DATE: July 31, 1998

A polynomial-time theory of black-box groups I

by
László Babai[1] and Robert Beals[2]
University of Chicago and University of Arizona

Abstract

We consider the asymptotic complexity of algorithms to manipulate matrix groups over finite fields. Groups are given by a list of generators. Some of the rudimentary tasks such as membership testing and computing the order are not expected to admit polynomial-time solutions due to number theoretic obstacles such as factoring integers and discrete logarithm. While these and other "abelian obstacles" persist, we demonstrate that the "nonabelian normal structure" of matrix groups over finite fields can be mapped out in great detail by polynomial-time randomized (Monte Carlo) algorithms.

The methods are based on statistical results on finite simple groups. We indicate the elements of a project under way towards a more complete "recognition" of such groups in polynomial time. In particular, under a now plausible hypothesis, we are able to determine the names of all nonabelian composition factors of a matrix group over a finite field.

Our context is actually far more general than matrix groups: most of the algorithms work for "black-box groups" under minimal assumptions. In a black-box group, the group elements are encoded by strings of uniform length, and the group operations are performed by a "black box."

1 Introduction

1.1 Outline of objectives

Let G be a finite group given by a list of generators. Our aim is to design asymptotically efficient algorithms to obtain structural information about G.

[1]Dept. Computer Science, University of Chicago, 1100 E 58th St, Chicago, IL 60637-1504. E-mail: `laci@@cs.uchicago.edu`

[2]Dept. Mathematics, University of Arizona, Tucson AZ 85721-0001. E-mail: `beals@@math.arizona.edu`

Typically G will be a matrix group over a finite field, but most of our algorithms work in the more general context of "black-box groups." In a *black-box group*, a "black box" performs the group operations on codewords representing the group elements. Each codeword has the same length n, the *encoding length.* (See Section 3.1 for details). This generalization, introduced in [BaSz], is critical to our work. Work in progress on statistical recognition of *simple* black-box groups will amplify the significance of the black-box approach (see Section 9). In particular, assuming success of that project, we shall be able to determine the names of all nonabelian composition factors of matrix groups over finite fields.

Our goal is to explore the power of *Monte Carlo* (randomized) *polynomial-time algorithms.* The rigorously proven performance guarantees accompanying the algorithms will include a user-prescribed bound ϵ on the probability of error and guaranteed polynomial time bounds. The complexity of such an algorithm will always be of the form $O(n^c \log(1/\epsilon))$ where n is the encoding length and c is a constant, so in polynomial time we can achieve an exponentially small probability of error. (A more detailed upper bound on the complexity may be available as a function of specific parameters of the input.)

Unfortunately some of the rudimentary tasks such as membership testing and determining the order of the group are not likely to be solvable in polynomial time for matrix groups over finite fields, due to their close association with hard problems in computational number theory such as factoring integers and discrete logarithm.

In spite of this significant handicap, we show that a great deal of structural information about black-box groups can be obtained in polynomial time. We emphasize, that, in contrast to previous work [Lu92, BeB], in this paper we do not assume an "oracle" (black box) for discrete logarithms, and our estimates do not involve quantities such as the largest prime divisor of the order [Lu92] or the minimum degree of permutation representations of composition factors [BeB]. Our algorithms are genuinely polynomial time (in the length of the input). The only concession we make is that for the most general black-box group results we assume that a superset of the primes dividing $|G|$ is available (preprocessing). However, for the most important applications (matrix groups over finite fields and their quotients), we get rid of this assumption and remain entirely within polynomial time.

In this paper we do not address questions of implementation. We believe, that, as has been the case in the past, the mathematical insights forced by the rigor of guaranteed polynomial time will lead, through further theoretical work as well as through the use of heuristic shortcuts, to algorithms with the potential of practical implementation[3].

1.2 Basic normal structure

We consider the following chain of characteristic subgroups of the finite group G:

$$1 \le \operatorname{Sol}(G) \le \operatorname{Soc}^*(G) \le \operatorname{Pker}(G) \le G. \tag{1}$$

Here $\operatorname{Sol}(G)$ is the *solvable radical* of G (largest solvable normal subgroup). We define $\operatorname{Soc}^*(G)$ by

$$\operatorname{Soc}^*(G)/\operatorname{Sol}(G) = \operatorname{Soc}(G/\operatorname{Sol}(G)) \tag{2}$$

where $\operatorname{Soc}(H)$ is the socle of H (product of the minimal normal subgroups of H). We say that a group H is *semisimple* if H is the product of *nonabelian* simple groups. We refer to $\operatorname{Soc}^*(G)/\operatorname{Sol}(G)$ as the *semisimple socle* of G. (Note that in general this is *not* the semisimple part of $\operatorname{Soc}(G)$.) Let

$$\operatorname{Soc}^*(G)/\operatorname{Sol}(G) = T_1 \times \cdots \times T_k \tag{3}$$

where the T_i are nonabelian simple groups. Let $\varphi : G \to \Sigma_k$ be the permutation representation of G via conjugation action on the set $\{T_1, \ldots, T_k\}$. (By Σ_k we denote the symmetric group of degree k.) We define the *permutation kernel* $\operatorname{Pker}(G)$ of G by

$$\operatorname{Pker}(G) = \ker(\varphi). \tag{4}$$

We make the following observations regarding the quotients of the characteristic chain (1).

- (i) $\operatorname{Sol}(G)$ is solvable;
- (ii) $\operatorname{Soc}^*(G)/\operatorname{Sol}(G)$ is semisimple (eqn. (3));
- (iii) $\operatorname{Pker}(G)/\operatorname{Soc}^*(G)$ is solvable; in fact $(\operatorname{Pker}(G)/\operatorname{Soc}^*(G))''' = 1$;
- (iv) $G/\operatorname{Pker}(G) \le \Sigma_k$, and $k \le \log|G|/\log 60$.

The reason for item (iii) is the observation that

$$\operatorname{Pker}(G)/\operatorname{Soc}^*(G) \le \operatorname{Out}(T_1) \times \cdots \times \operatorname{Out}(T_k). \tag{5}$$

This implies that $\operatorname{Pker}(G)/\operatorname{Soc}^*(G)$ is solvable according to Schreier's conjecture; in fact $(\operatorname{Pker}(G)/\operatorname{Soc}^*(G))''' = 1$ (cf. [CoCNPW]). Item (iv) is immediate from equation (3).

[3] A prime example is Luks's polynomial time algorithm for constructing the composition chain of a permutation group [Lu87]. A sequence of asymptotic improvements culminated in a nearly linear time algorithm (still based on Luks's basic outline) by Beals and Seress [BeS]. With appropriate heuristic shortcuts, Seress implemented the [BeS] algorithm in GAP [Sch, Se].

1.3 The main results

Let us now return to our basic object, a black-box group G of encoding length n. To simplify the presentation, we assume that a superset $\mathcal{P} \supseteq \pi(G)$ is also given, where $\pi(G)$ denotes the set of primes dividing $|G|$. This assumption is partly justified by requiring a *preprocessing* only, rather than the availability of an "oracle" to solve problems like discrete logarithm on-line.

Nonetheless, we consider the assumption of prime factors to be an unpleasant one. In Section 8 we shall see how to dispense with this assumption in the most important subcase of *black-box groups of characteristic p*. This subcase includes the subgroups of $GL(d, p^k)$ and their quotients by recognizable normal subgroups, assuming the group elements are encoded as $d \times d$ matrices over $GF(p^k)$.

Terminology. To *construct* (or synonymously, to *find*) a subgroup H of a black-box group G means to construct generators for H. To *recognize* H means to test, for any $g \in G$, membership of g in H. The cost of constructing H is the time or other cost measure (such as number of group operations) required to construct a set of generators for H. The cost of recognizing H is the maximum over $g \in G$ of the cost of the decision whether or not $g \in H$.

Unfortunately, efficient construction is not known to imply efficient recognition or vice versa. This is a major obstacle we have to face all along. We note that for *permutation groups*, construction does imply recognition in polynomial time (membership testing is available in polynomial time [Si78, FuHL]), but even for permutation groups, recognition does not seem to imply polynomial-time construction: the automorphism group of a graph is *recognizable* in polynomial time (membership testing is straightforward), yet we are unable to construct a set of generators in polynomial time. The latter task is equivalent to the graph isomorphism problem, not known to be solvable in polynomial time (cf. [Ba95, Section 6] for a survey of the graph isomorphism problem).

In this paper we shall try to *construct* the members of the chain (1), refine the chain to a composition chain, and identify the composition factors. Ideally, we would also want to *recognize* these subgroups. Certain "abelian obstacles" (Section 3.5) prevent us from completing this plan in polynomial time, but we claim considerable degree of success in the "nonabelian part" of the project.

By a *quasi-composition chain* of a group H we mean a subnormal chain $1 = H_m \triangleleft H_{m-1} \triangleleft \cdots \triangleleft H_0 = H$ such that all quotients H_{i-1}/H_i are abelian or simple nonabelian. In the case of an abelian quotient we allow the case

$H_{i-1} = H_i$ since we do not have membership testing. If we say that a quasi-composition chain is *known,* we mean that generators of each H_i are known, and furthermore we know which quotients are abelian and which are simple nonabelian.

First we note that $\mathrm{Sol}(G)$ is *recognizable* in Monte Carlo polynomial time, and therefore we can treat $G/\mathrm{Sol}(G)$ as a black-box group (Corollary 3.3). On the other hand, we are not able to *construct* $\mathrm{Sol}(G)$.

Theorem 1.1 *Given a black-box group G along with a set $\mathcal{P} \supseteq \pi(G)$, one can construct, in Monte Carlo polynomial time, the permutation kernel* $\mathrm{Pker}(G)$*, the semisimple socle* $\mathrm{Soc}^*(G)/\mathrm{Sol}(G)$*, the decomposition (3) of* $\mathrm{Soc}^*(G)/\mathrm{Sol}(G)$ *into a product of k simple groups, and the permutation representation $\varphi : G \to \Sigma_k$ with kernel* $\mathrm{Pker}(G)$*. Consequently one can construct, in Monte Carlo polynomial time, a quasi-composition chain of* $G/\mathrm{Sol}(G)$ *as well as a black-box representation of each nonabelian composition factor of G.*

(The last statement includes exhibiting a correspondence between the nonabelian composition factors of G and the corresponding members of the quasi-composition chain of $G/\mathrm{Sol}(G)$.)

We note that (iv) above implies $k \leq n/\log 60 < n/5$; therefore once φ has been computed, the extensive library of polynomial time algorithms for permutation groups becomes available for $G/\mathrm{Pker}(G)$ [BaLS, Lu93]. In particular, we can construct a composition chain of $G/\mathrm{Pker}(G)$ as well as permutation representations of the composition factors; moreover we can construct a presentation in terms of generators and relations.

"Constructing" φ would mean defining the φ-images of generators. However, we "construct" φ in a stronger sense: we set up a data structure which allows $\varphi(g)$ to be computed in polynomial time, for arbitrary $g \in G$. We do not assume here that we know how to construct g from the generators of G. This makes $\mathrm{Pker}(G)$ *recognizable within G:* given $g \in G$, we have $g \in \mathrm{Pker}(G)$ exactly if $\varphi(g) = 1$, and the latter is decidable in polynomial time.

We are, however, unable to recognize $\mathrm{Soc}^*(G)$, determine the order of the solvable group $\mathrm{Pker}(G)/\mathrm{Soc}^*(G)$, or even decide whether or not $\mathrm{Pker}(G) = \mathrm{Soc}^*(G)$.

The most elusive part of the project is the determination of the solvable radical $\mathrm{Sol}(G)$; we are unable to find a set of generators for $\mathrm{Sol}(G)$, or even to decide whether or not $\mathrm{Sol}(G) = 1$.

For *black-box groups of characteristic p* (Section 8.2), we eliminate the assumption that a superset $\mathcal{P}$ of primes is given.

Theorem 1.2 *Given a black-box group G of characteristic p, one can construct, in Monte Carlo polynomial time, all the objects listed in Theorem 1.1.*

We remark that for black-box groups of characteristic p we are able to make considerable inroads into $\operatorname{Sol}(G)$ and the only really elusive part that remains is $O_p(G)$, the largest normal p-subgroup. We give the details in a subsequent paper [BaB2].

1.4 Overview of the algorithm

We indicate the overall structure of the algorithm announced in Theorem 1.1.

