Constructive membership testing in black-box groups

P.E. Holmes, S.A. Linton, E.A. O'Brien, A.J.E. Ryba and R.A. Wilson

Abstract

We describe a method for constructive membership testing in black-box groups, using involution centralisers. If the group is of Lie type in odd characteristic, then the resulting Monte Carlo algorithm runs in polynomial time. The analysis of the algorithm raises interesting questions about the frequencies of certain configurations of elements in various classes of simple groups. Our implementation outperforms competing algorithms for important classes of examples.

1 Introduction

A vital component of many group-theoretic algorithms is an efficient solution of the *factorisation problem* which may be defined as follows: given an element g of a group $G = \langle X \rangle$, express g as a word in X.

We present a new algorithm to reduce this problem for a group G to the same problem in proper subgroups of G. In fact, we consider a slightly more general problem. We assume that we are given a group G and a subgroup H of G. Our algorithm reduces the problem of testing whether an arbitrary $g \in G$ is a member of H to instances of the same problem for $C_H(t)$ for three involutions $t \in H$. The algorithm is constructive, in the sense that if $g \in H$ then it returns a word for g in the generators of H.

Our algorithm addresses the factorisation problem for the class of *black box groups with an order oracle* [3]. In this model, group elements are represented by bit-strings of uniform length; the only group operations permissible are multiplication, inversion, and checking for equality with the identity element. If the order of an element can be computed in polynomial time, we say the group has an order oracle.

Permutation groups, groups of words with a confluent rewriting system, and matrix groups defined over finite fields are all covered by this model. In the case of matrix groups, an efficient order oracle is provided by the algorithm of Celler and Leedham-Green [11].

Well-analysed highly effective algorithms already solve the factorisation problem in two major classes of groups. Membership in a permutation group can be decided constructively by constructing *a base and strong generating set* (BSGS), a concept introduced by Sims [25]. Similarly

membership can be decided constructively for the class of soluble groups by constructing polycyclic presentations; see Sims [26].

In an attempt to solve the factorisation problem for matrix groups, Butler and Cannon generalised the notion of a base and strong generating set to such groups. Methods of constructing such a BSGS include the random Schreier–Sims algorithm of Leon [19] and the Todd–Coxeter Schreier–Sims algorithm of Butler and Cannon [8]. The resulting chain of subgroups—each a stabiliser of a suitable base point—allows the solution of the factorisation problem. Naturally, the BSGS technique is limited to those groups in which a suitable stabiliser chain can be found.

The *composition tree* approach and other algorithms developed in the matrix group recognition project [18] can be used to reduce the factorisation problem for arbitrary matrix groups to the case of non-abelian simple groups. Effective algorithms for the constructive recognition of the alternating groups have also been developed [6].

We therefore develop our algorithm mainly for the larger sporadic groups and for simple groups of Lie type, where neither BSGS techniques nor composition trees are effective, and focus our heuristics and complexity analysis on these groups. The algorithm works well for the sporadic groups, and for groups of Lie type over fields of odd order, where involutions and their centralisers are easy to construct. Its performance is poorer for groups defined over small fields of characteristic 2; for larger fields of even size, it fails completely, since it is difficult by random selection to find involutions in such groups.

The original application of the algorithm was as one step in the classification of conjugacy classes of subgroups of $E_7(5)$ isomorphic to the Rudvalis sporadic simple group [17]. An alternative factorisation algorithm which uses subset chains to solve the factorisation problem for black-box groups is under development by Ambrose *et al* [1].

The structure of the paper is as follows. In Section 2 we present our algorithm for reducing the factorisation problem to involution centralisers. In Section 3 we discuss the construction of the centraliser of an involution, which is an essential subroutine of our algorithm, and some of the interesting theoretical questions raised by it. In Section 4 we discuss how to find the required involutions, and how to direct the choice of involutions to produce the most useful centralisers. In Section 5 we discuss practical aspects of the choice of involutions in various representations. In Section 6 we discuss several approaches to solving the factorisation problem in an involution centraliser. In Section 7 we consider the complexity of the resulting algorithm. Finally, we report on the performance of our implementations.

2 The algorithm

Assume we are given a black-box group G with an order oracle, an element g of G, and a subgroup H of G. The following algorithm tests whether or not $g \in H$.

1. Find $h \in H$ such that $|gh| = 2\ell$. Now define $z = (gh)^\ell$.

2. Find x, an H-involution, such that $|xz| = 2m$. Now define $y = (xz)^m$.
3. Construct $X = C_H(x)$ and decide if $y \in X$.
4. If so, construct $Y = C_H(y)$ and decide if $z \in Y$.
5. If so, construct $Z = C_H(z)$ and decide if $gh \in Z$.

Note that $\langle x, z\rangle$ is D_{2m} having central involution $y = (xz)^m$. Hence y is in the centraliser of x and z is in the centraliser of y.

To see that the algorithm is constructive, observe that after step 1, we know a word for h in the generators of H. After step 2 we similarly know a word for x. In step 3 we can record words for the generators of X, and so the recursive call will give us a word for y. Similarly, in step 4 we can record words for the generators of Y and so obtain a word for z. Finally in step 5 we can record words for the generators of Z and so find a word for gh and hence a word for g.

The order algorithm of Celler and Leedham-Green [11] assumes that we can factorise certain large integers, which has consequences both for the practical performance and complexity of this algorithm. We can avoid these factorisations, using instead the *pseudo-order* of an element: a multiple of its order.

If g is an involution we can take h to be 1_G so that $z = g$ and so we avoid step 5. Observe that both x and y are involutions, so this remark applies to the subproblems solved at steps 3 and 4.

We anticipate that the algorithm will usually be applied in cases where we are confident that $G = H$. We may wish to prove this by expressing our generators for G as words in the generators for H, or we may have other elements of G which we need to factorise for other reasons.

In summary, there are three essential components of the algorithm:

1. finding suitable involutions;
2. given an involution, constructing its centraliser;
3. solving the factorisation problem in this centraliser.

We focus first on the problem of constructing a centraliser.

3 Constructing an involution centraliser

The centraliser of an involution in a black-box group having an order oracle can be constructed using an algorithm of Bray [9]. The generators of the centraliser are constructed from the generators of the group and the involution itself, making use of the following result.

