J. Symbolic Computation (2002) **11**, 1–000

Selecting base points for the Schreier-Sims algorithm for matrix groups

SCOTT H. MURRAY[†] AND E.A. O'BRIEN[‡]

[†] *Department of Mathematics, University of Chicago, Chicago, IL* 60637*, USA*

E-mail address: murray@math.uchicago.edu

[‡] *Centre for Mathematics and its Applications, School of Mathematical Sciences, Australian National University, Canberra, ACT* 0200*, Australia*

E-mail address: obrien@maths.anu.edu.au

(Received October 1994*)*

We consider the application of the Schreier-Sims algorithm and its variations to matrix groups defined over finite fields. We propose a new algorithm for the selection of the base points and demonstrate that it both improves the performance of the algorithm for a large range of examples and significantly extends the range of application. In particular, the random Schreier-Sims algorithm, with this enhancement, performs extremely well for almost simple groups.

1. Introduction

Sims (1970 and 1971) introduced a concept, the base and strong generating set, as a method of effectively representing the elements of a permutation group. It has proved of fundamental computational importance. He also described an algorithm, now known as the Schreier-Sims, for its construction.

Let G be a group which acts faithfully on the set Ω and let the stabiliser in G of the points $\beta_1, \beta_2, \ldots, \beta_i \in \Omega$ be denoted by $G_{\beta_1,\beta_2,\ldots,\beta_i}$. A *base* for G is a sequence of points $B = [\beta_1, \beta_2, \ldots, \beta_k]$ such that $G_{\beta_1,\beta_2,\ldots,\beta_k} = 1$. This determines a chain of stabilisers

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

where $G^{(i)} = G_{\beta_1,\beta_2,\ldots,\beta_{i-1}}$. A *strong generating set* corresponding to B is a subset S of G such that $G^{(i)} = \langle S \cap G^{(i)} \rangle$, for $i = 1, \ldots, k$. We use the abbreviation BSGS for base and strong generating set.

A central component of the BSGS construction is the computation of the *basic orbits* — namely, the orbit of the base point β_i under $G^{(i)}$ for $i = 1, \ldots, k$. The performance and range of application of the technique depends greatly on the sizes of these basic orbits.

Sims presented a deterministic algorithm to construct the required generating sets, using an application of Schreier's Lemma. A random algorithm, known as the random

0747–7171/90/000000 + 00 $03.00/0

Schreier-Sims, introduced by Leon (1980), finds generating sets for the stabilisers by considering random elements of the group, rather than using the Schreier generators. It is usually significantly faster than the original and provides smaller strong generating sets. In practice, it terminates when some *stopping condition* becomes true. This condition should be chosen to maximise the probability of finding a complete BSGS without considering many unnecessary random elements. Usually, we stop when a predetermined number, N, of consecutive random elements have all been found to be redundant as strong generators. If the random elements are uniformly distributed, the probability that we do not have a complete BSGS is now less than 2^{-N}. If the order of the group is known in advance, we can terminate when the product of basic indices reaches this value.

Butler (1976) first developed and applied the Schreier-Sims algorithm to matrix groups defined over a finite field. Suppose that G is a subgroup of $GL(d, q)$ for some prime power q. Then G acts faithfully on V, the d-dimensional vector space over $GF(q)$, with action on the right. We can now try to compute a BSGS for G, viewed as a permutation group on the vectors, where we take the base points to be standard basis vectors for V.

The central problem with this approach is that the basic orbits computed can be very large, since the number of vectors in V grows exponentially with d. It is not uncommon for the size of the first basic orbit to be the order of the group. Butler (1976 and 1979) considers the action of G on the one-dimensional subspaces of V. The use of these subspaces may reduce the size of the basic orbits by a factor of up to $q - 1$. Since the action of G on the subspaces need not be faithful, each subspace in the base is followed by a non-zero vector contained in it. This idea significantly extended the range of application of the variations of the Schreier-Sims algorithm, and they have become one important component of the machinery for computing with matrix groups. Implementations of these are available in the computational algebra systems GAP (Schönert *et al.*, 1994) and Magma (Bosma & Cannon, 1994). In practice, matrix group algorithms in Magma do not rely on a permutation representation for the matrix group but instead use, as a central component, the (permutation group) concept of a chain of subgroups or stabilisers; for details see, for example, Butler & Cannon (1982).