The first phase consist of the construction of the semisimple socle. For this phase, we work in the black-box group $\bar{G} = G/\operatorname{Sol}(G)$. We note that $\operatorname{Sol}(\bar{G}) = 1$.

First we need to find a minimal subnormal subgroup of $\bar{G}$. This is accomplished by a process of "blind descent" along a subnormal chain (Sections 6.3, 7.1).

The normal closure of a minimal subnormal subgroup is semisimple (because $\operatorname{Sol}(\bar{G}) = 1$). Our key auxiliary process splits a semisimple group into its simple factors (Section 5). This decomposition helps to extend our semisimple normal subgroup of $\bar{G}$ until the entire socle of $\bar{G}$ is found (Section 7).

The decomposition of $\operatorname{Soc}(\bar{G}) = \operatorname{Soc}^*(G)/\operatorname{Sol}(G)$ makes it easy to construct the permutation representation φ (Corollary 5.2). Given φ, standard permutation group techniques yield generators for $\operatorname{Pker}(G)$ (Proposition 3.4). Finally, a quasi-composition chain of $\bar{G}$ is easily constructed: we apply Luks's composition chain algorithm [Lu87] to $G/\operatorname{Pker}(G)$, followed by three steps of the derived series of $\operatorname{Pker}(G)/\operatorname{Sol}(G)$ and ending with a composition chain of $\operatorname{Soc}^*(G)/\operatorname{Sol}(G)$ which is obtained from the decomposition of $\operatorname{Soc}^*(G)/\operatorname{Soc}(G)$ (Section 5).

The proof of Theorem 1.2 requires the introduction of a set of "pretend-primes" to be used in determining the "pseudo-order" of elements (Section 8) to be used in our algorithms wherever computation of the order would be required. (Computing the order would require prime factorization, a problem not known to be solvable in polynomial time, cf. Section 8.1.)

Two results on finite simple groups, stated in Section 4, will play a critical role in the analysis of our algorithms. One is a bound by Landazuri and Seitz [LaS] stating that the degree of a representation of a Lie-type simple group in the wrong characteristic must be very large. We apply this result in several different contexts: in the analysis of "blind descent" (Lemma 6.3), in the selection of the "pretend-primes" for the calculation of "pseudo-order," and for handling the p'-part of the solvable radical [BaB2].

The other critical result states that for any prime r, elements of order not divisible by r occur with non-negligible frequency in every finite simple group (Theorems 4.2, 8.7). This statistical result is the key to breaking a direct product of simple groups to its factors and finding the semisimple socle (Sections 5, 7). Another application is to finding the center of a quasisimple group (see [BaB2]).

2 Brief history

2.1 Permutation groups

Charles Sims pioneered the design of efficient algorithms for the rudimentary tasks of managing permutation groups (membership, order, normal closure) based on his fundamental data structure of "strong generators" [Si71, Si78]. A polynomial time analysis of closely related algorithms was first given in [Ba79] in the context of an application to the Graph Isomorphism problem. [Ba79] employed randomization and introduced the term "Las Vegas." Motivated by [Ba79], a seminal paper by Furst, Hopcroft, and Luks [FuHL] designed a variant of Sims's algorithm and proved that it runs in polynomial time. Other polynomial-time versions of Sims's algorithm were described by Knuth [Kn] and Jerrum [Je]. Seress found that Sims's original version, too, runs in polynomial time [Se].

[FuHL] was followed by a succession of powerful polynomial-time algorithms for permutation groups. We highlight two results: Luks's algorithm to construct a composition chain along with permutation representations of the composition factors [Lu87], and Kantor's algorithm to construct the Sylow subgroups [Ka].

2.2 Complexity theory in black-box groups

The first paper addressing the complexity of basic computational problems for finite matrix groups was [BaSz]. That paper introduced the framework of black-box groups and proved that membership in subgroups of black-box groups is in the complexity class NP (has polynomial length verification) by proving that any element of a finite group G can be reached by at most $(1+\log |G|)^2$ group operations from any set of generators. The combinatorial idea of the proof became later the basis of the polynomial-time Monte Carlo algorithm for selecting nearly uniformly distributed random elements in a black-box group (Theorem 3.2).

Another result of [BaSz] asserts that determining the order of a black-box group is in the complexity class NP assuming the *"Short Presentation Conjecture"* which states that every finite simple group T has a presentation (in terms of generators and relations) of total bitlength $\leq (\log |T|)^c$, and that such a presentation can be constructed efficiently (in Las Vegas time $\leq (\log |T|)^c$), given the standard name of T. This conjecture has since been verified for all finite simple groups except for the three families of rank-1 twisted groups: the unitary groups $PSU(3,q) = {}^2A_2(q)$, the Suzuki groups $Sz(q) = {}^2B_2(q)$, and the Ree groups $R(q) = {}^2G_2(q)$ [BaGKLP]. A full proof of this conjecture will play an essential role in turning Monte Carlo algorithms for matrix groups into the more desirable Las Vegas variety (guaranteed no error), a significant conceptual and practical leap because of the use of unproven heuristics in random sampling from black-box groups and matrix groups (Section 3.3, cf. [Ba97]).

2.3 Polynomial time algorithms in matrix groups

The first polynomial-time algorithms for a host of problems on matrix groups appear in Luks's paper [Lu92]. In contrast to virtually all subsequent work, Luks's algorithms are deterministic. Luks showed that deciding solvability of matrix groups over finite fields is in polynomial time. Moreover he gave algorithms for membership testing and order, computing presentations, composition chain, as well as many other problems, for solvable matrix groups. Many of these algorithms are not polynomial time (in the bit-length of the input) but depend polynomially on the largest prime divisor of $|G|$, other than the characteristic. Some limitation of this kind seems indispensable for the membership and order problems because of the number theoretic obstacles mentioned in Section 3.5.

Algorithms for membership testing, order, computing presentations, composition chain, and other problems for (not necessarily solvable) finite matrix groups are given by Beals and Babai [BeB]. Unlike the algorithms of [Lu92], these algorithms are randomized (Las Vegas). As with Luks's algorithms, the timing is polynomial (in the bit-length of the input) in characteristic 0 (algebraic number fields), but depends polynomially on an additional parameter ν in characteristic p. The parameter $\nu = \nu(H)$ is defined as the largest integer k such that H has a composition factor $L \neq \mathbb{Z}_p$ which has no permutation representation of degree $< k$.

The large abelian composition factors represent a genuine obstacle (see

Section 3.5). We survey recent progress on how to get around some of the nonabelian obstacles in Section 9.

2.4 The Aschbacher classification

In this paper we use "polynomial time" as the criterion of efficiency. A more traditional approach measures efficiency by practical performance. We note here some important work in this direction. Many of the algorithms referred to below have not been shown to run in polynomial time. However, implementations in GAP and Magma are available with impressive test data; they work even in dimensions over 100.

Aschbacher has classified the maximal subgroups of the finite classical groups [As]. We give a summary of the Aschbacher classification: Let $G \leq GL(d,q)$, and let $Z = Z(GL(d,q)) \cap G$. Let V denote the vector space F_q^d. Then at least one of the following holds:

C1 G is reducible.

C2 G is imprimitive.

C3 G fixes a tensor decomposition $V = U \otimes W$.

C4 G preserves a symmetric tensor decomposition: $V = V_1 \otimes V_2 \otimes \cdots \otimes V_m$, where each V_i has the same dimension r, and $d = r^m$.

C5 G acts semi-linearly over an extension field F_{q^e}, so $V = F_{q^e}^{d/e}$ and $G \leq \Gamma L(d/e, q^e)$.

C6 Modulo Z, G is conjugate to a subgroup of $GL(d,q')$, for some $q' \mid q$.

C7 $d = r^m$ for some prime R, G normalizes an r-group R, of order r^{2m+1} or 2^{2m+2}. In the first case, R is extraspecial; in the second case, R is symplectic.

C8 $K' \leq G/Z \leq K$, where K is a projective general linear, symplectic, orthogonal, or unitary group in dimension d over F_q.

C9 $T \leq G/Z \leq \mathrm{Aut}(T)$, where T is nonabelian simple.

The "recognition project," led by C. R. Leedham-Green, is guided by the following computational problem related to the Aschbacher classification: given a list of generators for a subgroup H of $GL(n,q)$, find a maximal subgroup $G \leq GL(n,q)$ containing H. For several of the classes C1–C9, algorithms exist which seem to perform well in practice. The classes C1 and C5 are handled by the Meataxe [Par84, HoR94]. Since the Meataxe, the first significant step in the recognition project was the Neumann–Praeger algorithm [NeP, HoR92] for case C8 in the case K is $PGL(d,q)$. The other classical groups that can arise in C8 are handled by Niemeyer and Praeger [NiP98, NiP97].

In many cases, H has a normal subgroup N preserving some structure which is acted on by H. For example, in case C4, H permutes the V_i, and N is the kernel of this action. The SMASH algorithm [HLOR2], when given an element $x \in N \setminus Z$, attempts to find H in classes C2, C3, C4, and C7. SMASH does not solve, in general, the problem of finding such an x, though the SMASH authors have efficient heuristics for C2 [HLOR1]. And of course, G may intersect N trivially (or only in Z), in which case SMASH does not apply. Leedham-Green and O'Brien [LeO] have given another approach to C3.

Any implementation of our ideas should make use of the practical successes of the recognition project. The SMASH algorithm seems particularly suited to our needs, since we have, in many instances, algorithms for finding elements of nontrivial proper normal subgroups. Such an element, supplied to SMASH, may yield a permutation representation of G (in cases C2, C4, or C7), which is an important subgoal of our algorithm. Conversely, ideas from the theory of black-box group algorithms should be useful in the recognition project. In the case G is in class C9, what we have is, for practical purposes, a black-box group. In cases where SMASH is applicable, we present a well developed theory of how to find elements that are likely to lie in nontrivial proper normal subgroups (Section 6).

3 Algorithmic preliminaries

3.1 Black-box groups

A *black-box group* G is a finite group whose elements are encoded by $(0,1)$-strings ("codewords") of uniform length n [BaSz]. We call n the *encoding length* of the black-box group. This convention implies $|G| \leq 2^n$.

Group operations on the codewords are performed by a "black box" at unit cost. The operations are *multiplication, inversion, and identity testing* (decision whether or not a given string encodes the identity). A black-box group is *given* by a list of generators.

Significantly, we do not require the encoding to be unique. Our most important models of black-box groups are *quotients of matrix groups* over finite rings: $G = H/N$ where $N \triangleleft H \leq GL(d, R)$ for some finite ring R (usually a finite field). The elements of G are encoded as matrices (elements of H). Identity testing requires membership testing in N (magically, or by some algorithm).

We say that a subgroup H of the black-box group G is *recognizable in time t* if for every $g \in G$, membership of g in H can be tested in time t. Instead of "time" we can use other measures of complexity, e. g.. the number of group operations required. (The membership algorithm for H must not depend on how g is constructed from the generators of G. If the sequence of operations leading to g is required for the decision, we say that H is *weakly recognizable* in time t. However, we shall not need this concept here.) The following evident fact will be very useful and demonstrates the power of the black box model.

Proposition 3.1 *Let G be a black-box group and $H \triangleleft G$ a normal subgroup recognizable in time t (or at cost t). Then the black box for G can simulate a black box for G/N in time t per operation.* ■

In fact, the cost of the simulation is one step per multiplication/inversion, and t per identity query. Note that identity queries typically make up only a small fraction of the operations; they do not occur at all in the process of random sampling, one of the expensive routines.

If a black-box group of encoding length n is given by a list of k generators then the *bit-length of the input* is kn and therefore a polynomial-time algorithm makes $\leq (kn)^c$ steps for some constant $c > 0$.

Convention. We shall use the letter c to denote positive constants; separate occurrences of c may refer to *different* constants. We also use $c_1, c_2, \ldots$ to denote positive constants; repeated occurrences of c_i refer to the *same* constant. The letter ϵ denotes a positive quantity, not necessarily a constant, and separate occurrences of ϵ may refer to different values.

