Computing the Composition Factors of a Permutation Group in Polynomial Time

Eugene M. Luks[1]

Department of Computer and Information Science
University of Oregon
Eugene, OR 97403

Abstract

Given generators for a group of permutations, it is shown that generators for the subgroups in a composition series can be found in polynomial time. The procedure also yields permutation representations of the composition factors.

1) Research supported by National Science Foundation Grants DCR-8403745 and DCR-8609491

Introduction

The order of a permutation group G on n letters can be exponential in n. So it is natural, for computational purposes, to avoid listing the group elements, specifying G only by a generating set of permutations. But this, in turn, leads to the question of whether such a concise representation admits polynomial-time solutions to basic questions concerning the structure of the group. Motivated by links, suggested by Babai [3], to the computational complexity of the graph isomorphism problem, Furst, Hopcroft, and Luks [8] undertook the study of this issue. It was shown, in particular, that an algorithm of Sims [15] can be implemented in polynomial time, thus establishing several basic tools for complexity studies (see section 1). Additional results developed in [13] brought a significant instance of graph-isomorphism testing into polynomial time. In this paper, we extend the polynomial-time machinery, demonstrating, in particular, an algorithm for exhibiting the ''building blocks'' of the group, that is, its composition factors. For the correctness proof, Schreier's Conjecture (that the outer automorphism group of a finite simple group is solvable) is extracted from the classification of finite simple groups.

The composition factors of a group are essential tools in several computational problems, as they are in theory. Indeed, the results of this paper have already been applied extensively by Kantor [10],[11] to the polynomial-time construction of Sylow subgroups and related problems. Completing a cycle, there are applications also in the graph isomorphism problem ([5], [14]). The main result of this paper was announced in [13, section 4] and a sketch of the algorithm appeared in [5].

The results of this paper are also used by Babai, Luks, and Seress [4] in the parallelization of solutions to fundamental permutation group problems. In fact, they show the problem of computing composition factors is, itself, highly parallelizable, that is, it is in the class NC (solvable in time $(\log n)^{\text{constant}}$ using a polynomial number of processors).

Notation, definitions, and some known polynomial-time results are given in section 1. In section 2, we outline properties of primitive permutation groups that guide the algorithm; these properties are derived from the O'Nan-Scott Theorem (see [6]). Of independent interest is the problem, solved in polynomial time in section 3, of finding the centralizer of a normal subgroup of a permutation group. In section 4, we discuss the problem of finding a solvable normal subgroup, if one exists; this subcase of simplicity-testing uses only elementary ideas. The general simplicity test is given in section 5. If the group is non-simple, a proper normal subgroup is produced as witness. Finally, in section 6, it shown how to replace a non-maximal normal subgroup by a larger proper normal subgroup. Thus, one can obtain a maximal normal subgroup, and, by repeating the process, a composition series. The algorithm also returns faithful permutation representations of the composition factors.

We restrict our attention to the issue of polynomial time without digressing, at this point, to optimize the time bounds.

1. Preliminaries and basic algorithms

We write $H \le G$ to indicate that H is a subgroup of G and $H < G$ to indicate that H is a subgroup $\ne G$. If $H \le G$, we denote by G/H the set of right cosets of H in G. To specify that H is a normal subgroup of G we write $H \lhd G$. The *centralizer* in G of any subset A is $\{ \sigma \in G \mid \sigma\alpha = \alpha\sigma \text{ for all } \alpha \in A \}$ and is denoted $\mathrm{Ctr}_G(A)$. If $H \le G$, we denote by $\mathrm{Ncl}_G(H)$ the *normal closure of* H *in* G, that is, the smallest K such that $H \le K \lhd G$, and by $\mathrm{Cor}_G(H)$

the *core of* H *in* G, that is, the largest K such that $K \le H$ and $K \lhd G$. The socle of a group G, denoted $\mathrm{Soc}(G)$, is the group generated by all minimal normal subgroups of G. If Φ is a subset of a group G then $\langle \Phi \rangle$ denotes the subgroup generated by Φ. If $P = H_1 \times \cdots \times H_r$ is a direct product of groups each isomorphic to a fixed group H, a *diagonal* of P is a subgroup comprised of $\{ (f_1(\sigma), \ldots, f_r(\sigma)) \mid \sigma \in H \}$, where, for each i, $f_i : H \to H_i$ is a fixed isomorphism; typically the f_i do not require explication and we denote the diagonal subgroup by $\mathrm{Diag}(P)$.

