Polynomial-Time Isomorphism Test for Groups with no Abelian Normal Subgroups (Extended Abstract*)

László Babai**[1], Paolo Codenotti[2], and Youming Qiao ***[3]

[1] University of Chicago, laci@cs.uchicago.edu
[2] University of Minnesota, paolo@ima.umn.edu
[3] Institute for Theoretical Computer Science, Institute for Interdisciplinary Information Sciences, Tsinghua University, jimmyqiao86@gmail.com

Abstract. We consider the problem of testing isomorphism of groups of order n given by Cayley tables. The trivial $n^{\log n}$ bound on the time complexity for the general case has not been improved upon over the past four decades. We demonstrate that the obstacle to efficient algorithms is the presence of abelian normal subgroups; we show this by giving a polynomial-time isomorphism test for groups without nontrivial abelian normal subgroups. This concludes a project started by the authors and J. A. Grochow (SODA 2011). Two key new ingredient are: (a) an algorithm to test permutational isomorphism of permutation groups in time, polynomial in the order and simply exponential in the degree; (b) the introduction of the "twisted code equivalence problem," a generalization of the classical code equivalence problem by admitting a group action on the alphabet. Both of these problems are of independent interest.

Keywords: Group Isomorphism, Permutational Isomorphism, Code Equivalence.

1 Introduction, main results

The isomorphism problem for groups asks to determine whether or not two groups of order n, given by their Cayley tables (multiplication tables), are isomorphic. As pointed out long ago [7,15], if G is generated by k elements then isomorphism can be decided, and all isomorphisms listed, in time $n^{k+O(1)}$. Since $k \leq \log_2 n$ for all groups, this gives an $n^{\log_2 n+O(1)}$-time algorithm for all groups

* See http://people.cs.uchicago.edu/∼laci/papers for the full version of this paper.

** This collaboration was supported in part by László Babai's NSF Grants CCF-0830370 and CCF 1017781. The views expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

*** Youming's work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174.

and a polynomial-time algorithm for finite simple groups (because the latter are generated by 2 elements [20,1]). In spite of considerable attention to the problem over the decades, no general bound with a sublogarithmic exponent has been obtained. While the abelian case is easy ($O(n)$ by [9]), just one step away from the abelian case lurk the most notorious cases: nilpotent groups of class 2 (groups G such that the quotient $G/Z(G)$ is abelian, where $Z(G)$ is the center of G). No complete structure theory of such groups is known; recent work in this direction by James B. Wilson [22,23] commands attention. Recently, special classes of non-nilpotent solvable groups have been considered [11,16,4].

The group isomorphism problem is of great importance to computational group theory; heuristic methods (e. g., Cannon and Holt [6]) have been implemented in the Magma and GAP computational algebra systems for groups are given as permutation groups, represented by sets of generators. In this context the isomorphism problem is graph-isomorphism hard and therefore no subexponential ($\exp(n^{o(1)})$) worst-case algorithm can currently be expected (n is now the size of the permutation domain), while efficient practical algorithms remain a possibility.

While class-2 nilpotent groups have long been recognized as the chief bottleneck in the group isomorphism problem, this intuition has never been formalized. The ultimate formalization would be to reduce the general case to this case. As a first step, we consider a significant class without a complete structure theory at the opposite end of the spectrum: *groups without abelian normal subgroups.* Following Robinson [17], we call such groups *semisimple*[4].

In 2010, J. A. Grochow and the present authors started a project to test isomorphism of semisimple groups [3]. In that paper we observed, based on an elementary analysis of the group structure, that this problem can be solved in time $n^{\log \log n}$, and using a combination of additional structure theory and combinatorial/algorithmic techniques, we gave a polynomial time algorithm assuming the boundedness of certain parameters. The main result of the present paper, stated next, concludes the project.

Theorem 1. *Isomorphism of semisimple groups given by their Cayley tables can be decided in polynomial time.*

We note that semisimplicity of a given group can be decided in polynomial time (trivially for groups given by Cayley tables and nontrivially even for permutation groups given by generators [14]).

Our second main result concerns permutational isomorphism.

Definition 1. *Two permutation groups $G, H \leq S_k$ are* permutationally isomorphic *if $\exists \pi \in S_k$ such that $G^\pi = H$, where $G^\pi := \{\pi^{-1}\sigma\pi \mid \sigma \in G\}$.*

[4] We note that various authors use the term 'semisimple group' in several different meanings (see e. g. [21]). The definition we use conforms to the general practice in algebra that an algebraic structure (ring, algebra, inner product space, etc.) is semisimple if its 'radical' is trivial; each concept of 'radical' then corresponds to a notion of semisimplicity. In our case, the 'radical' is the solvable radical, i. e., the largest solvable normal subgroup.

Theorem 2. *Permutational Isomorphism of permutation groups $G, H \leq S_k$, given by lists of generators, can be decided in time* $\mathrm{poly}(|G|)\, c^k$*, for an absolute constant c.*

The proofs of these two main results are intertwined. First we solve the permutational isomorphism problem for the special case of *transitive groups* in a stronger sense (see Section 1.1); then we use this to solve our main problem, isomorphism of semisimple groups, via "twisted code equivalence." The general case of the permutational isomorphism problem follows via a simple reduction (cf. [3, Theorem 7.2]).

The two main ingredients that support Theorem 1 and 2 are: *permutational isomorphism of transitive permutation groups* and *twisted code equivalence.* We explain them in the next section.