3.2 Monte Carlo and Las Vegas algorithms

Our algorithms will use randomization; therefore the output will not necessarily be correct. However, the probability of error will be guaranteed to be less than ϵ, regardless of the input. The parameter ϵ is chosen by the user. The cost increases proportionally to $\log(1/\epsilon)$, so exponentially small error probability is achieved at polynomial cost. We should emphasize that probabilities are over the coin tosses made in the course of the execution of the algorithm. We do not assume any probability distribution over the input space: we perform "worst case" analysis (as opposed to "average case analysis").

We use the term "Monte Carlo algorithm" to refer to any randomized algorithm with a user-specified arbitrarily small error bound ϵ (which may be proven or heuristic). A *polynomial time Monte Carlo algorithm* is guaranteed, for any requested error margin $\epsilon > 0$, to stop in time $O(n^c \log(1/\epsilon))$ and have error probability $\leq \epsilon$.

"Las Vegas algorithms" form an important subclass of Monte Carlo algorithms: they never make an erroneous output, but with probability $\leq \epsilon$, they are allowed to report failure.

3.3 Random sampling from groups

By a *random element* of a finite set S we mean a nearly uniformly distributed random member of S. *Near-uniformity* means that every element of S has $(1 \pm \epsilon)/|S|$ chance to be selected; here $\epsilon > 0$ is a parameter chosen by the user. When we speak of selecting *several random elements*, we always assume *independent* random choices.

A critical tool in all algorithms under discussion is a method of selecting random elements from a black-box group. Our polynomial-time claims heavily depend on the following result.

Theorem 3.2 ([Ba91]) *An ϵ-uniformly distributed random element of a black-box group can be selected in Monte Carlo polynomial time at a cost of $O(n^c \log(1/\epsilon))$.*

This algorithm uses randomization but it makes no error in the sense that it is guaranteed to produce an ϵ-uniformly distributed element. Several *independent* elements from the same distribution can be obtained by repeating the experiment with a separate set of coin tosses. The actual result in [Ba91] gives a separate cost estimate for preprocessing and a considerably lower cost per random element generated (preprocessing costs $O(n^5)$ group operations and yields $\Theta(n)$ random elements; additional random elements cost $O(n)$ group operations each).

While this algorithm does not produce uniformly distributed random group elements, the slight deviation from uniformity has no significant effect on the analysis of our algorithms.

The "product replacement algorithm" [CeLMNO], a popular heuristic introduced by Charles Leedham-Green and Leonard Soicher, is another candidate for a polynomial-time method for nearly uniform selection from a finite group. However, very little has yet been proven in this direction (cf. [Ba97]). Of course in practical implementations of either of the above algorithms one uses much fewer operations than required for proven performance, and hopes for the best.

When efficient but unproven heuristics are invoked for sampling in groups under a randomized algorithm, Las Vegas algorithms are greatly superior to Monte Carlo since they never err, even if the distribution of the "random" group elements is not as nearly uniform as desired. The worst that can happen in such a case is that the algorithm reports failure more frequently then expected and thus a weakness of the sampling method is discovered. However, with Monte Carlo methods, the inadequacy of the sampling method may mean frequent erroneous outputs which may remain undetected.

Unfortunately, most of the algorithms in this area are not Las Vegas, and there is little hope for turning them into Las Vegas as long as some cases of the "Short Presentation Conjecture" (Section 2.2) remain open (cf. [Ba97]).

For a detailed discussion of the concepts involved in and possible pitfalls of Monte Carlo algorithms in finite groups, we recommend [Ba97].

3.4 Further rudiments: normal closure and applications

Recall our standard notation: G is a black-box group of encoding length n and therefore $|G| \le 2^n$. It follows that any set of more than n generators is redundant. In fact, in Monte Carlo polynomial time one can replace any set of generators by a list of $O(n)$ generators [BaCFLS], i. e., the number of generators is $\le cn$ for some constant c. Therefore we shall assume that all groups considered are given by $O(n)$ generators.

A fundamental polynomial-time Monte Carlo algorithm given in [BaCFLS] is constructing the *normal closure* $\langle S^G \rangle$ of a set $S \subseteq G$ (i. e., finding a set of $O(n)$ generators for $\langle S^G \rangle$). We note that this claim does not depend on nearly uniform random selection from G and uses the more elementary tool of "random subproducts."

As an immediate consequence, the following subgroups are constructible in polynomial Monte Carlo time: the *commutator subgroup* G', all members $G^{(i)}$ of the derived series, the *stable derivative* $G^{(\infty)} = G^{(n)}$ (the smallest normal subgroup such that $G/G^{(\infty)}$ is solvable), the lower central series. It follows that in polynomial Monte Carlo time one can decide solvability and nilpotence of a black-box group, as well as decide whether or not G is a p-group for a given prime p.

It follows that the following subgroups are *recognizable* in Monte Carlo polynomial time:

$O_p(G)$ – the largest normal p-subgroup of G,
$F(G)$ – the largest nilpotent normal subgroup (Fitting subgroup)
$\mathrm{Sol}(G)$ – the solvable radical of G (largest solvable normal subgroup),
$Z(G)$ – the center.

Indeed, to test membership in $\mathrm{Sol}(G)$, for instance, we test, given $g \in G$, whether or not the normal closure $\langle g^G \rangle$ is solvable. – We can also combine these operators; e. g., the inverse image of $Z(G/O_p(G))$ is also recognizable in Monte Carlo polynomial time.

We note an immediate consequence of particular importance:

Corollary 3.3 *If G is a black-box group then a black box for $G/\mathrm{Sol}(G)$ can be simulated in Monte Carlo polynomial time.*

Indeed, this follows by Proposition 3.1 in light of the recognizability of $\mathrm{Sol}(G)$. ∎

We note that this corollary works in spite of the fact that we are not able to find any elements of $\mathrm{Sol}(G)$ or even to tell whether or not $|\mathrm{Sol}(G)| = 1$.

Another important consequence concerns the kernel of a permutation representation. The following is folklore.

Proposition 3.4 *Let G be a black-box group and $\varphi : G \to \Sigma_k$ be a permutation representation. Suppose φ is given by the images of the generators of G. Then $\ker(\varphi)$ can be constructed in Monte Carlo polynomial time.*

Proof. Let $N = \ker(\varphi)$ and $H = G/N \le \Sigma_k$. Let S be the given set of generators of G and let $\bar{S}$ be the image of S under φ. Starting from $\bar{S}$, standard permutation group techniques produce a presentation of H in terms of a new set $S' \supseteq \bar{S}$ of generators and relations in terms of S' ([Si78, FuHL, Kn, Je]). Lift the same sequence of group operation to G, starting from S rather than from $\bar{S}$. Create a set R of elements of G by starting from the empty set and adding each $g \in G$ computed in the above process satisfying $\varphi(g) = 1$. (This procedure is deterministic.) It is easy to see that $N = \langle R^G \rangle$. ∎

3.5 Abelian obstacles

Abelian groups and abelian quotients represent obstacles to answering even the simplest rudimentary questions about black-box groups.

Assume the black-box group G is known to be elementary abelian of order p^k where p is known but k is unknown. Then it takes an exponential number of black-box operations to determine k, or, for large primes, even to decide whether or not $k = 1$ [BaSz]. Therefore linear algebra cannot be addressed strictly within the context of black-box groups.

For matrix groups of characteristic p, linear algebra of elementary abelian p-subgroups is straightforward. However, this is not the case for elementary abelian r-subgroups, where $r \neq p$.

The *constructive membership problem* asks not only whether some $g \in G$ belongs to a given subgroup $H \leq G$ or not, but if so, it asks to construct g from the generators of H. If we want to use Gaussian elimination for linear algebra in elementary abelian r-groups, we need constructive membership. However, constructive membership in a cyclic subgroup of order r in $\mathbb{F}_q^\times$ $(r|q-1)$ is precisely the *discrete log* problem which is not known (and not believed) to be solvable in (Monte Carlo) polynomial time. (For the best current algorithms, see [Lo], [AdD], [Ad].) Therefore, the order of an abelian matrix group over $\mathbb{F}_q$, consisting of diagonal matrices only, cannot be determined by known methods in polynomial time, even if the group is elementary abelian of known exponent.

Even determining the order of an element may not be feasible in polynomial time because of its close relationship to the problem of factoring integers, another problem of computational number theory not believed to be solvable in polynomial time. (We analyse the complexity of determining the order of elements in Sections 3.6 and 8.1). However, for most purposes, one can avoid using the exact order of an element, and be content with its "pseudo-order," based on a set of "pretend-primes" (Section 8.4).

Other "abelian obstacles" concern the discovery of certain abelian (or solvable) subgroups and quotients and may be unrelated to computational number theory. The two most important questions in this direction are stated in the "Open problems" section; see especially Problems 10.2 and 10.3.

The moral of the present paper is that, in a sense, the *only obstacles* to Monte Carlo polynomial-time computation in black-box groups are the "abelian obstacles."

3.6 Order, prime factors

The "primes of a group G" are the prime divisors of $|G|$. For our results about black-box groups without any further restriction, we shall assume that together with the black-box group G, a superset $\mathcal{P}$ of the primes of G is given. The combined bit-lengths of the primes in $\mathcal{P}$ is then part of the input length with respect to which our algorithms run in polynomial time.

The following is well known.

Claim 3.5 *Given a black-box group G with a superset $\mathcal{P}$ of its primes, we can determine the order of any element of $g \in G$ in polynomial time.*

Proof. First we construct an integer $m = \prod_{p\in\mathcal{P}} p^{k_p}$ such that $|g|$ divides m. For instance, $|G|$ divides m if we we choose $k_p = \lfloor n/\log p \rfloor$. (Smaller exponents will suffice if more information about G is available, cf. [CeL].) Given m, the following algorithm determines the order of g.

Algorithm ORDER($G, \mathcal{P}, g, m$)

```
Initialize: $\ell := m$            (Note: $g^\ell = 1$)
for $r \in \mathcal{P}$
    find largest $t$ such that $r^t|\ell$ and $g^{\ell/r^t} = 1$
    $\ell := \ell/r^t$
end
return $\ell$
```

Using repeated squaring for calculating powers, the **for** loop takes $O(\log m)$ multiplications, adding up to a total of $O(|\mathcal{P}| \cdot \log m) \le O(|\mathcal{P}|^2 \cdot n)$. This justifies the polynomial time claim. A divide-and-conquer trick described in [CeL] reduces the number of operations to $O(\log(1 + |\mathcal{P}|) \cdot \log m)$.

4 Two results on simple groups

In this section we state two group theoretic results which will be crucial for the analysis of our algorithms.

4.1 Smallest degree of representations in the wrong characteristic

Landazuri and Seitz [LaS] proved that if a simple group of T Lie type of characteristic $r \neq p$ is represented in characteristic p then the dimension of the representation must be very large: it is polynomially related to the *size* of the natural module on which T acts (projectively). A result of Feit and Tits [FeT] implies that the same conclusion holds for representations of extensions of T (i.e., groups with T as a homomorphic image). We state a consequence of their combined result.

Theorem 4.1 ([LaS, FeT]) *Let T be a finite simple group of Lie type of characteristic r. Let V be the natural module (in characteristic r) on which*

T acts projectively. Let H be a finite group which involves T (as a quotient of a subgroup). Suppose H has a faithful projective representation of dimension d in characteristic $p \neq r$. Then $|V| \leq d^{c_1}$ for some absolute constant c_1.

4.2 Some statistical group theory

Let r be a prime number. A group element is called *r-regular* if its order is not divisible by r. We shall make extensive use of the following statistical result on r-regular elements.

Theorem 4.2 ([BaPS]) *Let G be a finite simple group and r a prime number. Then at least a c/d fraction of the elements of G is r-regular where $c > 0$ is an absolute constant and d is the dimension of G, defined as follows. The dimension of the alternating group A_k is k, the dimension of a classical simple group is the dimension of the projective space on which the group acts; all other simple groups have dimension 1.*

We shall require a more detailed version of this result, stated as Theorem 8.7.

5 Splitting a semisimple group

We call a group G *semisimple* if G is the direct product of nonabelian simple groups.

If $G_1, \ldots, G_k$ is a family of black-box groups of total encoding length n then the direct product $G = G_1 \times \cdots \times G_k$ can be represented in the natural way as a black-box group of encoding length n. The cost of a group operation in G will be the sum of the costs of group operations in each G_i.

A *black-box decomposition* of a black-box group G into a direct product $G = G_1 \times \cdots \times G_k$ means the construction of black-box representations of each G_i.