The group of all permutations of the set X is denoted $\mathrm{Sym}(X)$. A permutation group $G \le \mathrm{Sym}(X)$ will always be given by a set of generating permutations. Frequently, we refer to *actions* of G on other sets, that is, homomorphisms $G \to \mathrm{Sym}(Y)$ that are not necessarily *faithful* (injective). Suppose G acts on Y. The image of $y \in Y$ under $\sigma \in G$ is denoted by y^σ. If $y_1, y_2, \ldots, y_r \in Y$ we denote by $G_{y_1 y_2 \cdots y_r}$ the subgroup $\{ \sigma \in G \mid \forall i,\ y_i^\sigma = y_i \}$. The *orbit* of $y \in Y$ is $\{ y^\sigma \mid \sigma \in G \}$. Then G acts *transitively* if Y consists of a single orbit. Suppose G acts transitively on Y. We say G acts *regularly* (or that G is *regular* when $G \le \mathrm{Sym}(Y)$) if $G_y = 1$ for any (and therefore all) $y \in Y$. A *G-block* (or *block* if the group is clear) is a proper subset Z of Y such that, for all $\sigma, \tau \in G$, Z^σ and Z^τ are disjoint or identical. We say that G acts *primitively* (or, when $G \le \mathrm{Sym}(Y)$, that G is *primitive*) if there are no blocks in Y of size > 1. If Z is a G-block then the induced set of blocks $\{ Z^\sigma \mid \sigma \in G \}$ is called a *block system in* Y. A block system is called *minimal* if G acts primitively on the collection of blocks.

We recall some of the permutation-group constructions that can be carried out in polynomial time:

Lemma 1.1 Given generators for $G \le \mathrm{Sym}(X)$, the following problems have polynomial-time solutions.

(1) Find the orbits of G.

(2) If G is transitive, find a minimal block system in X.

(3) Find the order of G.

(4) Given $\sigma \in \mathrm{Sym}(X)$, test whether $\sigma \in G$.

(5) Given $x_1, \ldots, x_r \in X$, find generators for $G_{x_1 \cdots x_r}$.

(6) Given generators for $H \le \mathrm{Sym}(X)$, test whether $H \le G$ and, if so, whether $H \lhd G$.

(7) Given generators for $H < G$, find generators for $\mathrm{Ncl}_G(H)$.

(8) Given an action $f : G \to \mathrm{Sym}(Y)$ (specified by the images in $\mathrm{Sym}(Y)$ of the generators of G), find generators for the kernel of f.

Discussion: A transitive closure algorithm (see, e.g., [1, chapter 5]) suffices for (1) (consider the directed graph on X induced by the generators). For (2), one finds any non-trivial block system (for example, fixing x find, for all y, the unique smallest block containing x and y by taking the component of x in the undirected graph whose edges are given by the G-orbit of $\{x, y\}$ in the set of all unordered pairs); Akinson [2] first observed this to be in polynomial time. Repeat, considering the action of G on the block system, until G acts primitively. Problems (3),(4),(5) were shown to be in polynomial time in [8] using, in effect, a version of Sims' algorithm [15]. Problem (6) follows directly from (4); for $H \lhd G$, it suffices to test for membership in H the conjugates of the generators of H by the generators of G. As observed in [8], Problem (7) is an

easy extension of (6) for, when one of the conjugates fails the membership test, increase H by adding this element to the generating set. Problem (8) is a corollary to (5) (for, consider the faithful embedding of G in $\mathrm{Sym}(X \cup Y)$ and fix all the points in Y).

2. Primitive permutation groups

We summarize some properties of primitive permutation groups (see, e.g., [6]).

Suppose $G \leq \mathrm{Sym}(X)$ is primitive. Any normal subgroup of G is transitive on X (its orbits are blocks for G). If non-trivial normal subgroups N, M centralize each other then both M and N are regular (the fact that M centralizes a transitive group implies $M_x = 1$, for any $x \in X$) and each is precisely the full centralizer in G of the other (for any $x, y \in X$, there is at most one permutation in X that commutes with a given transitive group and maps x to y); furthermore, identifying the (regular) action of M on X with the right-regular action of M on itself (i.e., by right multiplication), we see that the action of N must correspond to the left-regular action of M, and so N and M are isomorphic. It follows that G has either one or two minimal normal subgroups, and in the latter case, both are regular. In either case,

$$\mathrm{Soc}(G) = T_1 \times \cdots \times T_r,$$

where $T_1, \ldots, T_r$ are all isomorphic to a simple group T.

O'Nan and Scott have offered a classification of the structure and action of a primitive permutation group G according to the nature of $\mathrm{Soc}(G)_x$ (see [6, section 4] for proofs and additional details). We extract a subset of the O'Nan-Scott Theorem for use in our main algorithms. Observe that, if H acts transitively on X, then $|X| = |H|/|H_x|$, for any $x \in X$.

Proposition 2.1 Let G be a primitive subgroup of $\mathrm{Sym}(X)$. Then exactly one of the following holds,

Case 1. $\mathrm{Soc}(G)$ is regular or the direct product of two regular normal subgroups.

