COMPUTATIONAL COMPLEXITY AND THE CLASSIFICATION OF FINITE SIMPLE GROUPS

L. BABAI*
Dept. Algebra
Eötvös University
Budapest, Hungary

W. M. KANTOR
Dept. Mathematics
University of Oregon
Eugene, Ore.

E. M. LUKS
Dept. Computer and Info. Sci.
University of Oregon
Eugene, Ore.

Abstract

We address the graph isomorphism problem and related fundamental complexity problems of computational group theory.
The main results are these:

A1. A polynomial time algorithm to test simplicity and find composition factors of a given permutation group (COMP).
A2. A polynomial time algorithm to find elements of given prime order p in a permutation group of order divisible by p.
A3. A polynomial time reduction of the problem of finding Sylow subgroups of permutation groups (SYLFIND) to finding the intersection of two cosets of permutation groups (INT). As a consequence, one can find Sylow subgroups of solvable groups and of groups with bounded nonabelian composition factors in polynomial time.
A4. A polynomial time algorithm to solve SYLFIND for finite simple groups.

A5. An $n^{cd/\log d}$ algorithm for isomorphism (ISO) of graphs of valency less than d and a consequent improved moderately exponential general graph isomorphism test in $\exp(c\sqrt{n \log n})$ steps.

A6. A moderately exponential, $n^{c\sqrt{n}}$ algorithm for INT. Combined with A3, we obtain an $n^{c\sqrt{n}}$ algorithm for SYLFIND as well.

All these problems have strong links to each other. ISO easily reduces to INT. A subcase of SYLFIND was solved in polynomial time and applied to bounded valence ISO in [Lu1]. Now, SYLFIND is reduced to INT. Interesting special cases of SYLFIND belong to NP∩coNP and are not known to have subexponential solutions.
All the results stated depend on the classification of finite simple groups. We note that no previous ISO test had $n^{o(d)}$ worst case behavior for graphs of valency less than d. It appears that unless there is another radical breakthrough in ISO, independent of the previous one, the simple groups classification is an indispensable tool for further developments.

Supported in part by NSF Grants MCS 7903130 and MCS 8301756
* Currently visiting Dept. Computer and Info. Sci., University of Oregon, Eugene, Ore.

1. The classification

The final success of a major joint effort by a large number of mathematicians is the recent completion of the classification of finite simple groups (apart from the uniqueness of the Monster). For non-technical information on this achievement see [Co]. A more detailed account can be found in [Go].
There is no doubt that this result, the proof of which involves several thousand pages, will have an enormous impact on large sections of mathematics. The solutions of many old problems in group theory are known to follow, such as the nonexistence of 6-transitive permutation groups and indeed an essentially complete list of doubly transitive groups; Schreier's conjecture that the outer automorphism group of a finite simple group is solvable; the conjecture that every finite simple group is generated by two elements (cf. [Ca1],[Fe],[Ka2] for surveys of more consequences). Combinatorics has already greatly benefited from these results ([Ca2], [CPSS],[Ka2],[We],[Pá2]), as have other fields, such as model theory ([CL],[CHL]), number theory (see [Fe]) and universal algebra [Pá2].
The aim of the present paper is to discuss algorithmic consequences of the classification. At the present time, we are unable to give a polynomial time algorithm to solve such a seemingly harmless problem as finding an element of order p in a permutation group of order divisible by p, without invoking detailed knowledge of the classification.
The fundamental information about finite simple groups we shall need is the following (see [Ca1],[Go]). Apart from a finite number of "sporadic" groups (these shall never concern us), the finite simple groups fall into the following categories: cyclic groups of prime order, alternating groups and groups of Lie type. The groups of Lie type are matrix groups over finite fields. They are divided into a finite number of infinite families. Each family is parametrized by the dimension of the matrices and the order of the field. The classical groups form essentially four of these families (projective special linear, symplectic, orthogonal and unitary groups). The remaining ten families of groups of Lie type, the so-called exceptional groups, have bounded dimensions and are thus parametrized by only the order of the field.
Statements depending on the classification of finite simple groups will be marked by (CFSG).

2. Complexity problems in computational group theory

The two basic tools in analyzing group structure are composition factors and Sylow subgroups. Both can be regarded as building blocks of a group. Finding them is a problem analogous to factoring integers; we are "taking the group apart to see what makes it tick".
It may be of particular interest to note that

0272-5428/83/0000/0162$01.00 © 1983 IEEE

some natural subproblems of SYLFIND (finding Sylow subgroups) belong to NP∩coNP while no subexponential solution is known for them. An example : Does a given permutation group G have an element of order p^r where p is a given prime and p^r is the largest power of p dividing the order of G ?

Easy reductions show that the following problems are polynomial time equivalent: finding the centralizer of a permutation in a given group, finding the stabilizer of a given subset, finding the intersection of two permutation groups (given by generators), finding the intersection of two cosets of permutation groups (INT) [Lu2]. Graph isomorphism (ISO) is easily reduced to these. SYLFIND is also reducible to them (Section 8).

3. Dependence of the results on the classification

The results depend on the classification in varying ways.

The polynomial time simplicity test and the determination of the composition factors (Section 5) uses the classification through Schreier's conjecture stated in the previous section.

Finding elements of given prime order p, and more generally finding Sylow p-subgroups of simple groups in polynomial time, presently relies on detailed knowledge of properties of all families of finite simple groups (Sections 6,7). Part of this result, combined with the composition factors algorithm, yields a polynomial time algorithm for finding elements of given prime order in arbitrary permutation groups.

The reduction of SYLFIND to INT (Section 8) uses the Sylow subgroups of finite simple groups (A4). However, restricting the reduction to solvable groups and groups with bounded nonabelian composition factors, we find Sylow subgroups of such groups in polynomial time without using the classification. (This relies on the INT algorithm of [Lu1] combined with estimates from [Pa1] and [BCP].)

The timing analysis of the INT algorithm described in Section 10 requires estimates on the orders of primitive permutation groups. Using the recent elementary bounds of [Ba2],[Ba4], we obtain an $\exp(c\sqrt{n}\log^2 n)$ time bound for INT. Using the classification, one can essentially list all primitive permutation groups of order greater than $n^{\log n}$[Ca1], bringing down the INT running time to $n^{c\sqrt{n}}$. More significantly, this list provides a tool that might eventually lead to a subexponential (i.e. $\exp(n^{o(1)})$) INT algorithm and thereby to subexponential ISO and SYLFIND algorithms.

Due to an n to n^2 blowup in the ISO to INT reduction, the current INT algorithms do not yield a better than brute force ISO test. Notwithstanding, there is a moderately exponential, $\exp(c\sqrt{n}\log^{5/2} n)$ ISO test which refers to the elementary bounds mentioned above (see [BL]). A naive use of those consequences of the classification [Ca1] mentioned in the previous paragraph reduces the exponent of log n from 5/2 to 1. A further in-depth analysis, details of which will be outlined in Section 9, results in the best current bound, $\exp(c\sqrt{n}\log n)$. (It may be curious to note that the current bound for factoring n-digit integers looks precisely like this [Di]; the constant c was recently improved [Sch].) It is actually the underlying $n^{cd/\log d}$ ISO bound for graphs of valency less than d that makes significant use of the classification.

Finally we mention that somewhat surprisingly, the moderately exponential ISO and INT algorithms both serve as subroutines in a polynomial time algorithm. Namely, isomorphism of distributive lattices can be tested in $n^{3+o(1)}$ steps [Ba6].(Note that isomorphism of distributive lattices has only recently been brought down to polynomial time [BKL].)

