Notes on Polynomial-time Group Theory

William M. Kantor*
Mathematics Department, University of Oregon
Eugene, OR. 97403, USA
e-mail: kantor@math.uoregon.edu

1. Introduction

Throughout this paper let $G = \langle \Gamma \rangle$ be a subgroup of S_n generated by a subset Γ which may be assumed to be 'small' (say, of size $< n^2$; cf. Theorem 2.4iv). The goal will be to compute properties of G 'efficiently'. The notion of efficiency used here will be the existence of a polynomial-time algorithm. This is certainly not the only possible such notion: from a more pragmatic point of view it would be preferable to require that algorithms have been programmed and run well in practice for suitable values of n and for groups of greatest interest in group-theoretic research — as is the case, for example, in the group theory systems CAYLEY [7] and GAP [34]. On the other hand, the present polynomial-time emphasis occasionally leads to new methods that have more than just theoretical applicability. Moreover, insistence on polynomial time leads to a better understanding of why certain problems (such as intersecting subgroups) are hard and seem to require backtrack or other exhaustive search techniques (cf. §3).

After indicating some of the properties of G that can be obtained in polynomial time by using Sims' results or related algorithms (§2), I will turn to the more recent results due to Luks, Rónyai or myself [31, 35, 36, 19, 20, 23] (§4). The remainder of the paper consists of a survey of results. In all cases the details get fairly messy. The paper concludes with some remarks concerning similar types of questions about Galois groups (§7). Throughout the paper, relatively little will be proved, but easy proofs and some hints of methods will be provided.

The difficulties and interesting aspects of this subject can be illustrated by a standard result: Cauchy's Theorem that a finite group of order divisible by a prime p has an element of order p. The standard proofs of Cauchy's Theorem are essentially nonconstructive and are in no sense efficient; for example, they are obtained by viewing G as acting either on itself or on a set of size $> |G|$. Hence, an entirely different approach was required [17]. In practice, CAYLEY obtains an element of order p by randomly choosing elements of G until one is found of order divisible by p. This approach has recently been studied somewhat from the point of view of the probability of its success after relatively few random choices [15]; as with many of the recent work on algorithms for permutation

*Supported in part by NSF and NSA grants.

groups, that investigation rests heavily upon the classification of finite simple groups.

For the most part, this paper is concerned with a brief sampling of polynomial-time algorithms. However, at various points I will point out those algorithms, or variations on them, that are or appear to be practical — and have been or probably will be added to a group theory system. This is one of the most interesting and most recent developments in this area.

2. Fundamental algorithms

Since recursion or iteration will be used for subgroups of S_n, the following trivial result is very helpful.

Lemma 2.1. *If* $1 < H_1 < H_2 < \cdots < H_m \leq S_n$, *then* $m \leq n \log_2 n < n^2$.

Namely, by Lagrange's Theorem $2^i \leq |H_i| \leq n!$ for each i. As might be expected, a more detailed investigation produces a much better bound: $m < 2n$ [2, 8].

Let $G = \langle \Gamma \rangle \leq S_n = \mathrm{Sym}(X)$, where $X = \{1, 2, \ldots, n\}$. I begin with two simple results that give an indication of the meaning of 'polynomial time'. Note that in each situation an algorithm is needed which, for *any* G, produces the desired information. It must be emphasized that *all* subgroups of S_n mentioned throughout this paper are assumed to be specified by means of generating sets of permutations.

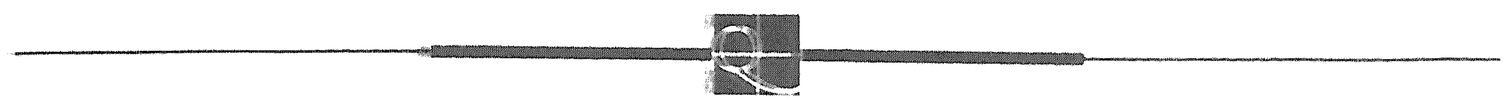
Recall that the orbits of G on X are the sets $\{x^g \mid g \in G\}$ for $x \in X$. (Here, x^g denotes the image of x under g.) These partition X. In particular, G is transitive on X if and only if there is just one orbit. Form the graph with vertex set X and edges $\{x, x^g\}$ for $g \in \Gamma$ and $x^g \neq x \in X$. This graph can be determined in time $O(n|\Gamma|)$. Since its connected components are just the orbits of G, this yields