Case 2. $\mathrm{Soc}(G)$ is nonabelian and is the unique minimal normal subgroup of G (so that G acts, by conjugation, transitively on $\#[T_1, \ldots, T_r]\#$). For any $x \in X$, $1 < \mathrm{Soc}(G)_x < \mathrm{Soc}(G)$ and

$$\mathrm{Soc}(G)_x = T_{1x} \times \cdots \times T_{rx} .$$

Thus, $|X| = (|T|/|T_{1x}|)^r$.

Case 3. $\mathrm{Soc}(G)$ is nonabelian and is the unique minimal normal subgroup of G. Also, $r = kl$, with $k > 1$, and upon renumbering of the T_i's, the sets $\#[T_{(i-1)k+1}, \ldots, T_{ik}\}$, $1 \leq i \leq l$ form blocks for the conjugacy action of G. For any $x \in X$,

$$\mathrm{Soc}(G)_x = D_1 \times \cdots \times D_l .$$

where D_i is a diagonal subgroup of $T_{(i-1)k+1} \times \cdots \times T_{ik}$. Thus, $|X| = |T|^{r-l}$.

We warn the reader that we have slightly rearranged the subcases of the O'Nan-Scott Theorem to suit our purposes, in particular, to isolate Case 1. Note that Case 1 includes the situation that $\mathrm{Soc}(G)$ is abelian, whence regular. It also subsumes the possibility that $\mathrm{Soc}(G)$ is nonabelian and regular; the discussion in [6] implicitly excludes this subcase.

3. Centralizer of a normal subgroup

The algorithm in the next section requires a procedure for finding $\mathrm{Ctr}_G(N)$, given $N \lhd G$. Private communication describing our polynomial-time solution to this problem has led to significant applications by Kantor [10],[11] and Kantor-Taylor [12], and has inspired an alternate approach for the special case of $\mathrm{Ctr}_G(G)$, i.e., the center of G, by Hoffman [9]. However, this original version has not appeared.

More generally, we need only assume G normalizes N. We offer a polynomial-time algorithm for

Problem 3.1 Centralizer of Normalized Group.
Input: Generators for $G, N \leq \mathrm{Sym}(X)$, where G normalizes N.
Output: Generators for $\mathrm{Ctr}_G(N)$.

For $\sigma \in \mathrm{Sym}(X)$, set

$$\Gamma_\sigma = \{ (x, x^\sigma) \mid x \in X \} \subset X \times X .$$

Then, with the natural action of $\mathrm{Sym}(X)$ on $X \times X$ ($(x,y)^\tau = (x^\tau, y^\tau)$), for $\tau \in \mathrm{Sym}(X)$, $\sigma\tau = \tau\sigma$ iff τ stabilizes Γ_σ. Thus, if $N = \langle \Psi \rangle$, τ centralizes N iff τ stabilizes Γ_ψ, for all $\psi \in \Psi$. The collection $\{ \Gamma^\psi \mid \psi \in \Psi \}$ induces a coloring of $X \times X$ in which two elements are colored alike iff they belong to precisely the same Γ^ψ's. Then, τ centralizes N iff τ is a "color automorphism" (preserves colors) on $X \times X$.

By the above, Problem 3.1 involves finding $\mathrm{Col}_G(Y)$, the subgroup of color automorphisms in a group G that acts on a colored set Y. It is unfortunate that no subexponential-time algorithm is known for $\mathrm{Col}_G(Y)$ in general, for graph isomorphism reduces to it as well [13]. However, we have additional information in the present problem, namely, since G normalizes N, G normalizes $\mathrm{Ctr}_G(N) = \mathrm{Col}_G(Y)$. Thus, we need only find $\mathrm{Col}_G(Y)$ in the instance when it is known (somehow) to be normal. More generally, we solve

Problem 3.2 Core of Color Automorphism Subgroup.
Input: Generators for $G < \mathrm{Sym}(Y)$, Y a colored set.
Output: Generators for $\mathrm{Cor}_G(\mathrm{Col}_G(Y))$.

The polynomial-time algorithm for Problem 3.2 uses

Proposition 3.3 Suppose Y is a colored set and $G < \mathrm{Sym}(Y)$. Then $\mathrm{Cor}_G(\mathrm{Col}_G(Y))$ is the kernel of an induced action of G on a set Z with $|Z| \leq |Y|$.