1.1 Technical ingredients

Theorem 3. *There exists an absolute constant c such that for all pairs of transitive permutation groups $G, H \leq S_k$ (a) the number of permutational automorphisms of G is at most $|G|\, c^k$; (b) we can* list *the set of all permutational isomorphisms of G and H in time $|G|\, c^k$.*

The proof involves a detailed group-theoretic study of the structure of transitive permutation groups. (See Section 4 for an outline of the proof.)

Another key ingredient is the concept of "twisted code equivalence," a generalization of the code equivalence problem, and a problem of independent interest. A *code* of length ℓ over a finite alphabet Γ is a subset of Γ^A for some set A with $|A| = \ell$. An *equivalence* of the codes $\mathcal{A} \subseteq \Gamma^A$ and $\mathcal{B} \subseteq \Gamma^B$ is a bijection $A \to B$ that takes $\mathcal{A}$ to $\mathcal{B}$. If $|\Gamma| = 2$ then the code is a Boolean function or hypergraph, so the code equivalence problem is a generalization of the hypergraph isomorphism problem. Slightly simplifying and extending Luks's C^ℓ dynamic programming algorithm for hypergraph isomorphism [12] to treat code equivalence, in [3] we gave an algorithm to test equivalence of codes of length ℓ over an alphabet Γ in time $(c|\Gamma|)^{2\ell}$, for an absolute constant c. In the present paper we introduce a generalization of this problem by allowing to permute the symbols by some group W acting on Γ, independently for each coordinate.

Definition 2 (Twisted code equivalence). *Let $\mathcal{A} \subseteq \Gamma^A$, $\mathcal{B} \subseteq \Gamma^B$ be codes of length ℓ over Γ. Let a group W act on Γ. Given a bijection $\pi : A \to B$ and a function $w : B \to W$, the pair (π, w) is a W-*twisted equivalence *of the codes $\mathcal{A}$ and $\mathcal{B}$ if by applying π to the coordinates of each codeword in $\mathcal{A}$ and then applying $w(b)$ to the entry in position b for each $b \in B$ we obtain the code $\mathcal{B}$.*

Generalizing the algorithm from [3] and improving its data management, we obtain the following result, proved in Section 3. Our algorithm uses a coset intersection subroutine and operates on rather large alphabets. In our case, W will have a low-degree faithful permutation representation, and we take advantage of this.

Theorem 4. *Let Γ be an alphabet, $W \leq \mathrm{Sym}(\Gamma)$, and assume a faithful permutation representation of W of degree d is given. The set of W-twisted equivalences of two codes $\mathcal{A}, \mathcal{B} \subseteq \Gamma^\ell$ can be found in time $c^{d\ell}\mathrm{poly}(|\mathcal{A}|, |W|, |\Gamma|)$.*

In the special case where $W = \{\mathrm{id}\}$, twisted code equivalence is simply code equivalence and the theorem gives a running time of $c^\ell\,\mathrm{poly}(|\Gamma|, |\mathcal{A}|)$. This improvement in the dependence on $|\Gamma|$ over the previous bound of $(c|\Gamma|)^{2\ell}$ from [3] is critical to our main result.

1.2 Strategy for the main result

Our approach is motivated by the Babai-Beals filtration of groups [2] (see [3, Section 7.5]). Specifically, the *socle* of a group is the product of its minimal normal subgroups. The socle of a semisimple group G is the direct product of nonabelian simple groups, and G acts on the set of simple factors of the socle by conjugation, producing a permutation group of degree at most $\log_{60}|G|$. (60 is the order of A_5, the smallest nonabelian simple group.)

First we observe that isomorphism of groups that are direct products of simple groups can be tested in polynomial time. So we can assume that our semisimple groups G and H have isomorphic socles. The second observation is that an isomorphism of the socles extends in at most one way to an isomorphism of G and H. Moreover, given an isomorphism of the socles, we can find the unique extension if it exists (Observation 8). Our next step is to identify isomorphic simple factors of the socles with a canonical copy. From now on we look for only those isomorphisms that respect the specific identification. We note that the number of identifications to consider is polynomially bounded ([3, Lemma 4.1]). We look at the conjugation action of the groups G and H on the set of simple factors of their socles. The orbits of this action correspond to the minimal normal subgroups. Our alphabets will be the isomorphism types of these actions under identification-preserving isomorphisms; these can be computed using our algorithm for transitive permutational isomorphism (Theorem 3). The problem then reduces to twisted code equivalence over these alphabets. The twisting groups consist of the identification-preserving automorphisms of the alphabets; they can be represented as permutation groups acting on the set of simple factors, thus giving a small value of d for the application of Theorem 4.

1.3 Organization of the paper

We first present the two technical ingredients: twisted code equivalence in Section 3, and permutational isomorphism of transitive groups in Section 4. We outline the proof of the main result, Theorem 1, in Section 5. Detailed proofs, and in some cases the detailed statements, appear in the full version of this paper.

2 Group-theoretic preliminaries

For a function $f : X \to Y$, we write x^f for the image of $x \in X$.