Proposition 2.2. *In polynomial time all orbits of* G *can be found.*

A *block* for the action of a transitive group G is a nonempty subset Y of X such that $Y \cap Y^g = \emptyset$ whenever $g \in G$ is such that $Y^g \neq Y$. (Then $\Sigma := \{Y^g \mid g \in G\}$, the *block system* determined by Y, is a G–invariant partition of X, and hence produces a new permutation group G^Σ that is a homomorphic image of G.) Each block containing the point x has the form x^H for a uniquely determined subgroup H of G containing the *stabilizer* $G_x := \{g \in G \mid x^g = x\}$; and, conversely, if $G_x \leq H \leq G$ then x^H is a block. The *trivial blocks* containing x are the sets $\{x\}$ and X, corresponding to the choices $H = G_x$ and G, respectively. If the only blocks are trivial then G is called *primitive* (this is the case if and only if G_x is a maximal subgroup of G).

Proposition 2.3 [1]. *Assume that* G *is transitive on* X.

(i) *In polynomial time one can determine all blocks of size* > 1 *that are minimal*



with respect to inclusion (subject to having size > 1*).*

(ii) *In polynomial time a block system* Σ *of size* > 1 *can be found such that* G^Σ *is primitive.*

PROOF. (i) Find the orbits of G in its natural action on the cartesian product X^2 of X with itself (namely, $g\colon (x,y) \mapsto (x^g, y^g)$). Each orbit other than the diagonal $\{(x,x) \mid x \in X\}$ determines a graph with vertex set X and edges $\{x,y\}$ for (x,y) or (y,x) in the orbit. The set of connected components of such a graph is a block system Σ, and each minimal block system arises in this manner.

(ii) Iterate (i), replacing X by Σ and G by G^Σ if $|\Sigma| < n$. (The number of iterations is at most $\log n$.) □

Note that it is not possible (in polynomial time!) to find *all* block systems (consider the regular permutation representation of an elementary abelian 2–group).

Write $G_{12\ldots i} = \{g \in G \mid g \text{ fixes } 1, 2, \ldots, \text{and } i\}$. The most important algorithm for this survey is *Sims' algorithm*, which produces the following:

THEOREM 2.4 [37, 11]. *In polynomial time the following can all be determined:*

(i) *For any given* $f \in S_n$*, whether or not* $f \in G$;

(ii) $|G|$;

(iii) *For* $i = 1, \ldots, n$*, a set* Δ_i *such that* $G_{12\ldots i} = \langle \Delta_i \rangle$ *and* $|\Delta_i| < n$*; and*

(iv) *A set* Γ' *such that* $G = \langle \Gamma' \rangle$ *and* $|\Gamma'| < n^2$.

A rough idea of this is as follows. Consider (i). First use (2.2) to find a word h in the generators Γ such that h and g agree on 1, so that $hg^{-1} \in G_1$. This produces a recursive procedure requiring generators for the subgroup fixing the first i of the permuted points for each $i \geq 1$. Sims showed how to modify an arbitrary Γ so that it includes generators for these subgroups, thereby producing what he termed a 'strong generating set'; this is Γ' in (iv), and each $\Delta_i \subseteq \Gamma'$. Moreover, Γ' contains a set of coset representatives for $G_{12\ldots i+1}$ in $G_{12\ldots i}$ for each i; the product over i of the sizes of these sets is $|G|$. For $|\Gamma| = O(n^2)$ the algorithm in [11] runs in time $O(n^6)$; a concise, self-contained and easily programmable $O(n^5)$ variant on this was given in [26] (another $O(n^5)$ algorithm is in [16]). Very recently, an $O(n^4 \log^c n)$ algorithm for finding $|G|$ has been obtained in [6]. This should have the effect of speeding up all of the algorithms occurring in later sections. Moreover, the algorithm in [26] has already been used in [10]; while that in [6] as well as related algorithms [4] are presently undergoing modifications and testing (in particular, for inclusion in GAP).

There are many useful consequences of (2.4). The simplest is

PROPOSITION 2.5 [11]. *Given* $\Delta \subset G$*, in polynomial time the smallest normal subgroup* $\langle \Delta^G \rangle$ *of* G *containing* Δ *can be found.*

There is a simple recursive procedure for this which the reader will have no difficulty devising (cf. (2.1)). Here is an outline of an algorithm using a different

approach, developed in [9] for both (2.5) and the following result:

PROPOSITION 2.6. *Given $G, H \leq S_n$ such that G normalizes H, the intersection $G \cap H$ can be found in polynomial time.*

PROOF OF (2.5) AND (2.6). Let $X^* = X \cup X'$, where X' is a copy of X disjoint from X. Let G and H act on X' exactly as they act on X; this produces actions of $G \times G$ and $H \times H$ on X^*. Let $K := \langle \{(g,g) \mid g \in G\} \cup (1 \times H) \rangle$. Then it is not difficult to check that

$\langle H^G \rangle \cap G$ is the projection into $\mathrm{Sym}(X)$ of the pointwise stabilizer of X' in K, and

$\langle H^G \rangle$ is the projection into $\mathrm{Sym}(X')$ of the pointwise stabilizer of X in K.

