DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume 28, 1997

Computing the Fitting Subgroup and Solvable Radical of Small-Base Permutation Groups in Nearly Linear Time

EUGENE M. LUKS AND ÁKOS SERESS

ABSTRACT. We present Monte Carlo algorithms for computing the centralizer of a normal subgroup, the maximal normal p-group for primes p, the Fitting subgroup, and the maximal solvable normal subgroup in a permutation group $G = \langle S \rangle$ acting on a set Ω. The running time of the algorithms can be expressed in the form $O(n|S|\log^c|G|)$, where n is the size of Ω. In particular, in the case of *small-base groups*, when $\log|G|$ is polylogarithmic as a function of n, the algorithms run in *nearly linear*, $O(n\log^{c'} n)$ time. If the order of G is known in advance, the algorithms can be upgraded to Las Vegas. The algorithms have been implemented in *GAP*.

1. Introduction

A remarkable recent development in the field of permutation-group computation has been the construction of a library of algorithms that run in time $O(n|S|\log^c|G|)$, given $G = \langle S \rangle \leq S_n$ [1], [2], [4], [5], [13], [15], [17], [18], [19]. We consider such algorithms to be *nearly linear*, the terminology justified by the following observation: if one is dealing with a family of groups G for which $\log|G| = O(\log^{c'} n)$, the running time is $O(n|S|\log^c n)$, which, up to only logarithmic factors, is a linear function of the input length $n|S|$. Indeed, such families of groups are commonly encountered and are referred to as *small-base groups.* Important classes of groups in the small-base category include all permutation representations of finite simple groups except the alternating groups.

In addition to their guarantee of a fast asymptotic running time, nearly linear-time algorithms appear to be well suited for implementation; in fact, many of them are already implemented in the *GAP* system [16]. Hence, they achieve the long sought goal of combining the theoretical and practical sides of permutation group manipulation.

The present paper describes some additions to the nearly linear-time library. For a group G and a prime number p, we denote by $O_p(G)$ the largest normal p-subgroup of G, by $O_\infty(G)$ the largest solvable normal subgroup (solvable radical)

1991 *Mathematics Subject Classification.* 20B40, 68Q40.

The first author acknowledges support by NSF Grant CCR-9013410. The second author was supported in part by NSF Grants CCR-9201303 and CCR-9503430.

This is the final version of this paper.

of G, and by $F(G)$ the largest nilpotent normal subgroup (Fitting subgroup) of G. Our main result is

THEOREM 1.1. *Given generators S for a permutation group $G \leq S_n$ and a prime p, the subgroups $O_p(G)$, $O_\infty(G)$, $F(G)$ can be computed by a Monte Carlo algorithm in time $O(n|S|\log^c|G|)$.*

A randomized algorithm is called *Monte Carlo* if there is only a small probability (prescribed by the user) that it returns a wrong answer. Doubling the running time squares the error probability. In our case, an error can occur only at the initial strong-generating-set computation (cf. Subsection 2.2). In particular, if $|G|$ is known in advance, in which case the correctness of the strong-generating-set construction can be verified, then the algorithm indicated in Theorem 1.1 never returns an incorrect answer. Hence, the algorithm is upgraded to *Las Vegas*: its output is always correct and it runs in nearly linear expected running time.

Polynomial-time algorithms for constructing $O_p(G)$ were first given by Kantor [9] and Neumann [14]. An alternative approach, and a construction of $O_\infty(G)$, was described by Luks in [11] (see also [12]). A critical point in the algorithm presented here is based on the method of [11]. In particular, we employ a recursive approach having constructed an action $\varphi : G \to S_n$, with $\ker(\varphi)$ an abelian p-group, the nature of the kernel guaranteeing that $\varphi(O_p(G)) = O_p(\varphi(G))$. The construction of φ can be carried out once we have any normal p-subgroup. (A similar procedure for $O_\infty(G)$ makes use of any normal solvable subgroup.) This useful normal subgroup is found as the normal closure of $C_N(M)$ (centralizer of M in N) for some successive terms $M \lhd N$ in a composition series of G and so we make use of the nearly linear time construction of composition series by Beals and Seress [4].

Observe that this approach also requires the efficient construction of centralizers of normal subgroups, which should be of independent interest.

THEOREM 1.2. *Given generators S for the permutation groups N, G with $N \lhd G$, the centralizer $C_G(N)$ can be computed by a Monte Carlo algorithm in time $O(n|S|\log^c|G|)$.*

The nearly linear constraint seems to prohibit standard methods for $C_G(N)$, e.g., in [10] this subgroup is realized as the kernel of an induced action on a set of size $O(n^2)$. Instead, we compute the kernel $\tilde{G}$ of an action of G on a set of size $\leq n$ and then the kernel of an action of $\tilde{G}$ on a set of size $\leq n$. The second kernel is $C_G(N)$.

Again, if $|G|$ is known in advance, then this algorithm can be upgraded to Las Vegas.

Implementations of the algorithms underlying Theorems 1.1 and 1.2 are available in the *GAP* [16] library.

2. Preliminaries

2.1. Group theory. For basic definitions and facts about groups and permutation groups, we refer the reader to [8] and [22]. We summarize the notation and terminology that we use.

For $H, G \leq \mathrm{Sym}(\Omega)$, $\langle H^G \rangle$ denotes the *G-closure* of H and $C_G(H)$ is the *centralizer* of H in G. For $H \leq G$, G/H denotes the set of right cosets of H in G; a *transversal* for H in G is a complete set of right coset representatives.