Proof: Let $K = \mathrm{Cor}_G(\mathrm{Col}_G(Y))$. For $C \subset Y$, $\sigma, \tau \in \mathrm{Sym}(Y)$, τ stabilizes C iff $\sigma^{-1}\tau\sigma$ stabilizes C^σ. Hence, if C is a color class and $\sigma \in G$, C^σ is stabilized by $\sigma^{-1}K\sigma = K$. It follows that K stabilizes the cells in the coarsest refinement of the given color partition that is compatible with the action of G (i.e., the generators of G map cells to cells). Denoting the collection of cells in that refinement by Z, we have then an induced action $f: G \to \mathrm{Sym}(Z)$ such that $K \leq \mathrm{kernel}(f)$. On the other hand, $\mathrm{kernel}(f) \leq \mathrm{Col}_G(Y)$ since the color classes are unions of cells in Z. It follows that $K = \mathrm{kernel}(f)$. □

From above remarks

Corollary 3.4 The center of $G \leq \mathrm{Sym}(X)$ is the kernel of an action of G on a set of size $\leq |X|^2$.

To solve Problem 3.2, and therefore Problem 3.1, in polynomial time, it suffices to observe that the refinement, Z, is obtainable in polynomial time. In fact, using [1, Algorithm 4.5], it costs $O(|Y| \log |Y|)$ to achieve compatibility of a partition with any given generator of G.

Remark. In practice, the center of $G < \mathrm{Sym}(X)$ is typically computed by cutting G down in stages, each of which centralizes an additional generator of the starting group. Though it seems to have efficient implementations, we note that there is no known polynomial-time algorithm following this approach. Indeed, finding $\mathrm{Ctr}_G(\sigma)$ for $\sigma \in \mathrm{Sym}(X)$ is polynomial-time equivalent to finding the subgroup of a permutation group that stabilizes a given subset (which, in turn, is at least as hard as graph isomorphism [13]). For, suppose we want to find the stabilizer in $G < \mathrm{Sym}(X)$ of $W \subset X$. Let G act naturally on the disjoint union $\bar{X} = X \dot{\cup} X$ of two copies of X, and let $\sigma \in \mathrm{Sym}(\bar{X})$ switch corresponding elements in the two copies of W while it fixes every other point. Then $\mathrm{Ctr}_G(\sigma)$ is precisely the stabilizer in G of W. The reverse reduction has been indicated earlier.

4. Solvable normal subgroup

We discuss testing for the presence of a solvable normal subgroup both for the intrinsic interest of this case and because it employs only elementary ideas. A key observation (Lemma 4.3) will play a role in the next section.

A polynomial-time algorithm is given for

Problem 4.1 Solvable Normal Subgroup (SNS).
Input: Generators for $G \leq \mathrm{Sym}(X)$.
Output: Generators for a non-trivial solvable normal subgroup of G or a report that none exists.

We first reduce the problem to one of locating *any* proper normal subgroup though only under the assumption that a non-trivial solvable normal subgroup exists. In this reduction we observe that the proper normal subgroup of G facilitates an effective divide-and-conquer procedure. For this, suppose we have $K \lhd G$. Then we could proceed as follows:

solve SNS for K
if some solvable $H \lhd K$ was found **then output** $\mathrm{Ncl}_G(H)$
else solve SNS for $\mathrm{Ctr}_G(K)$
  if some solvable $H \lhd \mathrm{Ctr}_G(K)$ was found **then output** $\mathrm{Ncl}_G(H)$
  else output "G does not have a non-trivial solvable normal subgroup"

To see why this reduction works, note that if H is solvable and $H \lhd N \lhd G$ then $\mathrm{Ncl}_G(H)$ is generated by solvable normal subgroups of N (the G-conjugates of H) and is solvable; on the other hand, any normal subgroup of G either intersects K non-trivially or centralizes K, so that a piece of any solvable normal subgroup must show up in K or in $\mathrm{Ctr}_G(K)$. We must also consider the effect of such reductions on the timing. Letting $t(G)$ denote the time required to solve SNS for G, the reduction gives

$t(G) \leq t(K) + n^c$, if K has a solvable normal subgroup,
$t(G) \leq t(K) + t(\mathrm{Ctr}_G(K)) + n^c$, otherwise,

where $n = |X|$ and the n^c subsumes the cost of elementary operations and finding normal

closures, as well as centralizer of a normal subgroup (section 3). We point out, if the second recursive call to SNS has to be employed, then we know that $K \cap \mathrm{Ctr}_G(K) = 1$ so that $|G| \geq |K| |\mathrm{Ctr}_G(K)|$. Thus, if we show that K can be found in polynomial time, say again within $O(n^c)$ steps, it will follow that $t(G) \leq O(\log(|G|)\, n^c$.

We may assume G is not cyclic of prime order. By the above, it suffices to indicate a polynomial-time algorithm for

Problem 4.2

Input: Generators for $G \leq \mathrm{Sym}(X), |G|$ not prime.

Output: Generators for a proper normal subgroup of G or a report that G does not have a non-trivial solvable normal subgroup.

Note, if G does not have a non-trivial solvable normal subgroup, the algorithm for Problem 4.2 may or may not come up with a proper normal subgroup.

