to the setting of groups, obtaining three different versions. Their Version III involves reducing to graphs and playing the pebble game on graphs, so we don't consider it further here. Versions I and II are both played on the groups G and H directly. Both Versions I and II may be generalized to allow Spoiler to pebble up to q group elements at a single round, for some $q \geq 1$. Mimicking the proof above for $q = 2$ for graphs, we have that $q = 3$ is sufficient to solve GPI in a single round. The distinguishing power, however, of the $q = 2$ game for groups remains unclear, and is the main subject of this chapter. As we are interested in the round complexity, we introduce the following notation.

Definition 4.2.1 (Notation for pebbles, rounds, arity, and WL version)**.** Let $k \geq 2, r \geq 1$, $q \in [3]$, and $J \in \{I, II\}$. Denote (k, r)-WL_J^q to be the k-pebble, r-round, q-ary Version J pebble game.

We refer to q as the *arity* of the pebble game, as it corresponds to the arity of generalized quantifiers in a logic whose distinguishing power is equivalent to that of the game:

Remark 4.2.2 (Equivalence with logics with generalized 2-ary quantifiers)**.** Hella [109] describes the game (essentially the same as our description, but with no restriction on number of pebbles, and a transfinite number of rounds) for general q at the bottom of p. 245, for arbitrary relational structures. We restrict to the case of $q = 2$, a finite number of pebbles and rounds, and the (relational) language of groups. Hella proves that this game is equivalent to first-order logic with arbitrary q-ary equantifiers in [109, Theorem 2.5].

We frequently use this observation without mention.

Observation 4.2.3. In the 2-ary pebble game, we may assume that Duplicator selects bijections that preserve inverses.

Proof. For suppose not. First, it is clear that Duplicator must select bijections that preserve the identity. Let $f : G \to H$ be a bijection such that $f(g^{-1}) \neq f(g)^{-1}$. Spoiler pebbles $(g, g^{-1}) \mapsto (f(g), f(g^{-1}))$. Now $gg^{-1} = 1$, while $f(g)f(g^{-1}) \neq 1$. So Spoiler wins. □

4.3 Higher-arity Weisfeiler-Leman-style coloring corresponding to higher arity pebble games

Given a k-tuple $\overline{x} = (x_1, \ldots, x_k) \in G^k$, a pair of distinct indices $i, j \in [k]$, and a pair of group elements y, z, we define $\overline{x}_{(i,j)\leftarrow(y,z)}$ to be the k-tuple $\overline{x}'$ that agrees with $\overline{x}$ on all indices besides i, j, and with $x'_i = y, x'_j = z$. If $i = j$, we require $y = z$, and we denote this $\overline{x}_{i\leftarrow y}$.

Two graphs Γ_1, Γ_2, with edge-colorings $c_i \colon E(\Gamma_i) \to C$ to some color set C (for $i = 1, 2$) are *color isomorphic* if there is a graph isomorphism $\varphi \colon V(\Gamma_1) \to V(\Gamma_2)$ that also preserves colors, in the sense that $c_1((u, v)) = c_2((\varphi(u), \varphi(v))$ for all edges $(u, v) \in E(\Gamma_1)$.

Definition 4.3.1 (2-ary k-dimensional Weisfeiler-Leman coloring)**.** Let G, H be two groups of the same order, let $k \geq 1$. For all k-tuples $\overline{x}, \overline{y} \in G^k \cup H^k$:

- (Initial coloring, Version I) $\chi_0^{2,I}(\overline{x}) = \chi_0^{2,I}(\overline{y})$ iff $\overline{x}, \overline{y}$ are partially isomorphic.
- (Initial coloring, Version II) $\chi_0^{2,II}(\overline{x}) = \chi_0^{2,II}(\overline{y})$ iff $\overline{x}, \overline{y}$ have the same marked isomorphism type.
- (Color refinement) Given a coloring $\chi \colon G^k \cup H^k \to C$, the color refinement operator R defines a new coloring $R(\chi)$ as follows. For each k-tuple $\overline{x} \in G^k$ (resp., H^k), we define an edge-colored graph $\Gamma_{\overline{x},\chi,i,j}$. If $i = j$, it is the graph on vertex set $V(\Gamma_{\overline{x},\chi,i,i}) = G$ (resp., H) with all self-loops and no other edges, where the color of each self-loop (g, g) is $\chi(\overline{x}_{i\leftarrow g})$. If $i \neq j$, it is the complete directed graph with self-loops on vertex set G (resp., H), where the color of each edge (y, z) is $\chi(\overline{x}_{(i,j)\leftarrow(y,z)})$. For an edge-colored graph Γ, we use $[\Gamma]$ to denote its edge-colored isomorphism class. We then define

$$R(\chi)(\overline{x}) = (\chi(\overline{x}); [\Gamma_{\overline{x},\chi,1,1}], [\Gamma_{\overline{x},\chi,1,2}], \ldots, [\Gamma_{\overline{x},\chi,k,k-1}], [\Gamma_{\overline{x},\chi,k,k}]) \,.$$

  That is, the new color consists of the old color, as well as the tuple of $\binom{k+1}{2}$ edge-colored isomorphism types of the graphs $\Gamma_{\overline{x},\chi,i,j}$.

The refinement operator may be iterated: $R^t(\chi) := R(R^{t-1}(\chi))$, and we define the *stable refinement* of χ as $R^t(\chi)$ where the partition induced by $R^t(\chi)$ on $G^k \cup H^k$ is the same as that induced by $R^{t+1}(\chi)$. We denote the stable refinement by $R^\infty(\chi)$.

Finally, for $J \in \{I, II\}$ and all $r \geq 0$, we define $\chi^{2,J}_{r+1} = R(\chi^{2,J}_r)$, and $\chi^{2,J}_\infty := R^\infty(\chi^{2,J}_0)$.

Remark 4.3.2. Since it was one of our stumbling blocks in coming up with this generalized coloring, we clarify here how this indeed generalizes the usual 1-ary WL coloring procedure. In the 1-ary "oblivious" k-WL procedure (see [98, §5], equivalent to ordinary WL), the color of a k-tuple $\overline{x}$ is refined using its old color, together with a k-tuple of multisets

$$(\{\{\chi(x_{1\leftarrow y}) : y \in G\}\}, \{\{\chi(x_{2\leftarrow y}) : y \in G\}\}, \ldots, \{\{\chi(x_{k\leftarrow y}) : y \in G\}\}).$$

For each i, note that two multisets $\{\{\chi(x_{i\leftarrow y}) : y \in G\}\}$ and $\{\{\chi(x'_{i\leftarrow y}) : y \in G\}\}$ are equal iff the graphs $\Gamma_{\overline{x},\chi,i,i}$ and $\Gamma_{\overline{x}',\chi,i,i}$ are color-isomorphic. That is, edge-colored graphs with only self-loops and no other edges are essentially the same, up to isomorphism, as multisets. Our procedure generalizes this by also considering graphs with other edges, which (as we'll see in the proof of equivalence below) are used to encode the choice of 2 simultaneous pebbles by Spoiler in each round of the game.

Theorem 4.3.3. *Let G, H be two groups of order n, with $\overline{x} \in G^k, \overline{y} \in H^k$. Starting from the initial pebbling $x_i \mapsto y_i$ for all $i = 1, \ldots, k$, Spoiler has a winning strategy in the k-pebble, r-round, 2-ary Version J pebble game (for $J \in \{I, II\}$) iff $\chi^{2,J}_r(\overline{x}) \neq \chi^{2,J}_r(\overline{y})$.*

Proof. By induction on r. The base case, $r = 0$, is built into the definition: the initial colors of $\overline{x}, \overline{y}$ agree iff $x_i = x_j \Leftrightarrow y_i = y_j$ and in Version I, $x_i x_j = x_\ell \Leftrightarrow y_i y_j = y_\ell$, or in Version II the pebbled map is a marked isomorphism. But these are precisely the conditions for Spoiler to lose without making a move, so both directions are established for $r = 0$.

Now suppose $r > 0$, and the equivalence is established for $r - 1$. For ease of notation, throughout the proof we denote colorings $\chi^{2,J}$ by χ^J instead, and we denote the graphs $\Gamma_{\overline{x},\chi^J_r,i,j}$ by $\Gamma_{\overline{x},r,i,j}$ instead ($q = 2$ and the dependence on J still being implied).

($\Leftarrow$) Suppose $\chi^J_r(\overline{x}) \neq \chi^J_r(\overline{y})$. If $\chi^J_{r-1}(\overline{x}) \neq \chi^J_{r-1}(\overline{y})$, then Spoiler has a winning strategy in $r - 1$ rounds by the inductive hypothesis. Otherwise, we have $\chi^J_{r-1}(\overline{x}) = \chi^J_{r-1}(\overline{y})$. But since their colors differ at the r-th round, there must exist $i, j \in [k]$ such that $\Gamma_{\overline{x},r-1,i,j}$ is not color-isomorphic to $\Gamma_{\overline{y},r-1,i,j}$. Spoiler picks up the pebbles i and j. Now, no matter what bijection φ Duplicator chooses, φ is not a color isomorphism on these graphs, so there exists an edge $(x_i', x_j') \in E(\Gamma_{\overline{x},r-1,i,j})$ whose color differs from that of the edge $(\varphi(x_i'), \varphi(x_j')) \in E(\Gamma_{\overline{y},r-1,i,j})$. (Note: there is no concern that the latter is not an edge of $\Gamma_{\overline{y},r-1,i,j}$, for φ is automatically an uncolored graph isomorphism: when $i \neq j$ the graphs are complete, and when $i = j$ the graphs consist of n isolated vertices with self-loops, so any bijection works.) Spoiler places the i-th pebble on $x_i' \mapsto \varphi(x_i') =: y_i'$ and the j-th pebble on $x_j' \mapsto \varphi(x_j') =: y_j'$.

By the definition of the edge coloring, the fact that the edges (x_i', x_j') and (y_i', y_j') receive different colors in the $\Gamma_{\bullet,r-1,i,j}$ graphs means that $\chi_{r-1}^J(\overline{x}_{(i,j)\leftarrow(x_i',x_j')}) \neq \chi_{r-1}^J(\overline{y}_{(i,j)\leftarrow(y_i',y_j')})$. Now, by the inductive hypothesis applied to $\overline{x}_{(i,j)\leftarrow(x_i',x_j')}$ and $\overline{y}_{(i,j)\leftarrow(y_i',y_j')}$, Spoiler can win in at most $r-1$ additional rounds.

($\Rightarrow$) Suppose $\chi_r^J(\overline{x}) = \chi_r^J(\overline{y})$. We show that Duplicator has a strategy that does not lose through round r. On the first round, Spoiler picks up pebbles i and j (not necessarily distinct). Since $\chi_r^J(\overline{x}) = \chi_r^J(\overline{y})$, we have that $\Gamma_{\overline{x},r-1,i,j}$ is color-isomorphic to $\Gamma_{\overline{y},r-1,i,j}$, say by the isomorphism $\varphi\colon G = V(\Gamma_{\overline{x},r-1,i,j}) \to V(\Gamma_{\overline{y},r-1,i,j}) = H$. Duplicator uses the isomorphism φ as their chosen bijection. Suppose Spoiler places the pebbles on $x_i' \mapsto \varphi(x_i') =: y_i'$ and $x_j' \mapsto \varphi(x_j') =: y_j'$. First, we claim that Duplicator has not yet lost. For the fact that φ is a color isomorphism means that the colors of the edge $(x_i', x_j') \in E(\Gamma_{\overline{x},r-1,i,j})$ and $(y_i', y_j') \in E(\Gamma_{\overline{y},r-1,i,j})$ are the same. But these colors are precisely $\chi_{r-1}^J(\overline{x}_{(i,j)\leftarrow(x_i',x_j')})$ and $\chi_{r-1}^J(\overline{y}_{(i,j)\leftarrow(y_i',y_j')})$. Since χ_{r-1}^J refines χ_0^J, we also have $\chi_0^J(\overline{x}_{(i,j)\leftarrow(x_i',x_j')}) = \chi_0^J(\overline{y}_{(i,j)\leftarrow(y_i',y_j')})$; since the 0-th coloring precisely matches the condition for Duplicator not to lose, Duplicator has not yet lost.

It remains to show that Duplicator can continue not to lose for an additional $r-1$ rounds. But this now follows from the inductive hypothesis applied to $\overline{x}_{(i,j)\leftarrow(x_i',x_j')}$ and $\overline{y}_{(i,j)\leftarrow(y_i',y_j')}$, for we have already established that they receive the same color under χ_{r-1}^J. □

Corollary 4.3.4. *For two groups G, H of the same order and any $k \geq 1$, the following are equiv-*

alent:

(1) *The 2-ary k-pebble game does not distinguish two groups G, H*

(2) *The multisets of stable colors on G^k and H^k are the same, that is, $\{\{\chi_\infty^{2,J}(\overline{x}) : \overline{x} \in G^k\}\} = \{\{\chi_\infty^{2,J}(\overline{y}) : \overline{y} \in H^k\}\}$*

(3) $\chi_\infty^{2,J}((1_G, 1_G, \ldots, 1_G)) = \chi_\infty^{2,J}((1_H, \ldots, 1_H))$.

The analogous result holds in the $q = 1$ case, going back to [48].

Proof. (2 ⇒ 3) Note that the identity tuples $(1, \ldots, 1)$ are the unique tuples of their color, since they are the only tuples $(x_1, \ldots, x_k)$ such that $x_i^2 = x_i$ for all i (Version I), and the only tuples for which $\langle x_1, \ldots, x_k \rangle = 1$ (Version II). Thus, if the multisets of stable colors are equal, the colors of $(1_G, \ldots, 1_G)$ and $(1_H, \ldots, 1_H)$ must be equal.

(3 ⇒ 2) Suppose $\chi_\infty^{2,J}((1_G, \ldots, 1_G)) = \chi_\infty^{2,J}((1_H, \ldots, 1_H))$. Claim: for all $0 \leq \ell \leq k$,

$$\{\{\chi_\infty^{2,J}(x_1, \ldots, x_\ell, 1_G, \ldots, 1_G) : x_1, \ldots, x_\ell \in G\}\} = \{\{\chi_\infty^{2,J}(x_1, \ldots, x_\ell, 1_H, \ldots, 1_H) : x_1, \ldots, x_\ell \in H\}\}. \tag{$*$}$$

For notational simplicity, for $\overline{x} \in G^\ell$, define $\overline{x}^+$ as the k-tuple $(x_1, \ldots, x_\ell, 1_G, \ldots, 1_G)$, and similarly *mutatis mutandis* for $\overline{y} \in H^\ell$.

For $\ell = 0$, $(*)$ is precisely our assumption (3).

Suppose $(*)$ is true for some $0 \leq \ell < k$. We show that it remains true for $\min\{\ell + 2, k\}$. For notational simplicity, we assume $\ell + 2 \leq k$; the proof in the case $k = \ell + 1$ is similar.

By the inductive hypothesis, there is a bijection $\varphi_\ell \colon G^\ell \to H^\ell$ such that for all $\overline{x} \in G^\ell$,

$$\chi_\infty^{2,J}(\overline{x}^+) = \chi_\infty^{2,J}(\varphi_\ell(\overline{x})^+)$$

For each $\overline{x} \in G^\ell$, by the definition of the coloring, there is a color-isomorphism $\psi = \psi_{\overline{x}} \colon \Gamma_{\overline{x}^+, \infty, \ell+1, \ell+2} \to \Gamma_{\varphi(\overline{x})^+, \infty, \ell+1, \ell+2}$ (technically the subscripts here should be "$\infty - 1$," but because it is the stable coloring, the colored graphs are the same as what we have written). By the definition of the colors on the edges, this means that for all $x_{\ell+1}, x_{\ell+2} \in G$, we have that $\chi_\infty^J(x_1, \ldots, x_\ell, x_{\ell+1}, x_{\ell+2}, 1_G, \ldots, 1_G) =$

$\chi_\infty^J(\varphi_\ell(\overline{x}), \psi(x_{\ell+1}), \psi(x_{\ell+2}))$ (where here we have slightly abused our parentheses, but in a way that should be clear). Thus, we may extend φ_ℓ to a color-preserving bijection $\varphi_{\ell+2}$ by defining

$$\varphi_{\ell+2}(\overline{x}, x_{\ell+1}, x_{\ell+2}) = (\varphi_\ell(\overline{x}), \psi_{\overline{x}}(x_{\ell+1}), \psi_{\overline{x}}(x_{\ell+2})) \qquad \forall \overline{x} \in G^\ell \forall x_{\ell+1}, x_{\ell+2} \in G.$$

Thus establishing $(*)$ for $\ell + 2$ (assuming $\ell + 2 \leq k$). This establishes the claim for all $\ell \leq k$, and thus that (3) implies (2).

(3 $\Leftrightarrow$ 1) The 2-ary k-pebble game starting from empty initial configurations is equivalent, move-for-move, with the 2-ary k-pebble game starting from the configuration $(1_G, 1_G, \ldots, 1_G) \mapsto (1_H, 1_H, \ldots, 1_H)$. By Theorem 4.3.3, the 2-ary k-pebble game thus does not distinguish G from H iff $\chi_\infty^{2,J}((1_G, \ldots, 1_G)) = \chi_\infty^{2,J}((1_H, \ldots, 1_H))$. □

Remark 4.3.5. For arbitrary relational structures with relations of arity $a+1$, the a-order pebble game may still be nontrivial, as pointed out in Hella [110, p. 6, just before §4]. Our coloring procedure generalizes in the following way to this more general setting, and the proof of the equivalence between the coloring procedure and Hella's pebble game is the same as the above, *mutatis mutandis.* The main change is that for an a-th order pebble game, instead of just considering a graph on edges of size 1 (when $i = j$) or 2 (when $i \neq j$), we consider an a'-uniform directed hypergraph, where each hyperedge consists of a list of a' vertices, for all $1 \leq a' \leq a$. This gives a coloring equivalent of the logical and game characterizations provided by Hella; this trifecta is partly why we feel it is justified to call this a "higher-arity Weisfeiler–Leman" coloring procedure. It would be interesting if there were also an algebraic equivalent, similar to the cellular algebras of Weisfeiler and Leman [196] (see also Higman's coherent algebras [111]) in the case of (ordinary) WL, but we leave that open for future development.

4.4 Equivalence between 2-ary (k, r)-WL Versions I and II

In this section we show that, up to additive constants in the number of pebbles and rounds, 2-ary WL Versions I and II are equivalent in their distinguishing power. For two different WL versions W, W', we write $W \preceq W'$ to mean that if W distinguishes two groups G and H, then so does W'.

Theorem 4.4.1. *Let $k \geq 2, r \geq 1$. We have that:*

$$(k,r)\text{-}WL_I^2 \preceq (k,r)\text{-}WL_{II}^2 \preceq (k+2,r+1)\text{-}WL_I^2.$$

Remark 4.4.2. This is a tighter connection in terms of rounds than that known for ordinary (1-ary) WL. In the case of $q = 1$, the best known round-tradeoff between Versions I and II of the pebble game is one pebble [48] and an additive $O(\log n)$ rounds (see Theorem 2.8.4). While it is possible that this $O(\log n)$ tradeoff can be improved, we believe it is unlikely that only an extra round suffices in the 1-ary case.

Proof. For the first statement, note that if $\overline{x} \mapsto \overline{y}$ is a marked isomorphism, then it is a partial isomorphism as well. After that, each color refinement step in the two versions are the same, so at whatever round Version I distinguishes G from H, Version II will distinguish G from H at that round (or possibly sooner).

For the second statement, suppose that (k,r)-WL$_{II}^2$ distinguishes G from H. Let r_0 be the minimum number of rounds after which Spoiler has a winning strategy in this game. We show how Spoiler can win in the $(k+2, r_0+1)$-WL$_I^2$ game. For the first r_0 rounds, Spoiler makes exactly the same moves it would make in response to Duplicator's bijections in the Version II game. At the end of the r_0 round, Spoiler has pebbled a map $\overline{x} \mapsto \overline{y}$ of k-tuples that is not a marked isomorphism, but may still be a partial isomorphism. On the next round, Spoiler picks up the two pebbles $k+1, k+2$. Let φ be the bijection Duplicator chooses. Since the pebbled map is not a marked isomorphism, there is a word w such that $w(\varphi(x_1), \ldots, \varphi(x_k)) \neq \varphi(w(\overline{x}))$. Let w be such a word of minimal length ℓ, and write $w(\overline{x}) = x_{i_1}^{\pm 1} w'(\overline{x})$, where w' has length $\ell - 1$. Spoiler places the two pebbles on $w'(\overline{x}) \mapsto \varphi(w'(\overline{x}))$ and $w(\overline{x}) \mapsto \varphi(w(\overline{x}))$. Now, since pebbles $i_1, k+1, k+2$ are on $x_{i_1}, w'(\overline{x}), w(\overline{x})$, resp., and $x_{i_1}^{\pm 1} w'(\overline{x}) = w(\overline{x})$ but $\varphi(x_i)^{\pm 1} \varphi(w'(\overline{x})) = w(\varphi(\overline{x})) \neq \varphi(w(\overline{x})$, Spoiler wins the Version I game. □

4.5 Descriptive Complexity of Semisimple Groups

In this section, we show that the $(O(1), O(1))$-WL_{II}^2 pebble game can identify groups with no Abelian normal subgroups, also known as semisimple groups.

We show that the second Ehrenfeucht–Fraïssé in Hella's hierarchy can identify both $\text{Soc}(G)$ and the conjugation action when G is semisimple. We first show that this pebble game can identify whether a group is semisimple. Namely, if G is semisimple and H is not semisimple, then Spoiler can distinguish G from H. This is essentially the same proof as Proposition 3.6.8 in the setting of 1-ary (classical) WL, but with a tighter control on rounds in the 2-ary case.

Proposition 4.5.1. *Let G be a semisimple group of order n, and let H be an arbitrary group of order n. If H is not semisimple, then Spoiler can win in the $(4, 2)$-WL_{II}^2 game.*

Proof. Recall that a group is semisimple if and only if it contains no Abelian normal subgroups. As H is not semisimple, $\text{Soc}(H) = A \times T$, where A is the non-trivial direct product of elementary Abelian groups and T is a direct product of non-Abelian simple groups. We show that Spoiler can win using at most 3 pebbles on the board and 2 rounds. Let $f : G \to H$ be the bijection that Duplicator selects. Let $a \in A$. So $\text{ncl}_H(a) \leq A$. Let $b := f^{-1}(a) \in G$, and let $B := \text{ncl}_G(b)$. As G is semisimple, we have that B is not Abelian. Spoiler first pebbles $b \mapsto f(b) = a$.

So there exist $g_1, g_2 \in G$ such that $g_1 b g_1^{-1}$ and $g_2 b g_2^{-1}$ do not commute (for B is generated by $\{gbg^{-1} : g \in G\}$, and if they all commuted then B would be Abelian). Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Spoiler pebbles (g_1, g_2) and $(f'(g_1), f'(g_2))$. As $\text{ncl}(f(b)) \leq A$ is Abelian, $f'(g_1)f(b)f'(g_1)^{-1}$ and $f'(g_2)f(b)f'(g_2)^{-1}$ commute. Spoiler now wins. □

We now apply Lemma 3.6.5 to show that Duplicator must map the direct factors of $\text{Soc}(G)$ to isomorphic direct factors of $\text{Soc}(H)$.