Theorem 5.1 *Let G be a black-box group given together with a set $\mathcal{P} \supseteq \pi(G)$. Assume that G is known to be semisimple. Then a black-box decomposition of G into its simple factors can be constructed in Monte Carlo polynomial time. Each simple factor will be represented as a black-box group.*

Note that such a decomposition immediately implies that a composition series of G can be constructed in polynomial time: namely, if $G = T_1 \times \cdots \times T_k$ is a black-box decomposition then the the family of prefixes $H_i = T_1 \times \cdots \times T_i$ is also constructed in polynomial time.

We state an important corollary.

Corollary 5.2 *Let G be a black-box group given together with a set $\mathcal{P} \supseteq \pi(G)$. Let $N = T_1 \times \cdots \times T_k$ be a given semisimple normal subgroup of G and let $\varphi : G \to \Sigma_k$ be the permutation representation of G defined by conjugation action on the set $\{T_1, \ldots, T_k\}$. Then φ can be constructed in the following strong sense: in Monte Carlo polynomial time we can set up a data structure which allows, for any $g \in G$, to compute $\varphi(g)$ in (deterministic) polynomial time.*

Proof of Corollary 5.2. By Theorem 5.1 we find generators for each T_i in Monte Carlo polynomial time. Let now $g \in G$ and $\sigma := \varphi(g) \in \Sigma_k$. Then $\sigma(i) = j$ exactly if T_i^g does not centralize T_j; this circumstance is verified by comparing the generators of each. ∎

The rest of this section is devoted to the proof of Theorem 5.1. Versions of this process and its analysis lead to our key algorithm: constructing the semisimple socle (Section 7).

Proof of Theorem 5.1. Let m be a known multiple of $|G|$, constructed as in the proof of Claim 3.5; all primes dividing m are from $\mathcal{P}$.

Let $m = \prod_{r \in \mathcal{P}} r^{k_r}$ and let $m_r = m/r^{k_r}$, i. e., m_r is the maximal r-free divisor of m.

Assuming G is semisimple, first we describe how to construct one of its simple factors.

Algorithm Construct_Simple_Factor(G, ν)

```
Initialize: $R := G$
repeat $\nu$ times
    choose random $g \in R$
    for $r \in \mathcal{P}$
        $h := g^{m_r}$
        if $h \neq 1$ then $R := \langle h^G \rangle$
    end
end
return $R$
```

Claim 5.3 *If $G \neq 1$ is semisimple then Algorithm* Construct_Simple_Factor(G) *returns a normal subgroup R of G which is simple with probability $\geq 1 - \epsilon$ assuming $\nu \geq c_2\sqrt{n}\log(1/\epsilon)$ where c_2 is an explicit constant.*

Note that these are proven worst-case guarantees; much fewer rounds may suffice in practice, and the theoretical estimates can be improved if information about the T_i is available.

Proof. Let $G = T_1 \times \cdots \times T_k$, T_i simple, nonabelian. For $I \subseteq \{1, \dots, k\}$, set $T_I = \prod_{i \in I} T_i$. We use the fact that the groups T_I are the only normal subgroups of G.

We refer to each iteration of the "repeat" loop as a "round." Let R_j denote the group R at the end of round j; $R_0 = G$. Observe that for all j we have $1 \neq R_j \leq R_{j-1}$ and $R_j \triangleleft G$.

We say that *component i is successful in round j* if either $R_j = T_i$ (which means we are done) or $R_j \cap T_i = 1$. Clearly, the output R_ν is simple exactly if each component is successful in some round.

Let δ_i denote the lower bound provided by Theorem 4.2 for the probability that for a prime r, a random element of T_i is r-regular. By the estimate given in Theorem 4.2 we have $\delta_i^{-2} \leq c_3 \log |T_i|$ and therefore

$$\sum_{i=1}^{k} \delta_i^{-2} \leq c_3 \log |G| \leq c_3 n \tag{6}$$

for an explicit constant c_3.

We claim, that, given any history of outcomes of earlier rounds, the probability that component i is not successful in round j is at most $1 - (59/60)\delta_i$.

Indeed, in order for this event to occur it is necessary that $R_{j-1} \geq T_i \times T_\ell$ for some $\ell \neq i$. Let us consider the random element $g = (g_1, g_2, \dots)$ selected in round j, where the $g_u \in T_u$ are the components of g with respect to our direct decomposition. We may model the random choice of g by first selecting g_ℓ at random, then g_i, and then the rest. With probability $\geq 59/60$, $g_\ell \neq 1$. In this case, let r be a prime dividing the order of g_ℓ. Then with probability $\geq \delta_i$ the order of g_i is not divisible by r. If this is the case, then $h = (h_1, h_2, \dots)$ satisfies $h_\ell \neq 1$ and $h_i = 1$. But this means $T_i \cap R_j = 1$ and component was successful in this round.

The probability that component i remains unsuccessful after ν rounds is therefore

$$\leq \left(1 - \frac{59\delta_i}{60}\right)^{\nu} < \exp\left(-\frac{59\delta_i \nu}{60}\right) = \exp\left(-\delta_i \sqrt{2c_3 n} \cdot 2\mu\right) < \left(\frac{1}{2\delta_i^2 c_3 n}\right)^{\mu} \tag{7}$$

where $\mu = 59\nu/(120\sqrt{2c_3 n}) > \nu/3\sqrt{c_3 n}$. Finally the probability that R_ν is not simple is less than

$$\sum_{i=1}^{k} \left(\frac{1}{2\delta_i^2 c_3 n}\right)^{\mu} < 2^{-\mu} < 2^{-\nu/3\sqrt{c_3 n}} \leq \epsilon \tag{8}$$

by inequality (6), assuming $c_2 = 3\sqrt{c_3}$. ∎

Suppose now that R is a given simple factor of the semisimple group G. Our next task is to construct the complement H of R in G defined by $G = R \times H$.

Algorithm CONSTRUCT_COMPLEMENT(G, R, ν)

```
Initialize: $H := 1$
repeat $\nu$ times
    choose random $g \in G$
    for $r \in \mathcal{P}$
        $h := g^{m_r}$
        if $[h, R] = 1$ then $H := H \cdot \langle h^G \rangle$
    end
end
return $H$
```

Claim 5.4 *If G is semisimple and $R \lhd G$ is simple then Algorithm* CONSTRUCT_COMPLEMENT(G, R) *returns a normal subgroup H of G such that $G = R \times H$ with probability $\geq 1 - \epsilon$ assuming $\nu \geq c_2\sqrt{n}\log(1/\epsilon)$ where c_2 is the same explicit constant as in Claim 5.3.*

Proof. As before, let $G = T_1 \times \cdots \times T_k$. We may assume $R = T_1$. Let $T = T_2 \times \cdots \times T_k$. Let H_j denote the group H at the end of round j; $H_0 = 1$. Observe that for all j we have $H_{j-1} \leq H_j \leq T$ and $H_j \lhd G$.

We say that *component $i \neq 1$ is successful in round j* if $H_j \geq T_i$. Clearly, the output H_ν is T exactly if each component $i = 2, \ldots, k$ is successful in some round.

We use the notation δ_i introduced in the proof of Claim 5.3. As before, we claim, that, given any history of outcomes of earlier rounds, the probability that component $i \neq 1$ is not successful in round j is at most $1 - (59/60)\delta_i$.

Indeed, as before, let us consider the random element $g = (g_1, g_2, \ldots)$ selected in round j ($g_i \in T_i$). We imagine this time that we first select g_i at random, then g_1, then the rest of the components. The rest of the proof proceeds exactly as the proof of Claim 5.3. ∎

Finally, to obtain the desired decomposition of the semisimple group G we separate a simple factor T_1, compute its complement H_1, and repeat the process with H_1 in the role of G. This completes the proof of Theorem 5.1. ∎

6 Finding a proper normal subgroup

Our first main goal is to construct a simple subnormal subgroup in $G/\operatorname{Sol}(G)$. This will be accomplished by a gradual descent along a subnormal chain. The steps of the process are given by the Monte Carlo algorithm $\textsc{Perm}(G, m)$ [BeB] which takes as inputs a black box group G and a positive integer m and produces one of the following outputs using $(m+n)^c$ group operations:

(a) a faithful permutation representation of G;
(b) a nontrivial normal subgroup $1 \neq H \triangleleft G$.

In case (b), H is not necessarily proper. However, the algorithm comes with the following guarantee:

Theorem 6.1 *If G has a proper subgroup of index $\leq m$ and case (b) is produced by Algorithm* $\textsc{Perm}(G, m)$ *then $H \neq G$ holds with large probability. If case (a) is produced then the degree of the permutation representation obtained is always $\leq m^c$.*

Note that the condition in the Theorem is equivalent to requiring that G has a nontrivial permutation representation of degree $\leq m$. An important corollary states that if a simple group G of Lie type is represented in the wrong characteristic then a faithful permutation representation of G can be found in Monte Carlo polynomial time (see Theorem 8.6).

The name of the algorithm owes to its attempts at finding a (not necessarily faithful) permutation representation of G. Although in general it will not find such a representation, it is likely that it will find a nonidentity element of the kernel of a non-faithful permutation representation of small degree if such a representation exists.

In this section we describe the algorithm Perm and outline the proof of Theorem 6.1.

6.1 The normal still

Since normal closures can be calculated in Monte Carlo polynomial time, a case (b) output will be produced once we found *any* nontrivial element of a proper normal subgroup.

In fact it suffices to produce an element which has a non-negligible chance ($\geq 1/m^c$) of belonging to a proper normal subgroup. Then after $O(m^c)$ trials, we expect to have encountered an element that actually does belong to a proper normal subgroup. We cannot decide which one, since we do not have membership testing, so we have no direct way of deciding whether a subgroup is proper.

We overcome this difficulty by "distilling" any list of candidate elements into a single nonidentity element guaranteed to belong to a proper a proper normal subgroup if any member of the list does.

It suffices to show how to do this with a list of length 2. The following is from [BeB]:

Lemma 6.2 *Given nonidentity elements a, b of a nonabelian black-box group G, we can in Monte Carlo polynomial time calculate a nonidentity element c of G such that if either a or b lies in a proper normal subgroup of G, then so does c.*

Proof. If $[a, b] \neq 1$ then we may take $c = [a, b]$, so assume a and b commute. Let T be a generating set for $\langle b^G \rangle$. If $[a, T] \neq 1$ then we may take $c = [a, t]$ for some $t \in T$, so assume a centralizes the normal closure of b. If a lies in the center of G we may take $c = a$. Otherwise, since a centralizes $\langle b^G \rangle$ but not all of G, we know $\langle b^G \rangle \neq G$ so we may take $c = b$. ■

We remark that for matrix groups, Lemma 6.2 can be strengthened: the only randomized part of the algorithm is testing if a centralizes $\langle b^G \rangle$. If G is a matrix group, then we can compute in deterministic polynomial time a basis consisting of elements of $\langle b^G \rangle$ for the matrix algebra generated by b^G. Then a centralizes $\langle b^G \rangle$ iff it commutes with every element of this basis.

6.2 Conjugacy classes large and small

We need the following useful statistical property of simple groups: certain powers of elements of some large conjugacy classes belong to small conjugacy classes.

Lemma 6.3 *Let T be a nonabelian simple group admitting a faithful permutation representation of degree $\leq m$. Let x be a random element of T, and let r be a randomly selected prime divisor of $|x|$. Let $y = x^{|x|/r}$. Then with probability $\geq 1/m^c$, y will lie in a conjugacy class of $\mathrm{Aut}(T)$ of cardinality $\leq m^{c_4}$, where c_4 is an absolute constant.*

Proof. If T is not an alternating or classical group, then Theorem 4.1 implies that $|T| \leq m^{c_4}$, so any conjugacy class is small enough.

For alternating groups, let us consider the case $T = A_k$ where k is even and not divisible by 3. In this case the cycle structure of x is $\{3, k-3\}$ with probability $1/(3k-9)$. If x is such an element, then with probability at least $1/k$, the prime 3 will be chosen for r, and y will be a 3-cycle (with conjugacy class size $O(k^3)$). Now $k \leq m$, justifying the claim. In the other cases, look for elements x with cycle structure $\{3, k-s\}$ for an appropriate value $3 \leq s \leq 6$.