The full symmetric group acting on the domain Ω is denoted by $\mathrm{Sym}(\Omega)$, while S_n denotes the symmetric group of degree n.. We say that G *acts* on a set Ω if a homomorphism $\varphi : G \to \mathrm{Sym}(\Omega)$ is given, usually φ is clear in context; the action is *faithful* if $\ker(\varphi) = 1$. If G acts on Ω and $\omega \in \Omega$, we denote by ω^G the *orbit* of ω under G, namely $\{\omega^g \mid g \in G\}$; we say G acts *transitively* on Ω if Ω consists of one orbit. If G acts transitively on Ω, a nonempty subset $\Delta \subseteq \Omega$ is called a *block of imprimitivity* for G if, for all $g \in G$, $\Delta^g = \Delta$ or $\Delta^g \cap \Delta = \emptyset$. If B is a sequence of points from Ω, G_B denotes the *pointwise stabilizer* of B in G. A group $G \leq \mathrm{Sym}(\Omega)$ is called *semiregular* if $G_\omega = 1$ for all $\omega \in \Omega$; if G is semiregular and transitive, it is called *regular.* For $G \leq \mathrm{Sym}(\Omega)$, we denote by fix(G) the set of fixed points of G, namely, $\{\omega \in \Omega \mid \forall g \in G : \omega^g = \omega\}$. If $\Delta \subseteq \Omega$ is the union of orbits of some $G \leq \mathrm{Sym}(\Omega)$, then we denote by $G|_\Delta \leq \mathrm{Sym}(\Delta)$ the *restriction* of G to Δ. The *wreath product* of $G \leq \mathrm{Sym}(\Omega)$ and $H \leq \mathrm{Sym}(\Delta)$ is denoted by $G \wr H$; we consider its action on the disjoint union of $|\Delta|$ copies of Ω (see, e.g., [8, p. 81] for details).

2.2. Algorithmic concepts. It is assumed that permutation groups are input and output via a list of generators.

In his pioneering work [20, 21], Sims introduced the notions of a base and strong generating set as the fundamental data structures for computing with permutation groups. A *base* for a permutation group $G \leq \mathrm{Sym}(\Omega)$ of degree n is a sequence $B = (\beta_1, \beta_2, .., \beta_M)$ of points from Ω such that $G_B = 1$. The *point stabilizer chain* of G relative to B is the chain of subgroups

$$G = G^{(1)} \geq G^{(2)} \geq \cdots \geq G^{(M+1)} = 1,$$

where $G^{(i)} = G_{(\beta_1,..,\beta_{i-1})}$. The base B is called *nonredundant* if there is strict inclusion $G^{(i)} > G^{(i+1)}$ for all $1 \leq i \leq M$. A *strong generating set* (SGS) for G relative to B is a set S of generators of G with the property that

$$\langle S \cap G^{(i)} \rangle = G^{(i)}, \text{ for } 1 \leq i \leq M+1.$$

With reference to some constant c, an (infinite) family $\mathcal{G}$ of permutation groups is called a family of *small-base groups* if all $G \in \mathcal{G}$ of degree n admit bases of size $O(\log^c n)$. Equivalently, $\log |G| = O(\log^{c'} n)$ for a constant c' and each $G \in \mathcal{G}$ of degree n.

Let $B = (\beta_1, ..., \beta_M)$ be a base of the group G and let $G = G^{(1)} \geq G^{(2)} \geq \cdots \geq G^{(M+1)} = 1$ be the corresponding point stabilizer chain. Moreover, let R_i denote a transversal for $G^{(i+1)}$ in $G^{(i)}$, $1 \leq i \leq M$. Each $g \in G$ can be written uniquely in the form $g = r_M r_{M-1} \cdots r_2 r_1$, $r_i \in R_i$. The process of factoring g in this form is called *sifting* or *stripping.*

In practical computation, the transversals R_i are not usually computed and stored explicitly; rather, they are encoded in a Schreier-tree data structure. Suppose that a base B and an SGS S for G relative to B are given. A *Schreier-tree data structure* for G is a sequence of pairs (S_i, T_i) called *Schreier trees*, one for each base point β_i, $1 \leq i \leq M$, where T_i is a directed labelled tree with all edges directed toward the root β_i, and with edge labels selected from the set $S_i := S \cap G^{(i)} \subseteq G^{(i)}$. The nodes of T_i are the points of the orbit $\beta_i^{G^{(i)}}$. The labels satisfy the condition that, for each directed edge from γ to δ with label h, $\gamma^h = \delta$. If γ is a node of T_i, then the sequence of the edge labels along the path from γ to β_i in T_i is a word in the elements of S_i such that the product of these permutations moves γ to β_i.

Thus each Schreier tree (S_i, T_i) defines *inverses* of the elements of a transversal for $G^{(i+1)}$ in $G^{(i)}$.

The reason for storing the inverses of transversal elements is that the sifting procedure uses these inverses. In some applications in the present paper, we need to recover the transversal elements themselves. This can be done, for example, by storing the inverses of the strong generators. Following the path from a node γ to the root β_i in T_i, and writing the inverses of the labels we encountered in the opposite order, we get a word representing a permutation which moves β_i to γ.

2.3. Some nearly linear-time algorithms. In this subsection, we describe some nearly linear-time algorithms used in our procedures. The fundamental result is that bases and strong generating sets can be computed.

THEOREM 2.1. (Babai, Cooperman, Finkelstein, Seress [1]) *Given a permutation group $G \leq S_n$, a nonredundant base B and an SGS for G relative to B can be computed by a Monte Carlo algorithm in nearly linear time. The Schreier-tree data structure computed by the algorithm supports membership testing in G in $O(n \log |G|)$ time.*

For fast sifting, it is crucial that the depths of trees in the Schreier-tree data structure are small. In fact, the algorithm in [1] constructs Schreier trees such that the sum of depths is $O(\log |G|)$. We call a Schreier-tree data structure *shallow* if the depth of each tree is at most $2 \log |G|$. A shallow Schreier-tree data structure supports membership testing in nearly linear time.

Lemma 2.2 in [1] states that, given $G = \langle S \rangle \leq \mathrm{Sym}(\Omega)$ and $\alpha \in \Omega$, group elements $g_1, ..., g_k$ can be computed in deterministic time $O(n|S| \log^2 |G|)$ for some $k \leq \log |G|$ such that the orbits α^G and $\alpha^{\langle g_1, ..., g_k \rangle}$ are equal, and the breadth-first-search tree for α^G, using the group elements $g_1, ..., g_k, g_1^{-1}, ..., g_k^{-1}$, has depth at most $2k \leq 2 \log |G|$. This implies that given a nonredundant base $B = (\beta_1, ..., \beta_M)$ and an SGS S relative to B for a group G, we can replace S in nearly linear deterministic time by another SGS T relative to B with $|T| \leq 2M \log |G|$ such that the Schreier-tree data structure obtained from T is shallow. Therefore, we can assume that all Schreier-tree data structures computed in our algorithms are shallow.