4. Notation and preliminaries

4.1. Basic group theory. We assume that the reader is familiar with some of the basic notions of group theory such as that contained in [Ro] (simple groups, composition factors, Sylow p-subgroups, center, conjugate elements and subgroups (notation: for H a subgroup and g an element of G, $H^g=g^{-1}Hg$), commutator subgroup = derived subgroup (denoted G'), derived series, solvability, direct products, the projective linear groups). [Ro] contains some of the basic material on permutation groups as well (orbits, stabilizers); we shall also need the beginning of [Wi].

We briefly review some standard notation.

An action of a group G on a set X is a homomorphism $g\mapsto g$ of G onto a subgroup G^X of Sym(X), the group of all permutations of X. The orbit of $x\in X$ is $x^G=\{x^g: g\in G\}$, where x^g is the image of x under g . Here, $|x^G|=|G:G_x|$ where G_x is the stabilizer $\{g\in G: x^g=x\}$. Let G_{xy} denote $G_x\cap G_y$.

If $x^G=X$ then G is said to be transitive on X. In this situation, one looks for G-invariant partitions of X. If there is no such nontrivial partition then G is called primitive ; otherwise G is imprimitive. The blocks of a G-invariant partition are called blocks of imprimitivity. G acts transitively on them. This action is primitive iff the partition is minimal (the blocks are maximal).

For $H\leq G$ (H a subgroup of G) and $g\in G$, the cosets gH and Hg of H will be called subcosets of G. The empty set will also be regarded as a subcoset of G. With this convention, the intersection of subcosets is always a subcoset. Right and left cosets make no difference since a left coset of H is a right coset of another subgroup, conjugate to H. For every subgroup $H\leq G$ there is a natural G-action on the set G/H of (left) cosets.

If ϕ: G-> Sym(X) and ψ: G-> Sym(Y) are two actions of the same group G then a map f: X->Y is a G-map if $\phi(g)f=f\psi(g)$ for every g in G.

Our "end of proof" mark is ##.

4.2. Algorithms. Throughout this paper, X will denote an n-set. The inputs of our algorithms will be permutation groups, usually acting on X.

Every permutation group will be given by a set of generating permutations. A nonempty subcoset Hg of Sym(X) can be represented by specifying a set of generators of H and a representative g.

Given a permutation group G the following can be found in polynomial time: the order of G, the pointwise stabilizer of a given subset, the derived series, the orbits of G, a minimal partition into blocks of imprimitivity ([Si],[At],[FHL]).

Our complexity estimates will be given in terms of n (the degree of the input group) rather than in terms of the input length. Of course, if an input group is represented by an unreasonably large number of generators, then the time required to remove redundant generators has to be added [Si],[FHL]. (It is easy to see that any set of n log n permutations contains redundant ones. A more involved argument shows that even 2n is too much [Ba5].)

Let $\mathcal{K}$ be a class of finite groups closed under isomorphisms, taking subgroups and factor groups. For any such $\mathcal{K}$ we shall be interested in the following four problems.

Finding Sylow subgroups (SYLFIND($\mathcal{K}$))
INPUT: a permutation group $G\in\mathcal{K}$ and a prime p.
OUTPUT: a Sylow p-subgroup of G.

Sylow conjugacy (SYLCONJ($\mathcal{K}$))
INPUT: a permutation group $G\in\mathcal{K}$, a prime p and two Sylow p-subgroups P(1) and P(2) of G.
OUTPUT: $g\in G$ such that $P(1)^g = P(2)$.

Coset intersection (INT($\mathcal{K}$))
INPUT: a set X, two subgroups G,H and two elements g,h of Sym(X) where at least one of G,H belongs to $\mathcal{K}$.
OUTPUT: the subcoset $Gg\cap Hh$ of Sym(X).

Color automorphism (CAUT(K))
INPUT: a set X, a coloring : $X \to \{colors\}$, a group $G \in K$ acting on X, and a permutation $g \in Sym(X)$
OUTPUT: the set of Gg-automorphisms of the coloring, i.e. $AUT(Gg,\phi) =$

$$\{h \in Gg: \phi(x)=\phi(x^g) \text{ for all } x \in X\}.$$

CAUT(K) is clearly a subcase of INT(K). They are actually polynomial time equivalent for any K [Lu2]. It follows from [Pa1] and [BCP] that the INT(K) algorithm given by [Lu1] works in polynomial time when K is the class of solvable groups or more generally a class of groups with bounded nonabelian composition factors. This result will be crucial for applications of the SYLFIND(K) to INT(K) reduction (see Cor. 8.2).(The actual restriction on K is even weaker, cf.Cor. 8.2.)

4.3. Complexity classes. By a moderately exponential function of u we mean a function bounded by $O(\exp(u^{1-c}))$ for some positive constant c.

An algorithm is moderately exponential if its running time is a moderately exponential function of $u=\log b$ where b is the running time of the natural brute force algorithm associated with the problem. If $b=n!$ (as in our examples) then $\log b$ can be replaced by n in this definition.

An algorithm is subexponential if its running time is $\exp(N^{o(1)})$ where N is the input length. This class is invariant under polynomial reductions, in contrast to the moderately exponential class.

The best known algorithms for factoring integers [Di],[Sch], graph isomorphism (section 9), Sylow subgroups and coset intersection for general groups (sections 8,10) are moderately exponential. Primality [Pom], isomorphism of projective planes [Mi] and of tournaments [BL] are subexponential.

5. Simplicity test and composition factors

We show

Theorem 5.1. (CFSG) Testing simplicity of permutation groups is in P.

Proof. We describe a polynomial-time algorithm REDUCE which accepts, as input, generators of a subgroup, G, of Sym(X), where X is a set of size n and outputs exactly one of the following:
(i) "G is simple."
(ii) Generators of a proper normal subgroup of G.
(iii) A faithful action of G on a set of size < n.

It is then clear how repeated application of REDUCE in case (iii) guarantees a polynomial-time simplicity test.

At several points in REDUCE it is useful to consider induced representations of G. In the following procedure, Y is a set, with $|Y| > 1$, on which G acts transitively. (G,X,n are global.)

```
procedure TEST_ACTION(Y)
  Y' <- a minimal G-block system in Y
  N  <- the kernel of the G-action on Y'
  if (N ≠ 1) then (output N; stop)
  if (|Y'| < n) then (output Y'; stop)
  return
```

REDUCE has seven major steps:

```
Step 1. Let Y be any non-trivial orbit
        Call TEST_ACTION(Y)

Step 2. If (G ≠ G') then
          if (G' ≠ 1) then (output G'; stop)
          else (output "Simple (abelian)"; stop)
Step 3. Let W be any subset of G of size n+1
        for each g,h in W
          N <- normal closure of <gh^-1>
          if (N ≠ G) then (output N; stop)
Step 4. Choose any x,y in X with x ≠ y
        Z <- the set of fixed points of G
        for each z in Z
          if (for some g in G, x^g = y, y^g = z)
            then fix such g
            Y <- (x,y,z,...), the cycle of g
            containing x
            Y" <- the set of G-images of Y (identifying cyclic permutations)
            call TEST_ACTION(Y")
Step 5. Choose any x in X
        for each y in X-{x}
          Y <- G-orbit of {x,y} (in the set of
          unordered pairs)
          call TEST_ACTION(Y)
Step 6. Choose any x in X
        for each y,z,w in X
          H <- <G_xy, G_zw>
          if (H ≠ G) then call TEST_ACTION(G/H)
Step 7. Output "Simple (non-abelian)"
```

Comments on proving the correctness of REDUCE: It is easy to check that, if REDUCE(G) outputs a subgroup then it has correctly identified a proper normal subgroup of G and if it outputs a set then it has constructed a smaller permutation domain. It is necessary to show that, if the output is "Simple" then G is, indeed, simple. Step 1 reduces the problem to the primitive case (now a standard technique). Hence, if G is abelian, it necessarily has prime order, easily justifying the proclamation "Simple" within Step 2. The problem, then, is to prove that, if the algorithm reaches Step 7 then G is simple. Passing Step 2 also means that $G = G'$. This was a natural test to perform since the derived group is normal and is easy to compute [FHL]. However, when all is said and done, we'll see that this step played a more subtle role. Step 3 detects the presence of a normal subgroup of small index ($\leq n$), covering a few special cases.