General group theory. For groups H, G, we write $H \leq G$ to say that H is a subgroup of G. Given groups G and H, $\mathrm{ISO}(G, H)$ denotes the set of $G \to H$ isomorphisms; $\mathrm{Aut}(G) = \mathrm{ISO}(G, G)$. The set $\mathrm{ISO}(G, H)$ is either empty or a coset of $\mathrm{Aut}(G)$. For $g \in G$, *conjugation by g* means the map $g : G \to G$ defined by $x \mapsto x^g := g^{-1}xg$. For $S \subseteq G$ and $g \in G$ we set $S^g = \{s^g \mid s \in S\}$.

Permutation groups. Let $\mathrm{Sym}(\Omega)$ denote the symmetric group acting on the set Ω, the group of all permutations of Ω. We write S_k for $\mathrm{Sym}([k])$ where $[k] = \{1, \ldots, k\}$. Permutation groups of *degree k* are subgroups of $\mathrm{Sym}(\Omega)$ with $|\Omega| = k$. A homomorphism $\varphi \colon G \to \mathrm{Sym}(\Omega)$ is called a *permutation representation* of G of degree $|\Omega|$; such a homomorphism defines a G-action $x \mapsto x^\pi := x^{\varphi(\pi)}$ on Ω ($x \in \Omega, \pi \in G$). We say that φ is *faithful* if it is injective. Let $\mathrm{Alt}(\Omega) \leq \mathrm{Sym}(\Omega)$ denote the *alternating group.* Let $G \leq \mathrm{Sym}(\Omega)$. The *orbit* of $x \in \Omega$ is the set $x^G := \{x^\pi : \pi \in G\}$. The *length* of an orbit is its size. A permutation group $G \leq \mathrm{Sym}(\Omega)$ is *transitive* if $x^G = \Omega$ for some (any) $x \in \Omega$. The *stabilizer* G_x of a point $x \in \Omega$ is $G_x = \{\pi \in G \mid x^\pi = x\}$.

Given a G-action on Ω, a nonempty set $B \subseteq \Omega$ is a *block of imprimitivity* (or simply "a block") if $(\forall \pi \in G)(B^\pi = B$ or $B \cap B^\pi = \emptyset)$. A transitive action is *primitive* if all blocks are trivial (the singletons or the whole domain Ω), and *imprimitive* otherwise. Let $G \leq \mathrm{Sym}(\Omega)$ and $H \leq \mathrm{Sym}(\Delta)$. The *wreath product* $G \wr H$ is a permutation group acting on $\Omega \times \Delta$ viewed as $|\Delta|$ copies of Ω. $|\Delta|$ copies of G act independently on each copy of Ω and H permutes the copies. This defines the *standard action* of this group. In its *product action*, the same group acts on Ω^Δ, such that the copies of G act on each coordinate and H permutes the coordinates.

Algorithms for permutation groups. For the purposes of computation, a permutation group $G \leq \mathrm{Sym}(\Omega)$ will be represented by a list of generators. Many basic computational tasks for permutation groups, including membership testing, finding the order of a group, finding pointwise stabilizers of subsets of the domain, finding blocks of imprimitivity, can be performed in polynomial time ([19,8,10], cf. [18]).

3 Twisted code equivalence

Let $\mathrm{EQ}_W(\mathcal{A}, \mathcal{B})$ denote the set of W-twisted equivalences of the codes $\mathcal{A}$ and $\mathcal{B}$. (See Section 1.1 for the definitions.) Note that this is either empty or a coset of the group $\mathrm{EQ}_W(\mathcal{A}, \mathcal{A}) \leq W \wr \mathrm{Sym}(A)$.

In this section we prove the following, more precise version of Theorem 4.

Theorem 5. *Let $\mathcal{A} \subseteq \Gamma^A$ and $\mathcal{B} \subseteq \Gamma^B$ be codes of length ℓ. Let $W \leq \mathrm{Sym}(\Gamma)$, and assume we are given a faithful permutation representation of W of degree d. Then $\mathrm{EQ}_W(\mathcal{A}, \mathcal{B})$ can be found in time $O(2^{\ell(d+1)}\,|W|\,|\Gamma|\,|\mathcal{A}|^2 \log |\mathcal{A}|)$.*

Proof. For a subset $U \subseteq A$ we call a function $y \colon U \to \Gamma$ a "partial string over A." The set U is the domain of y, denoted $\mathrm{dom}(y)$, and $|U|$ the *length* of y. For a partial string y over A, let $\mathcal{A}_y$ be the set of strings in $\mathcal{A}$ that are extensions of y. We make analogous definitions for $\mathcal{B}$.

We construct a dynamic programming table with an entry for each pair (y, z) of partial strings, y over A and z over B, of equal length such that $\mathcal{A}_y \neq \emptyset$, and $\mathcal{B}_z \neq \emptyset$. For each such pair (y, z), we store the set $I(y, z)$ of W-twisted equivalences of the restriction $\mathcal{A}^*_y$ of $\mathcal{A}_y$ to $A \setminus \operatorname{dom}(y)$ with the restriction $\mathcal{B}^*_z$ of $\mathcal{B}_z$ to $B \setminus \operatorname{dom}(z)$. Note that the $I(y, z)$ are either empty or cosets of the groups $\mathrm{EQ}_W(\mathcal{A}^*_y, \mathcal{A}^*_y) \leq W \wr \operatorname{Sym}(A \setminus \operatorname{dom}(y))$, and hence they can be stored efficiently.