Lemma 4.5.2. *Let G, H be finite groups of order n. Let $Fac(Soc(G))$ denote the set of simple direct factors of $Soc(G)$. Let $S \in Fac(Soc(G))$ be a non-Abelian simple group, with $S = \langle x, y \rangle$. If*

Duplicator selects a bijection $f\colon G \to H$ *such that:*

(a) $f(S) \not\cong \langle f(x), f(y)\rangle$*, then Spoiler can win in the* $(2,1)$*-*WL_{II}^2 *game; or*

(b) $f(S) \neq \langle f(x), f(y)\rangle$*, then Spoiler can win in the* $(4,2)$*-*WL_{II}^2 *pebble game.*

Note that the lemma does not require $f|_S\colon S \to f(S)$ to actually be an isomorphism, only that S and $f(S)$ are isomorphic. We also note that Lemma 3.6.9 only allows us to show that in the 1-ary case, the set of elements belonging to non-Abelian simple direct factors of $\text{Soc}(G)$ must map to the corresponding set in H. Lemma 4.5.2 establishes that in the 2-ary case, each such factor is preserved setwise.

Proof.

(a) If $S \not\cong \langle f(x), f(y)\rangle$, then Spoiler pebbles $(x, y) \mapsto (f(x), f(y))$ and immediately wins.

(b) Suppose that $S \cong \langle f(x), f(y)\rangle$. By part (a), we may assume that the map $(x, y) \mapsto (f(x), f(y))$ extends to an isomorphism; otherwise Spoiler immediately wins. So suppose that $f(S) \neq \langle f(x), f(y)\rangle$. Note that as $S \cong \langle f(x), f(y)\rangle$, we have that $|S| = |\langle f(x), f(y)\rangle|$. So $f(S) \subseteq \langle f(x), f(y)\rangle$ if and only if $f(S) = \langle f(x), f(y)\rangle$.

Now by the assumption that $f(S) \neq \langle f(x), f(y)\rangle$, there exists an element $b \in S$ such that $f(b) \not\in \langle f(x), f(y)\rangle$. Let $a \in S$ such that $a \neq 1$ and $f(a) \in \langle f(x), f(y)\rangle$. Spoiler pebbles $(a, b) \mapsto (f(a), f(b))$. Let $f' : G \to H$ be the bijection that Duplicator chooses on the next round. Let $g, h \in G$ such that $f'(g) = f(x)$ and $f'(h) = f(y)$. Spoiler now pebbles (g, h) and $(f(x), f(y))$. As $S \cong \langle f(x), f(y)\rangle$ by assumption, we have that $T := \langle g, h\rangle$ is isomorphic to S. Furthermore, as $f(a) \in \langle f(x), f(y)\rangle$, we may assume that $a \in \langle g, h\rangle$; otherwise, Spoiler immediately wins.

We now claim that $T \leq \text{Soc}(G)$. As $\text{Soc}(G)$ is normal in G, we have that $T \cap \text{Soc}(G)$ is normal in T. As T is simple and intersects with $\text{Soc}(G)$ non-trivially (namely, $a \in$

$T \cap \text{Soc}(G))$, it follows that that $T \leq \text{Soc}(G)$. Now as $S \trianglelefteq \text{Soc}(G)$ and $T \leq \text{Soc}(G)$, we have by similar argument (using the fact that $a \in S$) that $T = S$. So $S = \langle g, h \rangle$. Now $a, b \in S = \langle g, h \rangle$; however, $f(b) \notin \langle f(x), f(y) \rangle$. So Spoiler wins.

□

Proposition 4.5.3. *Let G be a semisimple group of order n, and let H be an arbitrary group of order n. Let $f : G \to H$ be the bijection Duplicator selects. If there exists $S \in Fac(Soc(G))$ such that $f(S) \notin Fac(Soc(H))$ or $f(S) \not\cong S$, then Spoiler may win in the $(4, 2)$-WL_{II}^2 pebble game.*

Proof. As G is semisimple, we have that $S = \langle x, y \rangle$ is non-Abelian. Let $f : G \to H$ be the bijection Duplicator selects. By Lemma 4.5.2, we may assume that $f(S) \cong S$ (though $f|_S$ need not be an isomorphism); otherwise, Spoiler can win with 2 pebbles and 1 round. Furthermore, we may assume that $f(S) = \langle f(x), f(y) \rangle$ setwise; otherwise, Spoiler can win with 4 pebbles and 2 rounds.

Suppose that $f(S)$ is not a direct factor of $\text{Soc}(H)$. Spoiler pebbles $(x, y) \mapsto (f(x), f(y))$. We now have the following cases.

- **Case 1:** Suppose that $f(S)$ is not contained in $\text{Soc}(H)$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. If $f'|_S$ is not an isomorphism, Spoiler immediately wins. As $S \triangleleft \text{Soc}(G)$, the normal closure $\text{ncl}(S)$ is minimal normal in G [124, Exercise 2.A.7]. As $f'(S)$ is not even contained in $\text{Soc}(H)$, we have by Lemma 3.6.5 that $\text{ncl}(f'(S))$ is not a direct product of non-Abelian simple groups, so $\text{ncl}(S) \not\cong \text{ncl}(f'(S))$. We note that $\text{ncl}(S) = \langle \{gSg^{-1} : g \in G\} \rangle$.

  As $\text{ncl}(f'(S))$ is not isomorphic to a direct power of S, there is some conjugate $T = gSg^{-1} \neq S$ such that $f'(T)$ does not commute with $f'(S)$, by Lemma 3.6.7. Yet since $S \trianglelefteq \text{Soc}(G)$, T and S do commute. Spoiler pebbles $g \mapsto f'(g)$. Since Spoiler has now pebbled x, y, g which generate $\langle S, T \rangle = S \times T \cong S \times S$ but the image is not isomorphic to $S \times S$, the map $(x, y, g) \mapsto (f(x), f(y), f'(g))$ does not extend to an isomorphism of $S \times T$. So Spoiler now wins, with a total of 3 pebbles and 2 rounds.

- **Case 2:** Suppose now that $f(S) \leq \mathrm{Soc}(H)$, but that $f(S)$ is not normal in $\mathrm{Soc}(H)$. By Lemma 4.5.2, we may assume without loss of generality that for each $T \in \mathrm{Fac}(\mathrm{Soc}(G))$, $f(T) \cong T$ (though $f|_T$ need not be an isomorphism) and, by the previous case, that $f(T) \leq \mathrm{Soc}(H)$. As $f(S)$ is not normal in $\mathrm{Soc}(H)$, there exists $T = \langle a, b \rangle \in \mathrm{Fac}(\mathrm{Soc}(H))$ such that T does not normalize $f(S)$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Again, we may assume $f'|_S$ is an isomorphism, or Spoiler wins with 2 additional pebbles and 1 additional round. Let $g = (f')^{-1}(a) \in \mathrm{Soc}(G)$ and $h = (f')^{-1}(b) \in \mathrm{Soc}(G)$. As S is normal in $\mathrm{Soc}(G)$, we note that $gSg^{-1} = S$ and $hSh^{-1} = S$. Spoiler pebbles $(g, h) \mapsto (a, b)$. The map $(x, y, g, h) \mapsto (f(x), f(y), a, b)$ does not extend to an isomorphism of $\langle x, y, g, h \rangle$ and $\langle f(x), f(y), a, b \rangle$, since g, h normalize $S = \langle x, y \rangle$ but a, b cannot both normalize $f(S) = \langle f(x), f(y) \rangle$ (since T does not). Spoiler now wins.

The result follows. □

Lemma 4.5.4. *Let G, H be groups of order n, let S be a nonabelian simple group in Fac(Soc(G)). Let $f, f' \colon G \to H$ be two bijections selected by Duplicator at two different rounds. If $f(S) \cap f'(S) \neq 1$, then $f(S) = f'(S)$, or Spoiler can win in the $(4,2)$-WL_{II}^2 pebble game.*

Proof. By Proposition 4.5.3, both $f(S)$ and $f'(S)$ must be simple normal subgroups of $\mathrm{Soc}(H)$ (or Spoiler wins with 4 pebbles and 2 rounds). Since they intersect nontrivially, but distinct simple normal subgroups of $\mathrm{Soc}(H)$ intersect trivially, the two must be equal. □

We next introduce the notion of weight.

Definition 4.5.5. Let $\mathrm{Soc}(G) = S_1 \times \cdots \times S_k$ where each S_i is a simple normal subgroup of $\mathrm{Soc}(G)$. For any $s \in \mathrm{Soc}(G)$, write $s = s_1 s_2 \cdots s_k$ where each $s_i \in S_i$, and define the *weight* of s, denoted $\mathrm{wt}(s)$, as the number of i such that $s_i \neq 1$.

Note that the definition of weight is well-defined since the S_i are the unique subsets of $\mathrm{Soc}(G)$ that are simple normal subgroups of $\mathrm{Soc}(G)$, so the decomposition $s = s_1 s_2 \ldots s_k$ is unique up to the order of the factors. (The following is essentially a particular instance of the "rank lemma"

Lemma 3.4.4, but as we are now in the setting of 2-ary WL we give the full proof, which also has tighter bounds.)

Lemma 4.5.6 (Weight Lemma)**.** *Let G, H be semisimple groups of order n. If Duplicator selects a bijection $f \colon G \to H$ that does not map Soc(G) bijectively to Soc(H), or does not preserve the weight of every element in Soc(G), then Spoiler can win in the $(4,3)$-WL_{II}^2 game.*

Proof. First, we show that f must send $\text{Soc}(G)$ into $\text{Soc}(H)$. For if not, then there is some $s \in \text{Soc}(G)$ with $f(s) \notin \text{Soc}(H)$. Among such s, choose one of minimal weight w; note that $w \geq 2$ since the weight-1 elements must get mapped into $\text{Soc}(H)$ by Proposition 4.5.3. Write $\text{Soc}(G) = S_1 \times \cdots \times S_k$ where the S_i are the simple normal subgroups of $\text{Soc}(G)$ and the first w of them are the support of s, that is, $s \in S_1 \times \cdots \times S_w$. Write $s = s_1 s_2 \cdots s_w$ with $s_i \in S_i$ and all $s_i \neq 1$. Since s was a minimum-weight element that gets mapped outside of $\text{Soc}(H)$, we have that $f(s_1)$ and $f(s_2 \cdots s_w)$ are both in $\text{Soc}(H)$, so their product is as well, and thus we have $f(s) = f(s_1 s_2 \cdots s_w) \neq f(s_1) f(s_2 \cdots s_w)$. Let $s' = s_2 s_3 \cdots s_w$. Spoiler now pebbles the pair $(s, s') \mapsto (f(s), f(s'))$.

On the next round, Duplicator picks a new bijection f' with $f'(s) = f(s) \notin \text{Soc}(H)$, $f'(s') = f(s') \in \text{Soc}(H)$, and, by Proposition 4.5.3, $f'(s_1) \in \text{Soc}(H)$ as well. Thus the product $f'(s_1) f'(s') = f'(s_1) f'(s_2 s_3 \cdots s_w)$ is also in $\text{Soc}(H)$, so we again have $f'(s) = f'(s_1 s_2 \cdots s_w) \neq f'(s_1) f'(s_2 s_3 \cdots s_w)$. Spoiler now pebbles s_1 and wins, in a total of 3 pebbles and 2 rounds.

Now we show that f must preserve weight. The identity is the unique element of weight 0, which must be sent to the identity because it is the unique element satisfying $e^2 = e$. The case of weight 1 is precisely Proposition 4.5.3.

Now, suppose that f does not preserve weight. Among the elements $s \in \text{Soc}(G)$ such that $\text{wt}(f(s)) \neq \text{wt}(s)$, choose one of minimal weight w (which must be ≥ 2, since we have already shown the weight-1 case). Without loss of generality, write $\text{Soc}(G) = S_1 \times \cdots \times S_k$ where the S_i are the simple normal subgroups of $\text{Soc}(G)$ and $s \in S_1 \times S_2 \times \cdots \times S_w$. Write $s = s_1 s_2 \cdots s_w$ with all $s_i \neq 1$. Since s is a smallest-weight element whose weight is not preserved, $f(s_2 s_3 \cdots s_w)$ must have weight precisely $w-1$. Now $f(s_1)$ has weight 1, so $f(s_1)f(s_2s_3\cdots s_w)$ has weight either $w-2$, $w-1$ or w. Since f is a bijection from the elements of weight $< w$ in $\text{Soc}(G)$ to the elements of weight $< w$ in $\text{Soc}(H)$, and $\text{wt}(f(s)) \neq w$, we must have $\text{wt}(f(s)) > w$ since f is a bijection. Thus $f(s_1s_2\cdots s_w) = f(s) \neq f(s_1)f(s_2s_3\cdots s_w)$, since the two elements have different weights. Now Spoiler pebbles the pair $(s, s_2s_3\cdots s_w) \mapsto (f(s), f(s_2\cdots s_w))$.

At the next round, Duplicator selects another bijection f'. We still have $\text{wt}(f'(s)) = \text{wt}(f(s)) > w$, $\text{wt}(f'(s_2\cdots s_w)) = \text{wt}(f(s_2\cdots s_w)) = w-1$, and (by Proposition 4.5.3) $\text{wt}(f'(s_1)) = 1$. Thus, again, we have $\text{wt}(f'(s_1)f'(s_2s_3\cdots s_w)) \in \{w, w-1, w-2\}$, so we must have $f'(s_1s_2\cdots s_w) = f'(s) \neq f'(s_1)f'(s_2\cdots s_w)$. Spoiler now pebbles s_1 and wins with 4 pebbles and 3 rounds. □

Lemma 4.5.7. *Let G and H be semisimple groups with isomorphic socles. Let $S_1, S_2 \in Fac(Soc(G))$ be distinct. Let $f : G \to H$ be the bijection that Duplicator selects. If there exist $x_i \in S_i$ such that $f(x_1x_2) \neq f(x_1)f(x_2)$, then Spoiler can win in the $(4,3)$-WL_{II}^2 pebble game.*

Proof. By Lemma 4.5.6, we may assume that $\text{wt}(s) = \text{wt}(f(s))$ for all $s \in \text{Soc}(G)$; otherwise, Spoiler wins with at most 4 pebbles and 3 rounds. As $f(x_1x_2)$ has weight 2, $f(x_1x_2)$ belongs to the direct product of two simple factors in $\text{Fac}(\text{Soc}(H))$, so it can be written $f(x_1x_2) = y_1y_2$ with each y_i in distinct simple factors in $\text{Fac}(\text{Soc}(H))$. Without loss of generality suppose that $y_1 \neq f(x_1)$. Spoiler pebbles $(x_1, x_1x_2) \mapsto (f(x_1), f(x_1x_2))$. Now $\text{wt}(x_1^{-1} \cdot x_1x_2) = 1$, while $\text{wt}(f(x_1)^{-1} \cdot f(x_1x_2)) \geq 2$. (Note that we cannot quite yet directly apply Lemma 4.5.6, because we have not yet identified a single element x such that $\text{wt}(x) \neq \text{wt}(f(x))$.)

On the next round, Duplicator selects another bijection f'. Spoiler now pebbles $x_2 \mapsto f'(x_2)$. Because $\text{wt}(x_1^{-1} \cdot x_1x_2) = 1$ but $\text{wt}(f(x_1)^{-1}f(x_1x_2)) \geq 2$, and f' preserves weight by Lemma 4.5.6, we have $f'(x_2) \neq f'(x_1)^{-1}f'(x_1x_2)$. Thus, the pebbled map $(x_1, x_2, x_1x_2) \mapsto (f'(x_1), f(x_2), f(x_1x_2))$ does not extend to an isomorphism, and so Spoiler wins with 3 pebbles on the board and 2 rounds. □

Recall from Section 3.6.1 that if G is semisimple, then $G \leq \text{Aut}(\text{Soc}(G))$. Now each minimal normal subgroup $N \trianglelefteq G$ is of the form $N = S^k$, where S is a non-Abelian simple group. So

$\mathrm{Aut}(N) = \mathrm{Aut}(S) \wr \mathrm{Sym}(k)$. In particular,

$$G \leq \prod_{\substack{N \trianglelefteq G \\ N \text{ is minimal normal}}} \mathrm{Aut}(N).$$

So if $g \in G$, then the conjugation action of g on $\mathrm{Soc}(G)$ acts by (i) automorphism on each simple direct factor of $\mathrm{Soc}(G)$, and (ii) by permuting the direct factors of $\mathrm{Soc}(G)$. Provided generators of the direct factors of the socle are pebbled, Spoiler can detect inconsistencies of the automorphism action. However, doing so directly would be too expensive as there could be $\Theta(\log |G|)$ generators, so we employ a more subtle approach with a similar outcome. By Lemma 4.5.6, Duplicator must select bijections $f : G \to H$ that preserve weight. That is, if $s \in \mathrm{Soc}(G)$, then $\mathrm{wt}(s) = \mathrm{wt}(f(s))$. We use Lemma 4.5.6 in tandem with the fact that the direct factors of the socle commute to effectively pebble the set of all the generators at once. Namely, suppose that $\mathrm{Fac}(\mathrm{Soc}(G)) = \{S_1, \ldots, S_k\}$, where $S_i = \langle x_i, y_i \rangle$. Let $x := x_1 \cdots x_k$ and $y := y_1 \cdots y_k$. We will show that it suffices for Spoiler to pebble (x, y) rather than individually pebbling generators for each S_i (this will still allow the factors to be permuted, but that is all).

Lemma 4.5.8. *Let G and H be semisimple groups with isomorphic socles, with $Fac(Soc(G)) = \{S_1, \ldots, S_m\}$, with $S_i = \langle x_i, y_i \rangle$. Let $f : G \to H$ be the bijection that Duplicator selects, and suppose that (i) for all i, $f(S_i) \cong S_i$ (though $f|_{S_i}$ need not be an isomorphism) and $f(S_i) \in Fac(Soc(H))$, (ii) for every $s \in Soc(G)$, $wt(s) = wt(f(s))$, and (iii) for all i, $f(S_i) = \langle f(x), f(y) \rangle$.*

Now suppose that Spoiler pebbles $(x_1 \cdots x_m, y_1 \cdots y_m) \mapsto (f(x_1 \cdots x_m), f(y_1 \cdots y_m))$. As f preserves weight, we may write $f(x_1 \cdots x_m) = h_1 \cdots h_m$ and $f(y_1 \cdots y_m) = z_1 \cdots z_m$ with $h_i, z_i \in f(S_i)$ for all i.

Let $f' : G \to H$ be the bijection that Duplicator selects at any subsequent round in which the pebble used above has not moved. If any of the following hold, then Spoiler can win in the WL_{II}^2 pebble game with 5 additional pebbles and 5 additional rounds:

(a) f' does not satisfy conditions (i)–(iii),

(b) there exists an $i \in [m]$ such that $f'(x_i) \notin \{h_1, \ldots, h_m\}$ or $f'(y_i) \notin \{z_1, \ldots, z_m\}$

(c) $f'|_{S_i}$ *is not an isomorphism*

(d) there exists $g \in G$ *and* $i \in [m]$ *such that* $gS_ig^{-1} = S_i$ *and for some* $x \in S_i$*,* $f'(gxg^{-1}) \neq f'(g)f'(x)f'(g)^{-1}$

Proof. We prove each part in turn:

(a) If f' does not satisfy (i)–(iii) then Spoiler can win with 4 additional pebbles and 3 additional rounds by, respectively, Proposition 4.5.3, Lemma 4.5.6, and Lemma 4.5.2.

(b) Suppose there exists an $i \in [m]$ such that for all $j \in [m]$, $f'(x_i) \neq h_j$. Spoiler pebbles $(x_i, x_1 \cdots x_m x_i^{-1}) \mapsto (f'(x_i), f'(x_1 \cdots x_m x_i^{-1}))$. Now $\text{wt}(x_1 \cdots x_m \cdot x_i^{-1}) = m - 1$, while $\text{wt}(f(x_1 \cdots x_m) \cdot f'(x_i)^{-1}) = m$, since $f'(x_i) \notin \{h_1, \ldots, h_m\}$ and $\text{wt}(f'(x_i)) = 1$ by (ii). Since f' preserves weight, $f'(x_1 \cdots x_m)f'(x_i)^{-1} \neq f'(x_1 \cdots x_m x_i^{-1})$. As all three of these elements have been pebbled, Spoiler wins. Similarly if instead some y_i has $f'(y_i) \notin \{z_1, \ldots, z_m\}$. In total, Spoiler used at most 4 additional pebbles and 3 additional rounds.

(c) Suppose that for some $i \in [k]$, $f'|_{S_i}$ is not an isomorphism. Then there is some word $w(x, y)$ such that $f'(w(x_i, y_i)) \neq w(f'(x_i), f'(y_i))$. Spoiler pebbles $g = w(x_i, y_i)$. Since we may assume $f'(1) = 1$, we have that $g \neq 1$. Let $f'' : G \to H$ be the bijection that Duplicator selects at the next round. Since $f'(S_i)$ and $f''(S_i)$ intersect at $f''(g) \neq 1$, by Lemma 4.5.4, we have $f'(S_i) = f''(S_i)$ (otherwise, Spoiler wins with 4 additional pebbles and 3 additional rounds).

By (b), we have that $f'(x_i) = h_j$ for some j (otherwise, Spoiler wins with 4 additional pebbles and 3 additional rounds). Since $f''(x_i) \in f''(S_i) = f'(S_i)$, and by (b) we must have $f''(x_i) \in \{h_1, \ldots, h_m\}$, only one of which is in $f'(S_i)$, we must have $f''(x_i) = h_j$ as well. Similarly, $f''(y_i) = f'(y_i) = z_j$. Spoiler now pebbles $(x_i, y_i) \mapsto (h_j, z_j)$. The pebbled map $(x_i, y_i, g) \mapsto (f'(x_i), f'(y_i), f'(g))$ does not extend to an isomorphism, and so Spoiler wins with at most 3 additional pebbles and 3 additional rounds (for a total of 4 additional pebbles and 4 rounds).

(d) By (c), we may assume that $f'|_{S_i}$ is an isomorphism. In this case, Spoiler pebbles $(g, gxg^{-1}) \mapsto (f'(g), f'(gxg^{-1}))$. Let $f'' : G \to H$ be the bijection that Duplicator selects on the next round. As we pebbled $gxg^{-1} \neq 1$, we have that $f''(S_i)$ and $f'(S_i)$ overlap at $f'(gxg^{-1})$, so we have $f''(S_i) = f'(S_i)$ by Lemma 4.5.4 (otherwise, Spoiler wins with 4 additional pebbles and 3 additional rounds).

As in the previous part, by (b), we have that $f''(x_i) = f'(x_i) = h_j$ and $f''(y_i) = f'(y_i) = z_j$ (or Spoiler wins with 4 additional pebbles and 4 additional rounds). Spoiler now pebbles $(x_i, y_i) \mapsto (h_j, z_j)$. Now, as $x \in \langle x_i, y_i \rangle$ there is a word w such that $x = w(x_i, y_i)$. Any isomorphism $\overline{f}$ extending our pebbled map $(x_i, y_i, g, gxg^{-1}) \mapsto (h_j, z_j, f'(g), f'(gxg^{-1}))$ must thus have $\overline{f}(x) = w(h_j, z_j) = f'(x)$, and thus the pebbled map does not extend to an isomorphism. Thus, Spoiler wins with at most 4 additional pebbles and 4 additional rounds (for a total of 5 additional pebbles and 5 additional rounds).

□

Lemma 4.5.8 provides enough to establish that Spoiler can force Duplicator to select at each round a bijection that restricts to an isomorphism on the socles.

Proposition 4.5.9. *(Same assumptions as Lemma 4.5.8.) Let G and H be semisimple groups with isomorphic socles, with Fac(Soc(G)) $= \{S_1, \ldots, S_m\}$, with $S_i = \langle x_i, y_i \rangle$. Let $f_0 : G \to H$ be the bijection that Duplicator selects, and suppose that (i) for all i, $f_0(S_i) \cong S_i$ (though $f_0|_{S_i}$ need not be an isomorphism) and $f_0(S_i) \in$ Fac(Soc(H)), (ii) for every $s \in$ Soc(G), wt(s) $=$ wt($f_0(s)$), and (iii) for all i, $f_0(S_i) = \langle f_0(x), f_0(y) \rangle$. Now suppose that Spoiler pebbles $(x_1 \cdots x_m, y_1 \cdots y_m) \mapsto (f_0(x_1 \cdots x_m), f_0(y_1 \cdots y_m))$.*

Let $f' : G \to H$ be the bijection that Duplicator selects at any subsequent round in which the pebble used above has not moved. Then $f'|_{Soc(G)} :$ Soc(G) $\to$ Soc(H) must be an isomorphism, or Spoiler can win in 4 more rounds using at most 6 more pebbles (for a total of 7 pebbles and 5 rounds) in the WL_{II}^2 pebble game.