Pointwise stabilizers can be found using (2.4). Thus, this computes $\langle H^G \rangle$; while if G normalizes H it computes $H \cap G$. □

COROLLARY 2.7 [11]. *The derived series and descending central series of G can be found in polynomial time. (Hence, solvability and nilpotence can be tested in polynomial time.)*

In CAYLEY [7] all intersections are handled in exactly the same manner, using a backtrack search: no distinction is made between the cases in which G does or does not normalize H. For further comments concerning intersections see §3. For now it is worth noting one further situation in which intersections can be found in polynomial time:

THEOREM 2.8 [30]. *Given an integer b, there is a polynomial-time algorithm which, when given $G, H \leq S_n$ such that all noncyclic composition factors of G have order $\leq b$, finds $G \cap H$.*

The algorithm for (2.8) is based in part on bounds in [3] for the order of primitive permutation groups all of whose composition factors are bounded as in (2.8). It requires time $O(n^{f(b)})$ with $f(b) \to \infty$ as $b \to \infty$. Thus, while polynomial-time, for large b it is perhaps unreasonably impractical. Nevertheless, it is important to see some consequences of (2.8):

COROLLARY 2.9 [30]. *Given an integer b, there are polynomial-time algorithms which, when given $G \leq S_n$ such that all noncyclic composition factors of G have order $\leq b$, finds*

(i) *the setwise stabilizer $G_Y = \{g \in G \mid Y^g = Y\}$ of any given $Y \subseteq X$, and*

(ii) *the centralizer $C_G(t)$ of any given $t \in S_n$.*

PROOF. Use (2.8) with $H = (S_n)_Y$ in (i) and $H = C_{S_n}(t)$ in (ii). □

Finally, note that it is not difficult to find the center $Z(G)$ in polynomial time. More generally, if $A \trianglelefteq G$ is given then $C_G(A)$ can be found in polynomial time [32]; for example, $C_G(A) = G \cap C_{S_n}(A)$ can be found using (2.6).

For an intricate description of many of the above results, see [14].

3. Graph Isomorphism

There are probably severe restrictions on what can be accomplished in polynomial time. Namely, consider the following four problems (where $G \leq S_n$ as usual):

(1) Given $H \leq S_n$, find $G \cap H$.
(2) Given a p-subgroup P of G (for a prime p), find its normalizer $N_G(P)$.
(3) Given an involution $t \in G$, find its centralizer $C_G(t)$.
(4) Given $Y \subset X$, find its setwise stabilizer G_Y.

Theorem 3.1. *If any of the problems* (1)–(4) *can be solved in polynomial time, then so can the* GRAPH ISOMORPHISM *problem.*

Here, GRAPH ISOMORPHISM is the following: Given two n–vertex graphs, decide whether or not they are isomorphic. The above somewhat surprising-looking result is due to Luks [30, 32]. Parts of the theorem and other similar results of Luks are described in [14].

Proof. Clearly, (3) is a special case of (2). Also, (4) is a special case of (1) since $G_Y = G \cap (S_n)_Y$ and $(S_n)_Y$ is easy to calculate.

It is not difficult to provide a polynomial-time reduction of (4) to (3). Namely, let $X' = \{1', \ldots, n'\}$ be a copy of X disjoint from X, and let G act on X' in the natural manner. Let E be the group $\langle (1,1') \rangle \times \cdots \times \langle (n,n') \rangle \cong \mathbb{Z}_2^n$. Then G normalizes E, and EG is a group. Let t be the involution in E interchanging y and y' for $y \in Y$ and fixing all other points in $X \cup X'$. Then $EG_Y = C_{EG}(t)$ and G_Y is the group induced on X by $C_{EG}(t)$, so that a polynomial-time algorithm for (3) produces one for (1).

It is also not hard to provide a polynomial-time reduction of GRAPH ISOMORPHISM to (1). This involves two stages: (i) a reduction from isomorphisms to automorphisms, and then (ii) a description of an automorphism group as an intersection.