In this paper we present a new algorithm for selecting base points for matrix groups. In summary, we select as base points some common eigenvectors for a collection of elements of the group and then add them to the base in a specific order. We developed a new implementation of the random Schreier-Sims algorithm to determine the effectiveness of our algorithm. Our investigations show that it extends significantly the range of application of the random Schreier-Sims algorithm. We are able to compute BSGS for a range of matrix groups where the existing implementations fail. In many other cases we can compute a BSGS in a significantly shorter period of time. We have frequently reduced the time taken by a factor in excess of 100. The improvement arises primarily because of the reduction in the sizes of the orbits. We expect that our base point selection method will be equally useful for other variations of the Schreier-Sims algorithm.

We studied various aspects of the random Schreier-Sims algorithm and record some of our observations. We rejected using a set consisting of vectors or subspaces as a base point since the storage requirements of the resulting basic orbit will always be at least as large as those of the orbit of any one of its elements. Since the storage required for and the time taken to echelonise a subspace is determined by its dimension, subspaces of dimensions larger than about four are not practically useful as base points. The existing one-dimensional space technique was highly effective for all soluble groups considered.

Random elements are used in two different ways in our work: we first use some to compute the base points, and we later use them instead of Schreier generators. We use the *product replacement* algorithm to generate random elements of a group. In this algorithm, a certain amount of preprocessing is first carried out, where we evaluate random words in the generators and store the results. When this preprocessing is complete, a new random element can be obtained at the cost of one matrix multiplication: namely, we multiply two of these stored words to obtain a new element, and also replace one of them by the product. The algorithm is studied in detail by Celler, Leedham-Green, Murray, Niemeyer & O'Brien (1996). The cost of the algorithm and the distribution of the elements generated is determined by the preprocessing step. We observed that the amount of preprocessing had little impact on the performance of the random Schreier-Sims algorithm in the cases reported. However, it is important to terminate only when about 30 consecutive elements are found to be redundant. Of course, here we are primarily interested in whether the generated elements lie in a particular subgroup.

In addition to its intrinsic interest, our work was motivated by another aim. Aschbacher (1984) classified the subgroups of $GL(d, q)$ into nine categories. As a first step, the "recognition project" seeks to develop algorithms to recognise the Aschbacher category of a matrix group, described by a collection of generating matrices. Recent work on aspects of the project includes the recognition algorithm for special linear groups of Neumann & Praeger (1992) and the algorithm for imprimitivity testing of Holt, Leedham-Green, O'Brien & Rees (1996).

Some of the algorithms developed do not deal efficiently with matrix groups of small degree. Let G be a subgroup of $GL(d, q)$ and let Z be its subgroup of scalar matrices; then G is *almost simple* if there is a non-abelian simple group T such that $T \leq G/Z \leq \mathrm{Aut}\ T$, the automorphism group of T. The almost simple groups comprise the last of the nine categories. Since its membership is diverse, we expect that the recognition of groups in this category will pose significant challenges. However, Liebeck (1985) proved that the maximal non-classical subgroups of $GL(d, q)$ have order at most q^{3d}, which is small by comparison with that of $GL(d, q)$. In a 1993 private communication, Praeger asked us whether the random Schreier-Sims algorithm performs particularly well for groups in the almost simple category. The evidence of Section 3 suggests that our improved version performs extremely well for some such groups. These observations suggest that variations of the Schreier-Sims algorithm will remain important in the structural investigation of matrix groups.

In other recent work, Cooperman, Finkelstein, York & Tselman (1994) have constructed a permutation representation of degree 9 606 125 for the sporadic simple group, Ly, acting on a conjugacy class of its subgroups of order 3. They propose to generalise this technique.

A good description of the Schreier-Sims algorithm (and related concepts) can be found in Bosma & Cannon (1992). We present our base point selection strategy in Section 2 and report on its performance in Section 3.

2. Selecting base points

We focus on the application of the Schreier-Sims algorithm to matrix groups defined over finite fields and present the new strategy to select base points which we expect *a priori* to have "small" orbits under the action of a subgroup in the stabiliser chain of a matrix group G.