Proof. By Lemma 4.5.6 f must map $\mathrm{Soc}(G)$ bijectively to $\mathrm{Soc}(H)$ and must preserve weights (otherwise, Spoiler wins with 4 pebbles and 3 rounds). Suppose that $f|_{\mathrm{Soc}(G)} \colon \mathrm{Soc}(G) \to \mathrm{Soc}(H)$ is not an isomorphism. The only way this is possible is if $f|_{\mathrm{Soc}(G)}$ is not a homomorphism. So there exist $s = s_1 s_2 \cdots s_k \in \mathrm{Soc}(G)$ and $s' = s'_1 s'_2 \cdots s'_k \in \mathrm{Soc}(G)$ such that

$$f(s_1 s_2 \cdots s_k) f(s'_1 s'_2 \cdots s'_k) \neq f(s_1 s'_1 s_2 s'_2 \cdots s_k s'_k).$$

We claim that this implies that there are $t_i \in S_i$ for all $i = 1, \ldots, k$ such that $f(t_1 t_2 \cdots t_k) \neq f(t_1) f(t_2) \cdots f(t_k)$. For suppose otherwise. Then the LHS of the preceding displayed equation is $f(s_1) f(s_2) \cdots f(s_k) f(s'_1) f(s'_2) \cdots f(s'_k)$, while the RHS is $f(s_1 s'_1) f(s_2 s'_2) \cdots f(s_k s'_k)$. But, by Lemma 4.5.8, f is an isomorphism on each simple normal subgroup of $\mathrm{Soc}(G)$ (otherwise, Spoiler wins with 4 additional pebbles and 5 additional rounds), so the latter is $f(s_1) f(s'_1) f(s_2) f(s'_2) \cdots f(s_k) f(s'_k)$. Furthermore, since the simple normal subgroups of $\mathrm{Soc}(G)$ commute with one another, their images must commute in $\mathrm{Soc}(H)$ (otherwise Spoiler can pebble two elements that commute in $\mathrm{Soc}(G)$ but whose images don't commute in $\mathrm{Soc}(H)$ and win), the LHS is equal to the same, a contradiction. Thus we have some $t_i \in S_i$ for $i = 1, \ldots, k$ such that $f(t_1 t_2 \cdots t_k) \neq f(t_1) f(t_2) \cdots f(t_k)$.

Suppose $t = t_1 \cdots t_k$ has weight w, and without loss of generality, re-index the S_i and t_i so that $t \in S_1 \times \cdots \times S_w$ with $t = t_1 t_2 \cdots t_w$. We still have $f(t_1 \cdots t_w) \neq f(t_1) \cdots f(t_w)$. If $f(t)$ is not in $f(S_1) \times f(S_{i_2}) \times \cdots \times f(S_{i_w})$ for any $(w-1)$ tuple of distinct indices $i_2, i_3, \ldots, i_w$, then Spoiler pebbles t and t_1. On the next round, Duplicator picks another f' with $f'(t_1) = f(t_1) \in S_1$ and $f'(t) = f(t) \notin f(S_1) \times f(S_{i_2}) \times \cdots \times f(S_{i_w})$ for any $(w-1)$ tuple of distinct indices. So $f'(t)$ is not the product of something in $f(S_1)$ with any element of weight $w-1$. But by Lemma 4.5.6, $f(t_2 t_3 \cdots t_w)$ must have weight $w-1$ (or Spoiler wins with 4 additional pebbles and 3 additional rounds), so we have $f'(t) \neq f'(t_1) f'(t_2 t_3 \cdots t_w)$. Spoiler may now pebble $t_2 t_3 \cdots t_w$ and win. Thus, $f(t)$ must be the product of something in $f(S_1)$ with some element of weight $w-1$. So there is some $w-1$-tuple of distinct indices $i_2, \ldots, i_w$ such that $f(t) \in f(S_1) \times f(S_{i_2}) \times \cdots \times f(S_{i_w})$.

Now repeat the same argument for any $j \in \{1, \ldots, w\}$ taking the place of 1. We thus find that for any such j, $f(t)$ must be in the direct product of $f(S_j)$ and $w-1$-many other $f(S_i)$'s.

Thus $f(t)$ must be in $f(S_1) \times f(S_2) \times \cdots \times f(S_w)$, since it has weight exactly w. So we have $f(t) = f(t_1')f(t_2') \cdots f(t_w')$ for some $t_i' \in S_i$. But we also have $f(t) \neq f(t_1)f(t_2) \cdots f(t_w)$. Since these decompositions are both unique, there must be some i such that $t_i \neq t_i'$. Without loss of generality (by re-indexing if needed), for simplicity of notation we may suppose that $i = 1$. Spoiler pebbles $(t, t_1) \mapsto (f(t), f(t_1))$. On the next round, Duplicator selects a new bijection f' with $f'(t) = f(t)$ and $f'(t_1) = f(t_1)$. Since S_1 is the unique simple normal subgroup of $\text{Soc}(G)$ containing t_1, and similarly $f(S_1)$ is the unique simple normal subgroup of $\text{Soc}(H)$ containing $f(t_1)$, by Lemma 4.5.4 we have $f'(S_1) = f(S_1)$ (or Spoiler wins with 4 additional pebble and 2 additional round).

Finally, we claim that $f'(t_1t_2 \cdots t_w) \neq f'(t_1)f'(t_2t_3 \cdots t_w)$. Suppose otherwise. Then by Lemma 4.5.6, $\text{wt}(f'(t)) = \text{wt}(f(t)) = w$, and $\text{wt}(f'(t_2t_3 \cdots t_w)) = w - 1$ (or Spoiler wins with 4 additional pebbles and 3 additional rounds). Now, if $f'(t_2t_3 \cdots t_w)$ has a non-identity component in the $f(S_1)$ factor of $\text{Soc}(H)$, then there must be some other $i \in \{2, \ldots, w\}$ such that $f'(t_2t_3 \cdots t_w)$ has trivial projection onto $f(S_i)$. Note that $f'(t_1) = f(t_1) \in f(S_1)$ also has trivial projection onto $f(S_i)$. But then this would contradict the fact that $\text{wt}(f'(t)) = w$. So the projection of $f'(t_2t_3 \cdots t_w)$ onto $f(S_1)$ (as a quotient of $\text{Soc}(H)$) must be trivial. Thus, the component of $f'(t) = f(t)$ in $f(S_1)$ is $f(t_1')$, but the component of $f'(t_1)f'(t_2 \cdots t_w)$ in $f(S_1)$ is precisely $f'(t_1) = f(t_1) \neq f(t_1')$. Thus $f'(t) \neq f'(t_1)f'(t_2 \cdots t_w)$ as claimed. As t, t_1 are already pebbled, Spoiler now pebbles $t_2t_3 \cdots t_w$, and wins. In total, Spoiler used at most 6 additional pebbles and 4 additional rounds. □

Remark 4.5.10. Brachter & Schweitzer [49, Lemma 5.22] previously showed that (1-ary) Weisfeiler–Leman can decide whether two groups have isomorphic socles. However, their results did not solve the search problem; that is, they did not show Duplicator must select bijections that restrict to an isomorphism on the socle even in the case for semisimple groups. This contrasts with Lemma 4.5.9, where we show that 2-ary WL effectively solves the search problem. This is an important ingredient in our proof that the $(7, O(1))$-WL_{II}^2 pebble game solves isomorphism for semisimple groups.

We obtain as a corollary of Lemma 4.5.8 and Lemma 4.5.9 that if G and H are semisimple, then Duplicator must select bijections that restrict to isomorphisms of $\text{PKer}(G)$ and $\text{PKer}(H)$.

Corollary 4.5.11. *Let G and H be semisimple groups of order n. Let $Fac(Soc(G)) := \{S_1, \ldots, S_m\}$, and suppose that $S_i = \langle x_i, y_i \rangle$. Let $x := x_1 \cdots x_m$ and $y := y_1 \cdots y_m$. and Let $f : G \to H$ be the bijection that Duplicator selects. Spoiler begins by pebbling $(x, y) \mapsto (f(x), f(y))$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. If $f'|_{\text{PKer}(G)} : \text{PKer}(G) \to \text{PKer}(H)$ is not an isomorphism, then Spoiler can win with 5 additional pebbles and 5 additional rounds in the WL_{II}^2 pebble game.*

Proof. We have the following cases.

- **Case 1:** Suppose that $f'(\text{PKer}(G)) \neq \text{PKer}(H)$. Then without loss of generality, there exists $g \in \text{PKer}(G)$ such that $f'(g) \notin \text{PKer}(H)$. Let $i \in [m]$ such that $f'(g) \cdot f'(S_i) \cdot f'(g)^{-1} \neq f'(S_i)$. Spoiler pebbles $(g, x_i) \mapsto (f'(g), f'(x_i))$. Let $f'' : G \to H$ be the bijection that Duplicator selects at the next round. As $x_i \mapsto f'(x_i)$ is pebbled, we have that $f''(S_i) = f'(S_i)$. Spoiler now pebbles $y_i \mapsto f''(y_i) = f(y_i)$. As $g \in \text{PKer}(G)$, we have that $g \cdot S_i \cdot g^{-1} = S_i$, while $f'(g) \cdot f'(S_i) \cdot f'(g)^{-1} \neq f'(S_i)$. So Spoiler wins in this case, with a total of 3 pebbles and 2 rounds.

- **Case 2:** Suppose now that $f'(\text{PKer}(G)) = \text{PKer}(H)$, but that $f'|_{\text{PKer}(G)}$ is not an isomorphism. As G and H are semisimple, so are $\text{PKer}(G)$ and $\text{PKer}(H)$. Thus, if $f'|_{\text{PKer}(G)}$ fails to be an isomorphism, then by Lemma 3.6.4, there exists $g \in \text{PKer}(G)$ and $s \in \text{Soc}(G)$ such that $f'(gsg^{-1}) \neq f'(g)f'(s)f'(g)^{-1}$. By Prop. 4.5.9, f' must be an isomorphism on the socle, so if we write $s = s_1 s_2 \ldots s_m$ with each $s_i \in S_i$, then we must have some $i \in [m]$ such that $f'(gs_ig^{-1}) \neq f'(g)f'(s_i)f'(g)^{-1}$. Since $g \in \text{PKer}(G)$, we have $gS_ig^{-1} = S_i$ as well. This case is handled precisely by Lemma 4.5.8 (d), in which only 5 additional pebbles and 5 additional rounds are required for Spoiler to win.

□

We now show that if G and H are not permutationally equivalent, then Spoiler can win.

Lemma 4.5.12. *(Same assumptions as Lemma 4.5.8.) Let G and H be semisimple groups with isomorphic socles, with $\mathit{Fac}(\mathit{Soc}(G)) = \{S_1, \ldots, S_m\}$, with $S_i = \langle x_i, y_i \rangle$. Let $f_0 : G \to H$ be the bijection that Duplicator selects, and suppose that (i) for all i, $f_0(S_i) \cong S_i$ (though $f_0|_{S_i}$ need not be an isomorphism) and $f_0(S_i) \in \mathit{Fac}(\mathit{Soc}(H))$, (ii) for every $s \in \mathit{Soc}(G)$, $\mathit{wt}(s) = \mathit{wt}(f_0(s))$, and (iii) for all i, $f_0(S_i) = \langle f_0(x), f_0(y) \rangle$. Now suppose that Spoiler pebbles $(x_1 \cdots x_m, y_1 \cdots y_m) \mapsto (f_0(x_1 \cdots x_m), f_0(y_1 \cdots y_m))$.*

Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. Suppose that there exist $g \in G$ and $i \in [m]$ such that $f'(gS_ig^{-1}) = f'(S_j)$, but $f'(g)f'(S_i)f'(g)^{-1} = f'(S_k)$ for some $k \neq j$. Then Spoiler can win with 4 additional pebbles and 4 additional rounds in the WL_{II}^2 pebble game.

Proof. We consider the following cases.

- **Case 1:** Suppose first that $i = j$. In this case, Spoiler pebbles $(g, x_i) \mapsto (f'(g), f'(x_i))$. Let $f'' : G \to H$ be the bijection that Duplicator selects at the next round. As $x_i \mapsto f'(x_i)$ is pebbled, we have that $f''(S_i) = f'(S_i)$ by Lemma 4.5.4 (or Spoiler wins with 4 additional pebbles and 2 additional rounds). Spoiler now pebbles $y_i \mapsto f''(y_i)$. The map $(g, x_i, y_i) \mapsto (f'(g), f'(x_i), f''(y_i))$ does not extend to a marked isomorphism, as conjugation by g sends S_i to itself, but conjugation by $f'(g)$ does not send $f'(S_i)$ to itself. Thus, in this case, Spoiler wins with at most 4 additional pebbles and 2 additional rounds.

- **Case 2:** Suppose $i \neq j$ but $i = k$. This is symmetric to the preceding case, by swapping the roles of G and H.

- **Case 3:** Suppose now that i, j, k are all distinct. By Lemma 4.5.7, we have that $f'(x_ix_j) = f'(x_i)f'(x_j)$ (or Spoiler wins with 4 additional pebbles and 3 additional rounds). Spoiler begins by pebbling $(g, x_ix_j) \mapsto (f'(g), f'(x_ix_j))$. Let $f'' : G \to H$ be the bijection that Duplicator selects at the next round. We have the following cases.

  * **Case 3(a):** Suppose that $f''(S_i) = f'(S_i)$. As $x_ix_j \mapsto f'(x_i)f'(x_j)$ is pebbled, we have

necessarily that $f''(x_j) \in f'(S_j)$, and hence that $f''(S_j) = f'(S_j)$ by Lemma 4.5.4. In this case, Spoiler pebbles $(x_i, y_i) \mapsto (f''(x_i), f''(y_i))$. As $gS_ig^{-1} = S_j$, but $f'(g)f''(S_i)f'(g)^{-1} \neq f''(S_j)$, the map $(g, x_i, y_i, x_ix_j) \mapsto (f'(g), f''(x_i), f''(y_i), f'(x_ix_j))$ does not extend to an isomorphism. So Spoiler wins with at most 4 pebbles and 2 rounds.

∗ **Case 3(b):** Suppose that $f''(S_i) \neq f'(S_i)$. As $x_ix_j \mapsto f'(x_i)f'(x_j)$ is pebbled, we have necessarily that $f''(S_i) = f'(S_j)$ and $f''(S_j) = f'(S_i)$. We now have the following sub-cases.

- **Case 3(b).i:** Suppose first that $gS_jg^{-1} = S_i$. By assumption, $gS_ig^{-1} = S_j$, but $f'(g)f'(S_i)f'(g)^{-1} = f'(S_k)$ (where $k \not\in \{i, j\}$). So the conjugation map $\sigma_g : a \mapsto gag^{-1}$ swaps S_i and S_j, while the conjugation map $\sigma_{f'(g)}$ does not swap $f'(S_i)$ and $f'(S_j)$. It follows that the map $(g, x_j, y_j, x_ix_j) \mapsto (f'(g), f''(x_j), f''(y_j), f'(x_ix_j))$ does not extend to an isomorphism. Spoiler now pebbles $(x_j, y_j) \mapsto (f''(x_j), f''(y_j))$. Thus, Spoiler wins with at most 4 additional pebbles and 2 additional rounds.

- **Case 3(b).ii:** Suppose now that $gS_jg^{-1} \neq S_i$. By assumption, as $gS_ig^{-1} = S_j$, we have that $g^{-1}S_jg = S_i$. Now recall that $g \mapsto f'(g)$ is pebbled. As the conjugation map $x \mapsto f'(g)^{-1} \cdot x \cdot f'(g)$ is a bijection and $f'(g)f'(S_i)f'(g)^{-1} = f'(S_k)$ (where again, $k \not\in \{i, j\}$), we have that $f'(g)^{-1}f'(S_j)f'(g) \neq f'(S_i)$.

  Suppose first that $f'(g)^{-1}f'(S_j)f'(g) = f'(S_j)$. Spoiler now pebbles $(x_i, y_i) \mapsto (f''(x_i), f''(y_i))$. As $gS_ig^{-1} \neq S_i$ and $f''(S_i) = f'(S_j)$, the map $(g, x_i, y_i, x_ix_j) \mapsto (f'(g), f''(x_i), f''(y_i), f'(x_ix_j))$ does not extend to an isomorphism. So Spoiler wins with at most 4 pebbles and 2 rounds.

  Suppose now that $f'(g)^{-1}f'(S_j)f'(g) \neq f'(S_j)$. By the first paragraph of Case 3(b).ii, we have that $f'(g)^{-1}f'(S_j)f'(g) \neq f'(S_i)$. As $g^{-1}S_jg = S_i$, we have that

$f''(g^{-1}x_jg) \in f''(S_i)$. So in particular, $f''(g^{-1}x_jg) \neq f'(g)^{-1}f''(x_j)f'(g)$. Spoiler now pebbles $(x_j, g^{-1}x_jg) \mapsto (f''(x_j), f''(g^{-1}x_jg))$. So Spoiler wins with at most 4 pebbles and 2 rounds.

□

Theorem 4.5.13. *Let G be a semisimple group and H an arbitrary group of order n, not isomorphic to G. Then Spoiler has a winning strategy in the $(9, O(1))$-WL_{II}^2 pebble game.*

Proof. If H is not semisimple, then by Proposition 4.5.1, Spoiler wins with 4 pebbles and 2 rounds. So we now suppose H is semisimple.

Let $\text{Fac}(\text{Soc}(G)) = \{S_1, \ldots, S_k\}$, and let x_i, y_i be generators of S_i for each i. Let f be the bijection chosen by Duplicator. Spoiler pebbles $(x_1x_2 \cdots x_k, y_1y_2, \ldots, y_k) \mapsto (f(x_1 \cdots x_k), f(y_1 \cdots y_k))$. On subsequent rounds, we thus have satisfied the hypotheses of Lemma 4.5.8 and Proposition 4.5.9. Spoiler will never move this pebble, and thus all subsequent bijections chosen by Duplicator must restrict to isomorphisms on the socle (or Spoiler wins with at most 7 pebbles and $O(1)$ rounds).

Recall from Lemma 3.6.4 that $G \cong H$ iff there is an isomorphism $\mu \colon \text{Soc}(G) \to \text{Soc}(H)$ that induces a permutational isomorphism $\mu^* \colon G^* \to H^*$. Thus, since $G \ncong H$, there must be some $g \in G$ and $s \in \text{Soc}(G)$ such that $f(gsg^{-1}) \neq f(g)f(s)f(g)^{-1}$. Write $s = s_1 \cdots s_k$ with each $s_i \in S_i$ (not necessarily nontrivial). We claim that there exists some i such that $f(gs_ig^{-1}) \neq f(g)f(s_i)f(g)^{-1}$. For suppose not, then we have

$$\begin{aligned} f(gsg^{-1}) &= f(gs_1g^{-1}gs_2g^{-1} \cdots gs_kg^{-1}) \\ &= f(gs_1g^{-1})f(gs_2g^{-1}) \cdots f(gs_kg^{-1}) \\ &= f(g)f(s_1)f(g)^{-1}f(g)f(s_2)f(g)^{-1} \cdots f(g)f(s_k)f(g)^{-1} \\ &= f(g)f(s_1 \cdots s_k)f(g)^{-1} = f(g)f(s)f(g)^{-1}, \end{aligned}$$

a contradiction. For simplicity of notation, without loss of generality we may assume $i = 1$, so we now have $f(gs_1g)^{-1} \neq f(g)f(s_1)f(g)^{-1}$.

We break the argument into cases:

(1) If $gs_1g^{-1} \in S_1$, then we have $gS_1g^{-1} = S_1$ (any two distinct simple normal factors of the socle intersect trivially), we have by Lemma 4.5.8 (d) that Spoiler can win with at most 5 additional pebbles (for a total of 7 pebbles) and 5 additional rounds (for a total of 6 rounds).

(2) If $gs_1g^{-1} \in S_j$ for $j \neq 1$ and $f(g)f(s_1)f(g)^{-1} \not\in f(S_j)$, we have by Lemma 4.5.12 that Spoiler can win with at most 4 additional pebbles (for a total of 6 pebbles) and 4 additional rounds (for a total of 5 rounds).

(3) Suppose now that $gs_1g^{-1} \in S_j$ for some $j \neq 1$ and $f(g)f(s_1)f(g)^{-1} \in f(S_j)$. Spoiler begins by pebbling $(g, gs_1g^{-1}) \mapsto (f(g), f(gs_1g^{-1}))$. Let $f' : G \to H$ be the bijection that Duplicator selects at the next round. As $gs_1g^{-1} \in S_j$ is pebbled, we have that $f'(S_j) = f(S_j)$ by Lemma 4.5.4 (or Spoiler wins with 4 additional pebbles and 2 additional rounds). Now by assumption, $gS_1g^{-1} = S_j$ and $f(g)f(S_1)f(g)^{-1} = f(S_j)$. So as $g \mapsto f(g)$ is pebbled, we claim that we may assume $f'(S_1) = f(S_1)$. For suppose not; then we have $g^{-1}S_jg = S_1$ but $f'(g)^{-1}f'(S_j)f'(g) = f(g)^{-1}f(S_j)f(g) = f(S_1) \neq f'(S_1)$. But then Spoiler can with win with 4 additional pebbles (for a total of 8 pebbles) and 4 additional rounds (for a total of 7 rounds) by Lemma 4.5.12. Thus we have $f'(S_1) = f(S_1)$.

In particular, we have that $f'(x_1) = f(x_1)$ and $f'(y_1) = f(y_1)$, by the same argument as in the proof of Lemma 4.5.8 (c). As $S_1 = \langle x_1, y_1 \rangle$, we have that $f'(s_1) = f(s_1)$, since they are both isomorphisms on the socle by Proposition 4.5.9. Spoiler now pebbles $(x_1, y_1) \mapsto (f'(x_1), f'(y_1))$. As the pebbled map $(g, x_1, y_1, gs_1g^{-1}) \mapsto (f(g), f'(x_1), f'(y_1), f'(gs_1g^{-1}))$ does not extend to an isomorphism, Spoiler wins. In this case, Spoiler used at most 8 pebbles and 7 rounds.

Note that the ninth pebble is the one we pick up prior to checking the winning condition. □

Chapter 5

Isomorphism Testing of Strongly Regular Graphs

The work in this chapter resulted in [143].

In this chapter, we investigate the parallel complexity of several isomorphism problems arising from quasigroups and certain families of combinatorial designs. We show (in particular) that (a) Latin square graphs, and (b) block-incidence graphs arising from special cases of Steiner 2-designs are not GI-hard under AC^0-computable many-one reductions. As a consequence of this work and prior work of Chattopadhyay, Torán, and Wagner [57], we have that the parallel complexity of these isomorphism problems is strictly easier than GI. Furthermore, in light of the fact that $\mathsf{AC}^0 = \mathsf{FO}$ [161], the results in this chapter suggest that in logics extending FO, certain families of strongly regular graphs may be definable using more succinct sentences than in the case of general graphs.

As with the work of Chattopadhyay, Torán, & Wagner [57], we show this by improving the parallel complexity of isomorphism testing in these classes of graphs to $\beta_2\mathsf{FOLL}$, which does not compute Parity. As Parity is AC^0-reducible to GI [189], this rules out AC^0-reductions (and more strongly, for any $i, c \geq 0$, we rule out $\beta_i\mathsf{FO}((\log \log n)^c)$-reductions).