(i) It may be assumed that each of the given graphs (V_i, E_i), with vertex-set V_i and edge-set E_i $(i = 1, 2)$, is connected (otherwise recursion could be used); it may also be assumed that $V_1 \cap V_2 = \emptyset$. Construct a $2n$–vertex graph (V, E) that is the union of the two graphs, so that (V, E) has exactly two connected components. If $\mathrm{Aut}(V, E)$ has an element interchanging these components then the two original graphs are isomorphic, while if each member of a generating set for $\mathrm{Aut}(V, E)$ sends each component to itself then the original graphs are not isomorphic.

(ii) Now consider an n–vertex graph (V, E). Let $V^{(2)}$ denote the set of all pairs of vertices. Since $S_n = \mathrm{Sym}(V)$ permutes the subsets of V it has a natural embedding as a subgroup G of $\mathrm{Sym}(V^{(2)})$. Let H be the subgroup of $\mathrm{Sym}(V^{(2)})$ consisting of those permutations sending E to itself, so that $H \leq \mathrm{Sym}(E) \times \mathrm{Sym}(V^{(2)} - E)$. Then $\mathrm{Aut}(V, E) = G \cap H$. Recall that each symmetric group S_k

has two easily found generators (a k–cycle and a transposition). Thus, $\mathrm{Aut}(V, E)$ is the intersection of groups having 2 and 4 generators, respectively. □

Note that an algorithm for intersections (1) was needed only in the very special case of two groups, one isomorphic to a symmetric group and the other to the direct product of two symmetric groups. Thus, the structure of G and H do not help in general (but compare (2.8)).

It is generally believed that there is no polynomial-time algorithm for GRAPH ISOMORPHISM. If that turns out to be the case then none of (1)–(4) can be accomplished in polynomial time. In any event, it should be evident that (1)–(4) must be avoided in the context of the present subject — except, of course, for the unlikely possibility that the study of polynomial-time group-theoretic algorithms might produce a solution to the GRAPH ISOMORPHISM problem.

Finally, note that (2.8) provides polynomial-time algorithms for (1), (3) and (4) when G is bounded as in (2.8) (e.g., if G is solvable). Until very recently no polynomial-time algorithm was known for (2) when G is solvable; one has just been found by Luks.

4. Further algorithms

There are a number of other elementary consequences of the results in §2 (see [23] for a summary of many of them). However, I will now move to more recent and more complicated results [31, 5, 35, 36, 19, 20, 23].

Let $G = \langle \Gamma \rangle$ be as usual. Recall that a sequence of subgroups $G = G_0 > \cdots > G_k = 1$ of subgroups of G is called a *composition series* if each term is normal in the preceding one and each quotient group G_i/G_{i+1} is simple; while $G = G_0 > \cdots > G_k = 1$ is called a *chief series* if each term is normal in G and there is no normal subgroup of G lying properly between two successive terms.

Theorem 4.1 (Luks [31]). *A composition series of G can be found in polynomial time.*

Note that it is not possible to find all composition series, as there may be too many of them (once again, consider the regular representation of an elementary abelian 2–group). Luks' algorithm is known to run in time $O(n^8)$, with the exponent due, to a large extent, to the $O(n^5)$ for (2.4) — and hence it can be made faster by using [6].

Corollary 4.2 [31].

(i) *Simplicity of G can be tested in polynomial time.*
(ii) *For each successive pair $A \trianglelefteq B$ in a given composition series, in polynomial time a set of size $\leq n$ can be found on which B/A acts faithfully.*

Proof. Since (i) is obviously taken care of by (4.1), consider (ii). WLOG $G = B$. Let Y be the set of orbits of A on X. Then G induces a group G^Y of permutations of Y. If $G^Y \neq 1$ output Y. WLOG $G^Y = 1$. Since G acts nontrivially on some member of Y, WLOG A is transitive on X. Now $G = AG_x$

for $x \in X$, so that $G/A \cong G_x/A_x$ and recursion can be applied to the pair G_x, $X - \{x\}$. (For a somewhat different argument, see [30], (3.2)].) □

THEOREM 4.3 (Rónyai [35, 36]). *A chief series of G can be found in polynomial time.*

OUTLINE. Using (4.1) it is easy to find a normal series for G each of whose factors is either (1) elementary abelian, or (2) the direct product of nonabelian simple groups permuted transitively by G. (Namely, consider the normal closures (2.5) of all the terms in (4.1).) Therefore, (4.3) can be viewed as a special case of the following situation. Given a set Δ of linear transformations of a finite vector space, find a Δ–irreducible subspace. Rónyai considers this latter problem in terms of the algebra of linear transformations generated by Δ. This is dealt with by an ingenious use of classical ideas concerning finite-dimensional algebras. □