Steps 4,5,6 are motivated by (non-classification-dependent) results on primitive permutation groups (see, especially, the O'Nan-Scott Theorem in [Cal]). Such results establish relations between the structure of a primitive action of G and the nature of N where N is the socle of G. Since N is necessarily transitive, $n = |N|/|N_x|$. If we can force an action in which the point stabilizers in N are increased, we thereby reduce the size of the domain. For example, suppose $N = 1$. We show, in such case, that the algorithm terminates in Step 4. The reason is: There is a unique element h in N such that $x^h = y$ and then $z = y^h$ is left fixed by G_{xy}. When we process z, g will exist. Though we may not have $g = h$, one can show that the cycle, Y, of g containing x is also a cycle of h. The set Y" will then consist of cycles of elements of N. Since the action of h on Y" has a fixed point (namely Y), TEST_ACTION(Y") will either output a kernel or a primitive action on a set of size less than $n = |N|$.

Steps 5 and 6 play analogous roles in reducing the domain for other possible actions of N.

Without reference to the classification, one verifies that the algorithm can get to Step 7 only if N is simple, which puts G between N and Aut(N), and $G = G'$ (recall, we passed Step 2). At that point the classification-dependent Schreier's "conjecture" is invoked. It implies, in such case, that $G = N$. ##

The simplicity test is more than the language recognition algorithm announced in Theorem 5.1. In the non-simple case a witness, a proper normal subgroup, is output. We can exploit this observation.

Theorem 5.2 (CFSG) Given generators for a permutation group, G, a composition series for G, including permutation representations of the quotients, can be constructed in polynomial time.

Proof. It suffices to be able to exhibit a maximal normal subgroup, N, of G and a representation of

G/N. If G is simple, then N = 1. Otherwise, the simplicity test returns a proper (not necessarily maximal) normal subgroup, N, of G. One can construct a non-trivial action of G whose kernel contains N: Let i be the least index such that G(i+1)N < G(i)N (here G(j) denotes the point stabilizer of the first j points in X); then G = G(i) acts on the left coset space G/G(i+1)N, N acting trivially. If the kernel of this action is larger than N then replace N by the kernel and repeat. Otherwise, we have constructed a faithful action of G/N. If G/N is not simple, find a proper normal subgroup, replace N by (the pullback in Sym(X) of) that subgroup and repeat. Otherwise, we have the required N and representation of G/N.##

We remark on three other applications of the above methods and results which are needed elsewhere in this paper.

In some instances, it is useful to go beyond the permutation representations guaranteed by Theorem 5.2 and construct the "natural" representation of a simple group. For example, the improved graph isomorphism test (section 9), requires the natural action of a permutation group which is isomorphic to the full alternating group (on some other set). This can be done efficiently:

Proposition 5.3. Given a permutation group, $G \leq Sym(X)$, it is possible to detect in polynomial time whether G is isomorphic to Alt(Y), for some Y, and, if so, to construct such Y.

Proof. The algorithm employs reductions to smaller sets as in REDUCE. As usual, we may assume that G is primitive. Suppose that $G \cong Alt(Y)$. Then X must be G-isomorphic either to $\binom{Y}{k}$, for some k, or to the set of partitions of Y into m equal parts. We indicate the procedure for detecting and converting the former; the other case is similar. If k = 1, then we know it and take Y = X. Assume then that k > 1 (and we may suppose n > 5 and $k \leq n/2$). It is not difficult to show that there exist k-tuples a,b,c in Y (equivalently, points in X) such that $H = \langle G_{ab}, G_{bc}, G_{ca} \rangle$ is precisely the stablilizer in Alt(Y) of a (k-1)-set. Thus, G/H is G-isomorphic to the set of all unordered (k-1)-tuples in Y. Hence, trying all triples a,b,c in X, we should find an instance where $|G/H| < |X|$, so that G/H is a smaller permutation domain; replace X by G/H and start over. ##

More generally, see Theorem 6.1.

The graph isomorphism test also requires the production of the socle of a certain permutation group. It happens, at that point, that the socle is known to be non-abelian and to be the unique minimal normal subgroup of G. In such a case, it is precisely the normal closure of the last term in a composition series of G and so it is computable in polynomial time. However, more generally

Proposition 5.4 (CFSG) Given a permutation group, G, the subgroup generated by the non-abelian minimal normal subgroups (the "non-abelian part" of the socle) can be found in polynomial time.

Proof. The following algorithm can be shown to produce the desired subgroup:

```
procedure  NA_SOC(G)
  N <- a maximal normal subgroup of G
  K <- the last term in the derived series of the
       centralizer in G of N
  if (K is simple) then output  <K,NA_SOC(N)>
  else output  NA_SOC(N)
```

The complexity of finding the abelian factor of the socle is open (see 11.1). We do note, however, that techniques like those in the simplicity test can be used for

Proposition 5.5. Given generators for a permutation group G, the presence of an abelian normal subgroup of G can be detected in polynomial time. In fact, if such exist, then generators for at least one can be determined.

There is a reduction to the primitive case (a bit trickier than usual) whereupon the algorithm proceeds much like Step 4 of REDUCE. We note that this result is independent of the classification.

Complete proofs of the results in this section will appear in [Lu3].

6. The Replacement Theorem

In many situations, one actually reduces to the case of a simple subgroup G of Sym(X). (Examples are found in Sections 7,8,9.) The following theorem can be used to switch to the most natural permutation representation of G in that case.

Theorem 6.1 (Replacement Theorem)(CFSG). There is a polynomial-time algorithm which, when given a simple group $G \leq Sym(X)$ with $|G| > n^{36}$, produces a set W with $|W| < n$ on which G acts such that one of the following holds :

(i) G is isomorphic to the alternating group on W, and acts on W as that group; or

(ii) G is isomorphic to a classical group defined on a vector space V, and G acts on W exactly as it does on the set of all 1-spaces of V.

Thus, if $G \cong PSL(d,q)$, $PSp(d,q)$, $P\Omega^{\pm}(d,q)$ or $PSU(d,q)$ then W is the set of points of a projective space PG(d-1,q).

Outline of the proof. As usual, we may assume that G is primitive on the n-set X. We may also assume that G is not a sporadic simple group, and therefore is an alternating group or a group of Lie type.