Prior to our work, there was little complexity-theoretic evidence to suggest that strongly-regular graphs are not GI-complete. Babai showed that there is no functorial reduction from GI to the isomorphism testing of strongly-regular graphs [25]. We note that almost all known reductions between isomorphism problems are functorial (c.f. [25]). An example where this is not the case is the reduction from 1-Block Conjugacy of shifts of finite type to k-Block Conjugacy [177, Theorem 18]. In the case of Quasigroup Isomorphism, Chattopadhyay, Torán, and Wagner improved the upper bound to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. In particular, they showed that for any $i, c \geq 0$, GI is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to Quasigroup Isomorphism [57]. We begin by summarizing the main results of this chapter.

Theorem 5.0.1. Latin Square Isotopy *is in* $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$*.*

Remark 5.0.2. This improves the previous bound of $\beta_2\mathsf{NC}^2$ due to Wolf [200].

To prove Theorem 5.0.1, we leverage cube generating sequences, in a similar manner as Chattopadhyay, Torán, & Wagner [57, Theorem 3.4]. After non-deterministically guessing cube generating sequences, we can check in $\mathsf{L} \cap \mathsf{FOLL}$ whether the induced map extends to an isotopy of the Latin squares.

Now for any $i, c \geq 0$, we have that $\beta_i\mathsf{FO}((\log \log n)^c)$ cannot compute Parity [57]. Thus, we obtain the following corollary.

Corollary 5.0.3. *For any* $i, c \geq 0$*,* GI *is not* $\beta_i\mathsf{FO}((\log \log n)^c)$*-reducible to* Latin Square Isotopy.

Latin squares yield a corresponding family of strongly regular graphs, known as *Latin square graphs*, where two Latin square graphs G_1 and G_2 are isomorphic if and only if their corresponding Latin squares $L(G_1)$ and $L(G_2)$ are main class isotopic [160, Lemma 3]. Miller previously showed that it is possible to recover the corresponding Latin square from a Latin square graph in polynomial-time [160], and Wolf strengthened the analysis to show that this reduction is in fact NC^1-computable [200]. A closer analysis yields that this reduction is actually AC^0-computable (see Lemma 5.1.8). Thus, we obtain the following corollary.

Corollary 5.0.4. *Deciding whether two Latin square graphs are isomorphic is in* $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$*. Consequently, for any* $i, c \geq 0$*,* GI *is not* $\beta_i\mathsf{FO}((\log \log n)^c)$*-reducible to isomorphism testing of Latin square graphs.*

Remark 5.0.5. It is also possible to construct a Latin square graph from a Latin square using an AC^0-circuit. Thus, Latin Square Isotopy and isomorphism-testing of Latin square graphs are equivalent under AC^0-reductions.

Remark 5.0.6. We sketch an alternative proof of Theorem 5.0.1 and Corollary 5.0.4. In Miller's [160] first proof that Latin Square Isotopy is solvable in time $n^{\log(n)+O(1)}$, Miller takes one Latin square L' and places it in a normal form. Miller shows that this step is polynomial-time computable. A closer analysis shows that it is in fact AC^0-computable: we may label the first row and first column as $1, 2, \ldots, n$. We then in AC^0 fill in the remaining cells of the normal form. For the second Latin square L, Miller places L in n^2 possible normal forms by guessing the element to label as 1. We may consider all such guesses in parallel, and so this step is also AC^0-computable. Miller then checks whether the normal form of L' and the normal form of L are isomorphic as quasigroups. This step is $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$-computable [57]. Thus, we have an AC^0-computable disjunctive truth table reduction from Latin Square Isotopy to Quasigroup Isomorphism, which places Latin Square Isotopy into $\beta_2\mathsf{L}\cap\beta_2\mathsf{FOLL}$. We note that Wolf's [200] proof that Latin Square Isotopy $\in \beta_2\mathsf{NC}^2$ followed a similar strategy as Miller's [160] proof that Quasigroup Isomorphism can be solved in time $n^{\log(n)+O(1)}$.

As we can recover a Latin square from a Latin square graph in AC^0 (see Lemma 5.1.8), we also have an AC^0-computable disjunctive truth table reduction from isomorphism testing of Latin square graphs to Quasigroup Isomorphism. So isomorphism testing of Latin square graphs belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$.

Instead, the proof we give of Theorem 5.0.1 (see Section 5.1) exhibits how the technique of cube generating sequences [57] can be fruitfully applied directly to isotopy (rather than isomorphism) of algebraic structures, which we do by combining cube generating sequences with Miller's [160] second proof that Latin Square Isotopy can be solved in time $n^{\log(n)+O(1)}$.

We now turn our attention to isomorphism testing of Steiner designs.

Theorem 5.0.7. *For any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log\log n)^c)$-reducible to isomorphism testing of*

Steiner $(t, t+1)$-designs.

We prove Theorem 5.0.7 in two steps. First, we establish the case of $t = 2$, which is the case of Steiner triple systems. It is well-known that Steiner triple systems correspond to quasigroups, and that this reduction is polynomial-time computable [160]. A careful analysis shows that this reduction is in fact AC^0-computable. Furthermore, we observe that the reduction in [36, Theorem 33] from isomorphism testing of Steiner t-designs to isomorphism testing of Steiner 2-designs is $\beta_2\mathsf{AC}^0$-computable. As a corollary, isomorphism testing of Steiner $(t, t+1)$-designs is $\beta_2\mathsf{AC}^0$-reducible to isomorphism testing of Steiner triple systems, and hence Quasigroup Isomorphism.

Steiner designs yield a family of graphs known as *block intersection graphs.* In the case of Steiner 2-designs, these graphs are strongly regular. When the block size is bounded, we observe that we may recover a Steiner 2-design from its block-incidence graph in AC^0. If the block size is not bounded, this reduction is TC^0-computable. In the case of Steiner triple systems, this yields the following corollary.

Corollary 5.0.8. *Let G_1, G_2 be block-incidence graphs arising from Steiner triple systems. We can decide whether $G_1 \cong G_2$ in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. Consequently, for any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of block-incidence graphs arising from Steiner triple systems.*

5.1 Latin Square Isotopy

In this section, we show that the Latin Square Isotopy problem is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. The key technique is to guess cube generating sequences A and B for the Latin square L_1, and cube generating sequences A', B' for the Latin square L_2. We then (attempt to) construct appropriate bijections $\alpha, \beta : L_1 \to L_2$, where α extends the map from $A \to A'$ and β extends the map from $B \to B'$. We can construct such bijections in $\mathsf{FOLL} \cap \mathsf{L}$ using the techniques from Chattopadhyay, Torán, and Wagner [57].

The key step remains in checking whether the map sending the product of each pair $(x, y) \in L_1 \times L_1$, $xy \mapsto \alpha(x)\beta(y)$ a bijection. Wolf approaches this in the following manner. First, construct sets $C = \{a_i b_i : a_i \in A, b_i \in B\}$ and $C' = \{a_i' b_i' : a_i' \in A', b_i' \in B'\}$. Then check whether the map $a_i b_i \mapsto a_i' b_i'$ extends to a bijection $\gamma : L_1 \to L_2$. Finally, check whether $\alpha(x)\beta(y) = \gamma(xy)$ for all $x, y \in L_1$. We note that Wolf is able to do this in NC^1 [200]. If C and C' are cube generating sequences, then we would be able to apply the technique of Chattophadyay, Torán, and Wagner [57] to compute the bijection γ in $\mathsf{FOLL} \cap \mathsf{L}$. However, C and C' need not be cube generating sequences. As an example in the case of groups, suppose that $B = A^{-1}$. Then B is a cube generating sequence. Furthermore, $a_i b_i = 1$ for each i, and so C has only the identity element. In general, we do not expect C and C' to be cube generating sequences, even if they do generate L_1 and L_2 respectively.

Instead of extending Wolf's technique, we extend Miller's technique in his second proof that Latin Square Isotopy is decidable in time $n^{\log(n)+O(1)}$. Miller constructs the relation $R = \{(xy, \alpha(x)\beta(y)) : x, y \in L_1\}$ and checks whether R is a bijection. If R is a bijection; then by construction, the triple (α, β, R) is an isotopy [160, Theorem 2, Proof 2]. Using the fact that A and B are cube generating sequences, we can compute x and y in $\mathsf{L} \cap \mathsf{FOLL}$. In the process of computing α and β, we obtain a data structure which allows us to compute $\alpha(x)$ and $\beta(y)$ in $\mathsf{L} \cap \mathsf{FOLL}$.

Theorem 5.1.1. Latin Square Isotopy *is in* $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$.

We begin with the following lemmas.

Lemma 5.1.2. *Let A, B be finite sets of the same size, and let $R \subseteq A \times B$ be a relation. We can decide whether R is a well-defined surjection (and as $|A| = |B|$, consequently whether R is a well-defined bijection) in* AC^0.

Proof. For each $b \in B$, define the relation:

$$Y(b) = \bigvee_{a \in A} 1_{(a,b) \in R}.$$

Observe that $Y(b)$ is AC^0-computable. Now R is surjective if and only if $Y(b) = 1$ for all $b \in B$, which is defined by the following condition:

$$\varphi := \bigwedge_{b \in B} Y(b).$$

Observe that φ is AC^0 computable. □

Lemma 5.1.3. *Let L_1 and L_2 be Latin squares of order n, and let $k = 4\lceil \log(n) \rceil$. Suppose that $(g_0, g_1, \dots, g_k)$ and $(h_0, \dots, h_k)$ are cube generating sequences for L_1 and L_2 respectively, with balanced parenthesization P. Deciding whether the map $g_i \mapsto h_i$ for all $i \in \{0, \dots, k\}$ extends to a bijection is in $\mathsf{L} \cap \mathsf{FOLL}$.*

Proof. For each $(g, h) \in L_1 \times L_2$, define $X(g, h) = 1$ if and only if there exists $(\epsilon_1, \dots, \epsilon_k) \in \{0, 1\}^k$ such that:

$$g := g_0 g_1^{\epsilon_1} \cdots g_k^{\epsilon_k},$$

$$h := h_0 h_1^{\epsilon_1} \cdots h_k^{\epsilon_k}.$$

Chattophadyay, Torán, and Wagner showed that the cube words for g and h can be computed in $\mathsf{L} \cap \mathsf{FOLL}$ [57], so $X(g, h)$ is computable in $\mathsf{L} \cap \mathsf{FOLL}$. We note that the FOLL bound follows from the fact that P is a balanced parenthesization. As the two quasigroups are the same size, the map on the quasigroups induced by $(g_0, g_1, \dots, g_k) \mapsto (h_0, \dots, h_k)$ is a well-defined bijection iff the induced map is injective iff the induced map is surjective. So it now suffices to check whether the induced map is surjective. By Lemma 5.1.2, we may check whether the induced map is surjective in AC^0. □

Proof of Theorem 5.1.1. Let $k = 4\lceil \log_2(n) \rceil$. We use $4k^2$ non-deterministic bits to guess cube generating sequences $A, B \subseteq L$ and $A', B' \subseteq L'$, where $A = \{a_0, a_1, \dots, a_k\}$, $B = \{b_0, b_1, \dots, b_k\}$, $A' = \{a'_0, a'_1, \dots, a'_k\}$, and $B' = \{b'_0, b'_1, \dots, b'_k\}$. We may then in $\mathsf{L} \cap \mathsf{FOLL}$ check the following.

- We first check that the map $a_i \mapsto a'_i$ extends to a bijection of $\langle A \rangle$ and $\langle A' \rangle$. In particular, we may check in $\mathsf{L} \cap \mathsf{FOLL}$ whether $L_1 = \text{Cube}(A)$ and $L_2 = \text{Cube}(A')$ (see [57, Theorem 5]). The procedure in Lemma 5.1.3 that decides whether the map $a_i \mapsto a'_i$ for all $i \in \{0, \dots, k\}$ extends to a bijection $L_1 \to L_2$, also explicitly computes a bijection $\varphi_A : L_1 \to L_2$ if one exists.

- We proceed analogously for the map $b_i \mapsto b'_i$ for all $i \in \{0, \dots, k\}$. In the case that $L_1 = \text{Cube}(B)$ and $L_2 = \text{Cube}(B')$, let $\varphi_B : L_1 \to L_2$ be the bijection computed by Lemma

5.1.3.

Suppose now that the bijections $\varphi_A, \varphi_B : L_1 \to L_2$ have been constructed. We now attempt to construct a relation $\varphi_C : L_1 \to L_2$ in such a way that φ_C is a bijection if and only if $(\varphi_A, \varphi_B, \varphi_C)$ is an isotopism. For each pair $(\ell, m) \in L_1 \times L_2$, define $X(\ell, m) = 1$ if and only if we define φ_C to map $\ell \mapsto m$. For each $(\epsilon_1, \ldots, \epsilon_k), (\nu_1, \ldots, \nu_k) \in \{0, 1\}^k$, we do the following:

(a) Compute:

$$g := a_0 a_1^{\epsilon_1} \cdots a_k^{\epsilon_k},$$

$$h := b_0 b_1^{\nu_1} \cdots b_k^{\nu_k}.$$

We note that computing g and h can be done in $\mathsf{L} \cap \mathsf{FOLL}$ (see Chattopadhyay, Toràn, and Wagner [57]).

(b) Compute $\ell := gh$, $\varphi_A(g)$, $\varphi_B(h)$, and $m := \varphi_A(g)\varphi_B(h)$. We set $\varphi_C(\ell) = m$, which we indicate by defining $X(\ell, m) = 1$. The computations at this stage are computable using an AC^0 circuit.

(c) It remains to check whether φ_C is a bijection. By Lemma 5.1.2, we may test whether φ_C is a bijection in AC^0.

Now φ_C was constructed so that $(\varphi_A, \varphi_B, \varphi_C)$ satisfies the homotopy condition, so, as they are also bijective, they are an isotopy. Thus, checking whether L_1 and L_2 are isotopic is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. □

Corollary 5.1.4. *For any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to* Latin Square Isotopy*.*

Miller [160] showed that isomorphism testing of Latin square graphs is polynomial-time reducible to the Latin square isotopy problem. Wolf [200] improved this bound, showing that isomorphism testing of Latin square graphs is NC^1 reducible to testing for Latin square isotopy. We recall Wolf's result below.

Lemma 5.1.5 (Wolf [200, Lemma 4.11])**.** *Let G be a Latin square graph derived from a Latin square of size n. We can retrieve a Latin square from G with a polynomial-sized NC circuit with $O(\log(n))$ depth.*

Remark 5.1.6. The statement of Lemma 4.11 in Wolf actually claims a depth of $O(\log^2(n))$. However, in the proof of Lemma 4.11, Wolf shows that only $O(\log(n))$ depth is needed [200].

In light of Wolf's result, Theorem 5.1.1, and Bruck's result that for $n > 23$, a pseudo-Latin square graph is a Latin square graph [52], we obtain the following corollary.

Corollary 5.1.7. *Isomorphism of pseudo-Latin square graphs can be decided in $\beta_2\mathsf{L}$.*

We next show that the reduction from [200, Lemma 4.11], which effectively parallelizes the reduction found in [160], can be implemented in AC^0. It follows that Latin Square Graph Isomorphism is also in $\beta_2\mathsf{FOLL}$. As for any $i, c \geq 0$, $\beta_i\mathsf{FO}((\log\log n)^c)$ cannot compute Parity, we obtain that for any $i', c' \geq 0$, GI is not $\beta_{i'}\mathsf{FO}((\log\log n)^{c'})$-reducible to Latin Square Graph Isomorphism.

Lemma 5.1.8. *Let G be a Latin square graph obtained from a Latin square of order n. We can recover a Latin square from G using an AC^0 circuit.*

Proof. By construction, two vertices u and v in G are adjacent precisely if u and v correspond to elements in the Latin square that are either in the same row, the same column, or have the same value. As a result, each row, each column, and the nodes corresponding to a fixed given value all induce cliques of size n. The algorithm effectively identifies these cliques and their relations to the other cliques in order to recover a Latin square.

Let $L = (\ell_{ij})_{1 \leq i,j \leq n}$ to be an $n \times n$ matrix, which our algorithm will populate with values for the Latin square. We proceed as follows.

(1) We begin by selecting two adjacent vertices x_1 and x_2. Without loss of generality, we may assume x_1 and x_2 belong to the same row of the Latin square. We next find the n vertices

$v_1, \ldots, v_n$ adjacent to both x_1 and x_2. Precisely, for a vertex v, we have the indicator

$$X(v) = E(x_1, v) \wedge E(x_2, v),$$

where $E(u, v) = 1$ precisely if u and v are adjacent. Exactly n indicators will evaluate to 1. All but two of these nodes form a clique of size n with x_1 and x_2. As $v_1, \ldots, v_n$ are adjacent to x_1 and x_2, it suffices to check which $n-2$ vertices of $v_1, \ldots, v_n$ form a clique of size $(n-2)$. For a given set $S \in \binom{[n]}{n-2}$, it suffices to check:

$$\bigwedge_{\substack{i,j \in S \\ i \neq j}} E(v_i, v_j),$$

which is AC^0-computable. There are $\binom{n}{n-2} \in \Theta(n^2)$ such sets to check. Thus, identifying the clique is AC^0-computable.

In parallel, we label the vertices of the clique as $x_3, \ldots, x_n$. One node not adjacent to any of $x_3, \ldots, x_n$ is labeled y_2.

(2) We associate ℓ_{1j} with x_j. Precisely, we set (in parallel) $\ell_{1j} = j$ for each $j \in [n]$. This step is AC^0-computable.

(3) We next find the n-clique associated with x_1 and y_2, and label the additional vertices as $y_3, \ldots, y_n$. By similar argument as in Step 1, this step is AC^0-computable. Here, we view $x_1, y_2, y_3, \ldots, y_n$ as the first column of the Latin square.

(4) For each $3 \leq i \leq n$, there exists a $3 \leq j \leq n$ such that x_i and y_j are adjacent. In particular, as x_i and y_j are neither in the same row nor the same column, it must be the case that x_i and y_j correspond to elements in the Latin square with the same value. It follows that our choice of j is in fact unique. We reorder $y_3, \ldots, y_n$ so that x_i is adjacent to y_i. This step is AC^0-computable.

(5) For $2 \leq j \leq n$, we associate ℓ_{j1} with y_j. Precisely, we set (in parallel), $\ell_{j1} = j$ for each $2 \leq j \leq n$. This step is AC^0-computable.

(6) For each of the remaining $(n-1)^2$ nodes z, we do the following in parallel:

  (a) If z is adjacent to x_1, then the edge $\{x_1, z\}$ is a value edge (as z is not in the same row or column as x_1). So there exist unique $i, j > 1$ such that z is adjacent to x_j and y_i. In this case, we set $\ell_{ij} = 1$. This case is AC^0-computable.

  (b) Suppose that z is not adjacent to x_1. As each row, each column, and each value induce an n-clique, there exist unique $i, j, k \in [n]$ such that z is adjacent to x_j, y_i, x_k, and y_k. Unless $i = j = k$, we set $\ell_{ij} = k$. If $i = j = k$, we do nothing at this step and defer to step 7. This case is AC^0-computable.

(7) We note that step 6b does not account for diagonal entries where $\ell_{ii} = i$. To this end, we do the following. For each $i \geq 2$, we (in parallel) set ℓ_{ii} to be the value that does not appear in row i. This step is AC^0-computable.

As we have a finite number of steps, each of which are AC^0-computable, it follows that we may recover a Latin square from G using an AC^0 circuit. □

Proposition 5.1.9. *Isomorphism testing of pseudo-Latin square graphs is in $\beta_2\mathsf{FOLL}$. In particular, for any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of pseudo-Latin square graphs.*

Proof. We may handle the cases when $n \leq 23$ in AC^0. So suppose $n \geq 23$, and let G and H be pseudo-Latin square graphs. As $n \geq 23$, we have by Bruck that G and H are Latin square graphs [52]. By Lemma 5.1.8, we may in AC^0 recover canonical Latin squares L_G and L_H corresponding to G and H. Now by [160, Lemma 3], $G \cong H$ if and only if L_G and L_H are main class isotopic. By Remark 5.0.6, we may place L_G (respectively, L_H) into a normal form corresponding to its main isotopy class in AC^0. By Theorem 5.1.1, we can test whether L_G and L_H are isotopic in $\beta_2\mathsf{FOLL}$. Chattopadhyay, Torán, and Wagner showed that $\beta_2\mathsf{FOLL}$ cannot compute Parity [57]. The result now follows. □

5.2 Isomorphism Testing of Steiner Designs

In this section, we show that for any $i, c \geq 0$, GI is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of several families of Steiner designs.

5.2.1 Nets

In this section, we show that for any $i, c \geq 0$, GI is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of nets or the corresponding strongly regular graphs. We note that projective and affine planes are special cases of nets.

Theorem 5.2.1. *Deciding whether two k-nets are isomorphic is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$.*

Proof. For $k = 0, 1, 2$, the pair (k, n) determines the net uniquely, and so deciding isomorphism is trivial in these cases. So assume that $n + 1 \geq k \geq 3$. Let $\mathcal{N}_1(n, k), \mathcal{N}_2(n, k)$ be nets. We now start by guessing three non-parallel lines $\ell_a, \ell_b, \ell_c \in L(\mathcal{N}_1)$ and three non-parallel lines $\ell_a', \ell_b', \ell_c' \in L(\mathcal{N}_2)$. By definition, no two lines in the same parallel class share any points in common, and two lines in different classes share exactly one point in common. As there are kn lines in $\mathcal{N}_i$, we only require $O(\log(kn))$ bits to identify a given line. As $k \leq n + 1$, in fact only $O(\log(n))$ bits are required.

We may identify whether two lines are disjoint in AC^0. In particular, we may check in AC^0 whether ℓ_a, ℓ_b, and ℓ_c (respectively, ℓ_a', ℓ_b', and ℓ_c') belong to different parallel classes. Suppose now that ℓ_a, ℓ_b, and ℓ_c (respectively, ℓ_a', ℓ_b', and ℓ_c') belong to different parallel classes. As there are $\binom{kn}{2}$ such pairs to check in a given $\mathcal{N}_i$, we may identify in AC^0 the lines belonging to the three parallel classes a, b, c in each $\mathcal{N}_i$.

Now our three parallel classes determine a net $X_1(n, 3)$ in $\mathcal{N}_1$ and a net $X_2(n, 3)$ in $\mathcal{N}_2$. We note that a net of degree 3 identifies a quasigroup (Latin square) up to isomorphism. As Quasigroup Isomorphism is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$ [57, Theorem 3.4], we may now in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$ decide whether $X_1 \cong X_2$. We note that [57, Theorem 3.4] actually yields an isomorphism $\varphi : X_1 \to X_2$. We may now in $\mathsf{L} \cap \mathsf{FOLL}$ check whether φ extends to an isomorphism of $\mathcal{N}_1$ and $\mathcal{N}_2$. The result follows. □

Corollary 5.2.2. *For any $i, c \geq 0$,* GI *is not $\beta_i \mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of two k-nets.*

Miller previously showed that isomorphism testing of nets and the corresponding net graphs are equivalent under polynomial-time reductions when $n > (k-1)^2$ [160, Theorem 8]. A closer analysis shows that this equivalence holds under TC^0-reductions in general; and that when k is bounded, the equivalence holds under AC^0-reductions.

Lemma 5.2.3. *Suppose that $G(V, E)$ is a k-net graph of order n and $n > (k-1)^2$. We can reconstruct the net $\mathcal{N}(n, k)$ associated with G in TC^0. If k is bounded, we reconstruct the net $\mathcal{N}(n, k)$ associated with G in AC^0.*

Proof. Suppose that $x_1, x_2 \in V(G)$ are adjacent. As there are kn lines, it suffices to show that in AC^0, we can identify the remaining vertices of $V(G)$ that are on the same line as x_1, x_2. We first note that we can, in AC^0, identify the vertices adjacent to both x_1 and x_2.