We start with full strings $y \in \mathcal{A}, z \in \mathcal{B}$ and work our way down to $\operatorname{dom}(y) = \operatorname{dom}(z) = \emptyset$, at which point we shall have constructed all $\mathcal{A} \to \mathcal{B}$ twisted W-equivalences. When y, z are full strings, we have $|\mathcal{A}_y| = 1$, $|\mathcal{B}_z| = 1$, and $I(y, z)$ is trivial.

Let y, z be proper partial strings of length h, and assume we have constructed $I(y', z')$ for all y', z' of length greater than h. To construct $I(y, z)$ we augment the domain of y by one index $r \in A \setminus \operatorname{dom}(y)$, and the domain of z by one index, $s \in B \setminus \operatorname{dom}(z)$. We fix r, and make all possible choices of $s \in B \setminus \operatorname{dom}(z)$. For each $s \in B \setminus \operatorname{dom}(z)$ and $\sigma \in W$, we will separately find the set of all elements of $I(y, z)$ that move s to r, and act on the symbol in that position by σ.

More formally, for $\gamma \in \mathit{\Gamma}$, and $r \in A \setminus \operatorname{dom}(y)$, let $y(r, \gamma)$ be the partial string extending y by γ at position r, and let $\sigma \in W$, $s \in B \setminus \operatorname{dom}(z)$. Given some $\pi : (A \setminus \operatorname{dom}(y) \setminus \{r\}) \to (B \setminus \operatorname{dom}(z) \setminus \{s\})$, let $\pi^* : (A \setminus \operatorname{dom}(y)) \to (B \setminus \operatorname{dom}(z))$ extend π by sending r to s; and for $w : (B \setminus \operatorname{dom}(z) \setminus \{s\}) \to W$, let $w^* : (B \setminus \operatorname{dom}(z)) \to W$ extend w by sending s to σ. Let $I^*(y, z\,;\, r \xrightarrow{(\gamma,\,\sigma)} s)$ be the set of all (π^*, w^*) for $(\pi, w) \in I(y(r, \gamma), z(s, \gamma^{\sigma^{-1}}))$. Then

$$I(y, z) = \bigcup_{s \in B} \bigcup_{\sigma \in W} \bigcap_{\gamma \in \mathit{\Gamma} : \mathcal{A}_{y(r,\gamma)} \neq \emptyset} I^*(y, z\,;\, r \xrightarrow{(\gamma,\,\sigma)} s).$$

If $z(s, \gamma^{\sigma^{-1}}) \in \mathcal{B}$, then we can look up the value of $I(y(r, \gamma), z(s, \gamma^{\sigma^{-1}}))$ in the table and use that to compute the corresponding I^*. If for some σ and γ, $z(s, \gamma^{\sigma^{-1}}) \notin \mathcal{B}$, then $I(y(r, \gamma), z(s, \gamma^{\sigma^{-1}}))$ is empty and so is I^*.

Analysis. We consider $\ell|\mathcal{A}|$ partial strings of $\mathcal{A}$ (all prefixes), and $2^\ell|\mathcal{A}|$ partial strings of $\mathcal{B}$ (we can assume $|\mathcal{A}| = |\mathcal{B}|$, otherwise we reject equivalence). So the number of table entries we store is at most $2^\ell\, \ell\, |\mathcal{A}|^2$. The cost of computing each dynamic programming entry is $\ell|\mathit{\Gamma}||W|$ coset intersection operations, and $\ell|\mathit{\Gamma}||W|$ times the cost of checking whether some $\mathcal{A}_{y'}$ is empty. Standard techniques allow us to compute the I^* and paste cosets together in polynomial time. The cost of coset intersection is $O(2^{\ell\, d})$ [12]. (Stronger bounds for coset intersection are available but not needed here.) The cost of checking if $\mathcal{A}_{y'}$ is empty is $\log|\mathcal{A}|$ if we add a preprocessing step to sort $\mathcal{A}$. □

We shall need a simple generalization of this result to multiple alphabets. (As before, we refer to the full version for all missing details.)

4 Permutational isomorphism for transitive groups

4.1 Further group-theoretic preliminaries

Permutational Isomorphism. If $G \leq \mathrm{Sym}(\Omega)$ and $H \leq \mathrm{Sym}(\Delta)$ are permutation groups, a bijection $\pi \colon \Omega \to \Delta$ is a *permutational isomorphism* from G to H if $G^\pi = H$. We denote the set of all $G \to H$ permutational isomorphisms by $\mathrm{PISO}(G, H)$; $\mathrm{PAut}(G) := \mathrm{PISO}(G, G)$. We say G and H are *permutationally isomorphic* if $\mathrm{PISO}(G, H) \neq \emptyset$. Each $\pi \in \mathrm{PISO}(G, H)$ induces an isomorphisim $\widehat{\pi} : G \to H$. Let $\widehat{\mathrm{PISO}}(G, H) := \{\widehat{\pi} \mid \pi \in \mathrm{PISO}(G, H)\}$.

Proposition 1. *Given $G, H \leq S_k$ and $f \in S_k$, we can decide in* $\mathrm{poly}(k)$ *time whether or not $f \in \mathrm{PISO}(G, H)$.* Proof: use membership testing.

Bounds on primitive groups. Let $G \leq S_k$. We call G a *giant*[5] if $k \geq 7$ and $G = S_k$ or A_k. These two groups are far larger than any other primitive group.