Let $x \in X$. Since $|G| > n$, results due to Bochert (see [Wi,p.41]) and in [Kal] lead to the conclusion that for some k,m,d,q either

(a) $G \cong A$, and G is the stabilizer of an r-subset of the k-set; or

(b) $G \cong A$, and G is the stabilizer of a partition into m equal parts of the k-set; or

(c) G is a classical group (cf. Section 1) defined on a vector space V of dimension d over the field of order q, and G is the stabilizer of an r-subspace of V.

Thus, we can identify X with a very natural set X. Namely, in case (a) X is the set of all r-subsets of the k-set; in case (b) X is the set of partitions of the k-set into m equal parts; in case (c) X is an orbit of the set of all r-subspaces of V. These cases can be dealt with independently of any assumption on the order of G. In cases (a) and (b) we have to replace X by the k-set. An algorithm achieving this has been indicated in Prop. 5.3. In case (c) we have to replace r by r=1. Once this has been done, the new set X might only consist of one of the G-orbits of 1-spaces of V, in which case the remaining orbits of 1-spaces also must be reconstructed. Details of (c) can be found in [Ka3]. ##

It seems likely that, in the near future, analogues of [Kal] will be proved for the exceptional groups of Lie type. This would produce a significant reduction of the exponent 36 in Theorem 6.1 (and, consequently, of the exponent 37 in the proof of Theorem 7.1).

7. Elements of prime order

The following result is an algorithmic version of an elementary theorem of Cauchy:

Theorem 7.1. Given a group $G \leq Sym(X)$ and a prime p dividing |G|, an element of order p can be found in polynomial time.

Proof. We begin with a reduction to the case of a simple group G. Use Theorem 5.1 to find a set X' on which G acts such that $G^{X'}$ is simple and $|X'| \leq n$; let M be the pointwise stabilizer of X'. If $p \mid |M|$, replace G by M. (By [Ba5], such a replacement can take place at most 2n times.) Thus, we may assume

that p divides the order of $G^{X'}$. If $g \in G$ and $|g^{X'}| = p$ then some power of g has order p. Consequently, we can replace G by $G^{X'}$ and assume that G is a nonabelian simple group.

In this situation, much more can be said [Ka3]:

Theorem 7.2. Given a simple group $G \leq Sym(X)$ and a prime p, a Sylow p-subgroup of G can be found in polynomial time.

However, we will continue the proof of Theorem 7.1. If $|G| \leq n^{36}$ then the order of each element of G can be found. (This part of the algorithm runs in time $O(n^{37})$. The remainder of the algorithm is much faster.) Now assume that $|G| > n^{36}$ and apply the Replacement Theorem in order to replace X by a new set, which we will again call X. At this point, G acts on X in an especially natural manner.

For example, if G is an alternating group then it is now A_n. If we identify X with $\{1,\ldots,n\}$, then $(1,\ldots,p)$ has order p.

A much more typical example is provided by the group G=PSL(d,q). Here, X can be regarded as the set of $(q^d-1)/(q-1)$ 1-spaces of the vector space V used to define G. Use Prop. 5.5 to find a nontrivial elementary abelian normal subgroup Q of G for some $x \in W$. This is a group of order q^{d-1}. If p divides q, any nontrivial element of Q has order p. Now suppose p does not divide q.

First find $x = x_1,\ldots,x_d \in X$ corresponding to an independent set of 1-spaces of V. Then find $c \in G$ inducing the cycle $(x_1,\ldots,x_d)$ on $\{x_1,\ldots,x_d\}$. Then the coset Qc consists of all elements of G arising from companion matrices of all the q^{d-1} monic polynomials f(t) of degree d over GF(q) having constant term $(-1)^d$. (Note that $|Qc| = q^{d-1} < |W| < n^2$, so that all elements of Qc can be listed.) Then some power of an element of Qc has order p. (Namely, when m(t) is the minimal polynomial of a matrix of multiplicative order p, the element of Qc corresponding to the polynomial $f(t) = m(t)(t-1)^{d-\deg m}$ has order divisible by p.)

The argument used in the case of the remaining classical groups is fairly similar. However, instead of linear algebra, it depends on results concerning algebraic groups [St, Thms. 9.4, 9.5, 9.12].##

Note, that, in fact, in the case of G=PSL(d,q), every element of G of order relatively prime to q is a conjugate in G to a power of an element of Qc. Similarly strong results hold for the remaining classical groups (again using [St]; see (11.4) in [Ka2]).

8. Sylow subgroups

In this section we reduce the Sylow subgroup problems to coset intersection. $\mathcal{K}$ denotes an arbitrary class of finite groups, closed under isomorphisms, taking subgroups and factor groups. For the definitions see subsection 4.2.

Theorem 8.1. (i) SYLCONJ($\mathcal{K}$) is polynomial time reducible to INT($\mathcal{K}$).

(ii) (CFSG) SYLFIND($\mathcal{K}$) is polynomial time reducible to SYLCONJ($\mathcal{K}$).

In fact we show that if we have an O(T(n)) algorithm for INT($\mathcal{K}$) (n is the degree of the input groups, cf. 4.2), then SYLCONJ($\mathcal{K}$) can be solved in $O(nT(n)+n^c)$ steps, and an $O(T'(n))$ SYLCONJ($\mathcal{K}$) algorithm can be turned into an $O(nT'(n)+n^c)$ algorithm for SYLFIND($\mathcal{K}$). In particular, moderately exponential INT algorithms result in moderately exponential SYLFIND (see Cor. 10.9), in contrast to the ISO to INT reduction [Lu1] where T(n) is replaced by $T(n^2)$.

Proof. (i) We are given a subgroup G of Sym(X), belonging to $\mathcal{K}$, a prime number p, and Sylow p-subgroups P(1) and P(2) of G; we want to find $f \in G$ with $P(1)^f = P(2)$. The algorithm will proceed in five steps and will refer to an INT($\mathcal{K}$) subroutine.

For $S \leq G$ let Orb S denote the set of orbits of $\langle S \rangle$.

1. If Orb P(1) ≠ Orb P(2), use INT($\mathcal{K}$) to find $g \in G$ such that $(\text{Orb } P(1))^g = \text{Orb } P(2)$.

Comments. It is straightforward to determine the group H of all elements of Sym(X) sending Orb P(i) to itself, and also to find $g' \in Sym(X)$ sending Orb P(1) to Orb P(2). Now find $g \in G \cap Hg'$. (This intersection is nonempty because of Sylow's theorem.) Then $\text{Orb } P(1)^g = \text{Orb } P(2)$.

2. Replace P(1) by $P(1)^g$ and then replace G by $\langle P(1), P(2) \rangle$.

Comment. Now Orb P(1) = Orb P(2) = Orb G.

3. If G is intransitive, divide and conquer.

Comment. If Y is an orbit of G, find g G such that $(P(1)^Y)^g = P(2)^Y$ and return to 2. Repeat this for all Y.

4. (We may now assume that G, P(1) and P(2) are transitive on X.)
Find Z(P(i)), the center of P(i). Pick $z(1) \in Z(P(1)) - \{1\}$.
Use INT($\mathcal{K}$) to test each $z(2) \in Z(P(2))$ to see if there is a $g \in G$ such that $z(1)^g = z(2)$. Find such a g and return to 2.

Comment. Since P(2) is transitive, Z(P(2)) is semiregular and so it has order ≤n. It is straightforward to find the centralizer H of z(1) in Sym(X) and to find $g' \in Sym(X)$ such that $z(1)^{g'} = z(2)$ for any $z(2) \in Z(P(2))$. Finally, $G \cap Hg'$ consists of those elements of g of G for which $z(1)^g = z(2)$. For some $z(2) \in Z(2)$, this intersection must be nonempty by Sylow's theorem.