Theorem 1 *If x is an involution in H, and w is an arbitrary element of H, then $[x, w]$ either has odd order $2k + 1$, in which case $w[x, w]^k$ commutes with x, or has even order $2k$, in which case both $[x, w]^k$ and $[x, w^{-1}]^k$ commute with x.*

Proof. In the first case $xw[x, w]^k = wx[x, w]^{k+1} = wx[x, w]^{-k} = w[x, w]^k x$ since x is an involution; in the second case $x[x, w^{\pm 1}]^k = x[x, w^{\pm 1}]^{-k} = [x, w^{\pm 1}]^{-k} x$. □

This theorem is used to convert a supply of independent nearly uniformly distributed random elements of H into a supply of elements of $C_H(x)$. While these are not, in general, nearly uniformly-distributed, we have the following result:

Theorem 2 *With the above notation, if w is uniformly distributed among the elements of the group for which $[x, w]$ has odd order, then $w[x, w]^k$ is uniformly distributed among the elements of the centraliser of x.*

Proof. If $w' = yw$, where $y \in C_H(x)$, then $[x, w'] = [x, w]$ so that $w'[x, w']^k = yw[x, w]^k$; so each element of $C_H(x)$ occurs exactly once as w runs through any coset. □

Thus if the odd order case occurs sufficiently often (with probability at least a positive rational function of the input size), then we can construct nearly-uniformly distributed random elements of the involution centraliser in polynomial time.

Our approach to constructing the centraliser of a given involution is to use Theorem 1 to obtain a supply of elements of the centraliser and to use a variety of heuristic methods, depending on our knowledge of the group H and our ability to analyse the group generated by the centraliser elements obtained so far, to decide when they generate the complete centraliser. We call this process *Bray's algorithm*.

Problems may arise in applying Bray's algorithm to construct involution centralisers in arbitrary groups. If $[z, w]$ almost always has even order, then we have limited control over the distribution of the centraliser elements that we create, and may not in practice be able to generate the complete involution centraliser. In particular it may happen that $C_G(z)$ is not generated by involutions; in this case we have no chance of success unless we find enough cases with $[x, w]$ of odd order.

In this application, we will usually apply the algorithm to simple groups. If H is known to be simple, then we conjecture that the probability of $[x, w]$ having odd order is not too small. More precisely, we formulate the following:

Conjecture 3 *If G is a finite simple group and x is an involution in G, then $[x, g]$ has odd order for at least a proportion $1/\log_2 |G|$ of the elements $g \in G$.*

Of course, in each sporadic group we can calculate explicitly the proportion of $[x, g]$ which have odd order. For every class of involutions x this proportion is always greater than 17%, and therefore the conjecture is true in this case, and Bray's algorithm completes rapidly.

For the groups of Lie type we content ourselves with the following slightly weaker conjecture, which nevertheless is sufficient to ensure that this step of our algorithm runs in Monte Carlo polynomial time.

Conjecture 4 *There exist constants $c_1, c_2 > 0$ such that if G is a finite simple group, of Lie type and rank r, and x is an involution in G then $[x, g]$ has odd order for at least a proportion c_1/r^{c_2} of the elements $g \in G$.*

We believe that we can take $c_2 = 2$. It may be possible to take $c_2 = 1$.

Parker and Wilson [24] prove the following version of the conjecture for classical groups defined over fields of odd characteristic.

Theorem 5 *There is an absolute constant c such that if G is a finite simple classical group, with natural module of dimension d over a field of odd order, and x is an involution in G, then $[x, g]$ has odd order for at least a proportion c/d of the elements $g \in G$.*

The exceptional groups require individual treatment, but the rank may now be treated as a constant, as may the order of the Weyl group. The essence of the proof is to find a suitable class of dihedral subgroup of twice odd order, and use a dimension-counting argument to show that there are enough involution pairs in these groups.

Theorem 6 *Conjecture 4 holds if G is an exceptional group over a field of odd order.*

Proof. First we consider the cases in which the central involution in the Weyl group fuses into the given class of G. It follows that such an involution t inverts every type of maximal torus T in G. We choose the following tori of odd order; in the case of $E_7(q)$, we take $(q^7 \pm 1)/2$ according as $q \equiv 1$ or $3 \bmod 4$.

G	T	$\lvert N(T)/T \rvert$	$\dim G$	$\dim C_G(t)$
${}^2G_2$	$q + \sqrt{3q} + 1$	6	7	3
G_2	$q^2 + q + 1$	6	14	6
${}^3D_4$	$q^4 - q^2 + 1$	4	28	12
F_4	$q^4 - q^2 + 1$	12	52	24
E_7	$(q^7 \pm 1)/2$	14	133	63
E_8	$q^8 - q^4 + 1$	24	248	120

In particular there are some nontrivial odd-order products of two involutions in the given class. We can ignore finitely many values of q, at the expense of possibly having to change the constants in the conjecture. Now for large q, the proportion of pairs of inverting involutions whose product is a regular semisimple element tends to 1. Therefore the number of pairs of involutions accounted for in this way is $\sim q^k/c$, where $k = 2\mathrm{rk}\,(G) + (\dim G - \mathrm{rk}\,(G)) = \dim G + \mathrm{rk}\, G$ and c is a constant, equal to $|N(T)/T|$ in all cases except E_7, where it is $4|N(T)/T|$. On the other hand, the total number of pairs of involutions in this class is $2(\dim G - \dim C_G(t))$. But

$\dim C_G(t)$ is the number of positive roots in these cases, so is $1/2(\dim G - \mathrm{rk}\, G)$, and therefore $2(\dim G - \dim C_G(t)) = k$. Hence the proportion of pairs of involutions whose product is a regular semisimple element in a torus of this type, tends to $1/c$ as q tends to infinity. This proves the theorem in these cases.

Before we treat the remaining cases, note that all we really need to check is that $2\dim T + \mathrm{codim}\, C_G(T) = 2\mathrm{codim}\, C_G(t)$, for a suitable torus T of (nearly) odd order inverted by t.

In $E_8(q)$ consider the subgroup $2{\cdot}(P\Omega_8^-(q) \times P\Omega_8^-(q))$ inside $2{\cdot}P\Omega_{16}^+(q)$. The involutions of type $-1^4 1^4$ in $\Omega_8^-(q)$ lift to involutions in the spin group, and it is straight-forward to calculate the trace of these involutions acting on the Lie algebra: this is 24, and therefore they are involutions of type A_1E_7. In particular $\mathrm{codim}\, C_G(t) = 112$. Also these involutions invert every maximal torus of O_8^-, in particular the cyclic torus of order q^4+1, of twice odd order. Finally, the centraliser of T is essentially $T \times D_4$ so has dimension $4 + 28 = 32$ and codimension 216. So $2\dim T + \mathrm{codim}\, C_G(T) = 8 + 216 = 224 = 2 \times 112$ as required.