In general, we can only assume that we know a generating set for G. Let A be a generator of G and let $g(x)$ be a factor of the characteristic polynomial of A which is irreducible over $GF(q)$. We call $v \in V$ an *eigenvector* if $v.g(A) = 0$. The subspace of all such eigenvectors is the *eigenspace* of A corresponding to $g(x)$. Note that we do not require that $g(x)$ be linear. If $g(x)$ has degree m, the size of the corresponding orbit is bounded by $q^m - 1$. Hence we choose as base points eigenvectors corresponding to a factor which has as small a degree as possible. Now suppose that $g(x) = x^m - \lambda$, for some $\lambda \in GF(q)$. Then the orbit $v^{\langle A \rangle}$ contains at most $m(q-1)$ points, since $v^{A^m} = \lambda v$ and $\lambda^{q-1} = 1$. We found that it is also useful to choose as base points eigenvectors corresponding to a factor which has as few non-zero terms as possible.

An eigenvector of one generator need not have a small orbit under the action of a member of the stabiliser chain. However, we can improve the probability of finding a small orbit by using a base point which is an eigenvector of more than one generator. First we calculate the set of eigenspaces of the generators and add to it all of the non-trivial intersections of these spaces. We order the elements of this set by considering the polynomial factors corresponding to each subspace and giving precedence to those spaces whose factors have the lowest degree and the smallest number of non-zero terms. We now choose one vector from each space in this order until we have found a base. When we select a vector as a base point, we always precede it in the base by the corresponding one-dimensional subspace, as suggested by Butler (1976).

It is also possible to consider the action of the group on complete subspaces, rather than selecting a vector from each. Occasionally, this can result in a reduction in orbit size sufficient to compensate for the increased time taken to calculate images.

One problem with this strategy is that the generating set for G usually contains very few elements; for example, every sporadic simple group can be generated by two elements. We can greatly improve our chances of finding a small orbit by also calculating eigenvectors of a number of random elements of the group. This not only improves our chances of finding a "good" eigenvector, but also allows us to choose base points which are simultaneously eigenvectors of a number of different group elements.

These techniques increase the range of application of the algorithm. However, in a few of the examples considered, we found that the base points chosen have larger orbits than the standard basis vectors. This happened particularly when the matrices in the group are sparse and the chosen eigenvector is not. In an attempt to address this problem, we used sparseness as one of the criteria for ordering the intersections, where we define the *sparseness* of a row-space to be the number of zeros in the row-reduced matrix of basis vectors divided by the dimension of the space.

Our base point algorithm is summarised below. The eigenspaces and intersections are sorted according to five criteria. Only the first of these has theoretical justification; the inclusion and order of the remaining criteria is largely based on experimentation. The final criterion, which has relatively little impact, is included because the dimension is critical when using complete intersections.

1 Calculate some random group elements.
2 Calculate eigenspaces of the generators and random elements.
3 Calculate all non-trivial intersections of the eigenspaces.
4 Order the eigenspaces and intersections by the following criteria:
   (a) Smallest degrees of corresponding factors.

(b) Smallest numbers of non-zero terms in corresponding factors.
(c) Largest number of eigenspaces intersected.
(d) Greatest sparseness.
(e) Smallest dimension.

5 Repeatedly take one vector from each subspace in turn until we obtain a base.

In seeking to order the collection of subspaces, we first order the eigenspaces according to the first two criteria, and then choose the best from the relevant factors as the corresponding factor for an intersection. Now we can sort all of the subspaces according to the five criteria.

Of course, it is possible that all of the computed characteristic polynomials are irreducible and the resulting intersection is the complete space. If this occurs, our method will choose a base consisting of standard basis vectors: it defaults to Butler's strategy.

3. Implementation and Performance

As part of this investigation, we developed a new implementation of the random Schreier-Sims algorithm for matrix groups defined over prime fields. The base selection algorithm is implemented in the MAGMA language and is linked to the main program, written in traditional C. The program provides the user with a high-level of control. For example, the base points can be computed using the MAGMA procedure, selected by the default strategy, or supplied by the user; and they can be vectors or subspaces of arbitrary dimension. The program is available on request from the authors, and our strategy is now part of the default used by the Schreier-Sims algorithm in MAGMA.