Lemma 1. *Let $G \leq S_k$ be primitive and let $H \leq S_k$. (a) If G is non-giant then* $|\mathrm{PAut}(G)| \leq \exp(\widetilde{O}(\sqrt{k}))$. *(The tilde hides polylog factors.)*
(b) If G is non-giant then we can list $\mathrm{PISO}(G, H)$ *in time* $\exp(\widetilde{O}(\sqrt{k}))$.
(c) We can find the coset $\mathrm{PISO}(G, H)$ *in quasipolynomial time* $(\exp(\mathrm{polylog}(k))$.

The proof requires the classification of finite simple groups via Cameron's classification of the large primitive groups [5].

Structure trees. For a transitive group $G \leq \mathrm{Sym}(\Omega)$, a *$G$-invariant tree* is a rooted tree whose set of leaves is Ω and to which the G-action extends as tree automorphisms. (Such extension is necessarily unique.) A G-invariant tree is a *structure tree* for G if every internal node has at least 2 children, and for every internal node u of the tree, the action of the stabilizer G_u on the set of children of u is primitive. The following observation will allow us to list all structure trees of a transitive group.

Lemma 2. *Let $G \leq \mathrm{Sym}(\Omega)$ be a transitive permutation group of degree k. Then (a) G has at most $k^{2\log k}$ structure trees; (b) we can list all structure trees of G in time $O(k^{2\log k + O(1)})$.*

Subdirect products, diagonals. Given groups $G_1, \dots, G_r$, we write $\pi_j \colon \prod_{i=1}^r G_i \to G_j$ for the projection map of the direct product onto the j-th factor. A *subdirect product* of the G_i is a subgroup $H \leq \prod_{i=1}^r G_i$ such that $\pi_j(H) = G_j$ for each j. A particularly important example of subdirect products is a diagonal. Let $V_1, \dots, V_r$ be isomorphic groups, $(\forall i)(V_i \cong T)$. A *diagonal* of $(V_1, \dots, V_r)$ is an embedding $\phi : T \hookrightarrow \prod_{i=1}^r V_i$ such that $\mathrm{Im}(\phi)$ is a subdirect product of the V_i. Its image is denoted $\mathrm{diag}(V_1 \times \cdots \times V_r)$. The *standard diagonal* of T^r is the map $\Delta : t \mapsto (t, \dots, t)$.

[5] We remark that S_k and A_k are primitive for $k \geq 3$. We look at $k \geq 5$ and $k \neq 6$ since A_k is simple when $k \geq 5$; and $\mathrm{Aut}(A_k) \cong S_k$ when $k \geq 4$, $k \neq 6$.

For permutation groups, analogously, we can define permutational diagonals. Let $V_i \leq \mathrm{Sym}(\Omega_i)$ (for $i \in [r]$) be permutation groups, permutationally isomorphic to $T \leq \mathrm{Sym}(\Xi)$. (In particular, for every i, $|\Omega_i| = |\Xi|$.) A *permutational diagonal* of $\prod_{i=1}^{r} V_i$ is a list of permutational isomorphisms $\phi_i \in \mathrm{PISO}(T, V_i)$ (so $\widehat{\phi}_i : T \to V_i$ is the corresponding isomorphism). Let $\widehat{\phi} : T \to \prod V_i$ be defined by $t \mapsto (t^{\widehat{\phi}_1}, \ldots, t^{\widehat{\phi}_r})$. We use $\mathrm{pdiag}(V_1, \times \cdots \times V_r)$ to denote the image $\widehat{\phi}(T)$ for some permutational diagonal of $(V_1, \ldots, V_r)$.

For example, given $G \leq \mathrm{Sym}(\Omega)$, consider the induced action of G^r on Ω^r. The standard diagonal T of G^r acting on $\{(\omega, \ldots, \omega) \mid \omega \in \Omega\}$ is a permutational diagonal defined by the identity bijections.

Fact 6 *Let $G \leq H_1 \times \cdots \times H_m$ be a subdirect product, where each H_i is a simple group. Then G is a direct product of diagonals.*

Fact 7 *Let $G \leq \mathrm{Alt}(\Omega_1) \times \cdots \times \mathrm{Alt}(\Omega_m)$, where $(\forall i)(|\Omega_i| \geq 5, \neq 6)$ be a subdirect product of alternating groups. Then G is a direct product of permutational diagonals.*

4.2 Transitive groups: outline of the proof of Theorem 3

Let $G \leq \mathrm{Sym}(\Omega)$ and $H \leq \mathrm{Sym}(\Delta)$ be transitive permutation groups of degree $k = |\Omega| = |\Delta|$. Our job is to list all their permutational isomorphisms in time $c^k|G|$. Our strategy is to fix a structure tree of G and work by induction on its depth. The base case is when G is primitive; this is settled by Lemma 1.

We call a structure tree T of G and a structure tree U of H *compatible* if their depth is the same and, for every ℓ, the primitive groups arising on level ℓ in G and H (as actions of the stabilizers of a node on level ℓ on the children of that node) are permutationally isomorphic.

By Lemma 2, we can try all structure trees of H that are compatible with the chosen structure tree of G. So we may assume we have fixed structure trees T and U on G and H, resp., and we are looking for permutational isomorphisms respecting them. Assume these trees have depth $d \geq 1$ (the root is level 0). Let G^* denote the action of G on $T(d-1)$: the set of nodes at level $d-1$; define H^* analogously. Assume by induction that the set $\mathrm{PISO}(G^*, H^*)$ is available. Now, for each $\pi \in \mathrm{PISO}(G^*, H^*)$ we wish to list the set $\mathrm{PISO}(G, H, \pi)$ of extensions of π to elements of $\mathrm{PISO}(G, H)$.