To solve Problem 4.2, we search for normal subgroups in the kernels of certain induced actions. The trick is to build a collection of such actions sufficient to bring out a proper kernel. We consider, first, the kernel of the action on a non-trivial orbit. If this kernel is non-trivial, we are done. Otherwise, G acts faithfully on the orbit, which, we may assume to be all of X. Breaking X into a minimal system of imprimitivity blocks, we consider the kernel of the action on the set of blocks. Again, if the kernel is trivial, we replace X by the set of blocks, on which G acts faithfully and, now, primitively.

We next construct a collection, $\{G \to B_z\}_z$, of at most n induced actions of G on domains each of size less than n^2. For this construction,

Fix any two distinct points $x, y \in X$.
Find all $z \in X$ such that
(i) z is a fixed point of G_{xy}, and
(ii) for some $\sigma \in G$, $(x, y)^\sigma = (y, z)$
Each such z gives rise to an action of G as follows:
fix a σ such that $(x, y)^\sigma = (y, z)$ and form the σ-orbit, O_z, of x;
viewing O_z as a directed cycle $(\cdots \to x \to y \to z \to \cdots)$, form the set Y_z of all G-images of O_z (the cycles $\cdots \to x^\rho \to y^\rho \to z^\rho \to \cdots$ for $\rho \in G$),
output a minimal G-block system B_z in Y_z.

A cycle O_z^ρ in the collection Y_z is completely determined by any edge, $u \to v$, in it, for an appearance of $u \to v \to w$ implies that (u, v, w) is in the G-orbit of (x, y, z), and so, by (i), w, and similarly the rest of O_z^ρ, is uniquely determined by $u \to v$. Thus, $|B_z| \leq |Y_z| < n^2$.

We need to show, under the assumption that G is primitive (on X) and has a non-trivial solvable normal subgroup, that G acts infaithfully on some B_z. For such G, $\mathrm{Soc}(G)$ is necessarily abelian and regular (Proposition 2.1), in particular, $|\mathrm{Soc}(G)| = |X| = n$. For further application we prove more generally

Lemma 4.3 Suppose that G has a proper normal subgroup, N, and that N acts regularly on X. Then none of the induced actions $G \to \mathrm{Sym}(B_z)$ are trivial. Further, if they are all faithful then at least one involves a permutation domain smaller than X.

Proof of Lemma 4.3: Observe that $|B_z| = 1$ only if $|Y_z| = 1$ and the latter would imply not only that the vertices of O_z comprise all of X, but also that G_x fixes all vertices in O_z (if either

failed there would be distinct images of O_z). Since N and G cannot both be regular, the actions are non-trivial. The regularity of N also means that for any $u, v \in X$ there is a unique $\nu \in N$ satisfying $u^\nu = v$. Suppose then $x^\mu = y$, where $\mu \in N$ and let $w = y^\mu$. Let $\tau \in G_{xy}$. Since $x^{\tau\mu\tau^{-1}} = y$ and $\tau\mu\tau^{-1} \in N$, the uniqueness of μ forces $\tau\mu\tau^{-1} = \mu$. Thus

$$w^\tau = y^{\mu\tau} = y^{\tau\mu} = y^\mu = w.$$

Hence, w satisfies (i) and (ii), so that a G-action will be constructed on a set B_w. It suffices to show that, if the kernel of $G \to \mathrm{Sym}(B_w)$ is trivial then $|B_w| < |X|$. Now we employ, in the construction of B_w, *any* $\sigma \in G$ that we find to satisfy $x^\sigma = y$ and $y^\sigma = w$. In general, $\sigma \neq \mu$. However, since both μ and $\sigma\mu\sigma^{-1}$ are elements of N that map x to y, we know $\mu = \sigma\mu\sigma^{-1}$. It follows that the orbits (directed cycles) of x under μ and σ are identical. (Thus, we shall have constructed an orbit of the element $\mu \in N$ without having any element of N in hand). Since μ fixes its own orbits, μ has a fixed point in Y_w, namely O_w, and the corresponding block is a fixed point for μ in the action on B_w. This is a primitive action of G. Hence, if it is faithful then N would act transitively on B_w, and since we know N does not act regularly on B_w, that would imply $|B_w| < |N| = |X|$. □

Finally, to complete the discussion of Problem 4.2, we show that if G is primitive on X and $\mathrm{Soc}(G)$ is abelian then G cannot act faithfully on a domain B_z with $|B_z| < |X|$. By construction, G acts primitively on B_z. If the action is faithful then $\mathrm{Soc}(G)$ would again act regularly. This is impossible on a smaller domain.

Of course, failure to find any non-trivial kernels in the collection of actions $\{G \to \mathrm{Sym}(B_z)\}_z$ would mean G does not have a non-trivial solvable normal subgroup.

5. Normal subgroups

We describe a polynomial-time algorithm for

Problem 5.1 Proper Normal Subgroup (PNS)
Input: Generators for $G \leq \mathrm{Sym}(X)$, $G \neq 1$.
Output: Generators for a proper normal subgroup of G or the report "G is simple".