In $E_7(q)$ we look inside $(SL_2(q) \circ 2\Omega_{12}^+(q).2).2$ at the involutions of type $(-1,1) \otimes (-1^2 1^{10})$ and calculate their trace on the Lie algebra to be 25. Thus $\mathrm{codim}\, C_G(t) = 54$. Such involutions can simultaneously invert $q \pm 1$ in $SL_2(q)$ and q^2+1 in $O_4^-(q)$ so we obtain a torus T of rank 3 and at most 4 times odd order, with centraliser of type $T \times D_4(q)$. Thus $2\dim T + \mathrm{codim}\, C_G(T) = 6 + 133 - 3 - 28 = 108 = 2 \times 54$ as required.

The other class of involutions in $E_7(q)$ can be dealt with again in $(SL_2(q) \circ 2\Omega_{12}^+(q).2).2$, this time looking at the involutions of type $-1^4 1^8$ in O_{12}^+. These have trace 5 on the Lie algebra so are of type A_1D_6, and have centraliser of codimension $133 - 3 - 66 = 64$ in G. They invert a torus of type q^4+1 (and so of twice odd order) inside $O_8^-(q)$, and therefore this torus has centraliser of type $T \times A_1(q)A_1(q^2)$, and dimension $4 + 3 + 6 = 13$. Finally we calculate $2\dim T + \mathrm{codim}\, C_G(T) = 8 + 133 - 13 = 128 = 2 \times 64$ as required.

In $E_6(q)$ or ${}^2E_6(q)$ we look inside the subgroup $({}^3D_4(q) \times (q^2 \pm q + 1)).3$, and find involutions inverting a torus of shape $q^4 - q^2 + 1$ (and hence odd order) inside ${}^3D_4(q)$. This torus has centraliser of dimension 6 only, and the involutions have centraliser codimension $78 - 3 - 35 = 40$. Finally, $2\dim T + \mathrm{codim}\, C_G(T) = 8 + 78 - 6 = 80 = 2 \times 40$ as required.

The other class of involutions in $E_6(q)$ or ${}^2E_6(q)$ can be dealt with by looking inside the subgroup of type A_1A_5 at an outer involution of type $(-1,1) \otimes (-1,1^5)$. This involution has trace 14 on the Lie algebra, so is of type T_1D_5. It inverts various tori T of rank 2 with centralisers $T \times GL_4(q)$ of dimension 18, and we have $\mathrm{codim}\, C_G(t) = 78 - 1 - 45 = 32$, so $2\dim T + \mathrm{codim}\, C_G(T) = 4 + 78 - 18 = 64 = 2\mathrm{codim}\, C_G(t)$ as required.

Finally consider the involutions of type B_4 in $F_4(q)$. First note that the pre-images in $2{\cdot}\Omega_9(q)$ of negatives of reflections in vectors of plus type lift to involutions in this class. They invert tori T of dimension 1, centralising $T \circ 2{\cdot}\Omega_7(q)$. In particular $\mathrm{codim}\, C_G(T) = 52 - 1 - 21 = 30$ and $\mathrm{codim}\, C_G(t) = 52 - 36 = 16$, so $2\dim T + \mathrm{codim}\, C_G(T) = 2 + 30 = 2 \times 16$ as required. We can choose T to have order $q \pm 1$, so of twice odd order. $\square$

Remark. For groups of Lie type in characteristic 2, "most" elements are regular semisimple elements, and therefore have odd order. It seems that "most" products of two conjugate involutions are regular semisimple elements of a suitable subgroup, and again have odd order. The case $PSL_n(q)$ can be proved by a modification of the argument in [24] and we expect that this argument can be generalised to the other classical groups.

In practice, producing elements that generate the centralisers of involutions of simple groups by this means is effective, although, as the open questions and conjectures above suggest a rigorous complexity analysis is some way off. On the other hand, this technique does not work well if the involution is in a normal subgroup of large index. It is particularly ineffective if the involution is in a normal 2-subgroup, and the involution centraliser has large index (or even if these properties hold in a quotient group).

The other problem arising is knowing when we have enough elements to generate the required involution centraliser. This is particularly acute when we have no *a priori* knowledge of the structure of the involution centralisers in H. However, in our application, we do not necessarily need the full involution centraliser – we just need to generate enough of the involution centraliser to contain the element that we are looking for. Our implementations used various heuristics to decide when we have constructed enough of the involution centraliser and add additional generators as necessary.

A variation of Theorem 1 allows us to test constructively for conjugacy of involutions in suitable groups. To decide conjugacy of x and y we construct random conjugates x_i of x, until we find $x_i y$ with odd order $2k+1$, say. In the dihedral group $D_{4k+2} = \langle x_i, y \rangle$, we can see that $(yx_i)^k$ conjugates x_i to y. If two random conjugates of x have a high enough probability of having product of odd order, this provides an effective method.

This observation enables us to construct involution centralisers as conjugates of centralisers already constructed, or taken from a database. In practice, if, after a small number of random selections, this method fails to find a product of odd order, we assume that x and y are non-conjugate, and use the original Bray algorithm to construct generators of $C_G(y)$.

4 Finding involutions

The first step of the algorithm constructs an involution by powering up an element of even order.

If our group has enough even order elements, we can choose which involutions to use in the first two steps of the algorithm. This is important because our algorithm reduces the problem being solved to the same problem in the centralisers of these involutions, and it may be more tractable in some centralisers than in others. For example, in the classical groups over fields of odd order, the involutions with the smallest centralisers are those whose eigenspaces on the natural module have dimension roughly half the dimension of the space.

Two choices are involved, one of the element h (and hence the involution z), and the other of the involution x (and hence the involution y). To what extent can we control which conjugacy classes the involutions x, y and z belong to?

Two central questions arise. Are there enough involutions in the desired classes and with the right properties? How do we identify which class an involution belongs to? We consider the first here, and the second in Section 5.

For the first choice (of the involution $z = (gh)^\ell$), we can try several $h \in H$ until we find one with $(gh)^\ell$ in our desired class of involutions. The second choice in the algorithm is of the involution x. This is a little more complicated, since we impose conditions on the two involutions x and $y = (xz)^m$ simultaneously. However, we can first find an involution x_0 in a suitable H-conjugacy class, and then take random conjugates x of x_0 until we find one such that y is in a suitable conjugacy class.