The method for constructing an SGS can be applied to a wide variety of tasks. In particular the following are straightforward consequences of Theorem 2.1.

LEMMA 2.2. *The following problems can be solved in nearly linear Monte Carlo time.*

(i) *Given $G \leq \mathrm{Sym}(\Omega)$ and an action $\varphi : G \to \mathrm{Sym}(\Delta)$, find the kernel of φ. In particular, the pointwise stabilizer of a block system on Ω can be computed.*
(ii) *Given $G \leq \mathrm{Sym}(\Omega)$ and $A \subset \Omega$, find the pointwise stabilizer G_A.*
(iii) *Given $H \leq G \leq \mathrm{Sym}(\Omega)$, find the normal closure $\langle H^G \rangle$.*
(iv) *Given $G \leq \mathrm{Sym}(\Omega)$, find the derived series of G.*

For $g \in G$, the g-image of a base of G uniquely determines g. It is also possible to reconstruct g from the image of the base.

LEMMA 2.3. *Let S be an SGS for $G \leq \mathrm{Sym}(\Omega)$ relative to B and let t denote the sum of depths of Schreier trees coding the coset representative sets along the point stabilizer chain of G. Then, given an injection $f : B \to \Omega$, it is possible to* *find a permutation $g \in G$ with $B^g = f$, or determine that no such element of G exists, in $O(t|\Omega|)$ deterministic time.*

There is no polynomial time algorithm known for computing the intersection of two permutation groups (see, e.g., [12]). However, generalizing a method from linear algebra for computing the intersection of subspaces, Cooperman, Finkelstein, and Luks [6] describe an algorithm which reduces the computation of $\langle H^G \rangle \cap G$ for $H, G \leq \mathrm{Sym}(\Omega)$ to the computation of a pointwise stabilizer in a permutation action of an appropriate group containing H and G. Combining with Theorem 2.1 and Lemma 2.2(ii), we obtain the following.

THEOREM 2.4. *Given $G, H \leq \mathrm{Sym}(\Omega)$, $|\Omega| = n$, the group $\langle H^G \rangle \cap G$ can be computed in $O(n \log^c(|\langle H^G \rangle||G|))$ Monte Carlo time. In particular, if H is normalized by G, then $H \cap G$ can be computed.*

In [10], Luks described a polynomial-time method for computing composition series in permutation groups. There is a nearly linear version of this result.

THEOREM 2.5. (Beals, Seress [4]) *Given $G = \langle S \rangle \leq S_n$, a composition series of G can be computed in Monte Carlo time $O(n|S| \log^c |G|)$. As in [10], the procedure also yields permutation representations of degree at most n for the composition factors of G.*

REMARK 2.6. The composition series constructions of [10] and [4] are dependent upon the classification of finite simple groups (specifically, they cite "Schreier's Conjecture"). However, while the use of a composition series simplifies both our exposition and the present implementation in *GAP*, the main results of this paper can be achieved without this device (see Remark 5.4). Alternately, Beals [3] has shown that one can avoid classification dependence in the construction of a composition series.

3. Centralizer of a normal subgroup

The purpose of this section is to prove Theorem 1.2. The problem is reduced to the transitive case. For N normal in a transitive G, $C_G(N)$ is constructed in two main stages. An action of G is constructed with a kernel $\tilde{G}$ (Lemma 3.4) and then an action of $\tilde{G}$ is constructed with kernel $C_G(N)$ (Lemma 3.6).

We start by recalling some well-known facts about centralizers of subgroups in the full symmetric group.

First we consider the case of transitive G. Thus, let $G \leq \mathrm{Sym}(\Omega)$ be transitive and fix $\alpha \in \Omega$. Observe that $G_\beta = G_\alpha$ for all $\beta \in \mathrm{fix}(G_\alpha)$, since $G_\alpha \leq G_\beta$ by the definition of $\mathrm{fix}(G_\alpha)$ and the transitivity of G implies that $|G_\alpha| = |G_\beta|$ (in general, the index of G_α in G is the size of the orbit of α). Then the following holds.

LEMMA 3.1. *Suppose $G \leq \mathrm{Sym}(\Omega)$ is transitive and $\alpha \in \Omega$. Then $C_{\mathrm{Sym}(\Omega)}(G)$ is semiregular and $\mathrm{fix}(G_\alpha)$ is an orbit of $C_{\mathrm{Sym}(\Omega)}(G)$.* □

Given $\beta \in \mathrm{fix}(G_\alpha)$, we can construct the (unique) $c = c(\alpha, \beta) \in C_{\mathrm{Sym}(\Omega)}(G)$ such that $\alpha^c = \beta$. For this, given arbitrary $\gamma \in \Omega$, γ^c is determined as follows. Take $g \in G$ such that $\alpha^g = \gamma$. Then

$$\gamma^c = \alpha^{gc} = \alpha^{cg} = \beta^g. \tag{3.1}$$

This construction is independent of the choice (g) of representative of the coset of G_α carrying α to γ, since if $\alpha^{g_1} = \alpha^{g_2}$ then $g_1 g_2^{-1} \in G_\alpha = G_\beta$ and so $\beta^{g_1} = \beta^{g_2}$.

Hence, given a transversal of G/G_α, encoded in a shallow Schreier tree, c can be computed in nearly linear time.

By semiregularity, $C_{\mathrm{Sym}(\Omega)}(G) = \langle U \rangle$ for any $U \subseteq C_{\mathrm{Sym}(\Omega)}(G)$ such that $\alpha^{\langle U \rangle} = \mathrm{fix}(\mathrm{G}_\alpha)$. Thus, to construct such U, start with $U = \emptyset$ and while there is some $\beta \in \mathrm{fix}(\mathrm{G}_\alpha) \setminus \alpha^{\langle \mathrm{U} \rangle}$, add the corresponding $c(\alpha, \beta)$ to U. As each such augmentation of U at least doubles $|\alpha^{\langle U \rangle}|$, we have to apply the construction above for at most $\log |C_{\mathrm{Sym}(\Omega)}(G)| \le \log n$ points $\beta \in \mathrm{fix}(\mathrm{G}_\alpha)$. Therefore, $C_{\mathrm{Sym}(\Omega)}(G)$ can be computed in nearly linear time as well.