For $i \in T(d-1)$ let Ω_i denote the set of children of i and let $G(i)$ denote the action of G_i, the stabilizer of i, restricted to Ω_i. Note that $\{\Omega_1, \ldots, \Omega_m\}$ is a maximal system of imprimitivity for G, where $m = |T(d-1)|$. For $j \in U(d-1)$, define Δ_j and $H(j)$ analogously. Since T and U (the structure trees) are compatible, all the $G(i)$ and $H(j)$ are permutationally isomorphic primitive groups. Let K be the pointwise stabilizer of $T(d-1)$ in G, and L the corresponding stabilizer in H. So $K \triangleleft G$ is the kernel of the restriction homomorphism $G \to G^*$; analogously for $L \triangleleft H$. Let $K(i)$ be the restriction of K to Ω_i; and $L(j)$ the restriction of L to Δ_j. We now distinguish two cases.

Case 1. $G(i)$ is not a giant. In this case, by Lemma 1, the number of permutational isomorphisms between $G(i)$ and $H(i^\pi)$ is at most c^h where $h = k/m = |\Omega_i|$. We can combine these in $(c^h)^m = c^k$ ways, settling this case.
Case 2. $G(i)$ is a giant. This case is more technical. For simplicity, in this case we shall assume $K(i) = \mathrm{Alt}(\Omega_i)$ in this outline. The basic observation is that $K(i) \lhd G(i)$ and therefore either $K(i) = \{\mathrm{id}\}$ (Case 2a) or $K(i)$ is a giant (Case 2b).
Case 2a. $K(i) = \{\mathrm{id}\}$. If this is true for some i, it is true for all. Therefore, $K = \{\mathrm{id}\}$, so G embeds in G^*, i. e., π has at most one extension. Let us call this unique extension φ if it exists. Let $x \in \Omega_i$; we wish to find $y = x^\varphi \in \Delta_{i^\pi}$. One can show that if such a y exists then it is unique, and to find such a y it is sufficient to consider the stabilizer G_x and look for a corresponding stabilizer H_y for some $y \in \Delta_{i^\pi}$.
Case 2b. $K(i) = \mathrm{Alt}(\Omega_i)$. Clearly, $\mathrm{PISO}(G, H) \subseteq \mathrm{PISO}(K, L)$. While this is always true, under Case 2b we shall also see that the set $\mathrm{PISO}(K, L; \pi)$ has "affordable size," so we can list $\mathrm{PISO}(K, L; \pi)$ and check each of its elements (Proposition 1). Assuming $h \geq 5$, $h \neq 6$, we can bound the number of extensions of π by $2^m|K|$, and list them in time $O(2^m|K|k)$.

5 Semisimple group isomorphism: the proof of Theorem 1

5.1 The framework

Recall our strategy from Section 1.2. If G is semisimple then $\mathrm{Soc}(G)$ is the direct product of its minimal normal subgroups; and each minimal normal subgroup is the direct product of isomorphic nonabelian simple groups. Both of these decompositions are unique. The conjugation action of G permutes the simple factors of the socle; this defines a permutation representation $G \to S_k$ where k is the number of simple factors of the socle. The kernel of this representation is denoted $\mathrm{Pker}(G)$; this is the first characteristic subgroup in the BB chain [2]. So $G/\mathrm{Pker}(G)$ is a permutation group of degree $k \leq \log_{60}|G|$. The orbits of this permutation representation correspond to the minimal normal subgroups of G; in particular, it is transitive exactly if G has a unique minimal normal subgroup (namely, the socle).

Lumping together isomorphic minimal normal subgroups, we obtain the product decomposition $\mathrm{Soc}(G) = \prod_{i=1}^{d} \prod_{j=1}^{z_i} N_{i,j} \cong \prod_{i=1}^{d} K_i^{z_i}$, where the $N_{i,j}$ are the minimal normal subgroups and $(\forall i, j)(N_{i,j} \cong K_i)$. The K_i are pairwise nonisomorphic. Let $N_{ij} = \prod_{h=1}^{t_i} V_{ijh}$ be the decomposition of N_{ij} into simple factors. Let $T_i \cong V_{ijh}$ be a canonical copy of the simple factors of $K_i \cong N_{ij}$.

Our first trick is to fix an isomorphism between T_i and each V_{ijh}, i.e., a diagonal of $\prod_{j,h} V_{ijh}$ for each i. Since T_i is generated by two elements, we have $|\mathrm{Aut}(T_i)| \leq |T_i|^2$, and therefore the number of diagonals we need to consider is at most $|\mathrm{Soc}(G)|^2$. Let ϕ and ψ be diagonals of $\mathrm{Soc}(G)$ and $\mathrm{Soc}(H)$, resp., with respect to their factorizations into simple factors. We write $\mathrm{ISOds}(G, H; \phi, \psi)$ for the set of $G \to H$ isomorphisms which respect these diagonals.