The effectiveness of these choices in practice depends on the proportions of elements gh (resp. xz) which power up to involutions of each class. For the sporadic groups, these proportions are constants which can be calculated from the character tables. In some cases, these proportions are zero: for certain choices of involution classes $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, there are no elements $x \in \mathcal{C}_1$, $z \in \mathcal{C}_2$, with xz powering to an element in $\mathcal{C}_3$. Hence we do not have a completely free choice of these three classes. In practice we normally want to choose x, y and z all to be elements of the largest class of involutions, in which case the probabilities are all positive. Indeed, in this case the probability that gh powers to an element in this class is at least $5/64 = 0.078125$, while the probability that xz powers to an element in this class is at least $6181967/148341375 \approx 0.041674$.

For groups of Lie type in odd characteristic, we have the following result:

Theorem 7 *There exists a constant $c > 0$ such that for every simple group G of Lie type in odd characteristic, of Lie rank r, and every conjugacy class C of involutions of G, the proportion of elements of G having a power in C is at least c/r^2.*

Proof. Our strategy is for each class of involutions to find a suitable class of maximal torus, such that at least some fixed proportion of regular semisimple elements (rss-elements) in that torus power up to an involution in the required class. By choosing a torus T with $|N(T)/T|$ bounded by a constant times r^2 in each case, we prove the result.

Consider first the classical groups, starting with the symplectic groups as they are the easiest to describe. Let $G \cong PSp_{2r}(q)$ with q odd, and let z be an involution in G. Then $C_G(z)$ is the inverse image in G of one of the following subgroups of $\widehat{G} = Sp_{2r}(q)$:

- $Sp_{2k}(q) \times Sp_{2r-2k}(q)$, with $0 < k < r/2$;
- $Sp_r(q) \wr C_2$, if r is even;
- $GL_r(q).2$, if $q \equiv 1 \bmod 4$;
- $GU_r(q).2$, if $q \equiv 3 \bmod 4$.

A Singer cycle in $Sp_{2k}(q)$ has order $q^k + 1$, so in the four cases we can choose maximal tori in $Sp_{2r}(q)$ of shape:

- $C_{q^k+1} \times C_{q^{r-k}+1}$, with $0 < k < r/2$;
- $C_{q^k+1} \times C_{q^k+1}$, with $k = r/2$ when r is even;
- C_{q^r-1};
- C_{q^r+1}.

In the case of the cyclic tori, our involution is the image of an element of order 4 in $Sp_{2r}(q)$, and at least half of the rss-elements in the torus power up to such an element. In the other two cases, we are interested in the powers of 2 which divide $q^k + 1$ and $q^{r-k} + 1$. If these powers are different (for example if r is odd) then at least half the rss-elements power up to our desired involution (modulo ± 1). On the other hand, if both are divisible by the same 2-power, we need to power up an element which has the full 2-power order in one factor but not the other. Again, half the rss-elements have this property.

Finally, in the four cases we have $N(T)/T$ isomorphic to $C_{2k} \times C_{2r-2k}$, $C_r \wr C_2$, C_{2r} and C_{2r}, all of which have order at most $2r^2$ as required. This completes the proof for symplectic groups.

We turn now to orthogonal groups. There are again essentially two kinds of involutions in $P\Omega_n(q)$, those which lift to involutions in $\Omega_n(q)$ and those which lift to elements of order 4. The former have centralisers which are essentially products of two smaller orthogonal groups, while the latter have centralisers which are essentially $GL_r(q).2$ or $GU_r(q).2$, where $n = 2r$. The same argument now works as in the symplectic case, except that orthogonal groups in odd dimensions do not have Singer cycles: in this case we use the Singer cycle in an orthogonal group in one dimension fewer.

Next consider the groups $G \cong PSL_n(q)$. Here the involution centralisers are the images of the following subgroups of $SL_n(q)$:

- $(SL_{2k}(q) \times SL_{n-2k}(q)).C_{q-1}$, for $0 < k < n/2$;
- $(SL_{2k-1}(q) \times SL_{n-2k+1}(q)).C_{q-1}$, for n even, $0 < k \leq n/2$, and $q \equiv 1 \bmod 2n_2$, where n_2 denotes the 2-part of n.
- $(SL_k(q) \times SL_k(q)).C_{q-1}.C_2$, where $n = 2k$ and either k is even or $q \equiv 1 \bmod 4$;
- $SL_k(q^2).C_{q+1}.C_2$, where $n = 2k$ and either $q \equiv 3 \bmod 4$ or n_2 does not divide $q - 1$.

In the last case we choose the Singer cycle, of order $q^n - 1/q - 1$, and note that our desired involution is the image of an element of order $2\gcd(n_2, q - 1)$ in this torus. Therefore at least half of the elements in this torus power up to our desired element.

In the second case, our desired involution is the image of an element of order $2n_2$ in the maximal torus which is a diagonal product of cyclic groups of order $q^{2k-1} - 1$ and $q^{n-2k+1} - 1$, of index $q - 1$ in the direct product. Half the rss-elements in this torus have even order in the quotient

C_{q-1}, so power up to our involution. The same argument applies in the first case when n is odd, and the third case when k is odd.

We are left with the first case when n is even and the third case when k is even. Here the torus is a diagonal product of cyclic groups of orders $q^{2\ell}-1$ and $q^{2m}-1$ where $2(\ell+m)=n$. But now if the 2-parts of ℓ and m are equal, most elements of the torus fail to power up to our desired involution. Therefore we must choose a different maximal torus, such as $C_{q^{2\ell-1}-1} \times C_{q^{2m}-1}$, in which half the rss-elements do now power to our involution.

The groups $PSU_n(q)$ can be treated in the same way.

This leaves us now with only the exceptional groups to consider. See Table 1 for the shapes of involution centralisers and tori that we use; the information about centralisers appears in [14], and about shapes of tori in [16]. In each case it is easy to see that at least a quarter of the rss-elements in the torus power up to our desired involution. In particular, if the torus is cyclic, then at least half its elements power to the involution. In the non-cyclic cases we have chosen tori such that the power of 2 dividing the two factors is either the same or differs by 1. □

Thus for groups of Lie type in odd characteristic we can choose the class of z in polynomial time.