Let H be the subgraph induced by these vertices together with x_1 and x_2. We note that H contains the maximum clique (of n vertices) containing both x_1 and x_2. The vertices of this clique have degree $\geq n-1$. As two adjacent vertices have $(n-2)+(k-1)(k-2)$ neighbors in common, there are $(k-1)(k-2)$ vertices of H that do not belong to this clique. Recall that a net has k parallel classes, and any two lines in a given parallel class have empty intersection. Furthermore, two lines from different parallel classes share exactly one point in common. Thus, each nonclique vertex of H is adjacent to exactly $(k-1)$ elements of the clique. Thus, each nonclique vertex of H has degree at most:

$$(k-1)+(k-1)(k-2)-1=(k-1)^2-1.$$

In general, using the difference in degrees, we may identify the clique and nonclique vertices in TC^0. If k is bounded, then we may identify the clique and nonclique vertices in AC^0. The result follows. □

We combine Lemma 5.2.3 with Bruck's result that for fixed k, pseudo-net graphs with sufficiently many vertices vertices are Net graphs to obtain the following.

Corollary 5.2.4. *For fixed k, isomorphism testing of pseudo-net graphs of order n and degree k is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. In particular, for any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of pseudo-net graphs of order n and degree k.*

5.2.2 Steiner Triple Systems

Theorem 5.2.5. *Deciding whether two Steiner triple systems are isomorphic is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$.*

Proof. Let $\mathcal{S}$ be a Steiner triple system of order n. We define a quasigroup Q on the set $[n]$, with the multiplication operation satisfying the following: (i) for each $x \in [n]$, define $x^2 = x$, and (ii) for each block $\{a, b, c\}$, define $ab = c$ (note that as the blocks are unordered, all such products $ba = c, ac = ca = b, bc = cb = a$ are required). The Steiner triple system determines Q up to isomorphism. In particular, this construction is AC^0-computable. As Quasigroup Isomorphism belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$ [57], it follows that deciding whether two Steiner triple systems are isomorphic is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. □

Proposition 5.2.6. *Let G be a Steiner graph on n vertices derived from a Steiner 2-design, in which each block has k points and $\sqrt{n} - 2 > (k-1)^2$. We can reconstruct the Steiner 2-design in TC^0. Furthermore, if k is bounded, then we may reconstruct the Steiner 2-design in AC^0.*

Proof. We closely analyze the proof of [184, Proposition 10]. We note that n is the number of blocks in the Steiner design. Let v be the number of points in the Steiner design, and let $R = (v-1)/(k-1)$ be the number of blocks containing a given point. Let B_1, B_2 be two blocks that intersect uniquely at the point p. There are $h - 2 + (s-1)^2$ blocks that intersect both B_1 and B_2. We note that $h - 2$ of these blocks also go through p, and the remaining $(s-1)^2$ blocks go through points other than p.

We note that p corresponds to the edge $\{B_1, B_2\} \in E(G)$. The h blocks that intersect p (including B_1, B_2) form a clique. Furthermore, these $h - 2$ blocks intersecting B_1, B_2 at p do not intersect with the remaining $(s-1)^2$ blocks that intersect with B_1, B_2 in points other than p. Let N be the set of mutual neighbors for B_1, B_2. Now if $h - 2 > (s-1)^2$, the h-clique is distinguished from the remaining $(s-1)^2$ blocks that don't contain p by their degrees in the induced subgraph $G[N]$. We may distinguish these vertices in TC^0 in general. If k is bounded, then we may distinguish these vertices in AC^0.

The fact that $h > \sqrt{n}$ was established in [184, Proposition 10]. □

We combine Proposition 5.2.6 with Bose's result that pseudo-STS graphs with strictly more than 67 vertices are STS graphs [46] to obtain the following.

Corollary 5.2.7. *Deciding whether two pseudo-STS graphs are isomorphic is in $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$. In particular, for any $i, c \geq 0$,* GI *is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of pseudo-STS graphs.*

5.2.3 Reduction to Steiner 2-Designs

Babai & Wilmes [36] previously exhibited a reduction from isomorphism testing of Steiner t-designs to the case of Steiner 2-designs. A careful analysis shows that their reduction is $\beta_2\mathsf{AC}^0$-computable. As a corollary, we obtain that GI is not AC^0-reducible to isomorphism testing of Steiner $(t, t+1)$-designs.

Observation 5.2.8 (c.f. [36])**.** If isomorphism testing of Steiner 2-designs belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$, then testing isomorphism of Steiner t-designs also belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$.

Proof. Let $\mathcal{D}_1 = (X_1, \mathcal{B}_1, I_1)$ and $\mathcal{D}_2 = (X_2, \mathcal{B}_2, I_2)$ be Steiner (t, k, v)-designs. We begin by guessing subsequences $a_1, \ldots, a_{t-2} \in X_1$ and $b_1, \ldots, b_{t-2} \in X_2$. While we think of $a_1, \ldots, a_{t-2}$ and $b_1, \ldots, b_{t-2}$ as determining subsets $A = \{a_1, \ldots, a_{t-2}\} \subseteq X_1$ and $B = \{b_1, \ldots, b_{t-2}\} \subseteq X_2$, we stress that we guess both the elements *and* an ordering. As $t \in O(\log n)$ (see the proof of [36, Theorem 30], in which Babai & Wilmes cite [170]), guessing A and B requires $O(\log^2 n)$ bits. We may write down the derived designs $\mathcal{D}_1(A)$ and $\mathcal{D}_2(B)$ in AC^0 (see Section 2.4 for the definition of a derived design).

Suppose first that $\mathcal{D}_1 \cong \mathcal{D}_2$. Let $\varphi : X_1 \to X_2$ be an isomorphism. Then for any $A \subseteq X_1$ of size $t-2$, the derived designs $\mathcal{D}_1(A)$ and $\mathcal{D}_2(\varphi(A))$ are isomorphic.

Conversely, let $\psi : X_1 \backslash A \to X_2 \backslash B$ be an isomorphism of the derived designs $\mathcal{D}_1(A) \cong \mathcal{D}_2(B)$. Define:

$$\widehat{\psi}(x) = \begin{cases} \psi(x) & : x \not\in A, \\ \\ b_i & : x = a_i \in A. \end{cases}$$

We may easily check in AC^0 whether $\widehat{\psi}$ is an isomorphism of $\mathcal{D}_1$ and $\mathcal{D}_2$. In particular, if isomorphism testing of Steiner 2-designs belongs to $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$, we may use the added non-determinism to guess an isomorphism of $\mathcal{D}_1(A) \cong \mathcal{D}_1(B)$ that lifts to an isomorphism of $\mathcal{D}_1 \cong \mathcal{D}_2$. In total, we are still using at most $O(\log^2 n)$ non-deterministic bits. □

We obtain the following corollaries.

Corollary 5.2.9. *The problem of deciding whether two Steiner* $(t, t+1)$*-designs are isomorphic is* $\beta_2\mathsf{AC}^0$*-reducible to the problem of finding an isomorphism of two Steiner triple systems.*

Now deciding isomorphism testing of Steiner triple systems is AC^0-reducible to Quasigroup Isomorphism (see Theorem 5.2.5). Furthermore, Chattopadhyay, Torán, and Wagner solved the search version of Quasigroup Isomorphism; that is, their $\beta_2\mathsf{L} \cap \beta_2\mathsf{FOLL}$ procedure for Quasigroup Isomorphism returns an isomorphism of the two quasigroups whenever an isomorphism exists. So in particular GI is not AC^0-reducible to the search version of Quasigroup Isomorphism [57]. We may use the isomorphism for the quasigroups to construct an isomorphism of the Steiner triple systems, which may in turn be used to construct an isomorphism of the two Steiner $(t, t+1)$-designs. We summarize this observation with the following corollaries.

Corollary 5.2.10. *The problem of deciding whether two Steiner* $(t, t+1)$*-designs are isomorphic is* $\beta_2\mathsf{AC}^0$*-reducible to the problem of finding an isomorphism of two quasigroups.*

Corollary 5.2.11. *For any* $i, c \geq 0$, GI *is not* $\beta_i\mathsf{FO}((\log \log n)^c)$*-reducible to the problem of deciding whether two Steiner* $(t, t+1)$*-designs are isomorphic.*

5.3 Conference Graphs

In this section, we consider the complexity of identifying conference graphs. A *conference* graph is a strongly regular graph with parameters $(n, (n-1)/2, (n-5)/4, (n-1)/4)$. For a graph G, a *distinguishing set* S is a subset of $V(G)$ such that for every $u, v \in V(G)$, we have that $u \in S, v \in S$, or $N(u) \cap S \neq N(v) \cap S$. We have the following observation.

Observation 5.3.1. Let G be a graph, and let S be a distinguishing set. After individualizing each vertex in S, we have that 2 rounds of the count-free Color Refinement algorithm (the initial coloring and a single refinement step) will assign each vertex in G a unique color.

Babai [37, Lemma 3.2] (take $k = (n-5)/4$) showed that conference graphs admit distinguishing sets of size $O(\log n)$ (in fact, almost all such subsets of this size are distinguishing). As a consequence of this and Observation 5.3.1, we obtain the following.

Theorem 5.3.2. *Let G be a conference graph, and let H be arbitrary. We can decide isomorphism between G and H in $\beta_2\mathsf{AC}^0$.*

Proof. We use $m := O(\log^2 n)$ non-deterministic bits to guess a sequence $S = (s_1, \ldots, s_m)$ for G– while it would suffice for $\{s_1, \ldots, s_m\}$ to be a distinguishing set for G, we only need that after individualizing the elements of S, two rounds of count-free Color Refinement will assign each vertex in G a unique color. There may be non-distinguishing sets that acccomplish this. Let $S' = (s'_1, \ldots, s'_m)$ be the vertices of H that were guessed. We now individualize S and S' so that $s_i \mapsto s'_i$ get the same color, and s_i, s_j get different colors whenever $i \neq j$. The individualization step is AC^0-computable. We now run two rounds of count-free Color Refinement, which is AC^0-computable (c.f., [104] and the discussion in Section 2.7). Lastly, we check that each vertex in G, H has a unique color, and whether the map induced by the colors is an isomorphism. This last step is AC^0-computable. The result now follows. □

In light of the previous work of Chattopadhyay, Torán, & Wagner [57], we obtain the following corollary.

Corollary 5.3.3. *For any $i, c \geq 0$, there is no $\beta_i\mathsf{FO}((\log \log n)^c)$-computable reduction from* GI *to isomorphism testing of conference graphs.*

Chapter 6

Relation Algebras

This work resulted in two papers: [8] is joint with Jeremy F. Alm, Saeed Moazami, Jorge Montero-Vallejo, Linda Pham, Dave Sexton, and Xioanan Xu; and [13], which is joint with Jeremy F. Alm.

In this chapter, we investigate the combinatorial complexity of the relation algebra we call A_n, which is obtained by splitting the non-flexible diversity atom of 6_7 (see [153] for the numbering) into n symmetric atoms. Let A_n denote the integral symmetric relation algebra with the atom $1'$ and diversity atoms r, b_1, …, b_n, where a diversity cycle is mandatory if and only if it involves the atom r (see Section 2.11). Let

$$f(n) = \min(\mathrm{Spec}(A_n)).$$

It was shown in [7] that $f(n)$ is finite for all n. Because representing finite integral relation algebras amounts to edge-coloring complete graphs with the diversity atoms, we will use the language of graph theory. We will refer to the flexible atom r as *red*, and $b_1, \ldots, b_n$ as our shades of blue.

In this chapter, we will show that $f(n) \leq 2n^{6+o(1)}$, which is the first polynomial bound and improves upon the previous bound due to Dodd & Hirsch [75]. In the process, we obtain stronger results regarding $\mathrm{Spec}(A_2) = \mathrm{Spec}(32_{65})$. Namely, we show that 1024 is in the spectrum. We also obtain improvements to the lower bound. The trivial lower bound is $f(n) \geq n^2 + 2n + 3$. We show that $f(n) \geq 2n^2 + \Omega(n\sqrt{\ln n})$, which holds for all $n \geq 2$. For smaller values of n, we obtain a slightly better lower bound of $f(n) \geq 2n^2 + 6n + 6$.

6.1 An upper bound on $f(n)$

In this section, we give a representation of 32_{65} over 1024 points, and then generalize to give representations of A_n for all n. Consider $G = (\mathbb{Z}/2\mathbb{Z})^{10}$, and consider the elements as bitstrings. Define:

$$R = \{x \in G : x \text{ has between one and six 1s}\}, \text{ and}$$
$$B = \{x \in G : x \text{ has at least seven 1s}\}.$$

This defines a group representation of 6_7, which is a subalgebra of 32_{65}. There exists a way of splitting B into B_1 and B_2 so that:

- $R + B_i = G \setminus \{0\}$, $i = 1, 2$;
- $B_i + B_i = R \cup \{0\}$, $i = 1, 2$;
- $B_1 + B_2 = R$.

This yields a group representation of 32_{65} over $2^{10} = 1024$ points, improving the previous smallest-known representation over $\binom{14}{7} = 3432$ points [11]. We note that while the representation given here is smaller, the representation over 3432 points in [11] has a nice, compact description.

J.F. Alm found the split with the aid of a computer search. He checked several million random splits. None of them worked, but some got "close". He took one of the close ones and tinkered with it for about three hours until it worked. The curious can view the process in the Jupyter notebook `32_65 splitting.ipynb` at `https://github.com/algorithmachine/RA-32-of-65`. The following Python 3 code can be used to verify that the given split yields a representation. (Bitstrings are encoded as integers between 0 and 1023.)

```
def s(X,Y): return {x^y for x in X for y in Y}
G = set(range(1024)); id = {0}; di = G-id
b = {127, 223, 239, 251, 253, 255, 367,
375, 381, 382, 431, 443, 446, 471, 475,
477, 478, 487, 491, 494, 499, 505, 509,
607, 635, 637, 639, 701, 702, 703, 719,
727, 733, 734, 743, 750, 751, 758, 763,
766, 815, 823, 827, 829, 847, 859, 862,
863, 877, 879, 883, 886, 887, 890, 893,
894, 919, 923, 925, 927, 935, 941, 943,
949, 950, 953, 954, 958, 979, 981, 982,
990, 991, 995, 1001, 1002, 1003, 1005,
1011, 1012, 1013, 1014, 1015, 1016,
1017, 1019, 1021, 1022}
a = s(b,b)-id; c = di-a-b
print ( s(a,a)==G, s(a,b)==s(a,c)==di,
s(b,b)==s(c,c)==a|id, s(b,c)==a )
```

We generalize this argument as follows:

Theorem 6.1.1. *For all $n \geq 2$, A_n is representable over $(\mathbb{Z}/2\mathbb{Z})^{3k+1}$ for sufficiently large $k \in \mathbb{N}$. In particular, for $n \geq 14$, it suffices to take $k = n$.*

Remark 6.1.2. Theorem 6.1.1 tells us that $f(n)$ is at most exponential in n. In contrast, the most one could obtain from [7] was that $f(n)$ was bounded above by (roughly) $\binom{15n^2}{n}$. See Figure 6.1.

Proof. We have already shown that for $k = 3$, A_2 can be realized over $(\mathbb{Z}/2\mathbb{Z})^{3k+1}$. We now argue that for $n \geq 3$, A_n can be realized over $(\mathbb{Z}/2\mathbb{Z})^{3k+1}$ for sufficiently large $k \in \mathbb{N}$. Our approach is to use the probabilistic method to show that, given a large enough representation of the relation algebra 6_7 over $(\mathbb{Z}/2\mathbb{Z})^{3k+1}$, the atom b can be partitioned into n parts, as A_n is obtained from 6_7 by splitting, as in [15].

Consider $G = (\mathbb{Z}/2\mathbb{Z})^{3k+1}$. For $x \in G$, let $x(i)$ denote the i^{th} coordinate of x. Let $|x|$ denote $|\{i : x(i) = 1\}|$. Denote $\text{supp}(x) = \{i : x(i) = 1\}$ to be the support of x. The key idea is the following partition of $G \setminus \{0\}$ into two sets R and B.

Let:

$$R = \{x \in G : 1 \leq |x| \leq 2k\} \text{ and}$$

$$B = \{x \in G : 2k+1 \leq |x| \leq 3k+1\}.$$

Then $R+R = G$, $R+B = G \setminus \{0\}$, and $B+B = R \cup \{0\}$. As we will see below, B is a sum-free set with high additive energy.

We now split both the "red" and "blue" atoms of 6_7 into n atoms and find a representation over a finite set. Namely, we split R and B into n parts $R_1, \ldots, R_n$ and $B_1, \ldots, B_n$ uniformly at random. We need to count the "witnesses" to the "needs" of each element. We will show that each need is witnessed at least 2^k times. Consider the following cases.

- **Case 1:** We count witnesses for $R \subseteq B+B$. Let $z \in R$, and denote $\ell := |z|$. We consider two sub-cases: whether $1 \leq \ell \leq k$, and whether $k+1 \leq \ell \leq 2k$.

  * **Case 1.1:** Suppose first that $1 \leq \ell \leq k$. We construct x, y randomly so that $z = x+y$. For each $i \in \text{supp}(z)$, we choose uniformly at random whether:

  $$x(i) = 1 \text{ and } y(i) = 0$$

  $$\text{OR}$$

  $$x(i) = 0 \text{ and } y(i) = 1.$$

  As $|z| = \ell$, this yields 2^ℓ possible selections. For the $k-\ell$ left-most indices $j \notin \text{supp}(z)$, we choose uniformly at random whether:

  $$x(j) = 1 = y(j)$$

  $$\text{OR}$$

  $$x(j) = 0 = y(j).$$

  As there are $k-\ell$ positions, there are $2^{k-\ell}$ possible selections. For the remaining $2k+1$ positions i, we let $x(i) = 1 = y(i)$. Thus, we have that $x, y \in B$. By the rule of product, we obtain $2^{\ell} \cdot 2^{k-\ell} = 2^k$ possible selections. It follows that there are at least 2^k witnesses for z.

- **Case 1.2:** Suppose now that $k+1 \leq \ell \leq 2k$. For the k *least* indices $i \in \text{supp}(z)$, we choose uniformly at random whether:

$$x(i) = 1 \text{ and } y(i) = 0$$

OR

$$x(i) = 0 \text{ and } y(i) = 1$$

For the remaining $\ell - k$ indices $i \in \text{supp}(z)$, let $x(i) = 1$ and $y(i) = 0$, or $x(i) = 0$ and $y(i) = 1$ in such a way that ensures that both x and y have at least $\ell - k$ 1's in coordinates in supp(z). Then for all indices $j \notin \text{supp}(z)$, let $x(j) = 1 = y(j)$. As there are 2^k possible selections for the first k coordinates, there are at least 2^k witnesses.

It follows that if $z \in R$, there are at least 2^k ways to witness z as the sum $x + y$, where $x, y \in B$.

- **Case 2:** Now let us consider witnesses to $B \subseteq B + R$. Let $z \in B$, and denote $\ell := |z|$. By the definition of B, we have that $2k + 1 \leq \ell \leq 3k$. We randomly construct $x \in B$, $y \in R$ so that $z = x + y$.

For the $2k+1$ indices $i \in \text{supp}(z)$ of *least* index, set $x(i) = 1$ and $y(i) = 0$. For the remaining $\ell - (2k+1)$ indices $i \in \text{supp}(z)$, we choose uniformly at random whether:

$$x(i) = 1 \text{ and } y(i) = 0$$

OR

$$x(i) = 0 \text{ and } y(i) = 1.$$

For each index $j \notin \text{supp}(z)$, we choose uniformly at random whether:

$$x(j) = 1 = y(j)$$

OR

$$x(j) = 0 = y(j).$$

Again, there were k random selections, so we have at least 2^k witnesses.

- **Case 3:** Next, let us consider witnesses to $B \subseteq R + R$. Let $z \in B$. We construct $x, y \in R$ so that $z = x + y$. For every $j \notin \text{supp}(z)$, set $x(j) = 0 = y(j)$. This is to ensure that $x, y \in R$. For the smallest k indices $i \in \text{supp}(z)$, we choose uniformly at random whether:

  $$x(i) = 1 \text{ and } y(i) = 0$$

  OR

  $$x(i) = 0 \text{ and } y(i) = 1.$$

  For the remaining indices $i \in \text{supp}(z)$, we choose uniformly at random whether:

  $$x(i) = 1 \text{ and } y(i) = 0$$

  OR

  $$x(i) = 0 \text{ and } y(i) = 1$$

  in such a way that ensures that neither x nor y receives more than $2k$ 1's. Clearly, there are at least 2^k witnesses.

- **Case 4:** Now we consider witnesses for $R \subseteq B + R$.

  Let $z \in R$, and denote $\ell := |z|$. We build $x \in B$, $y \in R$ so that $z = x + y$. We consider the following sub-cases, namely whether $1 \leq \ell \leq k$, and whether $k + 1 \leq \ell \leq 2k$.

  ∗ **Case 4.1:** First, consider the case where $1 \leq \ell \leq k$. For $i \in \text{supp}(z)$, set $x(i) = 1$ and $y(i) = 0$. For $j \notin \text{supp}(z)$, choose $2k + 1 - \ell$ of the $3k + 1 - \ell$ indices, and set

$x(j) = 1 = y(j)$. Set all others to $x(j) = 0 = y(j)$. Since $\binom{3k+1-\ell}{2k+1-\ell} \geq \binom{2k}{k} > 2^k$, we have at least 2^k witnesses.

- **Case 4.2:** Now consider the case where $k+1 \leq n \leq 2k$. For the smallest k indices $i \in \text{supp}(z)$, set $x(i) = 1$ and $y(i) = 0$. For the remaining $\ell - k$ indices $i \in \text{supp}(z)$, we choose uniformly at random whether:

$$x(i) = 1 \text{ and } y(j) = 0$$

OR

$$x(i) = 0 \text{ and } y(j) = 1.$$

Now we choose $k+1$ of the remaining $3k+1-\ell$ indices $j \notin \text{supp}(z)$. There are $\binom{3k+1-\ell}{k+1}$ choices, which ranges between $\binom{k+1}{k+1}$ and $\binom{2k+1}{k+1}$. Therefore there are at least $2^{\ell-k} \cdot \binom{3k+1-\ell}{k+1}$ witnesses. It is not hard to check that for $0 \leq N \leq k$, $\binom{k+1+N}{k+1} > 2^N$, and therefore $2^{\ell-k} \cdot \binom{3k+1-\ell}{k+1} > 2^{\ell-k} \cdot 2^{2k-\ell} = 2^k$.

- **Case 5:** Finally, we consider witnesses for $R \subseteq R + R$. Let $z \in R$, and denote $\ell := |z|$. We construct $x, y \in R$ so that $z = x + y$. We consider the following cases: whether $1 \leq \ell \leq k$, and whether $k+1 \leq \ell \leq 2k$.

  - **Case 5.1:** First, consider the case where $1 \leq \ell \leq k$. For each $i \in \text{supp}(z)$, set $x(i) = 1$ and $y(i) = 0$. Then for the smallest $2k$ indices outside of $\text{supp}(z)$, choose k of then. For each such selected j, set $x(j) = 1 = y(j)$, and $x(j) = 0 = y(j)$ otherwise. This gives at least $\binom{2k}{k} > 2^k$ witnesses.

  - **Case 5.2:** We next consider the case where $k+1 \leq \ell \leq 2k$. For each $i \in \text{supp}(z)$, we choose uniformly at random whether:

$$x(i) = 1 \text{ and } y(i) = 0$$

OR

$$x(i) = 0 \text{ and } y(i) = 1.$$

This gives $2^\ell > 2^k$ witnesses.