It is convenient to present the main idea in an algorithm for the following Problem 5.2. Clearly, repeated application when the output is of type (iii) yields an algorithm for PNS.

Problem 5.2
Input: Generators for $G \leq \mathrm{Sym}(X)$, $G \neq 1$.
Output: One of the following.
(i) The report "G is simple".
(ii) Generators for a proper normal subgroup of G.
(iii) A faithful action of G on a domain smaller than X.

In all but one case (when G has a normal subgroup of small index) outputs of types (ii) or (iii) are discovered in induced actions of G. Each constructed action gives rise to a primitive action in which we test the kernel and the size of the domain. The procedure TEST_ACTION formalizes these steps. The input, Y, is a set, with $|Y| > 1$, on which G acts transitively (it is assumed globally that G is faithfully represented on the set X and $n = |X|$).

procedure TEST_ACTION(Y)
begin
B := a minimal G-block system in Y ;
N := the kernel of the G-action on B ;
if $N \neq 1$ **then** (**output** N; **halt**);
if $|B| < n$ **then** (**output** $\#[G \to \mathrm{Sym}(B)]$; **halt**)
end

Algorithm for Problem 5.2:

Step 1. Y := any non-trivial orbit of G in X;
TEST_ACTION(Y)

Step 2. **if** $|G| = n$ **then** (**output** "G is simple (abelian)"; **halt**)

Step 3. A := any subset of G of size $n+1$;
for all $\sigma, \tau \in A, \sigma \neq \tau$, **do**
begin
$N := \mathrm{Ncl}_G(\langle \sigma\tau^{-1} \rangle)$;
if $N \neq G$ **then** (**output** N; **halt**)
end

Step 4. x, y := any two distinct points in X;
for all $z \in X$ **do**
if z is fixed by G_{xy} **and** $(x,y)^\sigma = (y,z)$ for some $\sigma \in G$ **then**
begin
fix any σ such that $(x,y)^\sigma = (y,z)$;
O_z := the orbit (directed cycle) of x under σ;
Y_z := the collection of G-images of O_z;
TEST_ACTION(Y_z)
end

Step 5. x := any point in X;
for all $y, z, w \in X$ **do**
begin
$H := \langle G_{xy}, G_{zw} \rangle$;
if $H \neq G$ **then** TEST_ACTION(G/H)
end

Step 6. x := any point in X;
for all $y \in X \setminus \#[x]\#$ **do**
begin
Z_y := the G-orbit of $\{x, y\}$ (in set of unordered pairs); TEST_ACTION(Z_y)
end

Step 7. **output** "G is simple (nonabelian)"

Proposition 5.3 The above algorithm solves Problem 5.2 in polynomial time.

Proof: That the algorithm can be implemented in polynomial-time follows easily from Lemma 1.1 and, for Step 4, section 4 (where it is shown that Y_z has polynomial size). We need to establish its correctness.

It is clear that the algorithm halts correctly if it encounters a proper normal subgroup (in Step 3 or in a call to TEST_ACTION) or a smaller domain (in TEST_ACTION). If Step 1 is passed then G acts primitively on X. If, at this point, $|G| = n$ then G acts regularly and so n must be prime (in the regular action any proper subgroup would correspond to an imprimitivity block); the algorithm would correctly halt at Step 2. It is sufficient now to show that, if the algorithm reaches Step 7, then G is simple.

Having passed Step 2, $|G| > n$, so that a set A will be available in Step 3. If Step 3 is passed, then G cannot have any proper normal subgroup N with $[G: N] \le n$, otherwise some distinct $\sigma, \tau \in A$ would be congruent modulo N whence $\mathrm{Ncl}_G(\langle \sigma\tau^{-1} \rangle) \le N < G$.

By Lemma 4.3, if Step 4 is passed then G does not have a regular normal subgroup, eliminating Case 1 of Proposition 2.1.

Claim 1: Suppose the action of G on X falls into Case 2 of Proposition 2.1 with $r > 1$ or Case 3 with $l > 1$. If Step 5 is reached, the algorithm will halt there (i.e., in a call to TEST_ACTION).

Proof of Claim 1: We extract from Proposition 2.1

(a) $\mathrm{Soc}(G) = N_1 \times N_2 \times \cdots \times N_m$, with $m > 1$
(b) G acts (by conjugation) transitively on $\{N_1, \ldots, N_m\}$.
(c) $\mathrm{Soc}(G)_x = N_{1_x} \times \cdots \times N_{m_x}$
(d) N_{i_x} is a proper subgroup of N_i ,

where

in Case 2: $m = r$, $N_i = T_i$, and $n = (|T|/|T_{1_x}|)^r$,

in Case 3: $m = l$, $N_i = T_{(i-1)k+1} \times \cdots \times T_{ik}$, $N_{i_x} = \mathrm{Diag}(N_i)$, and $n = |T|^{r-l}$.