For classical groups acting projectively on $\mathbb{F}_q^k$, Theorem 4.1 implies that $q^k \leq m^{c_1}$ where c_1 is the constant appearing in Theorem 4.1. So it suffices to show that with probability $\geq q^{-ck}$, the element y acts trivially on a subspace of bounded codimension.

We may assume $k \geq c$. (Each occurrence of c represents a different constant.) Now G contains a subgroup $H \cong G_1 \times G_2$ where G_1 is the same kind of classical group (or its central extension) acting on a space of bounded codimension, and G_2 is nontrivial; $|G : H| < q^{ck}$. Now Theorem 4.2 implies that if $x \in H$ then $y \in G_2$ with probability $\geq c/(kt)$ where t is the number of prime divisors of $|x|$. ∎

6.3 Blind descent: the Perm algorithm

We now give the algorithm for finding an element of a proper normal subgroup. This algorithm represents one step in the "blind descent" with the goal to reach a simple subnormal subgroup (Section 7.1).

Algorithm Perm1(G, m)

```
choose random $x, z \in G$
choose random prime $r$ dividing $|x|$
$y := x^{|x|/r}$,    $w := [y, z]$
**if** $w \neq 1$ **then return** $w$
**else** partially enumerate $y^G$
    **if** $|y^G| \leq m^{c_4}$ **then**
        **return** permutation representation of $G$ on $y^G$
    **else** report failure
```

Theorem 6.4 *Assume $G = G'$ and G has a nontrivial permutation representation of degree $\leq m$. Then with probability $\geq 1/m^{c_5}$, Algorithm* Perm1(G, m) *either returns an element $w \neq 1$ which belongs to a proper normal subgroup of G, or returns a nontrivial permutation representation of G of degree $\leq m^{c_4}$.*

Proof. It follows from the assumptions that G has a nonabelian simple quotient $T \leq \Sigma_m$. Let $\varphi : G \to T$ be an epimorphism. Note that φ, and indeed the isomorphism type of T, are not known to the algorithm.

By Lemma 6.3, we have a reasonable hope ($\geq m^{-c}$) that the conjugacy class of $\varphi(y)^T$ has at most $m^{c_4}/2$ elements. Now the probability that $w \in \ker(\varphi)$ is $1/|T : C_T(\varphi(y))| = 1/|\varphi(y)^T| \geq 2m^{-c_4}$.

Moreover, if $|y^G| > m^{c_4}$ then the probability that $w = 1$ is $1/|G : C_G(y)| = 1/|y^G| < m^{c_4}$. So overall the probability that the algorithm succeeds in the sense stated in the Theorem is $> m^{-c-c_4}$. ∎

Algorithm PERM(G, m)

```
flip coin
if heads then return $G'$
else repeat  $m^c \log(1/\epsilon)$ times
        $W := \emptyset$
        PERM1$(G, m)$
        if nontrivial permutation representation $\varphi$ found then
            if $\varphi$ faithful then return $\varphi$, exit
            else let $u \in \ker(\varphi)$, $u \neq 1$
            return $\langle u^G \rangle$, exit
        else if $w \neq 1$ then add $w$ to $W$
    end
    if $W \neq \emptyset$ then
        $u :=$ element distilled from $W$ (Section distill.sec)
        return  $\langle u^G \rangle$
    else return $G$
```

The proof of Theorem 6.1 is now immediate. ∎

The algorithm described is a very simple one and serves to justify the polynomial-time claim. There is a lot of room for improvement, both theoretical and heuristic. We comment on some of these in Section 6.4.

6.4 Speedups

It is possible to extend the PERM algorithm to work in some cases where no permutation representation of small degree exists. Suppose that, for some simple group T, we can test in t steps (group operations) if a given black-box group G is isomorphic to T. Then, essentially in t steps, we can construct a nontrivial normal subgroup of G if G is not isomorphic to T but has T as a homomorphic image. This was observed in [Be97]. Note that there are many black-box recognition algorithms which work for entire classes of groups; we do not have to treat each possible homomorphic image T separately.

Theorem 6.5 ([Be97]) *Suppose that the black-box group G has a nontrivial normal subgroup N with G/N isomorphic to the group T. Then from the transcript of a black-box algorithm which decides that G is not itself isomorphic to T, we can in Monte Carlo polynomial time compute an element of a nontrivial proper normal subgroup of G.*

By the *transcript* of the algorithm we simply mean the sequence of queries to the black-box, together with the responses, from a run of the algorithm.

This means that any improvements to the small/large conjugacy class algorithm for black-box simple groups automatically yield corresponding improvements to the algorithm for the problem of obtaining proper normal subgroups. For alternating groups, the small/large conjugacy class algorithm takes $O(k^3)$ steps, where k is the degree of the group, to achieve success probability $1/k$. This algorithm thus takes $O(k^4)$ steps to have a high $(1-\epsilon)$ success probability. However, a recent algorithm by Beals, Leedham-Green, Niemeyer, Praeger, and Seress [BeLNPS], constructs a permutation representation of black box alternating groups in $O(k \log k)$ steps. This is an improvement by a factor of nearly k^3, hence the algorithm for finding a nontrivial normal subgroups of groups with an alternating homomorphic image is sped up by the same factor.

In the terminology of Section 9.2, the above result means that black-box alternating groups A_k admit weak as well as strong *constructive recognition* in $O(k \log k)$ steps.

Weak constructive recognition of T is clearly sufficient for the condition of Theorem 6.5. Therefore the striking results to be discussed in Section 9.2 become relevant: if T is a classical group of dimension d over a field of order q then T, as a black-box group, admits (weak and strong) constructive recognition in time $(dq)^c$ [KaS]. (Compare this with the time q^{cd} required by the large/small conjugacy class method.)

7 Constructing the semisimple socle

In this section, we assume $\mathrm{Sol}(G) = 1$.

7.1 Blind descent: constructing a semisimple normal subgroup

The term "blind descent" [BeB] refers to the fact that we shall keep descending along a subnormal chain without ever being able to verify progress. Algorithm Perm represents one step of the blind descent.

Note that the permutation representation obtained if item (a) is produced has degree $\leq m^c$ for some constant c.

Recall that by $G^{(\infty)}$ we denote the stable derivative of G (cf. 3.4). The following algorithm performs the "blind descent."

Algorithm CONSTRUCT_SEMISIMPLE(G)

```
Initialize: $H := G$
repeat $n$ times
    $H := H^{(\infty)}$
    PERM($H, n/5$)
    if nontrivial normal subgroup found then
        $H :=$ such a normal subgroup
    else (faithful permutation representation of $H$ found)
        construct minimal subnormal subgroup of $H$, exit
end
return $H$
```

In the "**else**" branch we use Luks's composition chain algorithm for permutation groups [Lu87] (as improved by Beals–Seress [BeS]). We then exit the "**repeat**" loop and return the minimal subnormal subgroup found.

Claim 7.1 *Assume G is a black-box group satisfying* $\mathrm{Sol}(G) = 1$ *and $G \neq 1$. Then with large probability, the output of Algorithm* CONSTRUCT_SEMISIMPLE(G) *is a nontrivial semisimple subnormal subgroup of G.*

Proof. Assume for a that the PERM subroutine does not make any errors throughout the execution of the algorithm.

It is then clear that throughout the process, H is subnormal in G. Moreover $H \neq 1$ because G has no solvable subnormal subgroups. Let now H be the output. We need to prove that H is semisimple.

If the "**else**" branch was invoked at any time then H is in fact simple. So we may assume this did not occur.

Since no subgroup chain of G is longer than $\log |G| \leq n$, H is stable under the loop of the algorithm. Therefore H is perfect and H has no nontrivial permutation representation of degree $\leq n$.

Let $N = T_1 \times \ldots \times T_k$ be the socle of G and let $\varphi : G \to \Sigma_k$ be the permutation representation defined by the conjugation action of G on the set $\{T_1, \ldots, T_k\}$. Observe that $H \leq \ker(\varphi)$ since otherwise H had a nontrivial permutation representation of degree $\leq k < n/5$. So $H \leq M := \mathrm{Aut}(T_1) \times$

$\cdots \times \mathrm{Aut}(T_k)$. But $M^{(\infty)} = N$; therefore $H \leq N$. Any subnormal subgroup of N is normal in N and is of the form $\prod_{i \in I} T_i$ for some $I \subseteq \{1, \ldots, k\}$; therefore H is semisimple.

Our algorithm may err only if Perm errs; therefore the probability of error is at most $n\epsilon$ where ϵ is the error probability of the Perm runs. We have to choose $\epsilon < 1/n^2$, say, which is accomplished at a cost of $O(\log n)$-fold repetition. ∎

Now in order to construct a semisimple *normal* subgroup of G, take the normal closure of H.

7.2 Complementing a semisimple direct factor

Next we consider the situation that $G = N \times M$ where G is a black-box group and N is a semisimple normal subgroup of G. We continue to assume $\mathrm{Sol}(G) = 1$. Moreover, we assume that G and N are given (but M is not). We wish to find a nontrivial element of M or else decide that $M = 1$.

Using the algorithms of the preceding section, we may assume that we are given the decomposition $N = T_1 \times \cdots \times T_k$ where the T_i are simple.

Algorithm Discover_Complement(G, N, ν)

```
Initialize: $J := G$
for $i = 1$ to $k$ do
    if $[J, T_i] \neq 1$ then
        $J :=$ Construct_Complement$(J, T_i, \nu)$
        $J := \langle J^G \rangle$
end
return $J$
```

Claim 7.2 *If $G = N \times M$ where $N = T_1 \times \cdots \times T_k$, the T_i are simple, and* $\mathrm{Sol}(G) = 1$*, then* Algorithm Discover_Complement(G, N, ν) *returns a subgroup $J \triangleleft M$. If $M \neq 1$ then $J \neq 1$ with probability $\geq 1 - \epsilon$ assuming $\nu \geq c_2\sqrt{n}\log(1/\epsilon)$ where c_2 is the same explicit constant as in Claim 5.3.*

Proof. First we observe that if N is semisimple and M is an arbitrary group then any normal subgroup of $N \times M$ is of the form $K \times L$ where $K \triangleleft N$ and $L \triangleleft M$. Therefore J as well as the groups H arising in the inner (**repeat**) loop of Construct_Complement have the form $K \times L$ throughout the algorithm.

We claim that with high probability, $L \neq 1$ throughout the algorithm.

The reasoning is analogous to the proof of Claim 5.4. Consider round i of the **for**-loop, and within it, round j of the **repeat**-loop of the algorithm CONSTRUCT_COMPLEMENT(J, T_i, ν). Let $H_{i,j} = K_{i,j} \times L_{i,j}$ denote the group H of Algorithm CONSTRUCT_COMPLEMENT(J_i, T_i, ν) at the end of this round.

We say that *component i is successful in round* (i, j) if $T_i \cap K_{i,j} = 1$ and $L_{i,j} \neq 1$.

Let us assume that $L_{i,j-1} \neq 1$. This time we choose a random element $(x, y) \in K_{i,j-1} \times L_{i,j-1}$ by selecting $y \in L_{i,j-1}$ first. The probability that $y \neq 1$ is at least 59/60. Let, then, r be a prime dividing the order of y. Let $x = (g_1, \ldots, g_k)$. Now the probability that the order of g_i is not divisible by r is $\geq \delta_i$. If this is the case, component i is successful in this round. Therefore the probability that component i remains unsuccessful through ν rounds is bounded by the left-hand side of equation (7). Therefore the probability that at least one component remains unsuccessful during the entire algorithm is bounded by the left-hand side of equation (8). The result follows as before. ∎

7.3 Extending a semisimple normal subgroup

Suppose now that we have a black-box group G satisfying $\mathrm{Sol}(G) = 1$ and a semisimple normal subgroup N. We want either to increase N or decide that $N = \mathrm{Soc}(G)$ (and therefore no increase is possible).

Let $N = T_1 \times \cdots \times T_k$ where the T_i are simple. Let $\varphi : G \to \Sigma_k$ be the usual permutation representation on the set of simple factors. Let $K = \ker(\varphi)$ and let $L = K^{(\infty)}$. Note that K can be constructed in Monte Carlo polynomial time using the sifting technique for permutation groups [Si71, Si78, FuHL] and L is constructed in [BaCFLS].