Next, we treat the case of intransitive $G \le \mathrm{Sym}(\Omega)$. Since $C_{\mathrm{Sym}(\Omega)}(G)$ normalizes G, it permutes the orbits of G. We say that two orbits Δ_1, Δ_2 of G are *equivalent*, denoted $\Delta_1 \equiv \Delta_2$, if there is a bijection $\varphi : \Delta_1 \to \Delta_2$ such that for all $g \in G$ and $\delta \in \Delta_1$,

$$\delta^{g\varphi} = \delta^{\varphi g}. \tag{3.2}$$

Clearly, $\equiv$ is an equivalence relation. Also, it is clear that if $c \in C_{\mathrm{Sym}(\Omega)}(G)$ maps Δ_1 to Δ_2, then the restriction $c|_{\Delta_1}$ is a bijection satisfying (3.2). Conversely, if φ is a bijection satisfying (3.2) then φ induces an involution $\hat{\varphi}$ of $C_{\mathrm{Sym}(\Omega)}(G)$ such that $\hat{\varphi}|_{\Delta_1} = \varphi|_{\Delta_1}$, $\hat{\varphi}|_{\Delta_2} = \varphi^{-1}|_{\Delta_1}$, and $\hat{\varphi}$ fixes pointwise $\Omega \setminus (\Delta_1 \cup \Delta_2)$. Moreover, if $\Delta_1 \equiv \Delta_2$ then $C_{\mathrm{Sym}(\Delta_1)}(G|_{\Delta_1}) \cong C_{\mathrm{Sym}(\Delta_2)}(G|_{\Delta_2})$. From these observations, we obtain the following.

LEMMA 3.2. *Let $C_1, ..., C_t$ be the $\equiv$-equivalence classes of orbits of G, $|C_i| = k_i$, and, for each i, choose a representative $\Gamma_i \in C_i$. Then*

$$C_{\mathrm{Sym}(\Omega)}(G) \cong \prod_{i=1}^{t} (C_{\mathrm{Sym}(\Gamma_i)}(G|_{\Gamma_i}) \wr S_{k_i}).$$

□

If $\alpha, \beta \in \Omega$ with $G_\alpha = G_\beta$, then the orbits of α and β are equivalent (a well-defined bijection $\varphi : \alpha^G \to \beta^G$ satisfying (3.2) is given by $(\alpha^g)^\varphi = \beta^g$, for all $g \in G$). By the above,

LEMMA 3.3. *Let Δ_1, Δ_2 be orbits of G of the same size, $\alpha \in \Delta_1$. Then $\Delta_1 \equiv \Delta_2$ if and only if $\Delta_2 \cap \mathrm{fix}(\mathrm{G}_\alpha) \ne \emptyset$.* □

If α is the first base point of G then, using the set of fixed points of G_α and a shallow Schreier tree coding the transversal for G mod G_α, we can compute the orbits equivalent to the orbit of α and bijections between them in nearly linear time. Equivalence among other orbits can be determined by applying a base change algorithm to obtain a base starting with a point in the targeted orbit. However, if G has a lot of inequivalent orbits (but of the same size, so equivalence must be tested), then such computation of $C_{\mathrm{Sym}(\Omega)}(G)$ may require $\Theta(n^2)$ time even for small-base inputs.

We turn now to the computation of centralizers of normal subgroups. If $N \lhd G \le \mathrm{Sym}(\Omega)$ then $C_{\mathrm{Sym}(\Omega)}(N)$ is normalized by G so $C_G(N) = C_{\mathrm{Sym}(\Omega)}(N) \cap G$ is computable by Lemmas 3.1–3.3 and Theorem 2.4. However, as mentioned in the previous paragraph, $C_{\mathrm{Sym}(\Omega)}(N)$ may not computable in nearly linear time and/or it may not be a small-base group even if G is. Hence we have to follow a different tack.

Let B be a nonredundant base for G, $\{\Psi_1, \Psi_2, ..., \Psi_m\}$ the set of orbits of G that have non-empty intersections with B, $\Psi := \bigcup_{i=1}^{m} \Psi_i$, and G^*, N^* the restrictions of

G, N to Ψ, respectively. Since Ψ contains a base for G, G acts faithfully on Ψ and $G \cong G^*$. Any element of G^* can be uniquely extended to an element of G; moreover, by Lemma 2.3, the extension can be computed in nearly linear time, provided that the coset representatives along the point stabilizer chain corresponding to B are stored in a shallow Schreier-tree data structure. Hence it is enough to compute $C_{G^*}(N^*)$.

Clearly, $C_{G^*}(N^*)$ is contained in

$$K = \prod_{i=1}^{m} C_{G^*|_{\Psi_i}}(N^*|_{\Psi_i}),$$

K is normalized by G^*, and $|K| \leq |G|^m \leq |G|^{\log |G|}$. Hence, by Theorem 2.4, $C_{G^*}(N^*)$ is computable in nearly linear time as $C_{G^*}(N^*) = K \cap G^*$, and we reduced the computation of $C_G(N)$ to the computation of the groups $C_{G^*|_{\Psi_i}}(N^*|_{\Psi_i})$. So it suffices to solve the following problem.

Given a transitive $G = \langle S \rangle \leq \mathrm{Sym}(\Omega)$ and $N \lhd G$, (3.3)
compute $C_G(N)$ in $O(n|S| \log^c |G|)$ time.

Note that, although G is transitive in (3.3), N may have many orbits. Thus, $|C_{\mathrm{Sym}(\Omega)}(N)|$ may be too large in comparison to $|G|$, and so we still cannot apply Theorem 2.4 for computing $C_{\mathrm{Sym}(\Omega)}(N) \cap G$ in nearly linear time. Instead, we proceed via an algorithm based on the following Lemmas 3.4–3.6.