Now we are ready to compute the probability that our random partition $R_1, \ldots, R_n$ and $B_1, \ldots, B_n$ *fails* to be a representation. Let $z \in (\mathbb{Z}/2\mathbb{Z})^{3k+1} \smallsetminus \{\mathbf{0}\}$. If $z \in R_i$, then z has $3n^2$ "needs":

- $\forall i, j\ z \in R_i + R_j$
- $\forall i, j\ z \in R_i + B_j$
- $\forall i, j\ z \in B_i + B_j$.

If $z \in B_j$, then z has $2n^2$ "needs":

- $\forall i, j\ z \in R_i + R_j$
- $\forall i, j\ z \in R_i + B_j$.

So $3n^2$ is a bound on the number of "needs". Fix z, and let $x, y \in G$ such that $x + y = z$. The probability that the edge xy witnesses a fixed need is $1/n^2$. So the probability that the edge xy does *not* witness a fixed need is $(1 - 1/n^2)$. For a particular need, there are at least 2^k witnesses. As we color the edges uniformly at random, the probability that a fixed need of z is unsatisfied is at most:

$$\left(1 - \frac{1}{n^2}\right)^{2^k}.$$

As z has at most $3n^2$ needs, we have that

$$\Pr[z \text{ has an unsatisfied need}] \leq 3n^2 \left(1 - \frac{1}{n^2}\right)^{2^k}.$$

Thus

$$\begin{aligned}
\Pr[\exists z \text{ with an unsatisfied need}] &\leq \sum_z 3n^2 \left(1 - \frac{1}{n^2}\right)^{2^k} && (6.1)\\
&= 2^{3k+1} \cdot 3n^2 \left(1 - \frac{1}{n^2}\right)^{2^k} && (6.2)\\
&\leq 2^{3(k+1)} \cdot n^2 \left(1 - \frac{1}{n^2}\right)^{2^k}. && (6.3)
\end{aligned}$$

We want (6.3) to be less than 1, which is equivalent to its logarithm being less than zero:

$$\log\left(2^{3(k+1)}\cdot n^2\cdot\left(1-\frac{1}{n^2}\right)^{2^k}\right)<0 \iff 3(k+1)\log 2+2\log n+2^k\log\left(\frac{n^2-1}{n^2}\right)<0$$
$$\iff 3(k+1)\log 2+2\log n<2^k\log\left(\frac{n^2}{n^2-1}\right)$$

Now assuming $3\le n\le k$, we have that

$$\begin{aligned}3(k+1)\log 2+2\log n&<3(k+1)+k\\&\le 5k.\end{aligned}$$

Thus

$$\begin{aligned}2^k\cdot\log\left(\frac{n^2}{n^2-1}\right)&=2^k[\log(n^2)-\log(n^2-1)]\\&>2^k\cdot\frac{1}{n^2},\end{aligned}$$

where the last inequality is due to the fact that $\log(t+1)-\log(t)<1/t$, which follow by the concavity of log. So we need $5k<\frac{2^k}{n^2}$. Setting $k=n$, we have $5n^3<2^n$, which holds for all $n\ge 14$. Hence taking $k=n$ gives a non-zero probability that a random partition yields a representation. □

The construction in the proof of Theorem 6.1.1 provides the bound $f(n)\le 2^{3n+1}$ for $n\ge 14$. By fine-tuning our choice of k, we obtain polynomial bounds on $f(n)$. Let $k=(2+o(1))\log(n)$. We have by (6.2) that

$$\begin{aligned}\Pr[\exists z\text{ with an unsatisfied need}]&\le 2^{3k+1}\cdot 3n^2\left(1-\frac{1}{n^2}\right)^{2^k}\\&=2^{3(2+o(1))\log(n)+1}\cdot 3n^2\cdot\left(1-\frac{1}{n^2}\right)^{2^{(2+o(1))\log(n)}}\\&=2n^{6+o(1)}\cdot 3n^2\cdot\left(1-\frac{1}{n^2}\right)^{2^{(2+o(1))\log(n)}}\\&=6n^{8+o(1)}\cdot\left(1-\frac{1}{n^2}\right)^{2^{(2+o(1))\log(n)}}.\end{aligned}$$

Now we have that

$$\lim_{n\to\infty}6n^{8+o(1)}\cdot\left(1-\frac{1}{n^2}\right)^{2^{(2+o(1))\log(n)}}=0. \tag{6.4}$$

So choosing $k=(2+o(1))\log(n)$ yields the following:

Theorem 6.1.3. *For n sufficiently large, we have that $f(n) \leq 2n^{6+o(1)}$.*

We note that the threshold n_0 for which Theorem 6.1.3 applies is quite large. For instance, choosing $k = 3\log(n)$ yields that $f(n) \leq 2n^9$, which holds for all $n \geq 91$. So in choosing $k = (2 + o(1))\log(n)$, the bound of $f(n) \leq 2n^{6+o(1)}$ holds for all $n \geq n_0 \geq 91$, where n_0 depends on $o(1)$. This contrasts with the bound in Theorem 6.1.1, which holds for all $n \geq 14$. Furthermore, calibrating our choice of $k = c\log(n)$ failed to yield improvements on $f(3) \leq 2^{16}$ and $f(4) \leq 2^{19}$.

It is natural to ask whether modifying our choice of k in this construction will yield additional improvements in the upper bound for $f(n)$. If we take $k = 2\log(n)$ rather than $k = (2+o(1))\log(n)$, we have that

$$\lim_{n\to\infty} 6n^8 \cdot \left(1 - \frac{1}{n^2}\right)^{2^{2\log(n)}} = \infty.$$

The key reason behind this is that

$$\lim_{n\to\infty} \left(1 - \frac{1}{n^2}\right)^{2^{2\log(n)}} = \lim_{n\to\infty} \left(1 - \frac{1}{n^2}\right)^{n^2} = \frac{1}{e}.$$

This suggests that further analyzing the Boolean cube is unlikely to yield additional improvements on the upper bound for $f(n)$.

We also note that (6.4) yields that as $n \to \infty$, the probability that there exists $z \in (\mathbb{Z}/2\mathbb{Z})^{3k+1}$ with an unsatisfied need goes to 0. So with high probability, a random partition yields a representation of A_n. We record this observation with the following corollary.

Corollary 6.1.4. *Suppose that we split both the "red" and "blue" atoms of 6_7 into n atoms, as in the proof of Theorem 6.1.1. Namely, we split R and B into n parts $R_1, \ldots, R_n$ and $B_1, \ldots, B_n$ uniformly at random. With high probability, we have that such a random split is a representation of a relation algebra containing a subalgebra isomorphic to A_n.*

6.2 A lower bound for A_n

In this section, we consider representations of 32_{65} as edge-colorings of K_n with all mandatory triangles present and no all-blue triangles. Note that blue triangles are forbidden even if they contain

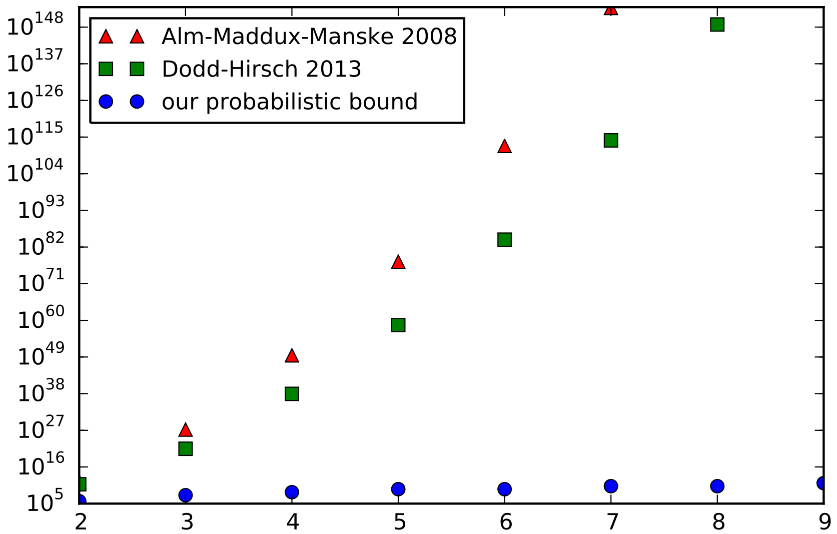

Figure 6.1: Upper bound on $f(n)$ vs. n

edges of differing shades of blue. In other words, every triangle must contain a red edge.

We now make our representation precise. Let $\rho : 32_{65} \to \text{Powerset}(U \times U)$, where $U = \{x_0, \dots, x_{n-1}\}$, be a representation. Then label the vertices of K_n with $\{x_0, \dots, x_{n-1}\}$, and let the color of edge $x_i x_j$ be the atom z such that $(x_i, x_j) \in \rho(z)$. We begin with some preliminary lemmas, that illustrate our counting strategy, including how we incorporate the classical Ramsey number $R(3, k)$ to obtain improvements. Ultimately, we will generalize this technique to obtain Lemma 6.2.8 and Proposition 6.2.14.

Lemma 6.2.1. $\text{Spec}(32_{65}) \subseteq \{11, \dots\} \cup \{\omega\}$.

Proof. There must be some red edge $x_0 x_1$. Any red edge has nine "needs". There must be nine points that witness these needs, which together with x_0 and x_1 make a total of 11 points. (See Figure 6.2.) □

We can easily obtain a slight improvement using the classical Ramsey number $R(4, 3)$.

Lemma 6.2.2. $\min \text{Spec}(32_{65}) \geq 12$.

Proof. We know that at least 11 points are required. Since $R(4, 3) = 9$, and there are no all-blue triangles, there must be a red K_4. Let $x_0 x_1$ be an edge in this red K_4. Then $x_0 x_1$ must have its red-red need met twice, hence there must be ten points besides x_0 and x_1. □

Lemma 6.2.3. *In any representation of 32_{65}, for every red edge there is a red K_4 that is vertex-disjoint from it. In particular, off of every red edge $x_0 x_1$ one can find the configuration depicted in Figure 6.3.*

Proof. Let $x_0 x_1$ be red, with witnesses to all needs as in Figure 6.2. Then $\{x_2, x_3, x_4, x_5\}$ induce a red K_4, since any blue edge among them would create an all-blue triangle with x_0 (and also with x_1). Furthermore, any edge running from any of x_2, x_3, x_4, x_5 to any of x_6, x_7, x_8, x_9 must be red, since any such blue edge would create an all-blue triangle with either x_0 (for x_7 and x_9) or x_1 (for x_6 and x_8). Thus we have the configuration depicted in Figure 6.3. □

Lemma 6.2.4. $12, 13, 14, 15, 16 \notin \text{Spec}(32_{65})$.

Proof. Consider the configuration depicted in Figure 6.3. The edge x_2x_5 is red. Then x_0 and x_1 both witness the light-blue-dark-blue need, while x_3, x_4, x_6, x_7, x_8 and x_9 all witness the red-red need. There are seven needs yet unsatisfied. The remaining vertex x_{10} could witness some need, but vertices x_{11} through x_{16} will have to be added. Thus there are at least 17 points. See Figure 6.4. □

Lemma 6.2.4 generalizes nicely as follows.

Theorem 6.2.5. *For all* n, $f(n) \geq 2n^2 + 4n + 1$.

Note that the trivial bound is $n^2 + 2n + 3$, roughly half the bound in Theorem 6.2.5.

Proof. Call the shades of blue b_1 through b_n. Fix a red edge x_0x_1. Let BB denote the set of vertices that witness a blue-blue need for x_0x_1, and let RB denote the set of vertices that witness either a red-blue need or a blue-red need for x_0x_1. BB induces a red clique, and all edges from BB to RB are red. Note that $|BB| = n^2$ Note that $|BB| = n^2$ and similarly $|RB| = 2n$. This gives the trivial lower bound of $n^2 + 2n + 3$.

Let $u \in BB$ witness b_1-b_1 for x_0x_1 and let $v \in BB$ witness b_2-b_2 for x_0x_1. The edge uv is red, hence has $(n+1)^2$ needs. Both x_0 and x_1 witness the same b_1-b_2 need, and all points in BB and RB (besides u and v) witness the red-red need. Hence there must be at least $(n+1)^2 - 2$ points outside of $\{x_0, x_1\} \cup BB \cup RB$. Hence there are at least $2 + n^2 + 2n + (n+1)^2 - 2 = 2n^2 + 4n + 1$ points. □

Remark 6.2.6. Note that if two points u, v satisfy a red-blue need for x_0x_1, then uv is necessarily red. Otherwise, uvx_1 would form a blue triangle. There are n points that satisfy the red-blue need for x_0x_1. As the points in BB form a red clique of size n^2, we obtain the following.

Corollary 6.2.7. *In any representation of* A_n, *the clique number of the red subgraph of the underlying graph of the representation is at least* $n^2 + n$.

We can further improve the lower bound on $f(n)$ using the Ramsey number $R(3, k)$ to analyze the size of the red clique containing x_0x_1.

Lemma 6.2.8. *$f(n) \geq 2n^2 + \Omega(n\sqrt{\ln(n)})$.*

Proof. We first recall the trivial lower bound from Theorem 6.2.5, as we will need to build on this initial structure. Call the shades of blue b_1 through b_n. Fix a red edge x_0x_1. Let BB denote the set of vertices that witness a blue-blue need for x_0x_1, and let RB denote the set of vertices that witness either a red-blue need or a blue-red need for x_0x_1. BB induces a red clique, and all edges from BB to RB are red. Note that $|BB| = n^2$ and $|RB| = 2n$. This gives the trivial lower bound of $n^2 + 2n + 3$.

Let $u \in BB$ witness b_1-b_1 for x_0x_1 and let $v \in BB$ witness b_2-b_2 for x_0x_1. The edge uv is red, hence has $(n+1)^2$ needs. Both x_0 and x_1 witness the same b_1-b_2 need, and all points in BB and RB (besides u and v) witness the red-red need. Hence there must be at least $(n+1)^2 - 2$ points outside of $\{x_0, x_1\} \cup BB \cup RB$. Hence there are at least $2 + n^2 + 2n + (n+1)^2 - 2 = 2n^2 + 4n + 1$ points.

Up to this point, the proof has been identical to Theorem 6.2.5. We now use Ramsey theory to strengthen the lower bound. Griggs showed that $R(3, m) < 2.4 \cdot m^2/\ln(m)$ [89]. So we have that for every n sufficiently large, there exists a $k > 0$ such that:

$$f(n) \geq 2n^2 + 4n + 1 > 2n^2 \geq 2.4 \cdot (kn)^2/\ln(kn) \geq R(3, kn). \tag{6.5}$$

Our goal is to find the largest possible k. As $f(n) \geq R(3, kn)$ and there are no blue triangles, we have by Ramsey's theorem that there exists a red clique of size kn. Considering

$$2n^2 \geq 2.4 \cdot (kn)^2/\ln(kn),$$

we obtain that

$$(5/6)\ln(kn) \geq k^2.$$

Taking $k = \sqrt{(5/6)\ln(n)}$ works for $n \geq 4$. Observe that for $n \geq 2$, we may take

$$k = \sqrt{(5/6)\ln(n) + (5/12)\ln(\ln(n)) - ((5/12)\ln(6/5) - \epsilon)},$$

where $\epsilon > 0$ (one may check this, such as with a computer algebra system). So there exists a red clique of size $\Omega(n\sqrt{\ln(n)})$. We may iterate on the argument outlined in the first two paragraphs, taking x_0x_1 to be on this red clique. As all the points in this red clique witness a red-red need for x_0x_1, the sets BB and RB must be disjoint from the red clique. Thus, by iterating on the argument in the first two paragraphs, we obtain

$$f(n) \geq 2n^2 + n \cdot \sqrt{(5/6)\ln(n) + (5/12)\ln(\ln(n)) - ((5/12)\ln(6/5) - \epsilon)} + 4n + 1.$$

The result now follows. □

Remark 6.2.9. We note that the red clique of size $\Omega(n\sqrt{\ln(n)})$ containing x_0x_1 is disjoint from the red clique of size $n^2 + n$ prescribed by [8, Corollary 18].

We also obtain a second lower bound, which works better in the case of small n.

Lemma 6.2.10. *For $n \geq 2$, we have that $f(n) \geq 2n^2 + 6n$.*

Proof. We have by Theorem 6.2.5 that $f(n) \geq 2n^2 + 4n + 1$. Now observe that:

$$2n^2 + 4n + 1 \geq \binom{3 + 2n + 1 - 2}{3 - 1} = \binom{2n+2}{2} \geq R(3, 2n + 1).$$

As there are no blue triangles, any representation of A_n has a red clique of size $2n + 1$. Take x_0x_1 to be a red edge on this clique. Iterating on the argument in the first two paragraphs yields an additional $2n - 1$ points. Thus, $f(n) \geq 2n^2 + 6n$. □

Remark 6.2.11. It follows that if $k < 21$, then $k \notin \text{Spec}(32_{65})$. This provides a combinatorial proof of a fact that was previously verified using a SAT solver in [8].

It is possible to further strengthen Lemma 6.2.10. For $n \geq 3$, we have that $2n^2 + 6n \geq \binom{2n+3}{2} \geq R(3, 2n + 2)$. So we may add an additional point to our red clique containing x_0x_1.

Proposition 6.2.12. *For $n \geq 3$, $f(n) \geq 2n^2 + 6n + 1$.*

In the case when $n = 3$, we have that $f(3) \geq 37$. As $R(3, 9) = 36$, we have that there is a red clique on 10 vertices. So we in fact get 1 additional point on the red clique containing x_0x_1. Thus, we obtain that $f(3) \geq 38$. Furthermore, as $f(n)$ is monotone, this analysis holds for all $n \geq 3$. So we obtain the following improvement to the above proposition.

Proposition 6.2.13. *For $n \geq 3$, $f(n) \geq 2n^2 + 6n + 2$.*

Now we note that $2 \cdot 2^2 + 6(2) + 1 = 21$. It was previously shown that if $k < 26$, then $k \notin \text{Spec}(A_2) = \text{Spec}(32_{65})$ [8]. Thus, we have the following.

Proposition 6.2.14. *For all $n \geq 2$, $f(n) \geq 2n^2 + 6n + 2$.*

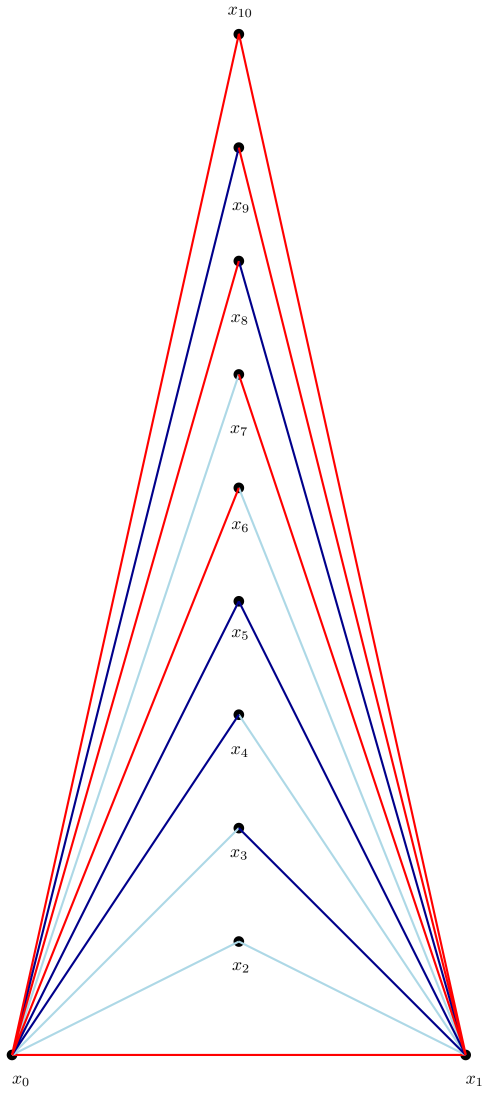

Figure 6.2: The needs of a red edge

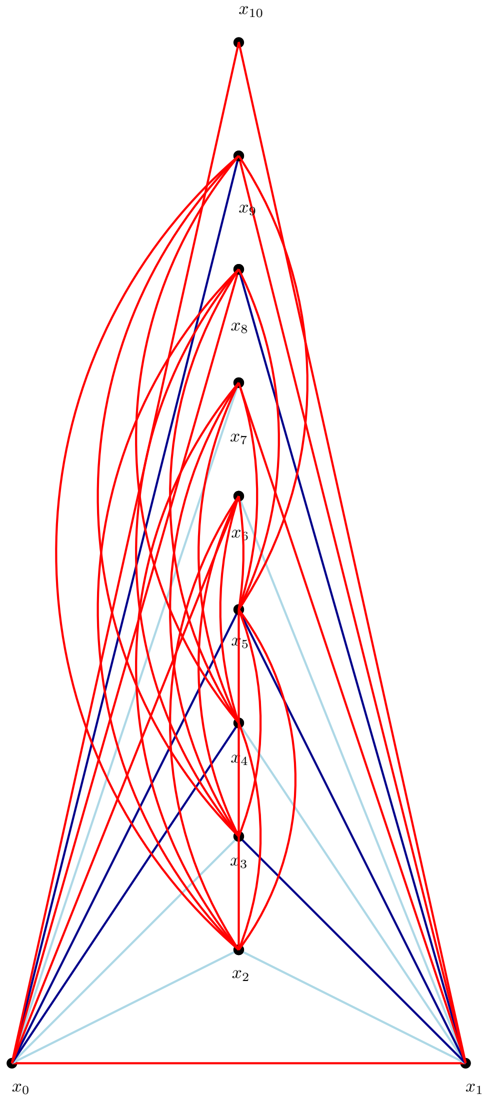

Figure 6.3: Subgraph which must appear off of any red edge

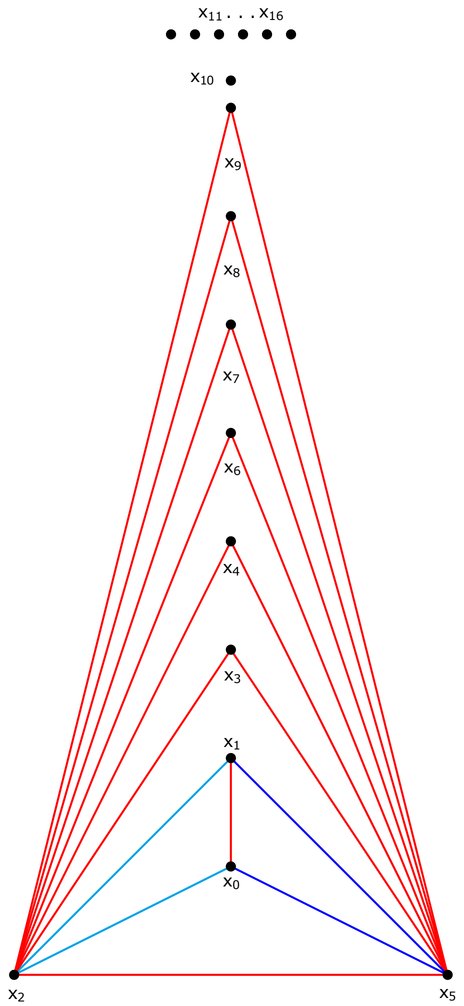

Figure 6.4: witnesses to the needs of x_2x_5

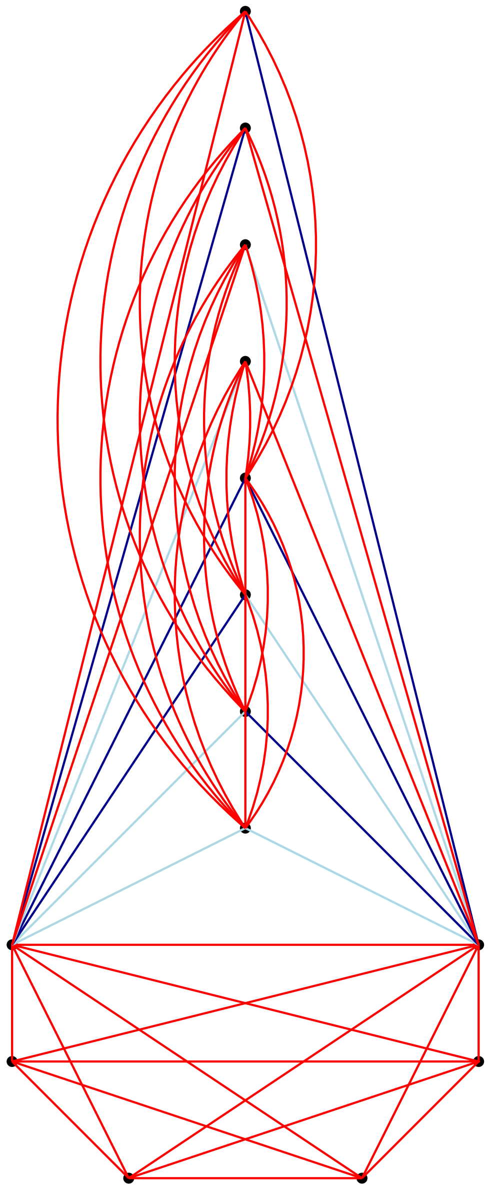

Figure 6.5: Mandatory subgraph with red K_6

Chapter 7

Conclusion and Open Problems

In this thesis, we investigated the combinatorial and logical complexities of several algebraic structures, including those arising from Latin squares and relation algebras. In Chapter 3, we investigated both the Weisfeiler–Leman dimension and iteration number [48] for several families of groups, including (i) coprime extensions $H \ltimes N$, where H is $O(1)$-generated and N is Abelian, and (ii) direct product decompositions. As a consequence, we showed that Weisfeiler–Leman serves as an L-isomorphism test for this family of coprime extensions, and that Weisfeiler–Leman can implicitly compute direct product decompositions in $O(\log n)$ rounds. Using the individualize-and-refine paradigm, we also showed that $\mathsf{quasiSAC}^1$ circuits with size $n^{O(\log \log n)}$ suffice to identify groups without Abelian normal subgroups. We showed that if furthermore the socle has $O(\log n/ \log \log n)$ non-Abelian simple direct factors, then all isomorphisms can be listed in FL. This improves upon the previous bound of FP [29].