Now we turn to the choice of the involutions x and y. As in the case of the sporadic groups, there are certain combinations of conjugacy classes for x, y and z which are impossible. For example, in $SL_d(q)$ (with d odd) if x and z are involutions whose (-1)-eigenspaces have dimensions k and m respectively, then the (-1)-eigenspace of y clearly cannot have dimension greater than $k+m$. It appears in practice that, subject to some restrictions of this kind, we can also get both x and y in our desired conjugacy classes in polynomial time. However, this depends on certain reasonable assumptions about the distribution of the elements xz in the group, which we have not been able to prove. We formulate the following:

Conjecture 8 *If $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ are three classes of involutions in a group of Lie type over a field of odd order, and Lie rank r, then the probability that random elements $x \in \mathcal{C}_1$, $z \in \mathcal{C}_2$ have $(xz)^k \in \mathcal{C}_3$ for some integer k, is either 0 or at least c/r, for some absolute constant c. Moreover, for each choice of $\mathcal{C}_1$ and $\mathcal{C}_2$ there exists a class $\mathcal{C}_3$ such that this probability is non-zero.*

In practice we do not need as much as this, as we can choose $\mathcal{C}_2$ to be equal to $\mathcal{C}_1$, and both of these can be relatively large classes of involutions. For example, an easy modification of the proof of Theorem 5 proves the following special case of the conjecture:

Theorem 9 *There is an absolute constant $c>0$ such that if (x,y) is a random pair of involutions of type $(-1^{2k},1^{n-2k})$ in $PSL_n(q)$, with q odd, then the probability that xy powers up to an involution of type $(-1^{4k},1^{n-4k})$ or $(-1^{4k-n},1^{2n-4k})$ is at least c/n^2.*

Similarly, we can modify the arguments of Theorem 6 to prove the following:

Table 1: Involution centralisers in simple exceptional groups, q odd

Group	Involution centraliser	Conditions	Torus
${}^3D_4(q)$	$(SL_2(q) \circ SL_2(q^3)).2$		$C_{(q-1)(q^3+1)}$
$G_2(q)$	$(SL_2(q) \circ SL_2(q)).2$		C_{q^2-1}
${}^2G_2(q)$	$C_2 \times PSL_2(q)$		C_{q-1}
$F_4(q)$	$(SL_2(q) \circ Sp_6(q)).2$		$C_{(q-1)(q^3+1)}$
	$2.\Omega_9(q)$		C_{q^4+1}
$E_6(q)$	$(C_{(q-1)/3} \circ 4.\Omega^+_{10}(q)).4$	$q \equiv 1 \bmod 12$	$C_{q-1} \circ_3 C_{q^5-1}$
	$(C_{q-1} \circ 4.\Omega^+_{10}(q)).4$	$q \equiv 5 \bmod 12$	$C_{q-1} \times C_{q^5-1}$
	$(C_{(q-1)/3} \circ 2.\Omega^+_{10}(q)).2$	$q \equiv 7 \bmod 12$	$C_{q-1} \circ_3 C_{q^5-1}$
	$(C_{q-1} \circ 2.\Omega^+_{10}(q)).2$	$q \equiv 11 \bmod 12$	$C_{q-1} \times C_{q^5-1}$
	$2.(PSL_2(q) \times PSL_6(q)).2$	$d = (3, q-1)$	$C_{(q+1)} \circ_d C_{(q^2+q+1)(q^3+1)}$
${}^2E_6(q)$	$(C_{q+1} \circ 2.\Omega^-_{10}(q)).2$	$q \equiv 1 \bmod 12$	$C_{q+1} \times C_{q^5+1}$
	$(C_{q+1} \circ 4.\Omega^-_{10}(q)).4$	$q \equiv 5 \bmod 12$	$C_{q+1} \times C_{q^5+1}$
	$(C_{(q+1)/3} \circ 2.\Omega^-_{10}(q)).2$	$q \equiv 7 \bmod 12$	$C_{q+1} \circ_3 C_{q^5+1}$
	$(C_{(q+1)/3} \circ 4.\Omega^-_{10}(q)).4$	$q \equiv 11 \bmod 12$	$C_{q+1} \circ_3 C_{q^5+1}$
	$2.(PSL_2(q) \times PSU_6(q)).2$	$d = (3, q+1)$	$C_{q+1} \circ_d C_{(q^2+q+1)(q^3+1)}$
$E_7(q)$	$(SL_2(q) \circ 2.\Omega^+_{12}(q).2).2$	$q \equiv 1 \bmod 4$	$C_{q-1} \circ_2 C_{q^6-1}$
	$(SL_2(q) \circ 2.\Omega^+_{12}(q).2).2$	$q \equiv 3 \bmod 4$	$C_{q+1} \circ_2 C_{q^6-1}$
	$2.PSL_8(q).4.2$	$q \equiv 1 \bmod 8$	$C_{(q^7-1)/2}$
	$2 \times PSU_8(q).2.2$	$q \equiv 3 \bmod 8$	$C_{(q^7+1)/2}$
	$2 \times PSL_8(q).2.2$	$q \equiv 5 \bmod 8$	$C_{(q^7-1)/2}$
	$2.PSU_8(q).4.2$	$q \equiv 7 \bmod 8$	$C_{(q^7+1)/2}$
	$(3.E_6(q) \circ C_{(q-1)/2}).S_3$	$q \equiv 1 \bmod 12$	$C_{(q^6+q^3+1)(q-1)/2}$
	$(E_6(q) \circ C_{(q-1)/2}).2$	$q \equiv 5 \bmod 12$	$C_{(q^6+q^3+1)(q-1)/2}$
	$({}^2E_6(q) \circ C_{(q+1)/2}).2$	$q \equiv 7 \bmod 12$	$C_{(q^6-q^3+1)(q+1)/2}$
	$(3.{}^2E_6(q) \circ C_{(q+1)/2}).S_3$	$q \equiv 11 \bmod 12$	$C_{(q^6-q^3+1)(q+1)/2}$
$E_8(q)$	$2.\Omega^+_{16}(q).2$		C_{q^8-1}
	$SL_2(q) \circ 2E_7(q)$		$C_{(q+1)(q^7-1)}$

Note: the notation $\circ_d$ means that the central product in which the subgroups C_d of the two factors are identified. Of course, as abstract groups these central products are isomorphic to direct products of smaller groups, but we use the central product notation to make it clear which elements of the torus are central in the universal group of Lie type.

Theorem 10 *There is an absolute constant $c > 0$ such that if $\mathcal{C}$ is any class of involutions in an exceptional group of Lie type over a field of odd order, then the probability that a random pair of elements from $\mathcal{C}$ has product of even order is at least c.*

We have already noted that there are better algorithms for the alternating groups, so we do not consider these groups further.