LEMMA 3.4. (i) *Let $\alpha \in \Omega$ be fixed, and let $A = \mathrm{fix}(N_\alpha)$. Then A is a block of imprimitivity for G.*
(ii) *Let $\mathcal{B}$ be the block system consisting of the G-images of A and let $\tilde{G}$ be the kernel of the G-action on $\mathcal{B}$. Then $C_G(N) \leq \tilde{G}$.*

PROOF. (i) By Lemmas 3.1, 3.2, and 3.3, A is an orbit of $C_{\mathrm{Sym}(\Omega)}(N)$. Since G normalizes $C_{\mathrm{Sym}(\Omega)}(N)$, the result follows.

(ii) The blocks in $\mathcal{B}$ are the orbits of $C_{\mathrm{Sym}(\Omega)}(N)$ so, in particular, are stabilized by $C_G(N)$. □

LEMMA 3.5. *Let $\Delta_1, \Delta_2, ..., \Delta_k$ be the orbits of N which have nonempty intersection with A and let $\Delta := \bigcup_{i=1}^k \Delta_i$. Then Δ is a block of imprimitivity for G.*

PROOF. Observe that Δ is the union of the N-images of A. Hence, in the G-action on $\mathcal{B}$, $\{\Delta_1, \Delta_2, ..., \Delta_k\}$ is an orbit of N, and therefore a block for G. Hence, $\bigcup_{i=1}^k \Delta_i$ is a block for the G-action on Ω. □

Next, choose $\alpha_i \in A \cap \Delta_i$ for $1 \leq i \leq k$. Since Δ is a block of imprimitivity for G, we can take G-images $\Delta^{g_1}, ..., \Delta^{g_m}$ of Δ which partition Ω. Recall that by Lemma 3.3, $\{\Delta_1, \Delta_2, ..., \Delta_k\}$ is an equivalence class of N-orbits. Hence the set of images $\Gamma := \{\alpha_i^{g_j} : 1 \leq i \leq k, 1 \leq j \leq m\}$ define a system of representatives of all N-orbits with the property that two N-orbits are equivalent if and only if their representatives are in the same orbit of $C_{\mathrm{Sym}(\Omega)}(N)$.

LEMMA 3.6. (i) *For each $g \in \tilde{G}$, there exists a unique $c_g \in C_{\mathrm{Sym}(\Omega)}(N)$ such that gc_g^{-1} fixes Γ pointwise.*

(ii) *The mapping $\varphi : \tilde{G} \to \mathrm{Sym}(\Omega)$ defined by $\varphi : g \mapsto gc_g^{-1}$ is a group homomorphism.*

(iii) *$C_G(N)$ is the kernel of the homomorphism φ.*

PROOF. (i) Recall that by Lemma 3.2,

$$C_{\mathrm{Sym}(\Omega)}(N) \cong \prod_{j=1}^{m} (C_{\mathrm{Sym}(\Delta_1^{g_j})}(N|_{\Delta_1^{g_j}}) \wr S_k). \tag{3.4}$$

Since $\tilde{G}$ fixes the blocks in $\mathcal{B}$, g fixes Δ^{g_j} for $1 \le j \le m$. There are k N-orbits in Δ^{g_j} and g permutes them; this permutation, regarded as an element of S_k, must also be induced by c_g in the wreath product in the j^{th} term of (3.4). Since g fixes the G-images of A and $C_{\mathrm{Sym}(\Delta_i^{g_j})}(N|_{\Delta_i^{g_j}})$ acts transitively on $A^{g_j} \cap \Delta_i^{g_j}$, for $1 \le i \le k$, $1 \le j \le m$, we can also choose c_g to fix Γ pointwise; since $C_{\mathrm{Sym}(\Delta_i^{g_j})}(N|_{\Delta_i^{g_j}})$ is semiregular, this choice is also uniquely determined.

(ii) Observe that for $g, h \in \tilde{G}$, $gc_g^{-1}hc_h^{-1} = gh(c_g^{-1})^h c_h^{-1}$. Here $(c_g^{-1})^h c_h^{-1} \in C_{\mathrm{Sym}(\Omega)}(N)$ and $gh(c_g^{-1})^h c_h^{-1}$ fixes Γ pointwise so, from the unicity of c_{gh}, $c_{gh}^{-1} = (c_g^{-1})^h c_h^{-1}$.

(iii) $C_G(N)$ is the kernel of the homomorphism φ since $\varphi(g) = 1$ if and only if $g = c_g$, i.e. $g \in \tilde{G} \cap C_{\mathrm{Sym}(\Omega)}(N) = C_G(N)$. □

Summarizing, the algorithm for computing $C_G(N)$ is the following.

```
procedure CNSinT($G, N$) [[Centralizer of Normal Subgroup in Transitive Group]]
  INPUT: $N, G \le \mathrm{Sym}(\Omega)$, $N \lhd G$, $G$ transitive on $\Omega$.
  OUTPUT: $C_G(N)$.
    Step 1. Compute a base $B$ and an SGS relative to $B$ for $G$. Compute an
            SGS relative to $B$ for $N$.
    Step 2. Let $\alpha$ be the first point of $B$. Compute $A := \mathrm{fix}(N_\alpha)$.
    Step 3. Compute the block system $\mathcal{B}$ consisting of the $G$-images of $A$, and
            $\tilde{G}$, the kernel of the $G$-action on $\mathcal{B}$.
    Step 4. Compute those $N$-orbits $\Delta_1, ..., \Delta_k$ which have nonempty intersec-
            tion with $A$, and compute $\Delta := \bigcup_{i=1}^{k} \Delta_i$. For $1 \le i \le k$, pick
            $\alpha_i \in A \cap \Delta_i$.
    Step 5. Compute $G$-images $\Delta^{g_1}, ..., \Delta^{g_m}$ which partition $\Omega$. Compute $\Gamma :=$
            $\{\alpha_i^{g_j} : 1 \le i \le k, 1 \le j \le m\}$.
    Step 6. For each generator $g$ of $\tilde{G}$, compute $c_g \in C_{\mathrm{Sym}(\Omega)}(N)$ such that
            $gc_g^{-1}$ fixes $\Gamma$ pointwise.
    Step 7. Compute and output the kernel of the homomorphism $\varphi : \tilde{G} \to$
            $\mathrm{Sym}(\Omega)$ defined by $\varphi : g \mapsto gc_g^{-1}$.
```

```
procedure CNS($G, N$) [[Centralizer of Normal Subgroup]]
    INPUT: $N, G \leq \mathrm{Sym}(\Omega)$, $N \lhd G$.
    OUTPUT: $C_G(N)$.
      Step 1. Construct a nonredundant base $B$ for $G$.
      Step 2. Let $\{\Psi_1, ..., \Psi_k\}$ be the $G$-orbits having nonempty intersection with
              $B$. Let $\Psi := \bigcup_{i=1}^{k} \Psi_i$.
              for $1 \leq i \leq k$ do
              compute $K_i := \mathrm{CNSinT}(G|_{\Psi_i}, N|_{\Psi_i})$.
      Step 3. Compute $C^* := \prod_{i=1}^{k} K_i \cap G|_{\Psi}$.
      Step 4. Extend the generators of $C^*$, considered as restrictions of elements
              of $G$, to act on $\Omega$.
              Output the extensions.
```