Claim 7.3 *$\mathrm{Soc}(G) \leq L$ and $L = N \times M$ for some $M \triangleleft G$.*

Proof. Clearly, $\mathrm{Soc}(G) \leq K$. Moreover, the image of K/N in $\mathrm{Aut}(N)/N$ under the conjugation action is $\leq \mathrm{Out}(T_1) \times \cdots \times \mathrm{Out}(T_k)$; the right hand side being solvable, the image of L is trivial. Therefore $L \leq N \cdot C_G(N) = N \times C_G(N)$. Any normal subgroup $L \geq N$ of a group of the form $N \times T$, N semisimple, has the form $N \times M$. ∎

Next we apply Algorithm DISCOVER_COMPLEMENT(L, N, ν). Assuming $N \neq \mathrm{Soc}(G)$ and therefore $M \neq 1$, by Claim 7.3 we obtain a nontrivial element $u \in M$. Let $H = \langle u^G \rangle$. Construct a nontrivial semisimple G-normal subgroup T in H; replace N by $N \times T$. Repeat $\leq n$ times to obtain $\mathrm{Soc}(G)$. ∎

8 How to do without prime factorization

8.1 The complexity of determining the order of elements

Proposition 8.1 *Without any knowledge about our black-box group other than its encoding length n, the determination of the order of an element may require exponential Monte Carlo time.*

Proof. Let p be a random prime number between 1 and 2^n and assume $G = \langle g \rangle$ is cyclic of order p, (p is unknown to us). Suppose our polynomial-time Monte Carlo algorithm makes t steps, including $\leq t$ queries regarding whether an element so far constructed is the identity. All such questions will be of the form "$g^j = 1$?" where $|j| \leq 2^t$. For $j = 0$ the answer of course is "yes," but the probability that p divides any other j that arises is less than $\nu(j)/\pi(2^n)$ where $\nu(j) = O(\log |j| / \log\log |j|) = O(t/\log t)$ is the number of distinct prime divisors of j and $\pi(2^n) \sim 2^n/(n \ln 2)$ is the number of primes $\leq 2^n$. Therefore the probability that a query with $p|j \neq 0$ will ever be asked is $< c(t/\log t)/(2^n/n)$. For this probability to be non-negligible, t must be exponentially large. ∎

On the other hand, the assumption that $\mathcal{P}$ is known is unpleasant since for linear groups it assumes factoring integers, a problem generally believed not to be solvable in polynomial time [Ad]. Even if $G \leq GL(1, p)$, we need to factor $p-1$ (p prime), and this task alone is equivalent to factoring arbitrary integers, as seen from the following result.

Theorem 8.2 (E. Bach, J. Shallit) *Factoring any integer can be reduced, in Monte Carlo polynomial time, to factoring integers of the form $p - 1$ (p prime) into their prime factors, assuming the Extended Riemann Hypothesis (ERH).*

Proof. Suppose we wish to factor the n-digit integer N into prime factors. Linnik's celebrated theorem asserts that there exists a prime $p \equiv 1 \pmod N$ such that $p < N^c$. Bach and Shallit show that such a prime p can be found in Monte Carlo polynomial time, assuming ERH [BacS, p. 241, Ex. 30]. Now factoring $p - 1$ into its prime factors will split N. ∎

We used the primes to determine the order of elements. We are unable to prove that determining the order elements of $GL(d, q)$ accessible to a Monte Carlo polynomial-time algorithm does actually require factoring integers of the form $p^i - 1$. But if we broaden our scope and consider linear groups over the rings $\mathbb{Z}/N\mathbb{Z}$, then factoring integers in fact becomes equivalent to finding the order of elements in such a group. Indeed, even determining the order of 1×1 invertible matrices over $\mathbb{Z}/N\mathbb{Z}$ is as hard as factoring N:

Proposition 8.3 (G. L. Miller [Mi]) *Factoring the composite integer N is reducible in Monte Carlo polynomial time to determining the order mod N of random integers k, $1 \le k \le N$, g.c.d.$(k, N) = 1$.*

This fact is the starting point of Peter Shor's polynomial-time algorithm for factoring integers on *quantum-mechanical* Turing-machines. Shor gives a nice explanation of Miller's result and provides additional background and literature on factoring [Sh, p. 1498]. ∎

These observations suggest that it is not possible to determine the order of elements of matrix groups in polynomial time. Nor is it necessary, however, at least for the class of black-box groups of greatest interest to us: "black-box groups of a known characteristic p."

8.2 Black-box groups of characteristic p

We say that G is a black-box group *of characteristic p* if G is a black-box group of some encoding length n and G is a section (quotient of a subgroup) of $GL(d, p)$ where[4] $d = \lfloor n/\log p \rfloor$. When we say that such a group G is *given*, we tacitly assume that p is known.

Proposition 8.4 *For q a power of the prime p, the subgroups of $GL(f, q)$ as well as their quotients are black-box groups of characteristic p as long as their elements are encoded as matrices over $\mathbb{F}_q$ or over a subfield of $\mathbb{F}_q$.*

Proof. If $q = p^k$ then $GL(f, q) \le GL(fk, p)$ and therefore $|G|$ divides $|GL(fk, p)|$. Moreover, $n \ge f^2 \log q = f^2 k \log p \ge fk \log p$ and therefore $fk \le d$. ∎

Remark 8.5 Note that any black-box group G becomes a black-box group of characteristic p (for any p) if we pad the encoding with a sufficiently long string of dummy symbols. However, for the purposes of polynomial-time algorithms, "padding" is limited to increasing the encoding length polynomially. In particular, the fact that G is a Lie-type simple group of characteristic p does not exclude the possibility of G being represented as a black-box group of characteristic $r \ne p$. However, in this case we have powerful tools to deals with G, as the following result shows.

Theorem 8.6 *Let G be a black-box group of characteristic r and suppose G is isomorphic to a simple group of Lie type of characteristic $p \ne r$. Then we can find a faithful permutation representation of G in polynomial time.*

[4]Throughout this paper, "log" refers to base 2 logarithms.

This result then implies that we can learn virtually everything about G in polynomial time: by a theorem of Kantor [Ka], we can find the standard name of G and its representation in its projective action on the natural module.

Proof. By the Landazuri–Seitz theorem (Theorem 4.1), G has a permutation representation of polynomially bounded degree, and therefore by Theorem 6.1 a faithful permutation representation of G can be computed in Monte Carlo polynomial time. ∎

8.3 More statistical group theory

We require a more complete version of Theorem 4.2, our main statistical tool about simple groups. For a set $S \subseteq G$ we write $S^G = \bigcup_{g\in G} S^g$ (the union of conjugates).

Theorem 8.7 ([BaPS]) *Let G be a finite simple group of Lie type. Then G has two cyclic maximal tori T_1, T_2 of relatively prime orders such that for each i, the set T_i^G has non-negligible density: $|T_i^G|/|G| \geq c/d$, where d is the dimension of the linear space on which G acts projectively and $c > 0$ is a constant.*

(For classical groups, the result holds with $c = 1/2$.) This result clearly implies Theorem 4.2 for simple groups of Lie type: if r divides $|T_1|$ then all elements of T_2^G are r-regular; otherwise all elements of T_1^G are r-regular.

The orders of the maximal tori in question are listed in [BaPS]. For a classical group $X_d(q)$ ($X \in \{PSL, PSp, PSU, P\Omega^{\pm}\}$; for the unitary groups, the field has order q^2) the orders of these tori are of the form $(k_1k_2)/(k_3k_4)$, where $k_1 = q^j \pm 1$ for $j \leq d$, $k_2 = 1$ or $k_2 = q \pm 1$, $k_3 = 1$ or $k_3 = q \pm 1$, and $k_4|d$. For exceptional groups $Y(q)$ we have numbers of similar form with $j \leq 7$, $k_3 \leq 3$, and $k_4 = 1$; moreover, $\Phi_j(q^s)$ for $s \leq 3$ and $j = 6, 3$ (where $\Phi_j(x)$ denotes the j-th cyclotomic polynomial), furthermore, $\Phi_j(q)$ for $j = 30, 15$ (these arise for $E_8(q)$), and finally the following integer factors of $\Phi_j(q^s)$ which we shall refer to as *semicyclotomic* factors of $q^j - 1$: $q \pm \sqrt{2q} + 1$ (factors of $\Phi_4(q)$ for $q = 2^{2t+1}$), $q \pm \sqrt{3q} + 1$ (factors of $\Phi_6(q)$ for $q = 3^{2t+1}$), and $q^2 + q + 1 \pm \sqrt{2q}(q + 1)$ (factors of $\Phi_6(q^2)$ for $q = 2^{2t+1}$).

8.4 Pseudo-order of elements in black-box groups of characteristic p

Let $\mathcal{P}$ be a set of pairwise relatively prime integers. Assume that $m = \prod_{p\in\mathcal{P}} p^{k_p}$ is a multiple of $|G|$. (We may take $k_p = \lfloor n/\log p \rfloor$ where n is the encoding length of G.) Let us call the members of $\mathcal{P}$ *pretend-primes*. Let us define the *pseudo-order* of $g \in G$ with respect to the set $\mathcal{P}$ of pretend-primes the smallest positive integer ℓ such that $g^\ell = 1$ and ℓ is a product of pretend-primes. Let $|g|_\mathcal{P}$ denote this quantity. Note that $|g|_\mathcal{P}$ is computed in polynomial time by Algorithm ORDER$(G, \mathcal{P}, g)$ (Section 3.6).

Celler and Leedham-Green [CeL] suggest that factoring the order of $GL(d, p)$ into easily computed pretend-primes will suffice in lieu of actual prime factorization for most applications. We turn this idea into a rigorous statement regarding the algorithms of this paper. Our pretend-primes will have to go slightly beyond the small primes and the cyclotomic factors $\Phi_j(p)$ of the integers $p^i - 1 = \prod_{j|i} \Phi_j(p)$ recommended by [CeL].

Let $\mathcal{L}$ be a set of positive integers. We define the *relatively prime refinement* of $\mathcal{L}$ as the smallest set $\mathcal{P}$ of pairwise relatively prime integers such that each member of $\mathcal{L}$ is a product of members of $\mathcal{P}$; and the sum of the elements of $\mathcal{P}$ should be maximal subject to this condition. We shall see that this set is unique and denote it by $\mathcal{P}(\mathcal{L})$.

For a set $\mathcal{M}$ of positive integers and an integer m, let us write $\mathcal{M} \vdash m$ if m is either the g.c.d. or the quotient of two members of $\mathcal{M}$. We say that $\mathcal{M}$ is closed under $\vdash$ if $\mathcal{M} \vdash m$ implies $m \in \mathcal{M}$. Let $\bar{\mathcal{M}}$ denote the closure of $\mathcal{M}$ (the smallest closed set containing $\mathcal{M}$).

Claim 8.8 *Let $\mathcal{L}$ be a set of positive integers. Then $\mathcal{P}(\mathcal{L})$ is the set of minimal elements with respect to divisibility of $\bar{\mathcal{L}} \setminus \{1\}$.*

(An element m of a set $\mathcal{N}$ of positive integers is *minimal* with respect to divisibility if no other element of $\mathcal{N}$ divides m.) The Claim includes the statement that $\mathcal{P}(\mathcal{L})$ is unique. We leave the easy proof to the reader. ∎

The following algorithm is folklore; it constructs $\mathcal{P}(\mathcal{L})$.

Algorithm REFINE$(\mathcal{L})$

```
Initialize: $\mathcal{P} := \mathcal{L}$
while not all pairs in $\mathcal{P}$ are relatively prime do
    pick $a, b \in \mathcal{P}, a \neq b$ such that $f :=$g.c.d.$(a, b) \neq 1$
    delete $a, b$ from $\mathcal{P}$, add $f, a/f, b/f$ to $\mathcal{P}$
end
return $\mathcal{P}$
```

We note that each round reduces the product of the elements of $\mathcal{P}$ by a factor of $d \geq 2$; therefore the process terminates in $\leq \sum_{m \in \mathcal{L}} \log m$ rounds. This is less than the length of the input (total bit-length of the integers

$m \in \mathcal{L}$), so the algorithm runs in polynomial time. We leave the easy proof of correctness to the reader. ∎

Remark 8.9 The "factor refinement problem" asks not only to produce the set $\mathcal{P}(\mathcal{L})$ but also to express the product $\prod_{m \in \mathcal{L}}$ as a product of elements of $\mathcal{P}(\mathcal{L})$. The Refine algorithm can easily be adapted for this purpose. The "factor refinement problem," and specifically the idea of the Refine algorithm, have a long history, going back to Stieltjes (1890). Bach, Driscoll, and Shallit [BacDS] give a detailed account of early as well as recent work and a multitude of applications. They also present a definitive result on the complexity of the problem which we state.