Most of our results do not carry over to the remaining class of simple groups: those of Lie type defined over fields of characteristic 2. In particular, Theorems 5 and 6 do not hold. Most elements are now regular semisimple elements, which have odd order, and the proportion of elements of even order in a group of Lie type over the field of size $q = 2^e$ is $O(q^{-1})$. Therefore the complexity of our algorithm in these cases is at least linear in q (since for example we may need to choose q random elements for step 1), and so is exponential in the size of the input.

For small fields of characteristic 2, we can at least obtain elements of even order, and power them up to obtain involutions. But the number of conjugacy classes of involutions increases as the Lie rank increases, and in only a bounded number of classes can we obtain members by constructing powers of random elements of even order. Moreover, the centralisers of involutions in these classes have very large normal subgroups of 2-power order making the factorisation problems in these subgroups intractable by any currently available means. Our algorithm is therefore unsuitable for this class of groups.

5 Identifying the class of an involution

Since our algorithm reduces the factorisation problem in G to three factorisation problems in involution centralisers, it makes sense to try to make these latter problems as easy as possible. The best way to do this is to choose involutions whose centralisers are small. Indeed, if we do not do this, it is possible for the depth of the recursion to be $O(d)$, where d is the dimension of the underlying vector space, and since the number of calls to the factorisation algorithm more than doubles at each level, we obtain an exponential-time algorithm.

In the classical groups, the involutions with the smallest centralisers are those whose eigenspaces on the natural module have dimensions roughly half the dimension of the space. Theorem 7 implies that in step 1 of the algorithm we need at most $O(d^2)$ trials to find an involution in a particular conjugacy class. By allowing a range of dimensions for the eigenspace, we can reduce this to $O(d)$.

Theorem 11 *Let λ be a constant, $0 < \lambda < 1/2$, and let G be a classical group over a field of odd order, with natural representation of dimension d. Then the proportion of elements of G which power to an involution (modulo scalars) whose eigenspaces have dimensions in the range λd to $(1-\lambda)d$ is $O(d^{-1})$.*

Proof. For each eigenspace dimension the proportion is $O(d^{-2})$, and there are $O(d)$ choices for this dimension. □

This raises the question: how do we ensure that our involutions are in these suitable conjugacy classes? Indeed, we really need a quick test to eliminate those which are not.

Since we construct the involutions as powers of other elements, it makes sense to try to identify the class of the involution without explicitly powering. To find involutions in particular classes, if we are working in the natural representation, we calculate the characteristic polynomial of our element gh (resp. xz), and look for cases where this polynomial is a product of two irreducible factors of roughly the same degree. Then with reasonably high probability the element in question powers up to an involution with the required property.

For large degree representations this strategy works well in practice; for small degrees it is not so important to specify the classes of involutions we use.

The other cases we need to consider are classical groups in representations other than the natural representation, and exceptional groups. In all cases we may assume that the characteristic of the representation is the defining characteristic of the group, for otherwise the degree of the representation is at least linear in the field size.

One strategy in these cases is to construct the involution centraliser using Bray's algorithm, and non-constructively recognise the composition factors using the methods of [5] or [22]. However, we can streamline this by exploiting the information we already have, in particular the order of the element which powers up to our involution.

Recall that a *primitive prime divisor* (ppd) is a prime divisor of $p^k - 1$ which does not divide $p^i - 1$ for $i < k$. By [22] ppds exist for all (p, k) except $(2, 6)$ and for $k = 1$ when p is a Mersenne prime, and $k = 2$ when p is a Fermat prime. These small values of k, however, are not relevant to us. A p^k-ppd element is an element of order divisible by a primitive prime divisor of $p^k - 1$.

In $G \cong SL_n(q)$ or $GL_n(q)$ where $q = p^e$, we look for a q^k-ppd element with k close to $n/2$, and power it up to an involution t. Now $H = C_G(t)$ is isomorphic to $(SL_k(q) \times SL_{n-k}(q)).(q - 1)$. In natural characteristic, we can discover the exact value of k by constructing H, restricting its action to the $+1$ or -1 eigenspace of t and non-constructively recognising the almost-simple group which results.

In the other classical groups a similar approach works. For the symplectic and orthogonal groups, we again look for a q^k-ppd element with k close to $n/2$. It then has a good chance of powering up to an involution negating a k-space. For the unitary groups, the calculations are slightly complicated by the fact that the underlying field has order q^2 rather than q. Therefore a q^k-ppd element no longer has to move a k-space: it could move just a $k/2$-space. In any case, it has a reasonable chance to power up to an involution that negates either a k-space or a $k/2$-space in the natural representation. If necessary, we can determine which, by finding its centraliser and then performing non-constructive recognition.

6 Working in the involution centralisers

Applying the algorithm requires three membership tests for elements in involution centralisers. A critical component in its success is the method of solving these subproblems. This choice may depend on the representation of the group. If the group is a matrix group, we can use the BSGS approach mentioned in Section 1, but this technique is limited to those groups in which a suitable stabiliser chain can be found. If the group is very small, simple algorithms can be applied. We discuss three other possible approaches and comment on their effectiveness.

6.1 The recursive approach

The conceptually simplest approach is to apply the algorithm recursively to each centraliser in turn as long as possible and then rely on simple methods.

One consequence is that we have to apply our algorithm to arbitrary iterated involution centralisers in our original group G, which may have a very wide range of structures. Many of our refinements to the algorithm do not apply in this more general context. For example, we know nothing about the structure of the arbitrary centralisers, so we cannot choose "nice" involutions within these.

To understand the limits of this approach, we consider the situations in which the algorithm may fail, and discuss how some of them can be addressed.

Firstly, we make no progress if any of the involutions x, y or z is central in H. Therefore we need first to reduce to $H/Z(H)$, then solve the problem in this quotient group, and finally lift back to H by testing all elements of the relevant coset of $Z(H)$. Of course, if $Z(H)$ is large (for example if H is abelian) this lifting can be time-consuming. However, membership testing in a large abelian group is known to be hard in the black-box context (although soluble in polynomial time in the matrix group context [20]), and this step cannot be avoided in any recursive implementation of our algorithm applied to arbitrary black-box groups. If we have a black box representation of H, we can easily derive one for $H/Z(H)$, so that this reduction is possible.

There are other types of group in which our algorithm does not improve on simple direct methods. Firstly, if $H/Z(H)$ has odd order then we fail at the first step. But in this case H is soluble, by the Feit–Thompson theorem, so soluble group methods would be more appropriate. Secondly, if $H/Z(H)$ does not contain a Klein 4-group, then we fail at the second step. But in this case, by Glaubermann's Z^*-theorem [13], H is the product of a soluble normal subgroup with the involution centraliser. Thirdly, if all involutions in $H/Z(H)$ are central, we fail to recurse to a smaller group. In this case we might as well work in $H/Z^\infty(H)$, and if this fails to contain any involutions, then again H is soluble.