Simplicity is one of the standard and most basic questions concerning a finite group. Almost as basic are Cauchy's and Sylow's Theorems. All of the standard proofs for the latter theorems either clearly do not produce polynomial-time algorithms or very likely do not. For example, the proof of Cauchy's Theorem via the 'class equation' is purely existential. Similarly, the most standard proofs of the existence of Sylow subgroups involve — in addition to Cauchy's Theorem — the use of normalizers or centralizers of p–subgroups of G, and these 'must' be avoided by §3. (On the other hand, the algorithm in CAYLEY builds up a Sylow subgroup by using centralizers.) Other proofs of Cauchy's Theorem or Sylow's Theorem involve the examination of potentially exponential-size subsets of G or an even larger set. Finally, the conjugacy part of Sylow's Theorem is standardly proved by a purely existential argument (a counting argument). Consequently, new techniques were required in order to obtain polynomial-time algorithms. Cauchy's Theorem was dealt with in [18]. Once again, the classification of finite simple groups was involved! However, unlike the situation with (4.1), detailed information was needed concerning such groups (cf. §6). In [25], polynomial-time algorithms were obtained for special cases of Sylow's Theorem, such as for solvable groups — and in the solvable case Hall's Theorem was also dealt with. These solvable group algorithms were later greatly expanded in [19] and incorporated into methods that led to the general case:

THEOREM 4.4 [19, 20]. *If p is a prime then the following can be found in polynomial time*:

(i) *Given a p–subgroup of G (possibly of order 1), a Sylow p-subgroup P of G containing it*;
(ii) *Given two Sylow p–subgroups of G, an element $g \in G$ conjugating the first one to the second*; *and*
(iii) *The normalizer $N_G(P)$ of a given Sylow p–subgroup P of G (in particular, all g in* (ii) *can be 'found': the coset $N_G(P)g$ is the set of all such conjugating elements).*

At present there is no Sylow algorithm that is both polynomial-time and practical. However, when the algorithm in [19] is specialized to solvable groups it becomes practical — and elementary. A version of the solvable group algorithm was studied in [12], and is now incorporated into CAYLEY. This may have been the first published instance where algorithms developed for polynomial-time group theory were faster than ones developed for purely practical purposes. The polynomial-time methodology must be different from standard approaches due to the fact that centralizers cannot in general be used in the polynomial-time context; and this in turn leads to new ideas some of which have practical applications. It was mentioned earlier that there are faster versions of (2.4) obtained in [6] using entirely different methods than approaches based on Sims' ideas, and that these have turned out to have significant practical value. There is thus evidence that a number of the polynomial-time algorithms mentioned in this paper are capable of leading to practical algorithms.

The methods used in the proof of (4.4) eventually depend on detailed information concerning all finite simple groups — not, for example, just on the finiteness of the number of sporadic simple groups. Some of this is discussed in §5. For now, I will give a hint of one aspect of the proof, involving an algorithmic version of the *Frattini argument* [13, p. 12]. Use (4.1) to find $M \triangleleft G$ with G/M simple. Use knowledge of simple groups in order to find a Sylow p–subgroup H/M of G/M, and replace G by H. This reduces (4.4i) to the case $|G/M| = p$. Recursively find a Sylow p–subgroup P of M. Let $g \in \Gamma$, $g \notin M$. Then, *assuming* that a polynomial-time algorithm for the conjugacy part (ii) of (4.4) is available (for M in place of G), find $m \in M$ such that $P^{gm} = P$. If $\langle g' \rangle$ is a Sylow p–subgroup of the cyclic group $\langle gm \rangle$, then $P\langle g' \rangle$ is a Sylow p–subgroup of G. This reduces the proof of (4.4i) to the proof of (4.4ii) together with a proof of the simple group case of (4.4i). Similar arguments reduce all of parts of (4.4) to considerations of simple groups. The Frattini argument is very powerful in group theory; it is also powerful in algorithmic contexts (see (6.1) for another example of this).