We also considered the weaker *count-free* variant of Weisfeiler–Leman, where we showed that $\Omega(\log n)$-WL is required to identify even Abelian groups. As a consequence, we obtain that FO does not capture all polynomial-time computable queries on even Abelian groups. Nonetheless, we successfully leveraged $O(\log \log n)$ rounds of the count-free WL Version I algorithm in tandem with bounded non-determinisim and a single $\mathsf{Majority}$ gate to obtain a $\beta_1\mathsf{MAC}^0(\mathsf{FOLL})$ upper bound for isomorphism testing of Abelian groups. This improves the previous bound of $\mathsf{TC}^0(\mathsf{FOLL})$ [57].

Our work in Chapter 3 leaves open the question as to whether groups without Abelian normal subgroups have bounded (or even $o(\log n)$) WL-dimension. We note that Weisfeiler–Leman corresponds to the first Ehrenfeucht–Fraïssé game in Hella's hierarchy [109, 110, 48]. We thus investigated the second Ehrenfeucht–Fraïssé game in Hella's hierarchy, which we refer to as the 2-*ary* game. While this game trivially handles Graph Isomorphism, it is not clear as to whether this game suffices to handle Group Isomorphism. Nonetheless, we showed that if G has no Abelian normal subgroups and $H \not\cong G$ is arbitrary, then Spoiler has a winning strategy in the 2-ary game with $O(1)$ pebbles and $O(1)$ rounds. Hella [109, 110] previously established that the 2-ary game is equivalent to the logic $\mathsf{FO}(Q)$, where Q is the set of all binary quantifiers. Thus, all groups without Abelian normal subgroups are identified by an $\mathsf{FO}(Q)$ formula with $O(1)$ variables and quantifier depth $O(1)$. Finally, we exhibited a novel Weisfeiler–Leman characterization of the 2-ary game, which we refer to as the 2-*ary WL*.

In Chapter 5, we showed that for any $i, c \geq 0$, GI is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to several isomorphism problems characterized by the generator enumeration technique, including Latin Square Isotopy, isomorphism testing of nets (which includes affine and projective planes), and isomorphism testing of Steiner $(t, t+1)$-designs. As a corollary, we obtained that GI is not $\beta_i\mathsf{FO}((\log \log n)^c)$-reducible to isomorphism testing of Latin square graphs, k-net graphs (for fixed k), and the block-intersection graphs arising from Steiner triple systems.

Finally, in Chapter 6, we investigated the minimum-sized representation of the relation algebra A_n (denoted $f(n)$), obtained by splitting the non-flexible diversity atom of 6_7 into n symmetric atoms. We showed that $2n^2 + \Omega(n\sqrt{\log n}) \leq f(n) \leq 2n^{6+o(1)}$. In the special case of $A_2 = 32_{65}$, we showed that $26 \leq f(2) \leq 1024$.

We conclude with some open problems.

7.1 Group Isomorphism

Question 7.1.1. What is the (1-ary) Weisfeiler–Leman dimension of groups without Abelian normal subgroups?

It would be of interest to address this question even in the non-permuting case when $G =$

PKer(G). Alternatively, establish an upper bound of $O(\log \log n)$ for the WL dimension of semisimple groups. These questions would form the basis of a WL analogue of [29], without needing individualize-and-refine.

In general, if an uncolored class of graphs is identified by WL, then so is the corresponding class of colored graphs. So if constant-dimensional WL identifies a class of graphs, it may readily be extended to an efficient canonization procedure (c.f., [102]). In the case of groups, it is not clear whether WL easily identifies colored variants. To this end, we ask the following.

Question 7.1.2. Does constant-dimensional (1-ary) Weisfeiler–Leman identify every colored Abelian group?

Our work on the 2-ary game also leaves open some interesting questions.

Question 7.1.3. Can the constant-dimensional 2-ary Wesifeiler–Leman algorithm be implemented in time $n^{o(\log n)}$?

Question 7.1.4. Show that the second Ehrenfeucht–Fraïssé game in Hella's hierarchy can identify coprime extensions of the form $H \ltimes N$ with both H, N Abelian (the analogue of [167]). More generally, an analogue of Babai–Qiao [34] would be to show that when $|H|, |N|$ are coprime and N is Abelian, that Spoiler can distinguish $H \ltimes N$ from any non-isomorphic group using a constant number of pebbles that is no more than that which is required to identify H (or the maximum of that of H and a constant independent of N, H).

Question 7.1.5. Let $p > 2$ be prime, and let G be a p-group with bounded genus. Show that in the second Ehrenfeucht–Fraïssé game in Hella's hierarchy, Spoiler has a winning strategy using a constant number of pebbles. This is a descriptive complexity analogue of [51, 125]. It would even be of interest to start with the case where G has bounded genus over a field extension $K/\mathbb{F}_p$ of bounded degree.

In the setting of groups, Hella's hierarchy collapses to some $q \leq 3$, since 3-ary WL can identify all ternary relational structures, including groups. It remains open to determine whether this hierarchy collapses further to either $q = 1$ or $q = 2$. Even if it does not collapse, it would also be of interest to determine whether the 1-ary and 2-ary games are equivalent. Algorithmically, this is equivalent to determining whether 1-ary and 2-ary WL are have the same distinguishing power.

Question 7.1.6. Does there exist an infinite family of non-isomorphic pairs of groups $\{(G_n, H_n)\}$ for which Spoiler requires $\omega(1)$ pebbles to distinguish G_n from H_n? We ask this question for the Ehrenfeucht–Fraïssé games at both the first and second levels of Hella's hierarchy.

Recall that the game at the first level of Hella's hierarchy is equivalent to Weisfeiler–Leman [55, 109, 110], and so a lower bound against either of these games provides a lower bound against Weisfeiler–Leman.

More generally, it would also be of interest to investigate Hella's hierarchy on higher arity structures. For a q-ary relational structure, the q-ary pebble game suffices to decide isomorphism. Are there interesting, natural classes of higher arity structures for which Hella's hierarchy collapses further to some level $q' < q$?

Finally, we wish to highlight a question that essentially goes back to [57], who showed that GpI cannot be hard under AC^0 reductions for any class containing Parity. In Theorem 3.7.9, we showed that count-free WL requires dimension $\geq \Omega(\log(n))$ to identify even Abelian groups. This shows that this particular, natural method does not put GpI into $\mathsf{FO}(\text{poly} \log \log n)$, though it does not actually prove GpI $\notin \mathsf{FO}(\text{poly} \log \log n)$, since we cannot rule out clever bit manipulations of the Cayley (multiplication) tables. While we think the latter lower bound would be of significant interest, we think even the following question is interesting:

Question 7.1.7. Show that GpI does not belong to (uniform) AC^0.

7.2 Strongly Regular Graphs

In Proposition 5.2.6, we showed that a Steiner 2-design can be recovered from its block-incidence graph in AC^0 when the block size is bounded. Otherwise, the procedure is TC^0-computable, as we need to distinguish vertices by their degrees. As a step towards ruling out $\beta_i\mathsf{FO}((\log \log n)^c)$

reductions from GI, it would be of interest to show that we can recover a Steiner 2-design from its block-incidence graph in a complexity class that cannot compute Parity. To this end, we ask the following.

Question 7.2.1. Can Steiner 2-designs of unbounded block size be recovered from their block-incidence graphs in AC^0?

It would also be of interest to find additional families of Steiner 2-designs where the isomorphism problem belongs to $\beta_i\mathsf{FO}((\log\log n)^c)$. As a starting point, we ask the following.

Question 7.2.2. For Steiner 2-designs with bounded block size, can we decide isomorphism in $\beta_2\mathsf{FOLL}$?

While a Steiner 2-design $\mathcal{D}$ admits generating set S of $O(\log v)$ points [33], we have that $\text{Aut}_S(\mathcal{D})$ may in general be non-trivial. This is the key barrier for the techniques here.

Babai & Wilmes [36] and Chen, Sun, & Teng [58] independently showed that Steiner 2-designs admit a set of $O(\log v)$ points where, once individualized, the color-refinement algorithm assigns a unique color to each point. A priori, it seems plausible that only $\text{poly} \log\log v$ iterations are required. However, color-refinement takes into account the multiset of colors, and so each round can be implemented with a TC^0-circuit. In particular, Babai & Wilmes rely crucially on counting [36, Target 2]. So we ask the following.

Question 7.2.3. Does there exist an absolute constant c, such that Steiner 2-designs admit a set S of $O(\log^c v)$ points where, after individuaizing S, the coloring from $\text{poly} \log\log n$ rounds of count-free color-refinement assigns each point a unique color?

In the remark following the proof of [37, Lemma 3.2], Babai outines a deterministic proof that leverages the greedy set cover algorithm (see [150]) to obtain a distinguishing set of the prescribed size. Now suppose that G and H are isomorphic, and the greedy set cover algorithm returns distinguishing sequences S and S' of the same size for G, H respectively. A priori, S and S' need not be *canonical* in the sense that there need not be an isomorphism $\varphi : V(G) \to V(H)$ mapping $\varphi(S) = S'$. Thus, we ask the following.

Question 7.2.4. Is it possible to deterministically construct a canonical distinguishing set of the size prescribed by [37, Lemma 3.2] for a graph in polynomial time?

An answer of yes to Question 7.2.4 in tandem with the work in Section 5.3 would immediately yield a polynomial-time isomorphism test for conference graphs. Babai's work [37] implies an $n^{O(\log n)}$-time algorithm, and to the best of my knowledge, no further improvements have been made to the runtime.

Alternatively, we ask the following.

Question 7.2.5. Let G be a conference graph. Does there exists a set of vertices S of size $O(1)$ such that, after individualizing S and running Color Refinement, each vertex of G would receive a unique color?

An answer of yes to Question 7.2.5 would also yield a polynomial-time isomorphism test for conference graphs, using the individualize-and-refine paradigm. Even finding a such a set S of size $o(\log n)$ would be a major advance, as it would yield an $n^{o(\log n)}$-time isomorphism test.

7.3 Relation Algebras

Question 7.3.1. Is $f(2) < 1000$?

Question 7.3.2. Is 32_{65} representable over $(\mathbb{Z}/2\mathbb{Z})^m$ for $m < 10$? The natural thing to try – using the construction from the proof of Theorem 6.1.1, with $k = 2$ (hence $m = 7$) – doesn't work; we checked all partitions. But there may some other representation.

Finally, we note that there is a considerable gap in the bounds $2n^2 + \Omega(n\sqrt{\log n}) \leq f(n) \leq 2n^{6+o(1)}$. It would be of interest to close this gap. In light of the discussion following Theorem 6.1.3, it is unlikely that the upper bound can be further improved by analyzing the Boolean cube. It appears that new ideas are required to improve either the upper or the lower bounds.

Bibliography

[1] A. Abdollahi, S. Akbari, and H.R. Maimani. Non-commuting graph of a group. Journal of Algebra, 298(2):468–492, 2006. doi:10.1016/j.jalgebra.2006.02.015.

[2] Sergei I. Adian. Algorithmic unsolvability of problems of recognition of certain properties of groups. Doklady Akademii Nauk SSSR, pages 533–535, 1955.

[3] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. Ann. of Math, 2:781–793, 2002. doi:10.4007/annals.2004.160.781.

[4] Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan. Exploiting symmetries for scaling loopy belief propagation and relational training. Mach. Learn., 92(1):91–132, 2013. doi:10.1007/s10994-013-5385-0.

[5] Wilhelm Ahrens. Mathematische unterhaltungen und spiele, volume 2. BG Teubner, 1918.

[6] A. A. Albert. Quasigroups. I. Transactions of the American Mathematical Society, 54(3):507–519, 1943. doi:10.2307/1990259.

[7] J. Alm, R. Maddux, and J. Manske. Chromatic graphs, Ramsey numbers and the flexible atom conjecture. Electron. J. Combin., 15(1):Research paper 49, 8, 2008. doi:10.37236/773.

[8] Jeremy Alm, Michael Levet, Saeed Moazami, Jorge Montero-Vallejo, Linda Pham, Dave Sexton, and Xiaonan Xu. Improved bounds on the size of the smallest representation of relation algebra 32_{65}. Algebra universalis, 83, 08 2022. doi:10.1007/s00012-022-00791-4.

[9] Jeremy F. Alm. 401 and beyond: improved bounds and algorithms for the Ramsey algebra search. J. Integer Seq., 20:17.8.4, 2017.

[10] Jeremy F. Alm. Personal communication, 2021.

[11] Jeremy F. Alm and David A. Andrews. A reduced upper bound for an edge-coloring problem from relation algebra. Algebra Universalis, 80(2):Art. 19, 11, 2019. doi:10.1007/s00012-019-0592-6.

[12] Jeremy F. Alm, David A. Andrews, and Michael Levet. Comer schemes, relation algebras, and the flexible atom conjecture, 2019. To Appear in Relational and Algebraic Methods in Computer Science (RAMiCS) 2023. doi:10.48550/ARXIV.1905.11914.

[13] Jeremy F. Alm and Michael Levet. Directed Ramsey and anti-Ramsey schemes and the flexible atom conjecture, 2022. doi:10.48550/ARXIV.1901.06781.

[14] Jeremy F. Alm and Jacob Manske. Sum-free cyclic multi-bases and constructions of Ramsey algebras. Discrete Applied Mathematics, 180:204–212, 2015. doi:https://doi.org/10.1016/j.dam.2014.08.002.

[15] H. Andréka, R. D. Maddux, and I. Németi. Splitting in relation algebras. Proc. Amer. Math. Soc., 111(4):1085–1093, 1991. doi:10.2307/2048576.

[16] Hajnal Andrèka, István Németi, and Steven Givant. Nonrepresentable relation algebras from groups. The Review of Symbolic Logic, 13(4):861–881, 2020. doi:10.1017/S1755020319000224.

[17] Sanjeev Arora and Boaz Barak. Computational Complexity: A Modern Approach. 01 2009. doi:10.1017/CBO9780511804090.

[18] V. Arvind and Piyush P. Kurur. Graph isomorphism is in SPP. Information and Computation, 204(5):835–852, 2006. doi:10.1016/j.ic.2006.02.002.

[19] James Aspnes, Richard Beigel, Merrick Furst, and Steven Rudich. The expressive power of voting polynomials. In Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing, STOC '91, page 402–409, New York, NY, USA, 1991. Association for Computing Machinery. doi:10.1145/103418.103461.

[20] Albert Atserias and Elitza Maneva. Sherali-Adams relaxations and indistinguishability in counting logics. In Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS '12, page 367–379, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2090236.2090265.

[21] L. Babai, W. M. Kantor, and E. M. Luks. Computational complexity and the classification of finite simple groups. In 24th Annual Symposium on Foundations of Computer Science (sfcs 1983), pages 162–171, 1983. doi:10.1109/SFCS.1983.10.

[22] L. Babai, E. Luks, and A. Seress. Permutation groups in NC. In STOC 1987, STOC '87, pages 409–420, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.28439.

[23] László Babai. Lectures on graph isomorphism, 1979. Mimeographed lecture notes.

[24] Laszlo Babai. On the order of uniprimitive permutation groups. Annals of Mathematics, 113(3):553–568, 1981. doi:10.2307/2006997.

[25] László Babai. On the automorphism groups of strongly regular graphs I. In Proceedings of the 5th Conference on Innovations in Theoretical Computer Science, ITCS '14, page 359–368, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2554797.2554830.

[26] 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.

[27] László Babai and Robert Beals. A polynomial-time theory of black box groups I. In Groups St Andrews 1997 in Bath, I, 1999.

[28] László Babai, Robert Beals, and Ákos Seress. Polynomial-time theory of matrix groups. In Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, STOC '09, page 55–64. Association for Computing Machinery, 2009. doi:10.1145/1536414.1536425.

[29] 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.

[30] László Babai, Paolo Codenotti, and Youming Qiao. Polynomial-time isomorphism test for groups with no abelian normal subgroups - (extended abstract). In International Colloquium on Automata, Languages, and Programming (ICALP), pages 51–62, 2012. doi:10.1007/978-3-642-31594-7_5.

[31] László Babai, Paul Erdös, and Stanley M. Selkow. Random graph isomorphism. SIAM Journal on Computing, 9(3):628–635, 1980. doi:10.1137/0209047.

[32] László Babai and Ludik Kucera. Canonical labelling of graphs in linear average time. In 20th Annual Symposium on Foundations of Computer Science (SFCS 1979), pages 39–46, 1979. doi:10.1109/SFCS.1979.8.

[33] László Babai and Eugene M. Luks. Canonical labeling of graphs. In Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, STOC '83, page 171–183, New York, NY, USA, 1983. Association for Computing Machinery. doi:10.1145/800061.808746.

[34] László Babai and Youming Qiao. Polynomial-time isomorphism test for groups with Abelian Sylow towers. In 29th STACS, pages 453 – 464. Springer LNCS 6651, 2012. doi:10.4230/LIPIcs.STACS.2012.453.

[35] 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.

[36] László Babai and John Wilmes. Quasipolynomial-time canonical form for Steiner designs. In STOC 2013, pages 261–270, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2488608.2488642.

[37] László Babai. On the complexity of canonical labeling of strongly regular graphs. SIAM Journal on Computing, 9(1):212–216, 1980. doi:10.1137/0209018.

[38] David A. Mix Barrington, Peter Kadau, Klaus-Jörn Lange, and Pierre McKenzie. On the complexity of some problems on groups input as multiplication tables. J. Comput. Syst. Sci., 63(2):186–200, 2001. doi:10.1006/jcss.2001.1764.

[39] David A. Mix Barrington and Pierre McKenzie. Oracle branching programs and Logspace versus P. Inf. Comput., 95(1):96–115, 1991. doi:10.1016/0890-5401(91)90017-V.

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

[41] 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.

[42] Hans Ulrich Besche, Bettina Eick, and Eamonn A. O'Brien. The groups of order at most 2000. Electronic Research Announcements of The American Mathematical Society, 7:1–4, 2001. doi:10.1090/S1079-6762-01-00087-7.

[43] Anton Betten, Michael Braun, Harald Fripertinger, Adalbert Kerber, Axel Kohnert, and Alfred Wassermann. Error-Correcting Linear Codes - Classification by Isometry and Applications. 01 2006. doi:10.1007/3-540-31703-1.

[44] Simon R. Blackburn, Peter M. Neumann, and Geetha Venkataraman. Enumeration of Finite Groups. Cambridge Tracts in Mathematics. Cambridge University Press, 2007. doi:10.1017/CBO9780511542756.

[45] Béla Bollobás. Distinguishing vertices of random graphs. In Béla Bollobás, editor, Graph Theory, volume 62 of North-Holland Mathematics Studies, pages 33–49. North-Holland, 1982. doi:10.1016/S0304-0208(08)73545-X.

[46] R. C. Bose. Strongly regular graphs, partial geometries and partially balanced designs. Pacific Journal of Mathematics, 13(2):389 – 419, 1963. doi:pjm/1103035734.

[47] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. J. Symbolic Comput., 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993). doi:10.1006/jsco.1996.0125.

[48] 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.

[49] Jendrik Brachter and Pascal Schweitzer. A systematic study of isomorphism invariants of finite groups via the Weisfeiler–Leman dimension, 2022. doi:10.4230/LIPIcs.ESA.2022.27.

[50] Peter A. Brooksbank, Joshua A. Grochow, Yinan Li, Youming Qiao, and James B. Wilson. Incorporating Weisfeiler–Leman into algorithms for group isomorphism. arXiv:1905.02518 [cs.CC], 2019.

[51] Peter A. Brooksbank, Joshua Maglione, and James B. Wilson. A fast isomorphism test for groups whose Lie algebra has genus 2. Journal of Algebra, 473:545–590, 2017. doi:10.1016/j.jalgebra.2016.12.007.

[52] R. H. Bruck. Finite nets. II. Uniqueness and imbedding. Pacific Journal of Mathematics, 13(2):421 – 457, 1963. doi:10.2140/pjm.1963.13.421.

[53] Harry Buhrman and Steven Homer. Superpolynomial circuits, almost sparse oracles and the exponential hierarchy. In R. K. Shyamasundar, editor, Foundations of Software Technology and Theoretical Computer Science, 12th Conference, New Delhi, India, December 18-20, 1992, Proceedings, volume 652 of Lecture Notes in Computer Science, pages 116–127. Springer, 1992. doi:10.1007/3-540-56287-7_99.

[54] David Burrell. On the number of groups of order 1024. Communications in Algebra, 50(6):2408–2410, 2022. doi:10.1080/00927872.2021.2006680.

[55] 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.

[56] 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.

[57] 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; ECCC Tech. Report TR10-117. doi:10.1145/2540088.

[58] Xi Chen, Xiaorui Sun, and Shang-Hua Teng. Multi-stage design for quasipolynomial-time isomorphism testing of steiner 2-systems. In Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, STOC '13, page 271–280, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2488608.2488643.

[59] Paolo Codenotti, Hadi Katebi, Karem A. Sakallah, and Igor L. Markov. Conflict analysis and branching heuristics in the search for graph automorphisms. In 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, pages 907–914, 2013. doi:10.1109/ICTAI.2013.139.

[60] Charles Colbourn and Jeffrey Dinitz. Handbook of combinatorial designs, second edition (discrete mathematics and its applications). 01 2007. doi:10.1201/9781420010541.

[61] M J Colbourn. An analysis technique for Steiner triple systems. Proc. Tenth Southeastern Conf. Combin., Graph Theory, Computing, page 289–303, 1979.