4. Quotients with small permutation degree

For iterative processes such as computing the upper central series, it would be useful to construct permutation representations of quotient groups. However, as shown by Neumann [14], there are examples of $G \leq S_n$, $N \lhd G$ such that G/N has no faithful permutation representation on less than $2^{n/4}$ points. On the positive side, it is proven by Easdown, Praeger [7], and Luks [11, 12] that if N is abelian then there exists a homomorphism $\varphi : G \to S_n$ such that $N \leq \ker(\varphi)$ and, in a sense, $\ker(\varphi)$ is not too big. Also, as observed in [11], φ can be constructed efficiently. We describe that construction in this section. In the next section, we shall apply it in the proof of Theorem 1.1.

THEOREM 4.1. *Let $G \leq \mathrm{Sym}(\Omega)$, $|\Omega| = n$ and let $N \lhd G$ be abelian. Then there exists a homomorphism $\varphi : G \to S_n$ such that $N \leq \ker(\varphi)$ and the prime divisors of $|N|$ and $|\ker(\varphi)|$ are the same. In particular, if N is a p-group then $\ker(\varphi)$ is an abelian p-group; moreover, if N is elementary abelian then $\ker(\varphi)$ is elementary abelian. Given generators for G and N, φ can be constructed in deterministic linear time.*

PROOF. Let $\Delta_1, ..., \Delta_k$ be the orbits of N, $\Omega = \bigcup_{i=1}^{k} \Delta_i$. Lemma 3.1 implies that a transitive abelian group is regular, so N acts regularly on each set Δ_i. In particular, $N|_{\Delta_i}$ has $|\Delta_i|$ elements. Let Σ_i denote the set of permutations in $N|_{\Delta_i}$, and let $\Sigma := \bigcup_{i=1}^{k} \Sigma_i$. Clearly, $|\Sigma| = n$. We shall define an action of G on Σ, satisfying the conclusion of the theorem.

Given $g \in G$ and $\sigma_i \in \Sigma_i$, we define $\sigma_i^g \in \Sigma$ as follows. Since $N \lhd G$, G permutes the orbits of N; suppose that $\Delta_i^g = \Delta_j$. Take any $x \in N$ with $x|_{\Delta_i} = \sigma_i$, and let $\sigma_i^g := x^g|_{\Delta_j}$. This mapping is well-defined since, if $x|_{\Delta_i} = y|_{\Delta_i}$ for some $x, y \in N$, then $x^g|_{\Delta_j} = y^g|_{\Delta_j}$. It is also clear that this construction defines a homomorphism $\varphi : G \to \mathrm{Sym}(\Sigma)$.

Now $N \leq \ker(\varphi)$, since N fixes its own orbits and commutes with $N|_{\Delta_i}$. Conversely, if $g \in \ker(\varphi)$ then g fixes the orbits of N and $g|_{\Delta_i}$ centralizes $N|_{\Delta_i}$. Since the regular subgroup $N|_{\Delta_i}$ is self-centralizing in $\mathrm{Sym}(\Delta_i)$, we conclude that $g|_{\Delta_i} \in N|_{\Delta_i}$. Hence the transitive constituents of N and $\ker(\varphi)$ are the same, implying the required relations between the prime factors and exponents.

To see the algorithmic feasibility of the construction, let us fix a representative $\delta_i \in \Delta_i$ in each orbit of N, and build a (breadth-first-search) Schreier tree T_i with root δ_i for N mod N_{δ_i}. In this tree, contrary to the usual Schreier-tree constructions, we direct the edges *away* from the root. Identifying the trivial permutation in Σ_i with δ_i, the elements of Σ_i naturally correspond to the elements of Δ_i. If $\sigma_i \in \Sigma_i$ corresponds to $\gamma \in \Delta_i$ then, in order to compute σ_i^g, it is sufficient to determine the image of δ_j under the permutation x_γ^g, where x_γ is the product of edge labels in T_i along the path from δ_i to γ. The path from δ_i to γ may have length $\Theta(|\Delta_i|)$, so computing one image $\delta_j^{x_\gamma^g}$ may require $\Theta(|\Delta_i|)$ time. However, if δ is the parent of γ in T_i and $\delta_j^{x_\delta^g}$ has already been computed, then $\delta_j^{x_\gamma^g}$ can be obtained in constant time. Thus, the entire image Σ_i^g can be constructed in $O(|\Delta_i|)$ time. □

5. Computing $O_p(G)$, $O_\infty(G)$, and $F(G)$

In this section, we prove Theorem 1.1. We shall refer to $O_\infty(G)$ as the *radical* of G and $O_p(G)$ as the *p-core* of G. Note that, since two solvable normal subgroups generate a solvable normal subgroup, there is a unique maximal solvable normal subgroup in G. Similarly, G has a unique maximal normal p-subgroup. By the same argument, we see that $O_\infty(G)$ and $O_p(G)$ are characteristic in G. Since the Fitting subgroup is the direct product of the p-cores, where p runs through the $O(\log|G|)$ prime divisors of $|G|$, it is enough to construct $O_p(G)$ and $O_\infty(G)$.