COROLLARY 4.5. *Given a subgroup G of S_n and a prime p, the largest normal p–subgroup $O_p(G)$ of G can be found in polynomial time.*

PROOF. If P is an intersection of some Sylow p–subgroups of G then successively test the elements $g \in \Gamma$ to see whether $P^g = P$ (using (2.4i)). If this fails for some g then find $P \cap P^g$ using (2.8), and replace P by $P \cap P^g$. Otherwise output P. □

Luks has devised a polynomial-time algorithm for finding $O_p(G)$ that does not use any information about simple groups [32]. This is an important type of development. Once theorems are known to be true by using the classification, it is very desirable to see if alternative proofs might be discovered that do not use the classification. In the context of algorithmic group theory, this is especially important, since algorithms obtained without the classification might be more practical than ones obtained using it. This is the case, for example, with (4.5).

5. Permutation representations of simple groups

Suppose that the group $G \leq \mathrm{Sym}(X)$ is known to be simple (cf. (4.1)). By the classification of finite simple groups, G is known. Sporadic groups have order $O(1)$, so that all questions concerning them are 'easy' in the polynomial-time context. Similarly, if $|G| < n^8$ then most questions can be answered by brute force. For example, Sylow subgroups can be built up recursively by the method frequently taught in basic undergraduate courses.

There is no problem reducing to the case in which G is primitive. Assume as well that $|G| \geq n^8$. Let $x \in X$. It is not hard to show that this implies that either $G \cong A_r$ and G_x is the stabilizer of a subset or a partition of the r–set into subsets of equal size, or G is classical and G_x is the stabilizer of a subspace (cf. [17, 18]). If $G \cong A_r$ call the underlying r–set the natural permutation representation of G; while if G is classical call the set of points of the underlying projective space the natural permutation representation of G. (Note that the assumption $|G| \geq n^8$ has eliminated all need to consider the exceptional simple groups of Lie type!)

Theorem 5.1 (Replacement Theorem [18]). *Given a simple subgroup G of S_n of order $\geq n^8$, there is a polynomial-time algorithm that finds the natural permutation representation of G (and that permutation representation has degree $< 2n$).*

This is proved using the geometry of the classical groups. The bound can be improved, with n^5 in place of n^8, using [29]. A potentially more practical polynomial-time algorithm for (5.1) is being investigated [24]. One goal of this is a more practical Sylow algorithm for an arbitrary G and 'large' n than is presently in the literature.

Much more can be obtained concerning the natural action when G is a classical group. The following appear in [18, 19].

Vector space: In polynomial time the underlying vector space V can be found. This involves a very classical method for coordinatizing projective geometry [38].

Matrices: A group of matrices inducing G can be found. Note that G itself may not act on the vector space: G only acts projectively. But of course, G is usually 'viewed' as a group of matrices or linear transformations, and this standard point of view is readily achieved using (5.1).

Forms: In the case of a symplectic, orthogonal or unitary group, a suitable form can be constructed on V.

Linear algebra: The preceding constructions have the effect of replacing permutation group considerations by 'ordinary' types of linear algebra. In view of this, it should come as no surprise that the simple group case of (4.4) can be deduced from the Replacement Theorem (by means of some rather tedious work — mostly linear algebra awkwardly phrased in terms of permutation groups). The much more interesting part of the proof of (4.4) is the reduction to the case of simple groups.

As noted above, I have avoided dealing with exceptional groups of Lie type by starting with a suitable inequality $|G| \geq n^8$. However, it would be interesting to have analogous algorithmic results for those groups as well, under the assumption that the given permutation representation is sufficiently natural (e.g., on a class of maximal parabolic subgroups).

6. Quotient groups

There are polynomial-time algorithms for finding many other properties of $G = \langle \Gamma \rangle \leq S_n$. Lists of these can be found in [19, 23], and will not be included here. Instead, I will briefly discuss the question of computations in quotient groups.

Let $G = \langle \Gamma \rangle \leq S_n$ be as usual, and let $K \trianglelefteq G$ be given. It is natural to ask for properties of the quotient group G/K. Unfortunately, G/K need not lie in S_r for any r of size polynomial in n, as the following example shows [33]. Let $n = 4m$, partition X into m sets of size 4, and let G be the direct product of m dihedral groups of order 8, each of which acts faithfully on one of the 4–sets and is the identity on the remainder of X. Let G/K be the quotient group obtained by identifying the centers of the m dihedral groups. Then G/K is an extraspecial group, and it is easy to use the complex representation theory or subgroup structure of this group in order to show that it does not have a faithful permutation representation on a set of polynomial size.

Thus, studying G/K is harder than studying G. Entirely new algorithms are needed. For example, while it is easy to find the center $Z(G)$, it seems to be very difficult to find $Z(G/K)$. In fact, even for p–groups it is by no means obvious how to go about finding the ascending central series (the descending central series is, however, easy; cf. (2.7)). In [23] the classification of finite simple groups was used to find $Z(G/K)$ for general G. More generally, in [23] analogues were found for G/K of every known polynomial-time algorithm for computations in G. A fairly typical though simple example is as follows.