Let k be the total bit-length of the integers in $\mathcal{L}$. Then a naive implementation of the Refine procedure runs in $O(k^3)$ time (bit-operations). Note that the above description of Refine does not specify how to keep track of which pairs remain not relatively prime and in what order to process them. By appropriately organizing this process, [BacDS] reduces the running time to $O(k^2)$.

Now here comes our **recipe for creating the set of pretend-primes** for our algorithms.

Given that our groups are sections of $GL(d,p)$ (quotients of subgroups), first we create a list $\mathcal{L}(d,p)$ of positive integers as follows.

Include in $\mathcal{L}(d,p)$ all primes ≤ 47 and the primes 59, 67, 71 (these are the primes occurring in sporadic simple groups). Include all primes $\leq d^{c_1}$ where c_1 is the constant in Theorem 4.1. (This takes care of many things, see below.) Further, include all cyclotomic factors $\Phi_i(p^j)$ for $ij \leq d$.

Finally, include the following semicyclotomic factors: If $p = 2$, include $2^{2t+1} \pm 2^{t+1} + 1$, $0 \leq t \leq (d-4)/8$, and $2^{4t+2} + 2^{2t+1} + 1 \pm 2^{t+1}(2^{2t+1}+1)$, $0 \leq t \leq (d-12)/24$. If $p = 3$, include $3^{2t+1} \pm 3^{t+1} + 1$, $0 \leq t \leq (d-6)/12$. This completes the list $\mathcal{L}(d,p)$.

Let now $\mathcal{P}(d,p)$ denote the relatively prime refinement of $\mathcal{L}(d,p)$. This will be our set of pretend-primes.

Corollary 8.10 *Let S be a nonabelian simple section of $GL(d,p)$. Then for any prime r, at least a c/d fraction of the elements of $g \in S$ has pseudo-order $|g|_{\mathcal{P}}$ relatively prime to r with respect to the set $\mathcal{P} = \mathcal{P}(d,p)$ of pretend-primes.*

Proof. If S is sporadic then $\pi(S) \subseteq \mathcal{P}$. The same holds if S is alternating of degree k since $k \leq d+1$, and also if S is of Lie type of characteristic

$s \neq p$ since in that case the natural module for S has order s^t for some t and $s^t \leq d^{c_1}$ by Theorem 4.1. If S is Lie type of characteristic p then let T_1 and T_2 be the maximal tori of S discussed in Theorem 8.7. But then, $|T_i|$ is a product of the cyclotomic and semicyclotomic factors included in $\mathcal{L}(d,p)$. (We note in this connection that $\Phi_i(x^j) = \prod_{h:h|j,gcd(h,t)=1} \Phi_{tj/h}(x)$.) Therefore if the prime r does not divide $|T_i|$ (which is true for at least one of T_1 and T_2) then the pseudo-order of elements of T_i^G with respect to $\mathcal{P}(d,p)$ is not divisible by r either (since the pseudo-order will divide $|T_i|$). ∎

It follows that our algorithms work correctly using the set $\mathcal{P}(d,p)$ of pseudoprimes. ∎

The following observation speeds up the refinement algorithm for the case $\mathcal{L} = \mathcal{L}(d,p)$.

Proposition 8.11 *Let r be a prime, q an arbitrary integer, $i > j$ positive integers. If $r|\Phi_i(q)$ and $r|\Phi_j(q)$ then $r|i$.*

Proof. The conditions imply that r divides g.c.d.$(q^i-1, q^j-1) = q^k - 1$, where $k =$g.c.d.(i,j). Now $q^i - 1$ is divisible by $\Phi_i(q)(q^k-1)$, therefore q is a multiple root of the polynomial $x^i - 1$ over $\mathbb{F}_r$. This implies that q is a root of the derivative, i. e., $r|iq^{i-1}$, and therefore $r|i$. ∎

As a consequence, the following version of the Refine procedure will succeed. For a prime r, the *r-free part* of an integer $a \neq 1$ is the integer $a' = a/r^k$ where r^k is the largest power of r which divides a.

Algorithm Simple-Refine(d,p)

```
Initialize: $\mathcal{P} := \mathcal{L}(d,p)$
for all primes $r \leq d$ do
    for all $a \in \mathcal{P}$ do
        replace $a$ by its $r$-free part
    end
end
return $\mathcal{P}$
```

Claim 8.12 *Algorithm* Simple-Refine(d,p) *returns the set $\mathcal{P}(d,p)$.*

Indeed, this is immediate from Proposition 8.11 and the fact that all primes $\leq d$ are included in $\mathcal{L}(d,p)$, with the additional observation that the pairs of semicyclotomic factors included in $\mathcal{L}(d,p)$ are trivially relatively prime. ∎

Remark 8.13 Given the set $\mathcal{L}(d,p)$, Algorithm SIMPLE-REFINE(d,p) runs in $O(d^3 \log p)$ bit operations. Note that the total number of bits of the input $\mathcal{L}(d,p)$ is $\Theta(d^2 \log p)$.

Remark 8.14 P. P. Pálfy kindly communicated the following result to us. *Under the conditions of Proposition 8.11, the g.c.d. of $\Phi_i(q)$ and $\Phi_j(q)$ is either 1 or a prime number r; in the latter case, $i = sr^k$ and $j = sr^\ell$ for some nonnegative integers s, k, ℓ where $s|r-1$.*

9 Statistical recognition of black box simple groups: a project

In this section we briefly review major progress on the following question.

Problem 9.1 *Given a black-box* simple *group G of characteristic p, what can we say about G in Monte Carlo polynomial time?*

This is a *promise problem:* we expect a correct answer only if G is indeed simple, but the fact of simplicity need not be verified. (See the Problem 10.2.)

9.1 Name-recognition

The simplest type of answer is *name-recognition:* we wish to tell which simple group G is isomorphic to (print the standard name of G). It now seems reasonable to expect that the following conjecture holds:

Conjecture 9.2 *Let G be a simple group of Lie type of characteristic p given as a black-box group. Assume p is known. Then one can compute the standard name of G in Monte Carlo polynomial time.*

Combined with Theorem 1.2 this conjecture implies that we can *list the names of all nonabelian composition factors (with multiplicity) of a black-box group of given characteristic in Monte Carlo polynomial time.* (If G is given as a black-box group of the wrong characteristic, then we can deal with G in a very strong sense, see Theorem 8.6.)

Conjecture 9.2 is addressed in [BaP]. In 1955, Artin introduced four basic invariants of simple groups of Lie type to distinguish simple groups by their orders [Ar]. For an update to include the classes discovered after Artin's paper, see [KiLST]. Two of Artin's invariants, called α and β in [Ar], turn out to be computable in Monte Carlo polynomial time for black-box simple groups of a given characteristic. (The other two do not seem to, since their computation would depend on finding p-singular elements in a group of Lie type of characteristic p, an elusive quest.) However, it turns out that α and β already "almost determine" G in the sense that at most 7 groups of Lie type of characteristic p share the same pair (α, β). Results of Niemeyer and Praeger [NiP98] on ppd-elements in classical groups and their extension to exceptional groups [BaPS] imply further breakup of these classes.

While the determination of the Artin invariants depends only on sampling the orders of the elements of the given black-box group, such a simple-minded approach will not separate the two infinite classes of simple groups of equal orders: $PSp(2k, q)$ and $P\Omega(2k+1, q)$, q odd. However, a breakthrough by Altseimer and Borovik eliminated this obstacle: *these two classes are distinguishable in Monte Carlo polynomial time* [AlB]. This result gives reason for optimism regarding the completion of the proof of Conjecture 9.2.

9.2 Constructive recognition

A much more ambitious goal is *strong constructive recognition*: given a black-box simple group T, find an isomorphism to a "natural representation," with the isomorphism efficiently computable in both directions. (The "natural representation" of Lie-type simple groups is its projective action on the natural module. The natural representation of the alternating groups is self-explanatory. For the purposes of the polynomial-time theory we are not concerned about sporadic groups.)

Weak constructive recognition requires finding a presentation in terms of generators and relations.

We note that strong constructive recognition implies weak constructive recognition, assuming the *Short Presentation Conjecture* mentioned in Section 2.2. Therefore this implication is known to hold for all finite simple groups except for the rank-1 twisted types [BaGKLP].

Sims's "strong generators" yield weak constructive recognition of permutation groups in polynomial time. This observation, combined with Theorem 6.4, yields weak recognition of simple black-box groups in Monte Carlo time polynomial in $n + m$ where n is the input length and m is the smallest degree of a permutation representation. For classical groups, Kantor [Ka] turns the permutation representation into "strong constructive recognition" in polynomial time.

A breakthrough in strong constructive recognition of black-box simple groups came with the recent paper by Cooperman, Finkelstein, and Linton [CoFL95] who solved this problem in Monte Carlo polynomial time for $PSL(d,2)$. Following work by Prabhav Morje on nearly linear time algorithms for Sylow subgroups in permutation groups [Mo], the result of [CoFL95] was extended by Bratus, Cooperman, Finkelstein, and Linton [BrCFL] to $PSL(d,q)$ for *tiny* q (see below) and finally in a monumental paper by Kantor and Seress [KaS] to all classical groups over *tiny fields*.

Following accepted terminology in the theory of computing, we say that an input parameter q is *tiny* if it is input in *unary* (rather than in binary), i. e., it contributes q rather than $\log q$ to the length of the input. So a polynomial-time algorithm for a $d \times d$ matrix group over a *tiny field* runs in time $O((dq)^c$ rather than $O((d \log q)^c$, as required if the field is not tiny (q is the order of the field).

(We note that over tiny fields, strong recognition automatically implies weak recognition since the Steinberg presentations [St] are of polynomial length as a function of dq (but not of $d \log q$).)

We mention that weak constructive recognition is provably hard (exponential time) for *elementary abelian black-box groups* [BaSz].

10 Open problems

We conclude with a list of open problems. All these problems represent bottlenecks in the polynomial-time attempts to obtain a more complete description of the normal structure of a black-box group G. For a further explanation of the connections we refer to [BaB2].

Problem 10.1 *(The p-singular element problem) Given a black-box group known to be isomorphic to $PSL(2,p^k)$ (or any other simple group of Lie type of characteristic p), find an element of order p.*

The difficulty is that when p^k is large then p-singular elements (elements of order divisible by p) become rare in these groups, so simple sampling is unlikely to find a p-singular element. This problem may hold the key to the *constructive recognition* (strong or weak) of $PSL(2,p^k)$ (and other simple groups), the most significant open problem in the area.

Problem 10.2 *(The p-core problem) Given a black-box group G of characteristic p, decide (in Monte Carlo polynomial time) whether or not $O_p(G) = 1$.*

This problem is open even in the case when $G = G'$, $O_p(G)$ is known to be an elementary abelian minimal normal subgroup, and $G/O_p(G) =$

$PSL(2, q)$. A Monte Carlo polynomial-time solution exists for $q = p$ (prime), but the question is open when $q = p^2$ [BabS]. Again, a solution to Problem 10.1 would solve this one. This problem indicates the difficulty of recognizing simplicity of a (nonabelian) black-box group. (For abelian black-box groups, deciding simplicity requires exponential time [BaSz], but for nonabelian black-box groups the problem could be easier.)

In contrast to this (affine) case, the case of central extensions is easy: *if G is a black-box group known to satisfy $G = G'$ and $G/Z(G)$ is known to be simple then $Z(G)$ can be found in Monte Carlo polynomial time.* (This is yet another simple algorithmic consequence of Theorem 4.2, cf. [BaB2].)

We note that for matrix groups $G \leq GL(d, p)$ (p prime), the quotient $G/O_p(G)$is itself a subgroup of $GL(d, p)$ and can be found in deterministic polynomial time. Indeed a composition chain of the G-module $\mathbb{F}_p^d$ can be constructed in polynomial time via Rónyai's algorithm for the radical of an algebra [Ró]. We restrict the G-action to the direct sum of the quotients of the composition chain. $O_p(G)$ is the kernel of this action. (A practical implementation would use the Holt–Rees generalization of the Meataxe [HoR94], avaliable in MAGMA [BoC].)