5. Let z=z(2) and $\hat{X}$=Orb z. Find f G such that $(P(1)^{\hat{X}})^f = P(2)^{\hat{X}}$. Then $P(1)^f = P(2)$.

Comments. By 4, $z \in Z(P(1)) \cap Z(P(2)) \leq Z(G)$, so that G acts on $\hat{X}$. Then f can be found recursively. The kernel of the homomorphism $G \to G^{\hat{X}}$ is a p-group. Therefore, the second sentence of 5 implies the third.

Timing. Step 1 is invoked at most once. At step 3, we add up the time needed to handle each orbit. For each recursive call at step 5, P(i) remains transitive and the value of n changes to n/p or less. The INT($\mathcal{K}$) subroutine is invoked once in step 1; and in step 4 at most m times for each orbit of size m. Thus, there are a total of $< m + m/p + m/p + \ldots < 2m$ calls in the case of an orbit of size m, totalling at $< 1 + 2n$ calls to Int($\mathcal{K}$).

(ii) This time we are given G and p, and we want to find a Sylow p-subgroup of G, using a SYLCONJ($\mathcal{K}$) subroutine.

1. Find a set Y such that G acts on Y, $|Y| \leq n$, and G^Y is simple. Let M be the kernel of $G \to G^Y$.

Comment. Use Theorem 5.2.

2. If p does not divide |G/M|, replace G by M.

3. If $p \neq |G/M|$, find a Sylow p-subgroup of G/M and find its complete inverse image H in G. Then replace G by H.

Comments. Since $G/M \cong G^Y$ is simple, Theorem 7.2 can be used. Note that H contains a Sylow p-subgroup of G.

4. Recursively find a Sylow p-subgroup P of M.

5. Let $g \in G-M$. Find $m \in M$ such that $(P^g)^m = P$.

Comments. At least one of the generators of G is outside M, so let one of them be g. To find m, use the SYLCONJ($\mathcal{K}$) subroutine.

6. Let <g'> be a Sylow p-subgroup of the cyclic group <gm>. Then <P,g'> is a Sylow p-subgroup of G.
Comment. $|\langle P,g'\rangle| = p|P|$.

Timing. Since p^n does not divide n!, there are fewer than n recursive calls for SYLFIND (step 4). Each time, SYLCONJ(K) is invoked once. This gives the stated time bound. ##

For more detailed proofs, see [KT].

In subsection 4,2 we noted that INT(K) can be solved in polynomial time for certain classes K. In fact, by [Lu1, p.61] and [BCP], we have the following

Corollary 8.2. SYLFIND(K) and SYLCONJ(K) have polynomial time algorithms if K is the class of groups all of whose noncyclic composition factors are of bounded order or of Lie type of bounded dimension.

Note that this corollary does not depend on CFSG, since the use of the COMP subroutine and the reference to finding Sylow subgoups in simple groups (steps 1 and 3 in the SYLFIND(K) algorithm) are restricted to groups from the class K.

In [KT] it is shown that, for G restricted as in Cor. 8.2, a given p-subgroup can be embedded in a Sylow p-subgroup in polynomial time, and polynomial-time versions of other group-theoretic theorems are also obtained. For example, for such groups G the largest normal p-subgroup of G is found in polynomial time. Even this problem remains open for general G, in spite of attempts in [Ka2] and [KT].

9. Graph isomorphism

The main result is

Theorem 9.1 (CFSG) Isomorphism of n-vertex, d-valent graphs can be tested in $n^{cd/\log d}$ steps.

Note, in particular, that the exponent is o(d) as $d \to \infty$.

For general graph isomorphism, a Zemlyachenko-valence- reduction ([ZKT], see also [Ba3]) breaks the problem into $n^{4\sqrt{n/\log n}}$ problems on $\sqrt{n \log n}$-valent graphs. Hence

Corollary 9.2 (CFSG) Graph isomorphism can be tested in $n^{\sqrt{cn/\log d}}$ steps.

In [Lu1], the isomorphism problem for d-valent graphs was reduced to CAUT(Γ_d) where Γ_d is the class of groups all of whose composition factors lie in S_{d-1}). It would appear that the cost associated with that approach is already prohibitive, considering the above goal, for it involves a set blow-up from n to n^d. Nevertheless, it is instructive to consider an improved CAUT(Γ_d) algorithm which achieves the time bound $n^{cd/\log d}$ on sets of size n. We refer the reader to [BL] for an indication on how to avoid the set blow-up. We point out below that a similar trick yields an application to INT.

We refer the reader to [Lu1] for the basic procedure. A fundamental observation is that available permutation group procedures [FHL] make it feasible to work "locally", stabilizing colors on one orbit at a time. Thus, a natural and efficient divide-and-conquer reduces the problem to the case when the group, G, acts transitively on the set, X. At that point, the idea is to restore intransitivity by dropping down to a suitable subgroup. (Cf. Lemma 10.4.) This subgroup reduction is guided by the induced primitive action on a minimal G-block system, B, in X. There is a constructible subgroup, P, of 'small' index which acts as a p-group on B (the efficacy of p-groups having been established in the trivalent case [Lu1,sec.2]). The size of this index governs the timing of the algorithm. For the purposes of [Lu1] it was sufficient and easy to observe that, for some function f, $[G:P] < m^{f(d)}$ (m = |B| = the number of blocks), for this implies an $n^{c+f(d)}$ bound on color automorphism. A review of the construction of f, citing only elementary group theory and number theory, reveals that $f(d) = O(d^2 \log d)$. We can now show that estimates available only through the classification reduce this to $f(d) = O(d)$. (Note: the latter implies an upper bound of n^{cd} on the original d-valent graph isomorphism test.) Furthermore, digging a little deeper into classification-dependent results, one can show that the index of P is actually bounded by $m^{cd/\log d}$ in all but 'one case'. That special case has sufficient combinatorial structure to allow one more divide-and-conquer trick.

Lemma 9.3. (CFSG) Let G be a primitive subgroup of Sym(B), |B|=m, with G in Γ_d. Then either

(i) G has a p-subgroup, P, of index $< m^{2d/\log d}$ or

(ii) the socle, N, of G is isomorphic to a direct product of t copies of Alt(D), for some t,D (with $|D| \leq d$) Moreover, with this identification, there is an N-isomorphism $B \to \binom{D}{k}^t$.

The proof of Lemma 9.3 involves analyses of each of the cases of the O'Nan-Scott Theorem (as in [Ca1], but taking into consideration number-theoretic consequences of the restriction to groups). (We conjecture that a much smaller bound is possible in case (i); see 11.3). For algorithmic purposes, it is necessary to construct the aforementioned subgroups and isomorphisms. The p-subgroup was constructed in [Lu1] in time [G:P]m and, as mentioned, the problem was decomposed into [G:P] problems for P. We now avoid this route for large [G:P], decomposing instead into [G:N] problems for N. One can find N in polynomial time (Proposition 5.4) Again, number-theoretic estimates yield

Lemma 9.4. Let G,N be as in case(ii) in Lemma 9.3. Then $[G:N] < m^{c \log d}$.

Thus, it is not particularly costly to pass to N. Now, although N is still transitive on B, it has, by virtue of the observed bijection $B \to \binom{D}{k}^t$, a particularly transparent structure. Furthermore, this structure is available. Extending Proposition 5.3, the following holds.