Proposition 6.1. *Given solvable subgroups A/K and B/K of G/K, their intersection $A/K \cap B/K = (A \cap B)/K$ can be found in polynomial time.*

Proof. If K is solvable, use (2.8). So WLOG K is not solvable, and hence in particular K is not nilpotent. Use (4.4) to find a Sylow subgroup Q of K that is not normal in K, and also to find $N_K(Q)$, $N_G(Q)$, $N_A(Q)$ and $N_B(Q)$. Recursively find $[N_A(Q)/N_K(Q)] \cap [N_B(Q)/N_K(Q)] = N_{A\cap B}(Q)/N_K(Q)$. By the Frattini argument, $A \cap B = K[N_{A\cap B}(Q)] = K[N_A(Q) \cap N_B(Q)]$. □

One interesting consequence of the consideration of quotient groups is the following result, which does not appear to have anything at all to do with quotient groups:

Theorem 6.2 [23]. *Given $H \leq G$, the largest normal subgroup $\cap\{H^g \mid g \in G\}$ of G contained in H can be found in polynomial time.*

Note that the intersection in (6.2) may involve exponentially many subgroups of G. A special case of this situation is finding $O_p(G)$, but the algorithm indicated in (4.5) is simpler — and, as already mentioned, this case of (6.2) has a polynomial-time solution not requiring the classification of finite simple groups [32].

7. Galois groups

Some of the methods described here have limited applicability to a superficially similar but much harder subject: polynomial-time Galois theory. There, one is given a finite extension K of $\mathbb{Q}$ and a polynomial $f \in K[x]$, and the basic problem is to find properties of the Galois group of f. For simplicity, assume that $f \in \mathbb{Z}[x]$ (although extensions of $\mathbb{Q}$ must eventually be considered as well). Then the problem is to determine $\mathrm{Gal}(f)$ in time that is polynomial in the number of (binary or decimal) digits required to write f. Analogues of (2.2) and (2.3) exist: f can be factored into irreducibles [28]; an extension $L = \mathbb{Q}(\alpha)$ of $\mathbb{Q}$ can be obtained with $f(\alpha) = 0$ (described as a vector space over $\mathbb{Q}$ with a distinguished basis and a multiplication rule for that basis); and, when f is irreducible, subfields of L can be specified in polynomial time that correspond to the blocks involved in (2.3) (specified as the sets of roots of explicitly constructed polynomials) [27]. Of course, it is then easy to test for 2–transitivity, or 5–transitivity, using factorizations (over number fields).

This situation is harder than the one in this paper because (i) Galois groups are determined only up to conjugacy in symmetric groups; (ii) hence, describing a nontrivial element of $G = \mathrm{Gal}(f)$ is difficult (except possibly for complex conjugation); and (iii) a splitting field of f generally has non-polynomial degree over $\mathbb{Q}$, and hence cannot be written (as a vector space over $\mathbb{Q}$) in polynomial time. In view of these difficulties, it is not surprising that no polynomial-time algorithm is known for determining $|G|$.

Only the following have been proved: In polynomial time it can be decided whether or not G is solvable [27], in which case all prime factors of $|G|$ can be found (but not the multiplicities to which they occur: it is an open problem to determine $|G|$ in polynomial time, even when it is known that G is a 2–group). Very weak analogues of (4.1) and (4.2) have been obtained [22, 21] when it is assumed that G acts primitively on the set of roots of f (based on analogues of (5.1)).

References

1. M.D. Atkinson (1975). An algorithm for finding the blocks of a permutation group. *Math. Comp.* 29, 911-913.
2. L. Babai (1980). On the length of subgroup chains in the symmetric group. *Comm. in Alg.* 14, 1729-1736.
3. L. Babai, P.J. Cameron and P. Pálfy (1982). On the order of primitive groups with restricted nonabelian composition factors. *J. Algebra* 79, 161-168.