Observation 8 ([3, Corollary 3.1]) *If G and H are semisimple groups then (a) any isomorphism $f : \mathrm{Soc}(G) \to \mathrm{Soc}(H)$ extends in at most one way to a $\overline{f} : G \to H$ isomorphism; and, (b) given f we can decide if $\overline{f}$ exists, and find it if it does, in polynomial time.*

Part (a) of the observation follows from a well-known fact [17, Claim 3.3.19, page 90] (cf. [6, sec. 3.1], [3, Lemma 3.1]). It follows that an isomorphism $\chi \in \mathrm{ISOds}(G, H; \phi, \psi)$ is uniquely determined by the permutational isomorphism it induces between $G/\mathrm{Pker}(G)$ and $H/\mathrm{Pker}(H)$.

5.2 Semisimple groups with a unique minimal normal subgroup

Corollary 1. *Let G and H be semisimple groups with a unique minimal normal subgroup. (a) $|\mathrm{Aut}(G)| \le |G|^{O(1)}$, and (b) we can list $\mathrm{ISO}(G, H)$ in time $|G|^{O(1)}$.*

Proof. Fix a diagonal ϕ of $\mathrm{Soc}(G)$ and consider all diagonals ψ of $\mathrm{Soc}(H)$; in each case, we shall compute $\mathrm{ISOds}(G, H; \phi, \psi)$. Because simple groups have 2 generators, the latter needs to be performed $\le |\mathrm{Soc}(G)|^2 \le |G|^2$ times (cf. [3, Lemma 4.1]).

By part (a) of Observation 8, every isomorphism $\chi \in \mathrm{ISOds}(G, H; \phi, \psi)$ is uniquely determined by the permutational isomorphism it induces between $G/\mathrm{Pker}(G)$ and $H/\mathrm{Pker}(H)$. Therefore the number of automorphisms respecting ϕ is at most the number of permutational automorphisms of $G/\mathrm{Pker}(G)$, which in turn is at most $c^k|G/\mathrm{Pker}(G)|$, by part (a) of Theorem 3. Since $k = O(\log |G|)$, this proves part (a). For part (b), we apply the algorithm in part (b) of Theorem 3 to the transitive permutation groups $G/\mathrm{Pker}(G)$ and $H/\mathrm{Pker}(H)$ and list all $\pi \in \mathrm{PISO}(G/\mathrm{Pker}(G), H/\mathrm{Pker}(H))$. For each such π, let f be the isomorphism of $\mathrm{Soc}(G)$ and $\mathrm{Soc}(H)$ determined by π and the diagonals ϕ, and ψ. For each such f, apply part (b) of Observation 8 to check whether f extends to an isomorphism $\overline{f} : G \to H$. □

5.3 All semisimple groups

In this subsection we outline the reduction of testing isomorphism of the semisimple groups G, H to twisted code equivalence. Since isomorphism of the socles and their direct decomposition to minimal normal subgroups are easy to test, we may assume that $\mathrm{Soc}(G) = \mathrm{Soc}(H)$ and they have the same decomposition into minimal normal subgroups, using the notation from Section 5.1. The conjugation action of G on $\mathrm{Soc}(G)$ embeds G into $\prod_i \prod_j \mathrm{Aut}(N_{ij})$; we call this embedding α, and let $G^* = \alpha(G)$; we define the embedding β and $H^* = \beta(H)$ analogously. We view G^* and H^* as permutation groups acting on $\mathrm{Soc}(G)$.

Having fixed diagonals as in Section 5.1, we are only interested in those permutational isomorphisms $G^* \to H^*$ which act on the socles by permuting the copies of the K_i and within them, permuting the copies of T_i, respecting their standard diagonals. Let X be the set of these $G^* \to H^*$ permutational isomorphisms (given as a coset). If we knew X, by [17, Claim 3.3.19, page 90], we can recover the coset of $G \to H$ isomorphisms respecting the diagonals by the formula $\alpha X \beta^{-1}$. We now describe how to find X.

Let G^*_{ij} denote the restriction of G^* to N_{ij}, and similarly H^*_{ij} the restriction of H^* to N_{ij}. By Cor. 1, we can compute the isomorphism types of the G^*_{ij} and H^*_{ij} under permutational isomorphisms in X. Let $\Gamma_1, \ldots, \Gamma_r$ be representatives of these isomorphism types. These will be our alphabets. For each i, j, pick an arbitrary isomorphism φ_{ij} of this type between G^*_{ij} and the corresponding representative, and analogously for H^*.

We create codes $\overline{G}$ and $\overline{H}$ over the alphabets Γ_i as follows. Let $\sigma \in G^*$, and let $\sigma_{ij} \in G^*_{ij}$ denote the restriction of σ to N_{ij}. The string associated with σ is $\overline{\sigma} = (\sigma_{ij}^{\varphi_{ij}})_{ij}$. Then $\overline{G} = \{\overline{\sigma} \mid \sigma \in G^*\}$. Define $\overline{H}$ analogously.

For $\ell \in [r]$, let W_ℓ be the group of automorphisms of Γ_ℓ induced by permutational automorphisms that preserve the standard diagonals. These automorphisms are determined by the permutation they induce on the set of simple factors. Thus if Γ_ℓ is acting on a copy of $K_i (= T_i^{t_i})$ then W_ℓ has a faithful permutation representation of degree t_i.

Let $\chi : G^* \to H^*$ be a permutational isomorphism respecting the standard diagonals (i. e., $\chi \in X$). So χ induces a permutation $\pi(\chi)$ of the minimal normal subgroups. $\pi = \pi(\chi)$ determines χ up to elements of the W_ℓ applied to each letter of the codes $\overline{G}$ and $\overline{H}$. In other words, the set X of $G^* \to H^*$ isomorphisms we are looking for corresponds exactly to the set of (W_ℓ)-twisted equivalences of $\overline{G}$ and $\overline{H}$.