Lemma 9.5. Let N be as in Lemma 9.3(ii). A suitable set D and N-isomorphism $B \to \binom{D}{k}^t$ is constructible in polynomial time.

The additional divide-and-conquer exploits this identification. We describe the idea in the case k=t=1 and then outline the extension.

Divide the set B (≅ D) in half arbitrarily, $B = B1 \cup B2$. Let N1 be the stabilizer in N of B1. Then the number of steps needed to find generators for N1 is bounded by a polynomial times [N:N1]. Note that $[N:N1] < 2^{|B|}$. We have again restored intransitivity and we exploit the fact, treating the orbits (which are, at most, half as large as the original set) serially. In focusing on an orbit, once again we split the set in half, etc., continuing until the target subset has cardinality 1. The end result is the replacement of a problem on a union of a set of m (= $|B| \leq d$) equal-sized blocks by less than 4^m ($\leq m^{2d/\log d}$) problems on the individual blocks.

For general k,t we make use of the naturally induced action of $N \cong Alt(D)^t$ on a set, C, formed by the disjoint union of t copies of D. Each element of $\binom{D}{k}$ is a subset of D and so the bijection

of of Lemma 9.5 induces a G-map Ψ:B -> {subsets of C}. As in the special case, the goal is to cut the group down so as to shrink orbits, zeroing in on a single element of B. Simply halving B appears, in the general case, inefficient. Instead, we look to the N-orbits in C (in the first round, with the full group N, these are just the copies of D). For purposes of recursion, we extract some essential aspects of the present problem. The present group N acts transitively on a collection, B, of blocks (the underlying target set, X, is the union of these blocks); N acts also on an auxiliary set, C, and there is an N-injection, Ψ, from B to the set of subsets of C. We proceed as follows. Fix x in B and define f: {N-orbits in C} -> {natural numbers} by $f(C') = |C' \cap \Psi(a)|$ (since N is transitive on B, f is independent of the choice of x). As long as $|B| > 1$, there must be some N-orbit, C', such that $0 < f(C') < |C'|$. We choose any such orbit, halve it, pass to cosets of the stabilizer of the halves then treat N-orbits (for the new N) in B serially. One shows, inductively, that the effect of this recursive procedure is to replace the problem for N on B by less than $|B|\prod(4f(C'))^{|C'|}$ problems on individual blocks in B, where the product is taken over all orbits, C', for which $0 < f(C') < |C'|$. Thus, the initial N,B problem leads to less than $(4k)^{\tilde{d}t}$ problems on the m blocks, where $\tilde{d} = |D| \leq d$. Finally, one shows that $(4k)^{\tilde{d}t} \leq m^{2\tilde{d}/\log \tilde{d}}$ for $\tilde{d} > 6$ ($m = (\tilde{d}_k)^t$ and we assume $k \leq \tilde{d}/2$). Hence, the problem on X involves less than $m^{2d/\log d + c}$ problems on sets of size $|X|/m$.##

We remark, finally that the above techniques are incorporable, also, into the INT(Γ_d) algorithm of [Lu1,p.61], resulting in

Theorem 9.6 (CFSG) Given subgroups G,H and elements g,h of Sym(X) with G in Γ_d the intersection of the cosets Gg and Hh can be found in $n^{cd/\log d}$

Details will appear in [Lu4].

10. Coset intersection

In the previous section we gave an outline of an INT(K) algorithm for certain classes K of groups. The methods described there do not appear to yield a general INT algorithm faster than c^n.

In this section we give a moderately exponential algorithm (cf. 4.3) for INT(K) where K comprises all finite groups and will therefore be omitted. First we have to treat CAUT, an important subcase of INT, separately.

Let G be a transitive permutation group acting on a set X. Let B={A1,...,At} be a system of blocks of imprimitivity for G with block size b ; bt=n. Let L(i) denote the restriction to Ai of the (setwise) stabilizer of Ai. These groups are isomorphic as permutation groups. There is a natural action f: G->Sym(B). Let K denote the kernel and R the image of f. Thus, R is a subgroup of Sym(B) and K is a subgroup of the direct product L(1) x ... x L(t); R≅G/K.

The situation where the methods of [Lu1] don't immediately give a moderately exponential CAUT algorithm is characterized by the property that b is small ($n^{\quad}$) and R contains the alternating group Alt(B). We begin with the structure theorem that enables us to handle this case. We refer to [Ba6] for the proofs of 10.2 and 10.3.

Definition 10.1. A subgroup H of G is a strong complement to K if
(i) HK = G ;
(ii) for any h in H and any block A in B, if A is invariant under h then it is fixed pointwise.

Remark. (ii) clearly implies that H∩K = {1} and thus H ≅ R.

Lemma 10.2. If R = Alt(B) and $t > 4b$, then K has a strong complement H. Such an H can be constructed in polynomial time.

Note that a strong complement does not necessarily exist if R = Sym(B), even if t is as large as n/2.

Before stating the structure theorem, we need some more notation. Let G be as above. Assume a strong complement H of K has been selected. Then H defines a unique bijection between each pair of blocks. This permits us to identify X with the product set A x T for some set A, where T={1,...,t}, A x {i}=Ai and the sets {a} x T (a∈A) are the orbits of H. For a permutation g∈Sym(A), let g(i) denote the corresponding permutation of Ai; and for a group F≤Sym(A), let F(i)={f(i):f∈F}. Note that our groups L(i) do arise from some L≤Sym(A) in this way. For a subset S of T, the direct power F^S will mean the product of {F(i):i∈S}, acting on ∪{Ai:i∈S}. Now we can say, that K is a subgroup of the direct power L^T.

By the restriction of the group G to a subset Y of X we mean the restriction to Y of the setwise stabilizer of Y in G.

Let $\tilde{G}$ and $\tilde{K}$ denote the restrictions of G and K, resp., to X-A1.

Theorem 10.3. Let G be as above and let S=T-{1}. Assume R contains Alt(B) and a strong complement H of K is given. Then L has a normal subgroup M such that
(i) M^S is a normal subgroup of $\tilde{G}$, and
(ii) $\tilde{K}/M^S$ is isomorphic to a subgroup of L/M.
M can be constructed in polynomial time.

We shall use INT(G,H) to denote the coset intersection problem restricted to those cases where the input groups are G and H. The following two tricks will be used frequently.

Lemma 10.4. Climbing Down. If K is a subgroup of index k of G then an instance of INT(G,H) reduces to k instances of INT(K,H).(We climb down to K.)

Lemma 10.5. Climbing Up. If G is a subgroup of index k of K then an instance of INT(G,H) reduces to one instance of INT(K,H) plus a number of steps, polynomial in n and k.(We climb up to K.)

Proof. 10.4 follows by representing a coset of G as a union of k cosets of K. As for 10.5, let the instance be the determination of the intersection of the subcosets Gg and Hh of Sym(X). Find Kg∩Hh; this is either empty or of the form Dd where D=K∩H. Find C=G∩H, a recognizable subgroup of index at most k in D, and represent D as CR where R is a set of (right) coset representatives of C in D. This can be done in time, polynomial in n and k [FHL]. If rd Gg for some r∈R then Gg∩Hh=Crd. Check this for each r. If every r is rejected then Gg∩Hh is empty.##

We start with an important subcase, involving wreath products.