4. L. Babai, G. Cooperman, L. Finkelstein and A. Seress (1991). Nearly linear time algorithms for permutation groups with a small base. *Proc. Int. Symp. Symbolic and Algebraic Computation*, 200-209.
5. L. Babai, W. M. Kantor and E. M. Luks (1983). Computational complexity and the classification of finite simple groups, *Proc. IEEE Symposium on Foundations of Computer Science*, 162-171.
6. L. Babai, E.M. Luks and A. Seress. *Managing permutation groups in $O(n^4 \log^c n)$ time* (in preparation).
7. J. J. Cannon (1984). An introduction to the group theory language Cayley. *Computational Group Theory* (ed. M. D. Atkinson), Academic Press 1984, 145-183.
8. P.J. Cameron, R. Solomon and A. Turull (1989). Chains of subgroups in symmetric groups. *J. Algebra* 127, 340-352.
9. G. Cooperman, L. Finkelstein and E. Luks (1989). Reduction of group constructions to point stabilizers. *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ACM Press, 351-356.
10. P. Diaconis, R. L. Graham and W.M. Kantor (1983). The mathematics of perfect shuffles. *Advances in Applied Mathematics* 4, 175-196.
11. M. Furst, J. Hopcroft and E. Luks (1980). Polynomial-time algorithms for permutation groups. *Proc. 21st I.E.E.E. Symp. Found. Comp. Sci.*, 36-41.
12. S.D. Glasby (1988). Constructing normalisers in finite soluble groups. *J. Symbolic Computation* 5, 285-294.
13. D. Gorenstein (1968). *Finite Groups.* Harper and Row, New York.
14. C.M. Hoffmann (1982). Group-Theoretic Algorithms and Graph Isomorphism. *Springer Lect. Notes in Comp. Sci.* 136.
15. I.M. Isaacs, W.M. Kantor and N. Spaltenstein *On the probability that a group element is p-singular* (in preparation).
16. M.R. Jerrum (1982). A compact representation for permutation groups. *Proc. 23rd IEEE Symp. Found. Comp. Sci.*, 126-133.
17. W.M. Kantor (1979). Permutation representations of the finite classical groups of small degree or rank. *J. Algebra* 60, 158-168.
18. W.M. Kantor (1985). Polynomial-time algorithms for finding elements of prime order and Sylow subgroups. *J. Algorithms* 6, 478-514.
19. W.M. Kantor (1985). Sylow's theorem in polynomial time. *J. Comp. Syst. Sci.* 30, 359-394.
20. W.M. Kantor (1990). Finding Sylow normalizers in polynomial time. *J. Algorithms* 11, 523-563.
21. W.M. Kantor (unpublished).
22. W.M. Kantor and E. Lander *Recognizing exponentially large Galois groups* (unpublished manuscript).
23. W.M. Kantor and E.M. Luks (1990). Computing in quotient groups. *Proc. 22nd ACM Symposium on Theory of Computing*, 524-534.
24. W.M. Kantor and T. Penttila (in preparation).
25. W.M. Kantor and D.E. Taylor (1988). Polynomial-time versions of Sylow's theorem. *J. Algorithms* 9, 1-17.

26. D.E. KNUTH (1991). Efficient representation of perm groups. *Combinatorica* 11, 33-43.
27. S. LANDAU and G.L. MILLER (1985). Solvability by radicals is in polynomial time. *J. Comp. Syst. Sci.* 30, 179-208.
28. A.K. LENSTRA, H.W. LENSTRA and L. LOVÁSZ (1982). Factoring polynomials with rational coefficients. *Math. Ann.* 261, 513-534.
29. M.W. LIEBECK (1985). On the orders of maximal subgroups of the finite classical groups. *Proc. LMS* 50, 426-446.
30. E.M. LUKS (1982). Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comp. Syst. Sci.* 25, 42-65.
31. E.M. LUKS (1987). Computing the composition factors of a permutation group in polynomial time. *Combinatorica* 7, 87-99.
32. E.M. LUKS (unpublished).
33. P.M. NEUMANN (1987). Some algorithms for computing with finite permutation groups. *Proceedings of Groups-St. Andrews* 1985 (Eds. E. F. ROBERTSON and C. M. CAMPBELL). London Math. Soc. Lect. Note 121, Cambridge U. Press, 59-92.
34. A. NIEMEYER, W. NICKEL and M. SCHÆNERT (1988). *GAP, Getting started and reference manual.* Aachen.
35. L. RÓNYAI (1985). Zero divisors and invariant subspaces. *Technical Report CIS-TR 85-12*, Department of Computer and Information Science, University of Oregon.
36. L. RÓNYAI (1987). Simple algebras are difficult. Proc. *ACM Symposium on Theory of Computing*, 398-408.
37. C.C. SIMS (1970). Computational methods in the study of permutation groups. *Computational Problems in Abstract Algebra* (ed. J. Leech), Pergamon Press, NY, 169-183.
38. O. VEBLEN and J.W. YOUNG (1916). *Projective geometry.* Ginn, Boston.