By (d), there is some $\nu \in N_2$ such that $x^\nu \ne x$. In running Step 5, one eventually encounters $y = x^\nu$. Suppose now $\tau \in G_{xy}$ and say $\tau^{-1} N_2 \tau = N_j$. We have

$$x^{\nu\tau^{-1}\nu^{-1}\tau} = y^{\tau^{-1}\nu^{-1}\tau} = y^{\nu^{-1}\tau} = x^\tau = x ,$$

so that $\nu\tau^{-1}\nu^{-1}\tau \in \mathrm{Soc}(G)_x$. But $\nu\tau^{-1}\nu^{-1}\tau \in N_2 N_j$. Since $\nu \notin N_{2_x}$, we conclude by (c) that j must be 2. Hence, in the action of G on $\{N_1, \ldots, N_m\}$, G_{xy} fixes N_2 . By (d) again, there is some $z \in X$ that is not fixed by N_{1_x}. In running Step 5, one eventually encounters such z as well as some w such that G_{zw} fixes N_2 . For this $\{x, y, z, w\}$ then, H fixes N_2 so that $H \ne G$ and the non-trivial action of G on G/H will be constructed and tested. Assume that the resulting primitive action on B (in performing TEST_ACTION(G/H)) is faithful. We shall show that $|B| < n$. For this, we consider the stabilizer G_H of the ''point'' H in the action of G on G/H ; of course, $G_H = H$. Now, since ν commutes with N_1, $N_{1_x} = N_{1_y}$, and so

$$N_{1_x} \le G_{xy} \le H = G_H .$$

Similarly, $N_{1_z} \le G_H$. By assumption $N_{1_x} \ne N_{1_z}$. But, since these subgroups have the same size, together they generate a strictly larger subgroup. We have

$$N_{1_x} < \langle N_{1_x}, N_{1_z} \rangle \le N_{1_H} \le N_{1_h} ,$$

where h denotes the block in B containing H (actually, the cosets in h comprise a maximal subgroup of G containing H). If the action of G on X is in Case 2, the primitive action on B is also in Case 2 (since, for example, a non-trivial subgroup of $T_1 = N_1$ fixes a point); in this case

$$|B| \le \frac{1}{2}\left(\frac{|T|}{|T_{1_h}|}\right)^r < \left(\frac{|T|}{|T_{1_x}|}\right)^r = \frac{n}{2} .$$

If the action of G on X is in Case 3, the primitive action on B is also in Case 3 (since, for example, N_{1_h} projects onto T_1 because N_{1_x} does); in this case, the fact that N_{1_h} is strictly larger than $N_{1_x} = \mathrm{Diag}(N_1)$, implies it must be the product of diagonal subgroups corresponding to a proper partition of $\{1,\ldots,k\}$ into cells of size $k' < k$. Hence, if $l' = r/k'$,

$$|B| = |T|^{r-l'} \leq |T|^{r-l}/2 = n/2.$$

This proves Claim 1.

Claim 2: Suppose the action of G on X falls into Case 3 of Proposition 2.1 with $l = 1$. So $n = |T|^{r-1}$ $(k = r)$. If Step 6 is reached, the algorithm will halt there.

Proof of Claim 2: Note that the sets Z_y constructed in Step 6 are non-trivial, i.e., $|Z_y| > 1$, otherwise $n = 2$ and the algorithm would have halted by Step 2. Let $\sigma \in T_1$ have order 2 (such σ exists by the Feit-Thompson Theorem [7]). Running Step 6, one eventually encounters $y = x^\sigma$ $(x^\sigma \neq x$ since $T_{1_x} = 1)$. Assuming that a kernel is not detected in the call TEST_ACTION(Z_y), a faithful primitive action $G \to \mathrm{Sym}(B)$ is constructed in which σ has a fixed point, namely the block containing $\{x, y\}$. This action then is in Case 2 of Proposition 2.1 and so $|B| = (|T|/|H|)^r$. where H is a proper subgroup of T. It remains to show that $|B| < n$. Suppose, to the contrary, $|B| \geq n$. Then

$$|T|^{r-1} = n \leq \left(\frac{|T|}{|H|}\right)^r \leq \frac{|T|^r}{2^r}.$$

Hence, $|T| \geq 2^r$ and so $n \geq 2^{r(r-1)} > r!$. Then the kernel, K, of the action of G on $\{T_1,\ldots,T_r\}$ has index $< n$. Also, since G acts transitively on $\{T_1,\ldots,T_r\}$, $K \neq G$. But, in such case, the algorithm would have halted in Step 3, outputting a proper normal subgroup. This proves Claim 2.