We used this implementation to test our base point strategy on various groups. We chose certain sporadic simple groups, almost simple groups, and soluble groups. We considered about 100 examples in total and found that our algorithm reduced the size of the largest orbit in about 60% of the cases. On only two occasions was this orbit larger than that constructed by the default.

In Table 1 we present the results of our tests for a range of examples which represent the various outcomes. Where possible, we describe the groups using the notation of the ATLAS (Conway *et al.*, 1985). The group labelled by $S(3.O'N)$ is a subgroup of $3.O'N$. The group denoted $GL(d,q) - i \wr p^m$ is a wreath product of the ith member of the library of soluble subgroups of $GL(d,q)$, constructed by Short (1992), with a p-subgroup of the symmetric group of degree p^m.

It is conceivable that the initial generating set influences the performance of the algorithm. In practice, "standard" generating sets for matrix groups are far from random. We have chosen commonly available generating sets for the sporadic and classical groups — those supplied as part of the data libraries or by functions in GAP or MAGMA. For each symmetric group, the permutation module over a prime field for its action on pairs of elements was first constructed, using as input the default generating set for the group supplied by MAGMA; this module was then split up using the Lux and Ringe implementation of the MeatAxe (see Ringe, 1992), and the resulting generating sets for the irreducible modules used. In Table 1 we identify the representation chosen for investigation only up to degree and field.

We provide results for two different methods. The default uses Butler's method of alternating one-dimensional subspaces and vectors in the base, where the subspaces are

Table 1. Comparison of performance of random Schreier-Sims algorithm

Group	d	q	Order	Time (seconds)		Largest Orbit	
				Default	Our method	Default	Our method
$S(3.O'N)$	27	7	77 760	37	8	25 920	405
$S(3.O'N)$	45	7	77 760	90	28	25 920	1 080
M_{12}	55	7	95 040	238	113	95 040	31 680
M_{12}	120	17	95 040	7 500	506	95 040	9 504
J_1	27	11	175 560	83	47	87 780	35 112
M_{22}	21	7	443 520	207	6	443 520	840
M_{22}	54	7	443 520	992	207	443 520	36 960
M_{22}	154	7	443 520	5 284	523	221 760	462
$3.M_{22}.2$	30	2	2 661 120	∞	46	∞	41 580
$2.J_2$	36	3	1 209 600	∞	18	∞	2 016
$2.J_2$	42	3	1 209 600	1 408	53	604 800	6 300
$2.J_2.2$	12	3	2 419 200	132	12	201 600	12 600
$3.J_3.2$	18	2	301 397 760	110	80	130 815	61 560
$2.HS.2$	112	3	177 408 000	3 794	108 595	134 400	739 200
$M^cL.2$	21	5	1 796 256 000	24	25	7 128	7 128
$He.2$	50	7	8 060 774 400	∞	175	∞	8 330
Ru	28	2	145 926 144 000	1 516	1 468	417 600	417 600
Co_3	22	3	495 766 656 000	∞	10	∞	276
$2.Suz.2$	12	3	1 793 381 990 400	52	223	32 760	232 960
Co_2	24	2	42 305 421 312 000	166	117	46 575	46 575
$F_4(2)$	26	2	3 311 126 603 366 400	∞	189	∞	69 615
S_9	28	13	9!	716	6	362 880	720
S_{13}	65	13	13!	∞	64	∞	286
S_{19}	153	19	19!	∞	1 768	∞	969
S_{20}	171	53	20!	∞	2 269	∞	5 040
S_{22}	21	3	22!	∞	2 190	∞	387 600
S_{25}	253	5	25!	∞	144 919	∞	71 820
S_{30}	28	31	30!	2 546	90	82 215	435
$L_3(4)$	63	11	20 160	41	44	630	630
$U_4(2)$	81	11	25 920	77	91	810	810
$L_2(81)$	82	41	265 680	89	87	82	82
$L_3(5)$	124	2	372 000	367	401	465	465
$PGU_3(5)$	20	3	378 000	447	8	378 000	1 750
$PGU_4(3)$	38	3	13 063 680	∞	15	∞	224
$Sp(6,3)$	50	3	9 170 703 360	∞	328	∞	17 496
$Sp(8,2)$	160	2	47 377 612 800	∞	2 461	∞	16 320
$Sp(6,7)$	21	7	273 457 218 604 953 600	104	133	19 608	19 608
$GL(2,7) - 3 \wr 3^3$	54	7	$2^{81}3^{13}$	460	438	54	54
$GL(4,3) - 50 \wr 3^3$	108	3	$2^{135}3^{40}$	6 916	6 161	216	216
$GL(5,3) - 2 \wr 3^3$	135	3	$2^{27}3^{13}11^{27}$	4 072	4 470	297	297
$GL(5,3) - 9 \wr 2^4$	80	3	$2^{95}5^{16}$	1 812	1 870	80	80
$GL(6,2) - 10 \wr 2^4$	96	2	$2^{31}3^{48}$	916	931	144	144
$GL(6,2) - 21 \wr 3^3$	162	2	$2^{27}3^{121}$	13 372	13 474	243	243
$GL(6,2) - 33 \wr 2^3$	48	2	$2^{39}3^{24}$	98	100	288	288