Theorem 4.1 was the key to the polynomial-time algorithms for computing $O_\infty(G)$ and $O_p(G)$ in [11]. We use the basic idea from [11] with some changes to achieve the nearly linear running time. The subgroups $O_p(G)$ and $O_\infty(G)$ are computed similarly, so we handle both of them simultaneously. We shall describe the construction of the radical, indicating the necessary changes for the construction of p-cores in double brackets [[...]].

First, let us consider the special case when G has a maximal normal subgroup N with $O_\infty(N) = 1$ [[$O_p(N) = 1$]]. This condition restricts severely the radical and p-core of G.

LEMMA 5.1. *Suppose that $N \lhd G$ is a maximal normal subgroup and $O_\infty(N) = 1$ [[$O_p(N) = 1$]]. If G/N is not cyclic [[not cyclic of order p]], then $O_\infty(G) = 1$ [[$O_p(G) = 1$]].*

PROOF. Since the radical [[p-core]] of G intersects N trivially, it must centralize N. In particular, if the radical [[p-core]] of G is non-trivial then $C_G(N)$ has non-trivial radical [[p-core]]. If $C_G(N) \le N$ then its radical [[p-core]] is trivial. If $C_G(N) \not\le N$ then $G = NC_G(N)$ and $C_G(N)/Z(N) \cong G/N$, so $C_G(N)$ could have non-trivial radical [[p-core]] only if G/N is cyclic [[is cyclic of order p]]. □

LEMMA 5.2. (i) *Suppose that $N \lhd G$ is a maximal normal subgroup, $O_\infty(N) = 1$, and G/N is cyclic. Then $O_\infty(G) = C_G(N)$.*

(ii) *Suppose that $N \lhd G$ is a maximal normal subgroup, $O_p(N) = 1$, and $|G/N| = p$. Then $O_p(G) \ne 1$ if and only if $C_G(N)$ is abelian and $p \,\big|\, |C_G(N)|$.*

PROOF. (i) Since $O_\infty(G) \cap N \le O_\infty(N) = 1$, we know $O_\infty(G) \le C_G(N)$. Suppose $C_G(N) \ne 1$. Since $C_G(N) \cap N = Z(N) \le O_\infty(N) = 1$, it follows from the maximality of N that $G = C_G(N)N$. Thus, $G/N = C_G(N)N/N \cong C_G(N)/(C_G(N) \cap N) \cong C_G(N)$, so that $C_G(N)$ is cyclic. Hence, $C_G(N) \le O_\infty(G)$.

(ii) If $C_G(N)$ is abelian and $p\big||C_G(N)|$ then the Sylow p-subgroup of $C_G(N)$ is a non-trivial normal p-subgroup of G, so that $O_p(G) \neq 1$. Suppose, conversely, that $O_p(G) \neq 1$. Since $O_p(G) \cap N \leq O_p(N) = 1$, we know $O_p(G) \leq C_G(N)$; in particular, $C_G(N) \not\leq N$, so that $G = C_G(N)N$. In this case, $C_G(N)/Z(N) \cong G/N \cong C_p$, so $p\big||C_G(N)|$; furthermore, $C_G(N) = \langle Z(N), g\rangle$ for any $g \in C_G(N) \setminus Z(N)$. But then $C_G(N)$ is abelian since g commutes with N and therefore with $Z(N)$. □

We shall also need the following simple observation.

LEMMA 5.3. *Suppose that $L \lhd H$, with H subnormal in G and with L solvable [[L a p-group]]. Then $\langle L^G\rangle$ is solvable [[$\langle L^G\rangle$ is a p-group]].*

PROOF. Let $H \lhd G_1 \lhd G_2 \lhd \cdots \lhd G_m = G$. Prove by induction on i that $\langle L^{G_i}\rangle$ is solvable [[p-group]]. □

Now we are ready to describe the construction of $O_\infty(G)$ [[$O_p(G)$]].

First, we produce *some* non-trivial solvable [[p–group]] $H \lhd G$ provided G has a non-trivial radical [[p-core]] (so that failure guarantees that the radical [[p-core]] is trivial). For this, we compute a composition series $1 = N_1 \lhd N_2 \lhd \cdots \lhd N_m = G$. Then we find the smallest index i such that N_i has a non-trivial radical [[p-core]] as follows. Suppose that $O_\infty(N_i) = 1$ [[$O_p(N_i) = 1$]]. If N_{i+1}/N_i is not cyclic [[not cyclic of order p]] then, by Lemma 5.1, we conclude that $O_\infty(N_{i+1}) = 1$ [[$O_p(N_{i+1}) = 1$]]. Otherwise, we compute $C_{N_{i+1}}(N_i)$, using the method of Section 3. If $C_{N_{i+1}}(N_i) = 1$ [[$C_{N_{i+1}}(N_i)$ not abelian with $p\big||C_{N_{i+1}}(N_i)|$]] then, by Lemma 5.2, $O_\infty(N_{i+1}) = 1$ [[$O_p(N_{i+1}) = 1$]]. Otherwise, set $L := C_{N_{i+1}}(N_i) \neq 1$ [[$L :=$ any non-trivial p-subgroup of $C_{N_{i+1}}(N_i)$, e.g., take any generator g with $p\big|\mathrm{order}(g)$ and set $L := \langle g^{\mathrm{order}(g)/p}\rangle$]]. By Lemma 5.3, $H = \langle L^G\rangle$ is a solvable normal subgroup [[normal p-group]] of G.

By taking the last non-trivial term in the derived series of H, we produce a non-trivial abelian normal subgroup [[abelian normal p–subgroup]] N of G. With respect to this N, we compute the homomorphism $\varphi : G \to S_n$ described in Theorem 4.1.