6.2 The database approach

In its original application, it was envisaged that the algorithm would be applied primarily to sporadic groups, which have a small number of conjugacy classes and where the involution centralisers are well-known. Hence it would be feasible to consult a database of the involution centralisers, containing sufficient information to solve the factorisation problem. An implementation along these lines was developed [15], but the limited gains did not seem to be worth the additional storage requirements and complications in the programming. In any case, such a proposal is impractical if we consider arbitrary simple groups.

6.3 The composition tree approach

Perhaps the most powerful approach is to construct a *composition tree* for each centraliser. Recall that Aschbacher [2] classified the maximal subgroups of $GL(d, q)$ into nine categories. In summary, a linear group preserves some natural linear structure and has a normal subgroup related to this structure, or it is almost simple modulo scalars. The on-going matrix group recognition project [18] explores the structure of a linear group by determining (at least one of) its categories in the Aschbacher classification. The homomorphisms and associated normal subgroups for the various categories are captured by constructing a composition tree, whose leaves are composition factors for the matrix group G. If we can write the image of an element of G in the generators of each composition factor of G, and so solve the factorisation problem for a composition factor, then we can construct the composition tree and solve the factorisation problem for G. We solve the factorisation problem for the leaves using a variety of approaches which depend on the concrete realisation of the leaf: in particular, if the leaf is a matrix group, we may apply the BSGS machinery discussed above.

If our input group is sporadic or of Lie type, then an involution centraliser is usually reducible. Hence, we can exploit the geometric structure of the involution centraliser and decide membership in the centraliser by constructing its composition tree. With this approach, we solve the factorisation problem for the centraliser by constructing its composition tree and solving the factorisation problem for each of its composition factors.

If a composition factor, described as a matrix group, is too large to solve the factorisation problem directly, we can now use our factorisation algorithm to reduce the problem to involution centralisers, and so on. If the composition factor is isomorphic to $PSL(2, q)$, then the algorithm of [12] provides a constructive solution in polynomial time to the factorisation problem for this factor.

7 The complexity of the algorithm

We have already ruled out applying our algorithm to the alternating groups, as better methods are available [6], and there are only finitely many sporadic groups, so they play no role in an

asymptotic complexity analysis. As an algorithm for arbitrary black box groups, or groups of Lie type over fields of characteristic 2, it is certainly exponential in the worst case.

Thus, the interesting case for complexity analysis is that of groups of Lie type in odd characteristic. Since any cross characteristic representation gives considerably longer input length and therefore better complexity, we can assume that the group is given as a matrix group in defining characteristic.

Theorem 12 *Assuming Conjecture 8, if the factorisation algorithm is applied to a group of Lie type in odd characteristic, it runs in time $O(d^4)$, excluding the recursive calls.*

Proof. In step 1, we may need $O(d)$ attempts to find z in a suitable class. Each attempt requires a multiplication, computation of a characteristic polynomial and its factorisation, which have complexity $O(d^3)$.

In step 2 a similar argument applies, except that we need to be sure that the distribution of the involutions to which xz powers is random enough. Conjecture 8 implies that we again need at most $O(d)$ attempts to find a suitable y.

Each time we calculate an involution centraliser, we need $O(d)$ attempts to find a commutator of odd order, and we need only a bounded number of these to obtain the full centraliser with high probability. □

Corollary 13 *Assuming Conjecture 8, running our algorithm recursively is still $O(d^4)$, excluding external calls to the composition tree algorithm.*

Proof. Each time we invoke the algorithm, we recurse to three more calls to the algorithm. However, as noted in Section 2, in two of these calls the element we are trying to factorise is itself an involution, which means we can dispense with steps 1 and 5. Therefore at depth k in the recursion we have exactly $2^{k+1} - 1$ calls to our algorithm.

Moreover, at depth k our group has become essentially a direct product of 2^k smaller classical groups. In terms of complexity, the worst case is when each involution splits the space up into two pieces, one twice the dimension of the other. Thus a step which took time d^4 at one level takes time $(d/3)^4 + (2d/3)^4 = 17d^4/81$ at the next level down.

Letting $\mu = 17/81$ we find that the total running time is bounded by d^4 times

$$\sum_{k=0}^{\infty}(2^{k+1} - 1)\mu^k = \frac{2}{1-2\mu} - \frac{1}{1-\mu} = \frac{1}{(1-2\mu)(1-\mu)},$$

since $\mu < 1/2$. Since this is a constant, our total running time is still $O(d^4)$. □

8 Implementation and performance

This algorithm uses random elements of various groups. For a discussion of algorithms used to produce such elements see [4, 10, 23]. In our implementations, we use the product replacement algorithm [10].

One remark relevant to any implementation is that, in practice, if a membership test in a group succeeds, then the resulting word is constructed and stored as a *straight line program*: the length of a word in a given generating set constructed in n multiplications and inversions can increase exponentially with n, whereas the length of the corresponding straight line program is linear in n. One may intuitively think of a straight line program for $h \in H = \langle X \rangle$ simply as a group word on X that evaluates to h, but is stored in compressed format. For further discussion of this point, see [18].

Two early versions of the algorithm were implemented: a database-driven version [15] in MAGMA [7], with a database covering a few of the smaller sporadic groups; and a recursive version in GAP [21], which used standard library functions for the base case.

These implementations accepted a group of arbitrary type, but of course their performance depended heavily on this type, since the costs of the basic operations vary. The second implementation revealed the problems of the purely recursive approach discussed above, in that the algorithm could become "stuck" in iterated involution centralisers with very large normal subgroups of 2-power order where the Bray algorithm struggled to obtain involution centralisers. Another problem was the difficulty of deciding when there were sufficient generators for an involution centraliser, and if not, which of the centralisers in the recursion was at fault. Nevertheless it was often successful in factoring elements of the sporadic group Co_1.

Our current implementation of the algorithm is in MAGMA [7]. One motivation for its development is to use the algorithm to solve for membership in composition factors of matrix groups. Hence the input to this implementation is an irreducible matrix group. It constructs (at most) three involution centralisers. A composition tree is now constructed for each centraliser, thus allowing a solution to the factorisation problem in each. The composition tree reduces the factorisation problem to almost simple groups. At this stage we may generate further calls to the factorisation algorithm. Alternatively, if the groups are sufficiently small, we invoke the Schreier–Sims algorithm (or its variations) to solve the problem.