[62] Marlene J. Colbourn and Charles J. Colbourn. Concerning the complexity of deciding isomorphism of block designs. Discrete Applied Mathematics, 3(3):155–162, July 1981. doi:10.1016/0166-218X(81)90012-3.

[63] Marlene J. Colbourn and Rudolf A. Mathon. On cyclic Steiner 2-designs. In C.C. Lindner and A. Rosa, editors, Topics on Steiner Systems, volume 7 of Annals of Discrete Mathematics, pages 215–253. Elsevier, 1980. doi:10.1016/S0167-5060(08)70182-1.

[64] Nathaniel A. Collins. Weisfeiler–Leman and group isomorphism, 2023. Undergraduate Thesis; In-Preparation. University of Coloardo Boulder.

[65] Nathaniel A. Collins and Michael Levet. Count-free Weisfeiler–Leman and group isomorphism, 2022. doi:10.48550/ARXIV.2212.11247.

[66] Stephen Comer. Combinatorial aspects of relations. Algebra Universalis, 18:77–94, 02 1984. doi:10.1007/BF01182249.

[67] Stephen D. Comer. Color schemes forbidding monochrome triangles. Proceedings of the fourteenth Southeastern conference on combinatorics, graph theory and computing (Boca Raton, Fla., 1983), 39:231–2366, 1983.

[68] Stephen A Cook and Pierre McKenzie. Problems complete for deterministic logarithmic space. Journal of Algorithms, 8(3):385–394, 1987. doi:10.1016/0196-6774(87)90018-6.

[69] D. G. Corneil and C. C. Gotlieb. An efficient algorithm for graph isomorphism. J. ACM, 17(1):51–64, January 1970. doi:10.1145/321556.321562.

[70] Francesca Dalla Volta and Andrea Lucchini. Generation of almost simple groups. J. Algebra, 178:194–223, 1995. doi:10.1006/jabr.1995.1345.

[71] Paul T. Darga, Mark H. Liffiton, Karem A. Sakallah, and Igor L. Markov. Exploiting structure in symmetry detection for CNF. In Proceedings of the 41st Annual Design Automation Conference, DAC '04, page 530–534, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/996566.996712.

[72] Bireswar Das and Shivdutt Sharma. Nearly linear time isomorphism algorithms for some nonabelian group classes. In René van Bevern and Gregory Kucherov, editors, Computer Science – Theory and Applications, pages 80–92, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-19955-5_8.

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

[74] 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.

[75] L. Dodd and R. Hirsch. Improved lower bounds on the size of the smallest solution to a graph colouring problem, with an application to relation algebra. Journal on Relational Methods in Computer Science, 2:18–26, 2013.

[76] Heinz-Dieter Ebbinghaus and Jörg Flum. Finite Model Theory. 01 1999. doi:10.1007/978-3-662-03182-7.

[77] Jack Edmonds. Paths, trees, and flowers. Canadian Journal of Mathematics, 17:449–467, 1965. doi:10.4153/CJM-1965-045-4.

[78] 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.

[79] Ben Elias, Lior Silberman, and Ramin Takloo-Bighash. Finding minimal permutation representations of finite groups. 2007. doi:10.48550/ARXIV.0705.4122.

[80] Ronald Fagin. Probabilities on finite models. The Journal of Symbolic Logic, 41(1):50–58, 1976. doi:10.2307/2272945.

[81] Lukas Fleischer. On the complexity of the Cayley Semigroup Membership Problem. In Rocco A. Servedio, editor, 33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA, volume 102 of LIPIcs, pages 25:1–25:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.25.

[82] Lance Fortnow and Joshua A. Grochow. Complexity classes of equivalence problems revisited. Inform. and Comput., 209(4):748–763, 2011. Also available as arXiv:0907.4775 [cs.CC]. doi:10.1016/j.ic.2011.01.006.

[83] Merrick Furst, John Hopcroft, and Eugene Luks. Polynomial-time algorithms for permutation groups. In 21st Annual Symposium on Foundations of Computer Science (SFCS 1980), pages 36–41, 1980. doi:10.1109/SFCS.1980.34.

[84] Vyacheslav Futorny, Joshua A. Grochow, and Vladimir V. Sergeichuk. Wildness for tensors. Lin. Alg. Appl., 566:212–244, 2019. Preprint arXiv:1810.09219 [math.RT]. doi:10.1016/j.laa.2018.12.022.

[85] Francois Le Gall. Efficient Isomorphism Testing for a Class of Group Extensions. In Susanne Albers and Jean-Yves Marion, editors, 26th International Symposium on Theoretical Aspects of Computer Science, volume 3 of Leibniz International Proceedings in Informatics (LIPIcs), pages 625–636, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.STACS.2009.1830.

[86] The GAP Group. GAP – Groups, Algorithms, and Programming, Version 4.12.2, 2022. URL: https://www.gap-system.org.

[87] Max Garzon and Yechezkel Zalcstein. On isomorphism testing of a class of 2-nilpotent groups. Journal of Computer and System Sciences, 42(2):237–248, 1991. doi:10.1016/0022-0000(91)90012-T.

[88] Steven Givant. Introduction to Relation Algebras: Relation Algebras, Volume 1. 01 2017. doi:10.1007/978-3-319-65235-1.

[89] Jerrold R Griggs. An upper bound on the Ramsey numbers $R(3, k)$. Journal of Combinatorial Theory, Series A, 35(2):145–153, 1983. doi:10.1016/0097-3165(83)90003-1.

[90] Joshua A. Grochow and Michael Levet. On the parallel complexity of group isomorphism via Weisfeiler-Leman. doi:10.48550/ARXIV.2112.11487.

[91] Joshua A. Grochow and Michael Levet. On the descriptive complexity of groups without Abelian normal subgroups, 2022. doi:10.48550/ARXIV.2209.13725.

[92] Joshua A. Grochow and Youming Qiao. Polynomial-time isomorphism test of groups that are tame extensions - (extended abstract). In Algorithms and Computation - 26th International Symposium, ISAAC 2015, Nagoya, Japan, December 9-11, 2015, Proceedings, pages 578–589, 2015. doi:10.1007/978-3-662-48971-0_49.

[93] 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.

[94] Joshua A. Grochow and Youming Qiao. Isomorphism problems for tensors, groups, and cubic forms: completeness and reductions. arXiv:1907.00309 [cs.CC], 2019. doi:10.48550/ARXIV.1907.00309.

[95] Martin Grohe. Isomorphism testing for embeddable graphs through definability. In Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, STOC '00, page 63–72, New York, NY, USA, 2000. Association for Computing Machinery. doi:10.1145/335305.335313.

[96] Martin Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. J. ACM, 59(5), nov 2012. doi:10.1145/2371656.2371662.

[97] Martin Grohe. Descriptive Complexity, Canonisation, and Definable Graph Structure Theory. 08 2017. doi:10.1017/9781139028868.

[98] Martin Grohe. The logic of graph neural networks. In LICS '21: Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science, 2021. Preprint arXiv:2104.14624 [cs.LG]. doi:10.1109/LICS52264.2021.9470677.

[99] Martin Grohe and Sandra Kiefer. A Linear Upper Bound on the Weisfeiler-Leman Dimension of Graphs of Bounded Genus. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), volume 132 of Leibniz International Proceedings in Informatics (LIPIcs), pages 117:1–117:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2019.117.

[100] Martin Grohe and Sandra Kiefer. Logarithmic Weisfeiler-Leman Identifies All Planar Graphs. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, 48th International Colloquium on Automata, Languages, and Programming (ICALP 2021), volume 198 of Leibniz International Proceedings in Informatics (LIPIcs), pages 134:1–134:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2021.134.

[101] Martin Grohe and Julian Mariño. Definability and descriptive complexity on databases of bounded tree-width. In Catriel Beeri and Peter Buneman, editors, Database Theory — ICDT'99, pages 70–82, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. doi:10.1007/3-540-49257-7_6.

[102] Martin Grohe and Daniel Neuen. Isomorphism, canonization, and definability for graphs of bounded rank width. Commun. ACM, 64(5):98–105, apr 2021. doi:10.1145/3453943.

[103] Martin Grohe and Martin Otto. Pebble games and linear equations. The Journal of Symbolic Logic, 80(3):797–844, 2015. doi:10.2307/43864249.

[104] Martin Grohe and Oleg Verbitsky. Testing graph isomorphism in parallel by playing a game. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part I, volume 4051 of Lecture Notes in Computer Science, pages 3–14. Springer, 2006. doi:10.1007/11786986_2.

[105] Yuri Gurevich and Saharon Shelah. On finite rigid structures. The Journal of Symbolic Logic, 61(2):549–562, 1996. doi:10.2307/2275675.

[106] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NeurIPS'17, page 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[107] Xiaoyu He and Youming Qiao. On the Baer–Lovász–Tutte construction of groups from graphs: Isomorphism types and homomorphism notions. Eur. J. Combin., 98:103404, 2021. doi:10.1016/j.ejc.2021.103404.

[108] Harald Andrés Helfgott, Jitendra Bajpai, and Daniele Dona. Graph isomorphisms in quasi-polynomial time, 2017. doi:10.48550/ARXIV.1710.04574.

[109] Lauri Hella. Definability hierarchies of generalized quantifiers. Annals of Pure and Applied Logic, 43(3):235 – 271, 1989. doi:10.1016/0168-0072(89)90070-5.

[110] Lauri Hella. Logical hierarchies in PTIME. Information and Computation, 129(1):1–19, 1996. doi:10.1006/inco.1996.0070.

[111] D. G. Higman. Coherent algebras. Linear Algebra Appl., 93:209–239, 1987. doi:10.1016/S0024-3795(87)90326-0.

[112] Graham Higman. Enumerating p-groups, II: Problems whose solution is PORC. Proceedings of the London Mathematical Society, s3-10(1):566–582, 1960. doi:10.1112/plms/s3-10.1.566.

[113] Robin Hirsch and Ian Hodkinson. Relation Algebras by Games. Elsevier Science B.V., 2002.

[114] D. Holt, B. Eick, and E. O'Brien. Handbook of Computational Group Theory. Chapman and Hall/CRC, 2005.

[115] Michael Huber. Computational complexity of reconstruction and isomorphism testing for designs and line graphs. Journal of Combinatorial Theory, Series A, 118(2):341–349, 2011. doi:10.1016/j.jcta.2010.06.006.

[116] Costas S. Iliopoulos. Computing in general abelian groups is hard. Theoretical Computer Science, 41:81–93, 1985. doi:10.1016/0304-3975(85)90061-1.

[117] Neil Immerman. Upper and lower bounds for first order expressibility. Journal of Computer and System Sciences, 25(1):76–98, 1982. doi:10.1016/0022-0000(82)90011-3.

[118] Neil Immerman. Relational queries computable in polynomial time. Inf. Control., 68(1-3):86–104, 1986. doi:10.1016/S0019-9958(86)80029-8.

[119] Neil Immerman. Descriptive complexity. Graduate texts in computer science. Springer, 1999. doi:10.1007/978-1-4612-0539-5.

[120] Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. In Alan L. Selman, editor, Complexity Theory Retrospective: In Honor of Juris Hartmanis on the Occasion of His Sixtieth Birthday, July 5, 1988, pages 59–81. Springer New York, New York, NY, 1990. doi:10.1007/978-1-4612-4478-3_5.

[121] Neil Immerman and Rik Sengupta. The k-dimensional Weisfeiler–Leman algorithm. arXiv:1907.09582 [cs.CC], 2019.

[122] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? Journal of Computer and System Sciences, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

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

[124] I. Isaacs. Finite Group Theory. American Mathematical Society, 2008.

[125] Gábor Ivanyos and Youming Qiao. Algorithms based on *-algebras, and their applications to isomorphism of polynomials with one secret, group isomorphism, and polynomial identity testing. SIAM J. Comput., 48(3):926–963, 2019. doi:10.1137/18M1165682.

[126] Jeffrey C. Jackson, Adam R. Klivans, and Rocco A. Servedio. Learnability beyond AC^0. In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, STOC '02, page 776–784, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.510018.

[127] P. Jipsen, R. D. Maddux, and Z. Tuza. Small representations of the relation algebra $\mathcal{E}_{n+1}(1,2,3)$. Algebra Universalis, 33(1):136–139, 1995. doi:10.1007/BF01190770.

[128] Bjarni Jónsson. On direct decompositions of torsionfree abelian groups. Mathematica Scandinavica, 5(2):230–235, 1957.

[129] T. Kavitha. Linear time algorithms for abelian group isomorphism and related problems. Journal of Computer and System Sciences, 73(6):986 – 996, 2007. doi:10.1016/j.jcss.2007.03.013.

[130] Neeraj Kayal and Timur Nezhmetdinov. Factoring groups efficiently. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikoletseas, and Wolfgang Thomas, editors, Automata, Languages and Programming, pages 585–596, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-02927-1_49.

[131] L. Khachiyan. Polynomial algorithms in linear programming. USSR Computational Mathematics and Mathematical Physics, 20:53–72, 1980. doi:10.1016/0041-5553(80)90061-0.

[132] Sandra Kiefer. Power and Limits of the Weisfeiler–Leman Algorithm. PhD thesis, RWTH Aachen University, 2019. URL: https://publications.rwth-aachen.de/record/785831/files/785831.pdf.

[133] Sandra Kiefer and Daniel Neuen. The power of the Weisfeiler-Leman algorithm to decompose graphs. SIAM J. Discret. Math., 36(1):252–298, 2022. doi:10.1137/20m1314987.

[134] Sandra Kiefer, Ilia Ponomarenko, and Pascal Schweitzer. The Weisfeiler–Leman dimension of planar graphs is at most 3. J. ACM, 66(6), November 2019. doi:10.1145/3333003.

[135] Victor Klee and George J. Minty. How good is the simplex algorithm. 1970.

[136] Johannes Köbler, Uwe Schöning, and Jacobo Torán. Graph isomorphism is low for PP. Comput. Complex., 2:301–330, 1992. doi:10.1007/BF01200427.

[137] Tomasz Kowalski. Representability of Ramsey Relation Algebras. Algebra universalis, 74, 11 2015. doi:10.1007/s00012-015-0353-0.

[138] Ludek Kucera. Canonical labeling of regular graphs in linear average time. SFCS '87, page 271–279, USA, 1987. IEEE Computer Society. doi:10.1109/SFCS.1987.11.

[139] R.E. Ladner, N.A. Lynch, and A.L. Selman. A comparison of polynomial time reducibilities. Theoretical Computer Science, 1(2):103–123, 1975. doi:10.1016/0304-3975(75)90016-X.

[140] Richard E. Ladner. On the structure of polynomial time reducibility. J. ACM, 22(1):155–171, January 1975. doi:10.1145/321864.321877.

[141] François Le Gall. Efficient isomorphism testing for a class of group extensions. In Proc. 26th STACS, pages 625–636, 2009. doi:10.4230/LIPIcs.STACS.2009.1830.

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

[143] Michael Levet. On the complexity of identifying strongly regular graphs, 2022. doi:10.48550/ARXIV.2207.05930.

[144] Michael Levet, Puck Rombach, and Nicholas Sieger. Logarithmic Weisfeiler–Leman and treewidth, 2023. doi:10.48550/ARXIV.2303.07985.

[145] Mark L. Lewis and James B. Wilson. Isomorphism in expanding families of indistinguishable groups. 4(1):73–110, 2012. doi:doi:10.1515/gcc-2012-0008.

[146] Yinan Li and Youming Qiao. Linear algebraic analogues of the graph isomorphism problem and the Erdös–Rényi model. In 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), pages 463–474, 2017. doi:10.1109/FOCS.2017.49.

[147] Leonid Libkin. Elements of Finite Model Theory. 2004. doi:10.1007/978-3-662-07003-1_1.

[148] P. Lindstrom. First order predicate logic with generalized quantifiers. Theoria, 32(3):186–195, 1966. doi:10.1111/j.1755-2567.1966.tb00600.x.

[149] 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.

[150] L. Lovász. On the ratio of optimal integral and fractional covers. Discrete Mathematics, 13(4):383–390, 1975. doi:10.1016/0012-365X(75)90058-8.

[151] Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. Journal of Computer and System Sciences, 25(1):42–65, 1982. doi:10.1016/0022-0000(82)90009-5.

[152] Eugene M. Luks. Permutation groups and polynomial-time computation. In Groups and Computation, volume 11 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pages 139–175. American Mathematical Society, 1993.

[153] R. Maddux. Relation algebras, volume 150 of Studies in Logic and the Foundations of Mathematics. Elsevier B. V., Amsterdam, 2006.

[154] Roger Maddux. Do all the Ramsey algebras exist? Talk presented at the AMS sectional meeting in Iowa City, March 18 2011.

[155] Peter N. Malkin. Sherali–Adams relaxations of graph isomorphism polytopes. Discrete Optimization, 12:73–97, 2014. doi:10.1016/j.disopt.2014.01.004.

[156] Rudolf Mathon. A note on the graph isomorphism counting problem. Information Processing Letters, 8(3):131–136, 1979. doi:10.1016/0020-0190(79)90004-8.

[157] Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. Journal of Symbolic Computation, 60:94–112, 2014. doi:10.1016/j.jsc.2013.09.003.

[158] Ralph McKenzie, George McNulty, and Walter Taylor. Algebras, Lattices, Varieties. Volume I, volume 72. AMS Chelsea Publishing: An Imprint of the American Mathematical Society, 06 1988. doi:10.2307/3618961.

[159] Alan H. Mekler. Stability of nilpotent groups of class 2 and prime exponent. The Journal of Symbolic Logic, 46(4):781–788, 1981. doi:10.2307/2273227.

[160] 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.

[161] David A. Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. Journal of Computer and System Sciences, 41(3):274–306, 1990. doi:10.1016/0022-0000(90)90022-D.

[162] Cristopher Moore, Alexander Russell, and Leonard J. Schulman. The symmetric group defies strong Fourier sampling. SIAM Journal on Computing, 37(6):1842–1864, 2008. doi:10.1137/050644896.

[163] Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. In Proceedings of the 34th International Conference on Neural Information Processing Systems, NeurIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[164] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. doi:10.1609/aaai.v33i01.33014602.

[165] Daniel Neuen and Pascal Schweitzer. An exponential lower bound for individualization-refinement algorithms for graph isomorphism. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018, pages 138–150. ACM, 2018. doi:10.1145/3188745.3188900.

[166] A. Neumaier. Strongly regular graphs with smallest eigenvalue $-m$. Archiv der Mathematik, 33:392–400, 1979. doi:10.1007/BF01222774.

[167] 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.

[168] Michael O. Rabin. Recursive unsolvability of group theoretic problems. Annals of Mathematics, 67(1):172–194, 1958. doi:10.2307/1969933.

[169] F. P. Ramsey. On a problem of formal logic. *Proceedings of the London Mathematical Society*, s2-30(1):264–286, 1930. `doi:10.1112/plms/s2-30.1.264`.

[170] Dijen K. Ray-Chaudhuri and Richard M. Wilson. On t-designs. *Osaka Journal of Mathematics*, 12(3):737–744, 1975. `doi:10.18910/7296`.

[171] D. Robinson. *A Course in the Theory of Groups*. Springer, 1982.

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

[173] Benjamin Rossman. Ehrenfeucht-Fraïssé Games on Random Structures. In Hiroakira Ono, Makoto Kanazawa, and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information and Computation, 16th International Workshop, WoLLIC 2009, Tokyo, Japan, June 21-24, 2009. Proceedings*, volume 5514 of *Lecture Notes in Computer Science*, pages 350–364. Springer, 2009. `doi:10.1007/978-3-642-02261-6\_28`.

[174] C. Savage. An $O(n^2)$ algorithm for abelian group isomorphism. Technical report, North Carolina State University, 1980.

[175] Douglas C. Schmidt and Larry E. Druffel. A fast backtracking algorithm to test directed graphs for isomorphism using distance matrices. *J. ACM*, 23(3):433–445, July 1976. `doi:10.1145/321958.321963`.

[176] Uwe Schöning. Graph isomorphism is in the low hierarchy. *Journal of Computer and System Sciences*, 37(3):312 – 323, 1988. `doi:10.1016/0022-0000(88)90010-4`.

[177] Tyler Schrock and Rafael Frongillo. Computational complexity of k-block conjugacy. *Theoretical Computer Science*, 856:21–40, 2021. `doi:10.1016/j.tcs.2020.12.009`.

[178] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. `doi:10.1137/S0097539795293172`.

[179] S. S. Shrikhande. The Uniqueness of the L_2 Association Scheme. *The Annals of Mathematical Statistics*, 30(3):781 – 798, 1959. `doi:10.1214/aoms/1177706207`.

[180] Tyler Shrock. *On The Complexity of Isomorphism In Finite Group Theory and Symbolic Dynamics*. PhD thesis, University of Colorado Boulder, 2019. URL: `https://scholar.colorado.edu/concern/graduate_thesis_or_dissertations/db78td05x`.

[181] 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`.

[182] Michael Sipser. *Introduction to the Theory of Computation*. Course Technology, Boston, MA, third edition, 2013.

[183] 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, 1987, New York, New York, USA*, pages 77–82. ACM, 1987. `doi:10.1145/28395.28404`.

[184] Daniel A. Spielman. Faster Isomorphism Testing of Strongly Regular Graphs. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96, page 576–584, New York, NY, USA, 1996. Association for Computing Machinery. doi:10.1145/237814.238006.

[185] Bangsheng Tang. Towards Understanding Satisfiability, Group Isomorphism and Their Connections. PhD thesis, Tsinghua University, 2013. URL: http://papakonstantinou.org/periklis/pdfs/bangsheng_thesis.pdf.

[186] D. R. Taunt. Remarks on the isomorphism problem in theories of construction of finite groups. Mathematical Proceedings of the Cambridge Philosophical Society, 51(1):16–24, 1955. doi:10.1017/S030500410002987X.

[187] The Sage Developers. SageMath, the Sage Mathematics Software System (Version x.y.z), 2023. https://www.sagemath.org.

[188] Jacquez Theévenaz. Representations of finite groups in characteristic p^r. J. Algebra, 72:478–500, 1981. doi:10.1016/0021-8693(81)90305-7.

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

[190] Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber, editors, Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA, pages 137–146. ACM, 1982. doi:10.1145/800070.802186.

[191] Valery Vasil'ev, Maria Grechkoseeva, and V. Mazurov. Characterization of the finite simple groups by spectrum and order. Algebra and Logic, 48:385–409, 12 2009. doi:10.1007/s10469-009-9074-9.

[192] T.C. Vijayaraghavan. Classifying certain algebraic problems using Logspace counting classes. PhD thesis, HBNI, 2008. URL: https://www.imsc.res.in/xmlui/handle/123456789/118.

[193] Narayan Vikas. An $O(n)$ algorithm for abelian p-group isomorphism and an $O(n \log n)$ algorithm for abelian group isomorphism. Journal of Computer and System Sciences, 53(1):1–9, 1996. doi:10.1006/jcss.1996.0045.

[194] 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.

[195] 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.

[196] B. Yu. Weisfeiler and A. A. Leman. Reduction of a graph to a canonical form and an algebra arising during this reduction, 1968. English translation available at https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf.

[197] Boris Weisfeiler. On construction and identification of graphs. 1976. doi:10.1007/BFb0089374.

[198] James B. Wilson. Existence, algorithms, and asymptotics of direct product decompositions, I. Groups - Complexity - Cryptology, 4(1), Jan 2012. doi:10.1515/gcc-2012-0007.

[199] 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.

[200] 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.

[201] V. N. Zemlyachenko, N. M. Korneenko, and R. I. Tyshkevich. Graph isomorphism problem. Journal of Soviet Mathematics, 29:1426–1481, 1985. doi:10.1007/BF02104746.

[202] Complexity zoo. URL: https://complexityzoo.net.