Finally, we recursively compute $M = O_\infty(\varphi(G))$ [[$M = O_p(\varphi(G))$]]. Since $\ker(\varphi)$ is abelian [[an abelian p-group]], $\varphi^{-1}(M)$ is $O_\infty(G)$ [[$O_p(G)$]]. The inverse image, $\varphi^{-1}(M)$, is generated by $\ker(\varphi)$ together with liftings of the generators of M, the liftings obtained, say, via Lemma 2.3.

(Note that the composition-series computation need not be repeated for $\varphi(G)$. We have $1 = \varphi(N_1) \trianglelefteq \varphi(N_2) \trianglelefteq \cdots \trianglelefteq \varphi(N_m) = \varphi(G)$, with $\varphi(N_{i+1})/\varphi(N_i) \cong N_{i+1}/N_i$ or $\varphi(N_{i+1})/\varphi(N_i) \cong 1$.)

This completes the description of the algorithm, proving Theorem 1.1. □

REMARK 5.4. As mentioned in Remark 2.6, the use of composition factors can be avoided. In [11, 12], it is shown that one can make use of any proper normal subgroup N in a recursive construction of a suitable H as above. Namely, recursively seek a solvable [[a p-group]] L such that $1 < L \trianglelefteq N$. If this succeeds, set $H := \langle L^G\rangle$. Otherwise similarly seek a solvable [[a p-group]] L such that $1 < L \trianglelefteq C_G(N)$. A second failure implies $O_\infty(G) = 1$ [[$O_p(G) = 1$]]. Critical now to the nearly-linear timing is the observation that the second recursive call needs to be made only when $O_\infty(N) = 1$ [[$O_p(G) = 1$]] in which case $|N||C_G(N)| \leq |G|$ [[$|N|_p|C_G(N)|_p \leq |G|_p$, where $|X|_p$ denotes the p-part of $|X|$]]. Finally, one shows, under the assumption that $O_\infty(G) \neq 1$, one can get by with relatively elementary (in particular, not dependent on the simple groups classification) portions of the methods of [10].

References

1. L. Babai, G. Cooperman, L. Finkelstein, and Á. Seress, *Nearly linear time algorithms for permutation groups with a small base*, Proc. 1991 ACM-SIGSAM Inter. Symp. on Symbolic and Algebraic Comp., 1991, pp. 200–209.
2. R. Beals, *Constructing blocks of imprimitivity in nearly linear time for small-base groups*, in: Groups and Computation, L. Finkelstein, W. Kantor ed., DIMACS Series on Discrete Math. and Theor. Computer Science **11**, AMS 1993, pp. 17–26.
3. R. Beals, *An elementary algorithm for computing the composition factors of a permutation group*, Proc. 1993 ACM-SIGSAM Inter. Symp. on Symbolic and Algebraic Comp., 1993, pp. 127–134.
4. R. Beals and Á. Seress, *Structure forest and composition factors for small base groups in nearly linear time*, Proc. 24th ACM STOC (1992), pp. 116–125.
5. G. Cooperman and L. Finkelstein, *A random base change algorithm for permutation groups*, J. Symb. Comp. **17** (1994), pp. 513–528.
6. G. Cooperman, L. Finkelstein, and E.M. Luks, *Reduction of group constructions to point stabilizers*, Proc. 1989 ACM-SIGSAM Inter. Symp. on Symbolic and Algebraic Comp., 1989, pp. 351–356.
7. D. Easdown and C. E. Praeger, *On minimal faithful permutation representations of finite groups*, Bull. Aust. Math. Soc. **38** (1988), 207–220.
8. M. Hall, Jr., The Theory of Groups, Macmillan, New York 1959.
9. W. M. Kantor, *Sylow's theorem in polynomial time*, J. Comp. and Syst. Sci. **30** (1985), pp. 359–394.
10. E.M. Luks, *Computing the composition factors of a permutation group in polynomial time*, Combinatorica **7** (1987), pp. 87-99.
11. E. M. Luks, *Lectures on Polynomial-Time Computation in Groups*, Northeastern University, Boston, 1990.
12. E. M. Luks, *Permutation groups and polynomial-time computation*, in: Groups and Computation, L. Finkelstein, W. Kantor ed., DIMACS Series on Discrete Math. and Theor. Computer Science **11**, AMS 1993, pp. 139–175.
13. P. Morje, *On nearly linear time algorithms for Sylow subgroups of small-base permutation groups*, this Proceedings.
14. P. M. Neumann, *Some algorithms for computing with finite permutation groups*, Proc. of Groups-St Andrews 1985, E.F. Robertson, C.M. Campbell ed., London Math. Soc. Lect. Note 121, Cambridge Univ. Press 1987, pp. 59–92.
15. F. Rákóczi, *Fast recognition of nilpotent permutation groups*, Proc. 1995 ACM-SIGSAM Inter. Symp. on Symbolic and Algebraic Comp., 1995, pp. 265–269.
16. M. Schönert et. al., *GAP: Groups, Algorithms, and Programming*, Lehrstuhl D für Mathematik, RWTH Aachen, 1992.
17. M. Schönert and Á. Seress, *Finding blocks of imprimitivity in small-base groups in nearly linear time*, Proc. 1994 ACM-SIGSAM Inter. Symp. on Symbolic and Algebraic Comp., 1994, pp. 154–157.
18. Á. Seress, *Nearly linear time algorithms for permutation groups: an interplay between theory and practice*, Acta Appl. Math., to appear.
19. Á. Seress, Permutation Group Algorithms, book manuscript.
20. C. C. Sims, *Computational methods in the study of permutation groups*, in: Computational Problems in Abstract Algebra, J. Leech, ed., Pergamon Press 1970, pp. 169–183.
21. C. C. Sims, *Computation with Permutation Groups*, Proc. Second Symposium on Symbolic and Algebraic Manipulation, S.R. Petrick, ed., ACM, New York, 1971, pp. 23–28.
22. H. Wielandt, Finite Permutation Groups, Acad. Press, New York 1964.

Department of Computer and Information Science, University of Oregon, Eugene, OR 97403
E-mail address: luks@cs.uoregon.edu

Department of Mathematics, The Ohio State University, Columbus, OH 43210
E-mail address: akos@math.ohio-state.edu