Analysis. Fixing diagonals will only add a factor of $|\mathrm{Soc}(G)|^2 \leq |G|^2$. The algorithm for groups with a unique minimal normal subgroup (part (b) of Corollary 1) takes polynomial time. The length of the codes is the number of minimal normal subgroups, which is $O(\log |G|)$. The alphabets Γ_ℓ are subgroups of $G^* \cong G$ and hence $(\forall \ell)(|\Gamma_\ell| \leq |G|)$. The groups W_ℓ have polynomial size by part (a) of Corollary 1, and faithful permutation representations of degree t_i. Therefore the permutation group where we will perform coset intersection will have a permutation representation of degree $k = \sum_{i,j} t_i$, the number of simple factors of $\mathrm{Soc}(G)$, which is $O(\log |G|)$. Finally the size of the codes themselves is the order of the groups. Therefore the total running time of the twisted code equivalence algorithm is polynomial. $\square$

6 Comparison with prior work

A 2003 paper by Cannon and Holt [6] describes a practical method to test isomorphism of permutation groups. Sec. 3 of their paper is dedicated to semisimple groups, underlining the significance of this class. Naturally, our framework is based on the same simple structural observations regarding the socle as theirs (Sec. 5.1); the most notable common element is part (a) of Obs. 8. After these initial observations, the two algorithms diverge in accordance with their very different goals: [6] describes heuristic algorithms with no performance guarantees and with reference to programs that use backtracking which would count as illegal steps for us; [6] reports practical efficiency. We devise algorithms which take time, polynomial in the *order* of the group, a prohibitive cost in their context.

References

1. M. Aschbacher and R. Guralnick. Some applications of the first cohomology group. *J. Algebra*, 90(2):446–460, 1984.
2. L. Babai and R. Beals. A polynomial-time theory of black-box groups I. In *Groups St Andrews'97 in Bath*, *LMS Lect. Notes*, 260, pp 30–64. Cambr. U. Press, 1999.
3. L. Babai, P. Codenotti, J. A. Grochow, Y. M. Qiao. Code equivalence and group isomorphism. In *Proc. 22nd SODA*, pp. 1395–1408, 2011.
4. L. Babai, Y. M. Qiao. Polynomial-time isomorphism test for groups with abelian Sylow towers. In *29th STACS*, pp. 453–464. Springer LNCS 6651, 2012.
5. P. J. Cameron. Finite permutation groups and finite simple groups. *Bull. London Math. Soc.*, 13(1):1–22, 1981.
6. J. J. Cannon, D. F. Holt. Automorphism group computation and isomorphism testing in finite groups. *J. Symb. Comput.*, 35:241–267, 2003.
7. V. Felsch and J. Neubüser. On a programme for the determination of the automorphism group of a finite group. In *Proc. Conf. on Computational Problems in Algebra, Oxford, 1967*, pages 59–60. Pergamon Press, 1970.
8. M. L. Furst, J. Hopcroft, E. M. Luks. Polynomial-time algorithms for permutation groups. In *Proc. 21st FOCS*, pages 36–41. IEEE Comp. Soc., 1980.
9. T. Kavitha. Linear time algorithms for abelian group isomorphism and related problems. *J. Comput. Syst. Sci.*, 73(6):986–996, 2007.
10. D. E. Knuth. Efficient representation of perm groups. *Combinat.*, 11:57–68, 1991.
11. F. Le Gall. Efficient isomorphism testing for a class of group extensions. In *26th STACS*, pp. 625–636, 2009.
12. E. M. Luks. Hypergraph isomorphism and structural equivalence of boolean functions. In *Proc. 31st ACM STOC*, pages 652–658. ACM Press, 1999.
13. E. M. Luks and T. Miyazaki. Polynomial-time normalizers for permutation groups with restricted composition factors. In *13th ISAAC*, pp. 176–183, 2002.
14. E. M. Luks and Á. Seress. Computing the Fitting subgroup and solvable radical for small-base permutation groups in nearly linear time. In *Workshop on Groups and Computation II*, DIMACS Series in DMTCS, pp. 169–181, 1991.
15. G. L. Miller. On the $n^{\log n}$ isomorphism technique. *10th STOC*, pp. 51–58, 1978.
16. Y. M. Qiao, J. M. N. Sarma, B. Tang. On isomorphism testing of groups with normal Hall subgroups. In *Proc. 28th STACS*, pages 567–578, 2011.
17. D. J. S. Robinson. *A Course in the Theory of Groups.* Springer, 2nd ed., 1996.
18. Á. Seress. *Permutation Group Algorithms.* Cambridge Univ. Press, 2003.
19. C. C. Sims. Computation with permutation groups. In S. R. Petrick, editor, *Proc. 2nd Symp. Symb. Algeb. Manip.*, pages 23–28. ACM Press, 1971.
20. R. Steinberg. Generators for simple groups. *Canad. J. Math.*, 14:277–283, 1962.
21. M. Suzuki. *Group Theory II.* Springer, 1986.
22. J. B. Wilson. Decomposing p-groups via Jordan algebras. *J. Algebra*, 322:2642–2679, 2009.
23. J. B. Wilson. Finding central decompositions of p-groups. *J. Group Theory*, 12:813–830, 2009.