Definition 10.6. Let P≤Sym(A), Q≤Sym(T). Let us think of Q as acting on the Cartesian product A x T by permuting the second components. The wreath product (cf.[Ro]) P wr Q is defined as P^TQ, acting on A x T.(Both P^T and Q act on A x T.)

Lemma 10.7. If C,D≤Sym(A), G = C wr Sym(T), H = D wr Sym(T), g,h Sym(X) where X = A x T then the determination of Gg∩Hh reduces to t^2 instances of INT(C,D).

Outline of the proof. This situation is analogous to the reduction of the isomorphism problem of disconnected graphs to pairwise comparisons of

their connected components. If $gh^{-1} \notin Sym(A)$ wr Sym(T) then $Gg \cap Hh$ is empty. So we may assume g=1, $h \in Sym(A)$ wr Sym(T). We shall use the language of b-ary relations (subsets of the b-th Cartesian power, where b=|A|). Let w be the list of elements of A x {1} in any order, and let $R(1) = w^G$, $R(2) = w^{Gg}$, $S(1) = w^H$, $S(2) = w^{Hh}$. It is easy to see that G is the automorphism group of the b-ary relational structure (X;R(1)) and therefore Gg is the set of isomorphisms between (X;R(1)) and (X;R(2)) and similarly for Hh. It follows that $Gg \cap Hh$ is the set of isomorphisms between the structures (X;R(1),S(1)) and (X;R(2),S(2)). Both structures are disconnected and their connected components share the same underlying sets A1,...,At. A pairwise comparison of two such components is an instance of INT(C,D).##

The main procedure has two phases. In the first phase, we solve the subcase CAUT(G). The input is a subcoset Gg of Sym(X) and a coloring f of X.

Procedure CAUT(G)
1. If G is intransitive, divide and conquer.
2. Find a minimal block system B. (We shall use the notation of Lemma 10.2.)
3. If t<4b or R does not contain Alt(B) then climb down to K.
4. If R=Sym(B) then climb down to the inverse image of Alt(B) under the homomorphism G->R.
5. (Now R=Alt(B) and t>4b, so 10.2 applies.) Find a strong complement H to K. Find M as in Theorem 10.3.
6. Climb down to the setwise stabilizer of A1 in G.(This group has two orbits: A1 and X-A1.)
7. (Divide and conquer; first solve on X-A1. This is an instance of CAUT($\tilde{G}$).) Let $\tilde{H}$ be the restriction of H to X-A1 and $H^* = \tilde{H}\tilde{M}^S$ (S=T-{1}). Climb down to H^*.

Comment. The index of H^* in $\tilde{G}$ is not more than |L/M|, by Theorem 10.3, and $H^* = M$ wr Alt(S).
8. Climb up to M wr Sym(S).
9. Use Lemma 10.7 to finish on X-A1.
10. Use brute force on A1.

Timing. Let $m = [2\sqrt{n}]$. Let p(n) denote the maximum order of primitive permutation groups of degree n, not containing the alternating group A_m. Set p(n)=1 if no such groups exist. Let $q(n) = \max\{m!, p(1), \ldots, p(n)\}$. It is easy to see that the running time will be less than q(cn). By [Ba2] and [Ba4],

$$q(n) < \exp(4\sqrt{n}\log^2 n)$$

for large n. This estimate does not require CFSG. Using CFSG one obtains

$$q(n) < n^{\sqrt{n}} \quad \text{(for large n)}$$

[Ca1]. Moreover, it follows from CFSG that in the bottleneck case, R has to be essentially an induced alternating group. More precisely, Lemma 9.3(ii) must hold for R. This well defined structure raises the hope of a possible improvement.

Now we turn to the actual INT algorithm. The input is a pair of subcosets Gi.g(i) (i=1,2) of Sym(X).

Procedure INT(G1,G2)
1. Use CAUT(Gi) to reduce to the case when the orbits of G1 and G2 coincide. Then, by divide-and-conquer, reduce to the case when both groups are transitive.
2. Find minimal block systems Bi for Gi (i=1,2). (We shall use the notation of Lemma 10.2 for both groups, augmenting each symbol to indicate the corresponding group.)
3. If t(i)<4b(i) or Ri does not contain Alt(Bi) then climb down to K(i) (i=1,2).
4. (Now Ri>Alt(Bi) and t(i)>4b(i). Our goal is to align B1 and B2.) View B1 and B2 as equivalence relations. Consider the graphs W=(X;B1,B2) and $W' = (X; B1^{g(1)}, B2^{g(2)})$. (The edges of these graphs have colors 1,2 or both. Isomorphisms preserve colors by definition. Clearly, $G1g(1) \cap G2g(2) \le ISO(W,W')$ where the right side denotes the set of W->W' isomorphisms.)
Use CAUT(Gi) to reduce Gi.g(i) unless every component of W and W' has equal size.
5. (Now each component of W and W' has the same size w. These components are unions of blocks of Gi.)
If b(i)<w<n then reduce Gi.g(i) to the subcoset sending components of W to components of W'. (This can be done in linear time by adjusting g(i) and reducing Ri.)
6. If b(1)=b(2)=w (i.e. B1=B2 and $B1^{g(1)} = B2^{g(2)}$) then perform steps 4 to 8 of CAUT(Gi) (i=1,2). (These steps of CAUT(Gi) do not refer to the coloring f.) Use Lemma 10.7 to finish on X-A1 (A1=A11=A12).
Use brute force on A1.
7. If w=n (i.e. W, W' are connected) then choose $x \in X$ and climb down to Gi . Let us define the product relations Pj by setting P1=B1; P(j+1)=PjB2 if j is odd and PjB1 if j is even. Find j such that x has valence between $\sqrt{n}$ and n/2 in the graph (X,Rj). (Such j exists because $b(i) < \sqrt{n}/2$.) Let Pj' be the corresponding relation defined by W'. (Clearly, the intersection of our cosets sends Pj to Pj'.)
Let i=j mod 2. Reduce Gi.g(i) to its subcoset sending xPj to x'Pj' where xPj is the set of Pj-neighbors of x, and $x' = x^{g(i)}$. (This takes linear time.)

Comment. This takes us back to the case of intransitive Gi. Observe, however, that the orbits of Gi have lengths $< n-\sqrt{n}$. Hence this step will not be repeated more than $O(\sqrt{n})$ times. It costs a factor of n each time, because of climbing down to the stabilizer of x.

Timing. Similarly to CAUT, the number of steps is bounded by q(cn). The effect of step 7 is only a factor of $n^{\sqrt{n}}$ (see the Comment). We thus conclude:

Theorem 10.8. The intersection of two subcosets of Sym(X) is computable in
(i) $\exp(c\sqrt{n}\log^2 n)$ steps;
(ii)(CFSG) $n^{c\sqrt{n}}$ steps.

Combining this result with Theorem 8.1 we obtain

Corollary 10.9. (CFSG) SYLFIND and SYLCONJ are solvable for general groups in $n^{c\sqrt{n}}$ steps.

11. Problems and comments.

11.1. In Section 5 we indicated how to find, in polynomial time, the non-abelian part of the socle as well as how to test for the non-triviality of the abelian part. In [Ka3] it is shown that the algorithm for Proposition 5.5 can be extended to produce, in polynomial-time a normal p-subgroup (if one exists) for any given p. However, the complexity of the following related problems remain open:
(i) Find the abelian part of the socle.
(ii) Find a minimal normal p-subgroup.
(iii) Find a maximal normal p-subgroup. (Surprisingly, this is reducible to (ii) [Ka3]).
(iv) Find the maximal solvable normal subgroup.