By the above, if Step 7 is reached, the action of G on X must fall into Case 2 with $r = 1$. Thus, $\mathrm{Soc}(G)$ is a nonabelian simple group, T. Then G is faithfully embedded in $\mathrm{Aut}(T)$ ($\sigma \to$ inner automorphism induced by σ; this is faithful since the center of G must be trivial, else it would intersect $\mathrm{Soc}(G)$ non-trivially). Also $G = G'$ (derived group), otherwise a maximal normal subgroup containing G' would have prime index $\leq n$ and the algorithm would not have passed Step 3. (We could also have inserted a step that simply outputs G' if its a proper subgroup [8]). To conclude, finally, we invoke Schreier's Conjecture, that is, the outer automorphism group of a finite simple group is solvable, which is known to hold by virtue of the classification of finite simple groups. It follows that $G = T$. □

6. Finding composition factors

A polynomial-time algorithm is given for

Problem 6.1 Composition Factors.
Input: Generators for $G \leq \mathrm{Sym}(X)$.
Output: Generators for the composition factors of G, and a representation of each composition factor on a set of size $\leq |X|$.

It suffices to show how to find generators for a maximal normal subgroup N of any given G together with a permutation representation of G/N.

If G is not simple, the algorithm for PNS (Problem 5.1) returns some proper normal subgroup M. Denote by $G_{(i)}$ the subgroup of G that fixes each of the the first i points of X (with $G_{(0)} = G$). Let j be the smallest integer such that $G_{(j+1)}M \neq G_{(j)}M$. Then G acts on

$$Y = G/G_{j+1}M = G_{(j)}M/G_{(j+1)}M\ ,$$

and $|Y| \leq [G_{(j)}: G_{(j+1)}] \leq |X| - j$. Let K be the kernel of this action, so $M \leq K$. We have in Sym(Y) a faithful action of G/K. Thus, we may use the algorithm for PNS to test simplicity of G/K. If G/K is simple, K is a maximal normal subgroup and we are done. Otherwise, PNS returns a proper normal subgroup of G/K which we may pull back to a subgroup L of G that is not contained in K (in fact, by retaining the action of G-elements on X through all computations on Y, L is constructed in the process of solving PNS). Replace M by LK and repeat.

We remark that the problem of faithfully representing G/N for general $N \lhd G$ appears more difficult. In fact, we do not know a reasonable criterion for deciding when G/N has a ''small'' faithful action.

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, ''The Design and Analysis of Computer Algorithms'', Addison-Wesley, Reading, Mass., 1974.

[2] M.D. Atkinson, *An algorithm for finding the blocks of a permutation group,* Math. Comp. 29 (1975), 911-913.

[3] L. Babai, *Monte Carlo algorithms in graph isomorphism testing,* Tech. Rep. 79-10, Dép. Math. et Stat., Univ. de Montréal, 1979.

[4] L. Babai, E.M. Luks, and A. Seress, *Permutation groups in NC,* to appear in Proc. 19th ACM Symp on Theory of Computing (1987).

[5] L. Babai, W.M. Kantor, and E.M. Luks, *Computational complexity and the classification of finite simple groups,* Proc. 24th IEEE Symp. Found. Comp. Sci. (1983), 162-171.

[6] P. Cameron, *Finite permutation groups and finite simple groups,* Bull. London Math. Soc., 13 (1981), 1-22.

[7] W. Feit and J.G. Thompson, *Solvability of groups of odd order,* Pacific J. Math, 13 (1963), 775-1029.

[8] M. Furst, J. Hopcroft, and E. Luks, *Polynomial-time algorithms for permutation groups,* Proc. 21st IEEE Symp. Found. Comp. Sci. (1980), 36-41.

[9] C.M. Hoffman, ''Group-Theoretic Algorithms and Graph Isomorphism,'' Lecture Notes in Comp. Sci. 136, Springer, Berlin, 1982.

[10] W.M. Kantor, *Sylow's theorem in polynomial time,* J. Comp. Syst. Sci. 30 (1985), 359-394.

[11] W.M. Kantor, *Polynomial-time algorithms for finding elements of prime order and Sylow subgroups,* J. Algorithms 6 (1985), 478-514.

[12] W.M. Kantor and D.E. Taylor, *Polynomial-time versions of Sylow's theorem,* J. Algorithms, to appear.

[13] E.M. Luks, *Isomorphism of graphs of bounded valence can be tested in polynomial time,* J. Comp. Syst. Sci. 25 (1982), 42-65.

[14] E.M. Luks, *The complexity of fixed valence graph isomorphism and the implications for general graph isomorphism*, in preparation.

[15] C.C. Sims, *Computational methods in the study of permutation groups*, in ‘‘Computational Problems in Abstract Algebra’’, (ed. J. Leech), Pergamon, New York, 1970, 169-183.