The preceding problem considered the key question in the case when an abelian group was sitting "under" a nonabelian simple group. The next problem concerns the converse: an abelian (or solvable) group on the "top."

Problem 10.3 *(Outer automorphism problem) Given a black-box group of characteristic p known to satisfy $T \leq G \leq \operatorname{Aut} T$, recognize T within G in Monte Carlo polynomial time. (I. e., we ask for membership testing in T for any $g \in G$.)*

References

[Ad] L. M. Adleman: Algorithmic Number Theory: The Complexity Contribution. *Proc. 35th IEEE FOCS*[5], 1994, pp. 88–113.

[AdD] L. M. Adleman, J. DeMarrais: A subexponential algorithm for discrete logarithms over all finite fields. *Proc. CRYPTO'93*, 1993.

[AlB] Christine Altseimer, A. Borovik: On the recognition of $PSp(2n, q)$ and $P\Omega(2n + 1, q)$ for q odd. Manuscript, 1997.

[Ar] E. Artin: The orders of the classical simple groups. *Comm. Pure Appl. Math.* **8** (1955) 455-472.

[5]FOCS: IEEE Symposium on Foundations of Computer Science. IEEE Computer Society Press.

[As] M. Aschbacher: On the maximal subgroups of the finite classical groups. *Invent. Math.* **76** (1984), 469–514.

[Ba79] L. Babai: Monte Carlo algorithms in graph isomorphism testing. Université de Montréal Tech. Rep. DMS 79-10, 1979 (pp. 42)

[Ba91] L. Babai: Local expansion of vertex-transitive graphs and random generation in finite groups. *Proc. 23rd ACM STOC*[6], 1991, pp. 164–174.

[Ba95] L. Babai: Automorphism groups, isomorphism, reconstruction. Chapter 27 of the *Handbook of Combinatorics*, R. L. Graham, M. Grötschel, L. Lovász, eds., North-Holland – Elsevier, 1995, pp. 1447–1540.

[Ba97] L. Babai: Randomization in group algorithms: conceptual questions. In: [G&C2], pp. 1–16.

[BaB2] L. Babai, R. Beals: A polynomial-time theory of black box groups 2. *In preparation.*

[BaBR] L. Babai, R. Beals, D. Rockmore: Deciding finiteness of matrix groups in deterministic polynomial time. *Proc. ISSAC'93*, Kiev, ACM Press, 1993, pp. 117–126.

[BaCFLS] L. Babai, G. Cooperman, L. Finkelstein, E. M. Luks, Á. Seress: Fast Monte Carlo algorithms for permutation groups. *J. Computer and System Sciences* **50** (1995), 296-307. (Preliminary version appeared in *Proc. 23rd ACM STOC,* 1991, pp. 90–100.)

[BaCFS] L. Babai, G. Cooperman, L. Finkelstein, Á. Seress: Nearly linear time algorithms for permutation groups with a small base. *Proc. ISSAC'91 (Internat. Symp. on Symbolic and Algebraic Computation)*, Bonn 1991, pp. 200–209.

[BaGKLP] L. Babai, A. J. Goodman, W. M. Kantor, E. M. Luks, P. P. Pálfy: Short presentations for finite groups. *J. Algebra* **194** (1997), 79-112.

[BaLS] L. Babai, E. M. Luks, Á. Seress: Computing composition series in primitive groups. In: [G&C1], pp. 1–16.

[BaP] L. Babai, P. P. Pálfy: L. Babai, P. Pálfy: Recognizing finite simple groups by order statistics. Manuscript, 1998.

[BaPS] L. Babai, P. P. Pálfy, J. Saxl: On the number of p-regular elements in simple groups. Manuscript, 1998.

[BabS] L. Babai, A. Shalev: Finding p-singular elements in affine groups. In preparation.

[BaSz] L. Babai, E. Szemerédi: On the complexity of matrix group problems I. *Proc. 25th IEEE FOCS,* 1984, pp. 229–240.

[6]STOC: ACM Symposium on Theory of Computing. ACM Press.

[BacDS] E. Bach, J. Driscoll, J. O. Shallit: Factor refinement. *J. Algorithms* **15** (1993), 199–222.

[BacS] E. Bach, J. O. Shallit: *Algorithmic Number Theory, Vol. I: Efficient Algorithms.* MIT Press, 1996.

[Be95] R. Beals: Algorithms for matrix groups and the Tits alternative. *Proc. 36th IEEE FOCS,* 1995, pp. 593–602.

[Be97] R. Beals: Towards polynomial time algorithms for matrix groups. In: [G&C2], pp. 31–54.

[BeB] R. Beals, L. Babai: Las Vegas algorithms for matrix groups. *Proc. 34th IEEE FOCS,* 1993, pp. 427–436.

[BeLNPS] R. Beals, C. Leedham-Green, Alice Niemeyer, Cheryl Praeger, Á. Seress: A *mélange* of algorithms for recognising black-box alternating and symmetric groups, in preparation.

[BeS] R. Beals, Á. Seress: Structure forest and composition factors for small base groups in nearly linear time. *Proc. 24th ACM STOC*, 1992, pp. 116–125.

[BoC] W. Bosma, J. Cannon: MAGMA *Handbook.* Sydney, 1993.

[BrCFL] S. Bratus, G. Cooperman, L. Finkelstein, S. A. Linton: Constructive recognition of a black box group isomorphic to $GL(n, q)$. In preparation.

[Ca] R. W. Carter: *Simple Groups of Lie Type*, Wiley, New York, 1989.

[CeL] F. Celler, C. Leedham-Green: Calculating the order of an invertible matrix. In: [G&C2], pp. 55–60.

[CeL] F. Celler, C. Leedham-Green: A non-constructive recognition algorithm for the special linear and other classical groups. In: [G&C2], pp. 61–68.

[CeLMNO] F. Celler, C. R. Leedham-Green, S. H. Murray, Alice C. Niemeyer, E. A. O'Brien: Generating random elements of a finite group. *Comm. Algebra* **23** (1995), 4931–4948.

[CoCNPW] J. H. Conway, R. T. Curtis, S. P. Norton, R. A. Parker, R. A. Wilson: *ATLAS of Finite Groups.* Clarendon Press, Oxford, 1985.

[CoF] G. Cooperman, L. Finkelstein: Combinatorial tools for computational group theory. In: [G&C1], pp. 53–86.

[CoFL95] G. Cooperman, L. Finkelstein, S. Linton: Constructive recognition of a black box group isomorphic to $GL(n, 2)$. In: [G&C2], pp. 85–100.

[FeT] W. Feit, J. Tits: Projective representations of minimum degree of group extensions. *Can. J. Math.* **30** (1978), 1092–1102.

[FrR85] K. Friedl, L. Rónyai: Polynomial time solutions of some problems in abstract algebra. *Proc. 17th ACM STOC*, 1985, pp. 153–162.

[FuHL] M. L. Furst, J. Hopcroft, E. M. Luks: Polynomial-time algorithms for permutation groups. *Proc. 21st IEEE FOCS*, 1980, pp. 36–41.

[G&C1] *Groups and Computation.* Proc. 1991 DIMACS Workshop. (L. Finkelstein and W. M. Kantor, eds.) DIMACS Ser. in Discr. Math. and Theor. Comp. Sci. Vol 11, A. M. S. 1993.

[G&C2] *Groups and Computation II.* Proc. 1995 DIMACS Workshop. (L. Finkelstein and W. M. Kantor, eds.) DIMACS Ser. in Discr. Math. and Theor. Comp. Sci. Vol 28, A. M. S. 1997.

[HLOR1] D. F. Holt, C. R. Leedham-Green, E. A. O'Brien, Sarah Rees: Testing matrix groups for primitivity. *J. Algebra* **184** (1996) 795–817.

[HLOR2] D. F. Holt, C. R. Leedham-Green, E. A. O'Brien, Sarah Rees: Computing matrix group decompositions with respect to a normal subgroup. *J. Algebra* **184** (1996) 818–838.

[HoR92] D. F. Holt, Sarah Rees: An implementation of the Neumann–Praeger algorithm for the recognition of special linear groups. *J. Experimental Math.* **1** (1992), 237–242.

[Je] M. Jerrum: A compact representation for permutation groups. *J. Algorithms* **7** (1986), 60–78.

[HoR94] D. F. Holt, Sarah Rees: Testing modules for irreducibility. *J. Austral. Math. Soc.* **57** (1994), 1–16.

[Ka] W. M. Kantor: Sylow's theorem in polynomial time. *J. Comp. Sys. Sci.* **30** (1985), 359–394.

[KaS] W. M. Kantor, Á. Seress: Black box classical groups. Manuscript, 1997. 152 pages. *Trans. AMS*, to appear.

[KiLST] W. Kimmerle, R. Lyons, R. Sandling, D. N. Teague: Composition factors from the group ring and Artin's theorem on orders of simple groups. *Proc. London Math. Soc.* **60** (1990) 89–122.

[KlL] P. Kleidman, M. Liebeck: *The Subgroup Structure of the Finite Classical Groups.* LMS Lecture Notes 129, Cambridge Univ. Press, 1990

[Kn] D.E. Knuth: Efficient representation of perm groups. *Combinatorica* **11** (1991), pp. 33–44. (Preliminary version circulated since 1981.)

[LeO] C. R. Leedham-Green, E. A. O'Brien: Tensor products are projective geometries. *J. Algebra* **189** (1997) 514–528.

[Leon] J. S. Leon: On an algorithm for finding a base and strong generating set for a group given by generating permutations. *Math. Comp.* **35** (1980), 941–974.

[Lo] R. Lovorn: *Rigorous, subexponential algorithms for discrete logarithms over finite fields.* Ph.D. thesis, University of Georgia, 1992.

[LaS] V. Landazuri, G. M. Seitz: On the minimal degrees of projective representations of the finite Chevalley groups. *J. Algebra* **32** (1974), pp. 418–443.

[Lu82] E. M. Luks: Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Sys. Sci.* **25** (1982), 42–65.

[Lu87] E. M. Luks: Computing the composition factors of a permutation group in polynomial time. *Combinatorica* **7** (1987), pp. 87–99.

[Lu92] E. M. Luks: Computing in solvable matrix groups. *Proc. 33rd IEEE FOCS*, 1992, pp. 111–120.

[Lu93] E. M. Luks: Permutation groups and polynomial-time computation. In: [G&C1], pp. 139–175.

[Mi] G. L. Miller: Riemann's hypothesis and tests for primality. *J. Comput. Sys. Sci.* **13** (1976), 300–317.

[Mo] P. Morje: *A nearly linear algorithm for Sylow subgroups of permutation groups.* Ph. D. Thesis. Ohio State University, 1996.

[NeP] P. M. Neumann, Cheryl E. Praeger: A recognition algorithm for special linear groups. *Proc. London Math. Soc.* **65** (1992), 555–603.

[NiP97] Alice Niemeyer, Cheryl E. Praeger: Implementing a recognition algorithm for classical groups. In: [G&C2], pp. 273–296.

[NiP98] Alice C. Niemeyer, Cheryl E. Praeger: A recognition algorithm for classical groups over finite fields. *Proc. London Math. Soc.* **77** (1998), 117–169.

[Par84] R. A. Parker: The computer calculation of modular characters (the meat-axe), Computational Group Theory (London, New York) (M.D. Atkinson, ed.), (Durham, 1982), Academic Press, 1984, pp. 267–274.

[Ró] L. Rónyai: Computing the structure of finite algebras. *J. Symbolic Comp.* **9** (1990), 355–373.

[Sch] M. Schönert *et al.: GAP – Groups, Algorithms, and Programming.* Lehrstuhl D für Mathematik, RWTH Aachen, Germany, 1994.

[Se] Á. Seress: *Permutation Group Algorithms.* Book, in preparation.

[Sh] P. W. Shor: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comp.* **26** (1997), 1484–1509.

[Si71] C. C. Sims: Computation with permutation groups. *Proc. Second Symp. on Symbolic and Algebraic Manipulation* (New York) (S.R. Petrick, ed.), ACM, 1971, pp. 23–28.

[Si78] C. C. Sims: Some group-theoretic algorithms. *In: Springer Lecture Notes in Math.* **697** (1978), 108–124.

[St] R. Steinberg: *Lectures on Chevalley groups.* Mimeographed notes. Yale University, 1967.