11.2 Investigations of 11.1(ii) lead naturally to matrix group problems for we may assume we already have an elementary abelian normal p-subgroup, whence the containing group acts on it via linear transformations. Though we suspect 11.1(ii) is actually easier than the following, we wonder about the complexity of
(i) Find an irreducible subspace of a linear group over a finite field (given again by generators). In fact, test irreducibility.

This problem and 11.1 inspire

(ii) Find the radical of a matrix ring over a finite field.

11.3 The estimate $m^{cd/\log d}$ in Lemma 9.3(i) comes from bounding the product of all prime factors $\leq d$ in $(p-1)(p^2-1)\ldots(p^d-1)$ where p is prime and $m=p^d$ (see [Lu1,3.2]). It seems likely, however, that a smaller bound can be established ($m^{c \log^2 d}$?). Note, that this would still leave one $n^{cd/\log d}$ bottleneck in d-valent graph isomorphism, but that corresponds to case(ii) in Lemma 9.3 where we already know a lot about the group.

11.4 We pointed out (section 2) that the following problem is in $NP \cap coNP$.

Let p^r be the highest p-power dividing $|G|$. Does G have an element of order p^r ?
Is there a subexponential deterministic algorithm? Suppose r is not maximal. Is the problem still in $NP \cap coNP$? Another problem in $NP \cap coNP$ is testing whether the Sylow p-subgroups are abelian. Can this be done subexponentially?

11.5 An oracle for SYLCONJ does not, in itself, seem to offer much help in finding all conjugating elements (if P(1) = P(2), this is the normalizer of P(1)). This appears unusual for permutation group problems. Are Sylow normalizers computable, say, given an oracle for INT?

11.6 Problems put in P only through CFSG are candidates for future investigation. Is there a more elementary approach to finding a p-element? The simplicity test seems awfully close to avoiding CFSG. Can one bypass Schreier's conjecture?

11.7 An essential tool in studying permutation groups is the use of representations for quotient groups, G/N. These are available if N is maximal normal (Theorem 5,2) Can one find a 'small' representation for G/N in general? The algorithmic interpretation of this question clearly must be preceded by the theoretical one. This, we editorialize, is typical of what is now going on in this subject. Complexity concerns have motivated new theoretical problems as well as new perspectives on old tools.

References

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

[AS] M.Aschbacher, L.L.Scott, to appear

[Ba1] L.Babai, Monte-Carlo algorithms in graph isomorphism testing, preprint, Univ. Montreal 1979

[Ba2] L.Babai, On the order of uniprimitive permutation groups, Annals of Math. 113 (1981), 553-568

[Ba3] L.Babai, Moderately exponential bound for graph isomorphism, Proc. Conf. FCT'81, Szeged (Hungary), Springer Lect. Notes in Comp. Sci. 117 (1981), 34-50

[Ba4] L.Babai, On the order of doubly transitive permutation groups, Invent. Math. 65 (1982), 473-484

[Ba5] L.Babai, On the length of subgroup chains in the symmetric group, in preparation

[Ba6] L.Babai, Permutation group intersection in $n^{c\sqrt{n}}$ time, in prep.

[BCP] L.Babai, P.J.Cameron, P.P.Pálfy, On the orders of primitive groups with restricted nonabelian composition factors, J.Alg. 79(1982), 161-168

[BKL] L.Babai, P.Klingsberg, E.M.Luks, Distributive lattice isomorphism is in polynomial time, to appear in Proc. S-E.Conf. Comb. Graph Thy, Comp.,1983

[BL] L.Babai, E.M.Luks, Canonical labeling of graphs, Proc. 15th ACM Symp.Thy. of Comp., Boston 1983, 171-183

[Ca1] P.J.Cameron, Finite permutation groups and finite simple groups, Bull. London Math. Soc. 13 (1981), 1-22

[Ca2] P.J.Cameron, There are only finitely many finite distance transitive graphs of given valency greater than two, Combinatorica 2(1982), 9-13

[CPSS] P.J.Cameron, C.E.Praeger, J.Saxl, G.M.Seitz, The Sims conjecture for primitive permutation groups, Bull.London Math.Soc., in print

[CHL] G.Cherlin, L.Harrington, A.Lachlan, $\aleph_0$ categorical $\aleph_0$ stable structures, submitted

[CL] G.Cherlin, A.Lachlan, Stable finitely homogeneous structures, submitted

[Co] J.H.Conway, Monsters and moonshine, Math. Intelligencer 2(1979/80) 164-171

[Di] J.D.Dixon, Asymptotical fast factorization of integers, Math. Comp. 36(1981),255-260

[Fe] W.Feit, Some consequences of the classification of the finite simple groups, Proc. Symp. Pure Math. 37 (1980), 175-181

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

[Go] D.Gorenstein, Finite Simple Groups: An Introduction to their Classification, Plenum, N.Y. 1982

[Ho] C.M.Hoffmann, Group-Theoretic Algorithms and Graph Isomorphism, Springer Lect. Notes in Comp. Sci. 136, 1982

[Ka1] W.M.Kantor, Permutation representations of the finite classical groups of small degree or rank, J. Algebra 60 (1979), 158-168

[Ka2] W.M.Kantor, Some consequences of the classification of finite simple groups, Proc. Montreal Monster Conf. 1982, to appear

[Ka3] W.M.Kantor, Polynomial time algorithms for finding elements of prime order and Sylow subgroups, submitted

[KT] W.M.Kantor, D.E.Taylor, Polynomial time versions of Sylow's theorem, in preparation

[Lu1] E.M.Luks, Isomorphism of graphs of bounded valence can be tested in polynomial time, J.C.S.S. 25 (1982), 42-65

[Lu2] E.M.Luks, The complexity of permutaion group problems, in preparation

[Lu3] E.M.Luks, Testing simplicity of permutation groups, in preparation

[Lu4] E.M.Luks, On the complexity of fixed valence graph isomorphism and the implications for general graph isomorphism, in preparation

[Mi] G.L.Miller, On the $n^{\log n}$ isomorphism technique, Proc. 10th ACM Symp. Thy. Comp.(1978), 51-58

[Pál] P.P.Pálfy, A polynomial bound for the orders of primitive solvable groups, J. Alg. 78 (1982), 127-137

[Pá2] P.P.Pálfy, Applications of group theory in group theory and universal algebra, Ph.D.Thesis, Budapest 1982 (in Hungarian)

[Ro] J.J.Rotman, The theory of groups,An Introduction, Allyn and Bacon, Boston 1973

[Sch] C.P.Schnorr, Refined analysis and improvements on some factoring algorithms, J.Algorithms 3(1982), 101-127

[Si] C.C.Sims, Some group-theoretic algorithms, Springer Lect. Notes in Math. 697 (1978), 108-124

[St] R. Steinberg, Regular elements of semisimple algebraic groups, Publ. Math. I.H.E.S. 25 (1965), 281-312

[We] R.M.Weiss, The nonexistence of 8-transitive graphs, Combinatorica 1 (1981), 309-311

[Wi] H.Wielandt, Finite permutation groups, Acad. Press 1964

[ZKT] V.N.Zemlyachenko, N. Kornienko, R.I.Tyshkevich, Graph isomorphism problem (in Russian), The Theory of Computation I, Notes Sci. Sem. LOMI 118, Leningrad 1982