Let g, h be two involutions in a group G. Recall from Section 3 that a variation of Bray's algorithm allows one to decide if g is conjugate to h; if so, we obtain a conjugating element. If we repeatedly test for membership in the same group, then we store the chosen involutions and their associated composition trees; as a preliminary step in a new membership test, we decide if the new involutions are conjugate to the known ones; if so, we do not need to construct a new composition tree.

Our algorithm is competitive with the standard BSGS machinery for matrix groups of "moderate" dimension. If the matrix group has no subgroup of reasonable index, our algorithm is currently the only practical approach. For example, the largest proper subgroup of J_4 has index about

10^8; our algorithm readily completes a constructive membership test in the 112-dimensional representation over $GF(2)$.

The computations reported in Table 2 were carried out using MAGMA V2.10 on a Pentium III 800 MHz processor. The input to the algorithm is an irreducible representation of a matrix group. In the column entitled "Time", we list the CPU time in seconds (averaged over three runs) needed to solve the factorisation problem for a random element of the group.

We report on the application of the algorithm to some of the larger sporadic groups and to certain classical groups. The strategy outlined in Section 4 to direct the choice of involution works well. For example in $SL_{20}(5)$ the three centralisers obtained are $SL_n(5)$, with $n = 8$, 10 or 12. A further recursion then reduces to $n \leq 6$, and an invocation of a random Schreier–Sims algorithm now completes the task. None of these examples completed using the random Schreier–Sims algorithm in MAGMA.

Name	d	q	Time
J_4	112	2	20.1
$SL(20,5)$	20	5	10.0
$G_2(3^5)$	7	3^5	1.0
Ly	111	5	120
Th	248	3	4300
$Sp(10,9)$	10	9	6.1

Table 2: Performance of implementation for a sample of groups

References

[1] Sophie Ambrose, Max Neunhöffer, Cheryl E. Praeger and Csaba Schneider, Generalised sifting in black-box groups, preprint 2004.

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

[3] László Babai and Endre Szemerédi, On the complexity of matrix group problems, I, *Proc. 25th IEEE Sympos. Foundations Comp. Sci.*, pp. 229–240, 1984.

[4] László Babai, Local expansion of vertex-transitive graphs and random generation in finite groups, *Theory of Computing*, (Los Angeles, 1991), pp. 164–174. Association for Computing Machinery, New York, 1991.

[5] László Babai, William M. Kantor, Péter P. Pálfy and Ákos Seress, Black box recognition of finite simple groups of Lie type by statistics of element orders, *J. Group Theory* **5** (2002), 383–401.

[6] Robert Beals, Charles R. Leedham-Green, Alice C. Niemeyer, Cheryl E. Praeger, and Ákos Seress, A black-box group algorithm for recognizing finite symmetric and alternating groups. I, Trans. Amer. Math. Soc. **355**, no. 5, 2097–2113, 2003.

[7] Wieb Bosma, John Cannon, and Catherine Playoust, The MAGMA algebra system I: The user language, *J. Symbolic Comput.*, **24**, 235–265, 1997.

[8] Gregory Butler and John J. Cannon, Computing in permutation and matrix groups I: Normal closure, commutator subgroups, series, *Math. Comp.*, **39**, 663–670, 1982.

[9] J. N. Bray, An improved method of finding the centralizer of an involution, *Arch. Math. (Basel)* **74** (2000), 241–245.

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

[11] Frank Celler and C.R. Leedham-Green, Calculating the order of an invertible matrix, In *Groups and Computation II*, volume 28 of *Amer. Math. Soc. DIMACS Series*, pages 55–60. (DIMACS, 1995), 1997.

[12] M.D.E. Conder, C.R. Leedham-Green, and E.A. O'Brien, Constructive recognition of PSL(2, q), accepted to appear *Trans. Amer. Math. Soc.*, 2003.

[13] Daniel Gorenstein, *Finite Groups*, Harper's Series in Modern Mathematics. Harper & Row, New York, Evanston, London, 1968.

[14] Daniel Gorenstein, Richard Lyons, and Ronald Solomon, The classification of the finite simple groups. Number 3. Part I, American Mathematical Society, Providence, RI, 1998.

[15] P.E. Holmes, A Matrix Group Recognition Algorithm and a New Construction of the Monster, MSc thesis, University of Birmingham, 2000.

[16] W.M. Kantor and A. Seress, Prime power graphs for groups of Lie type, *J. Algebra* **247** (2002), 370–434.

[17] P.B. Kleidman, U. Meierfrankenfeld and A.J.E. Ryba, $Ru < E_7(5)$, *Comm. Algebra* **28**, 3555–3583, 2000.

[18] Charles R. Leedham-Green, The computational matrix group project, in *Groups and Computation*, III (Columbus, OH, 1999), 229–247, Ohio State Univ. Math. Res. Inst. Publ., **8**, de Gruyter, Berlin, 2001.

[19] Jeffrey S. Leon, On an algorithm for finding a base and strong generating set for a group given by generating permutations, *Math. Comp.*, 20:941–974, 1980.

[20] E.M. Luks, Computing in solvable matrix groups, *Proc. 33rd IEEE Symp. Found. Comp. Sci.*, 111-120, 1992.

[21] The GAP Group, *GAP – Groups, Algorithms, and Programming, Version 4.3*; 2002, (`http://www.gap-system.org`).

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

[23] Igor Pak (2001), What do we know about the product replacement algorithm?, *Groups and Computation* III, Ohio State Univ. Math. Res. Inst. Publ., (Ohio, 1999). de Gruyter, Berlin.

[24] C.W. Parker and R.A. Wilson, p-cores in black-box groups, in preparation.

[25] Charles C. Sims, Computational methods in the study of permutation groups, In *Computational problems in abstract algebra*, pages 169–183, Oxford, 1970. (Oxford, 1967), Pergamon Press.

[26] Charles C. Sims, *Computation with finitely presented groups*, Cambridge University Press, 1994.

P.E. Holmes, Department of Mathematics, The University of Birmingham, Birmingham B15 2TT, UK.

S.A. Linton, Centre for Interdisciplinary Research in Computational Algebra, University of St Andrews, St Andrews, Fife KY16 9SS, UK.

E.A. O'Brien, Department of Mathematics, University of Auckland, Auckland, New Zealand.

A.J.E. Ryba, Department of Computer Science, City University of New York, Flushing, NY 11367, USA.

R.A. Wilson, School of Mathematical Sciences, Queen Mary, University of London, London E1 4NS, United Kingdom.

Last updated 11th November 2004.