generated by standard basis vectors. Our implementation initially uses the same base points, but if an orbit exceeds a user-supplied size — taken to be 10 000 in our tests — it is discarded and subsequent base points are computed using the algorithm described in Section 2. Clearly, in a few cases — for example, $Sp(6,7)$ and $2.Suz.2$ — this size was not optimal and it may be sensible to set a larger bound. We computed the action of $2.HS.2$ on a 4-dimensional subspace. The total number of generators and random elements considered in selecting base points is usually 20; however, we considered 40 for the symmetric group examples.

All of the CPU times reported are in seconds and the tests were carried out on a Sparc Station 10/51, with 128 MB of RAM. We list the size of the largest basic orbit for each example, since this is the most significant factor in determining the effectiveness of the random Schreier-Sims algorithm. The basic orbits are searched using a linear-probe hash algorithm (see Knuth, 1973), with hash tables of default size 1 000 000. When an orbit exceeds this length, the program aborts; this is indicated by the ∞ symbol.

Our implementation can use either of the stopping conditions mentioned in Section 1. For all examples reported in Table 1, the algorithm terminated after 30 consecutive random elements were found to be redundant. The choice of $N = 30$ allowed us to find a complete BSGS in all of the examples given. Since we do not use uniformly random elements, smaller values of N are often insufficient.

References

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

Bosma, Wieb, Cannon, John (1992), "Structural computation in finite permutation groups", *CWI Quarterly*, **5**(2), 127–160.

Bosma, Wieb, Cannon, John (1994), *Handbook of* MAGMA *functions*. Department of Pure Mathematics, Sydney University.

Butler, Gregory (1976), "The Schreier Algorithm for Matrix Groups", SYMSAC '76, *Proc. ACM Sympos. symbolic and algebraic computation*, (New York, 1976), pp. 167–170. Association for Computing Machinery, New York.

Butler, Gregory (1979), *Computational Approaches to Certain Problems in the Theory of Finite Groups*, PhD thesis. University of Sydney.

Butler, Gregory, Cannon, John J. (1982), "Computing in Permutation and Matrix Groups I: Normal Closure, Commutator Subgroups, Series", *Math. Comp.*, **39**, 663–670.

Celler, Frank, Leedham-Green, Charles R., Murray, Scott H., Niemeyer, Alice C., O'Brien, E.A. (1996), "Generating random elements of a finite group", *Comm. Algebra*.

Conway, J.H., Curtis, R.T., Norton, S.P., Parker, R.A., and Wilson, R.A. (1985), *Atlas of finite groups*. Clarendon Press, Oxford.

Cooperman, Gene, Finkelstein, Larry, York, Bryant, Tselman, Michael (1994), "Constructing Permutation Representations for Large Matrix Groups", *Proceedings of International Symposium on Symbolic and Algebraic Computation* ISSAC '94, (Oxford), pp. 134–138. ACM Press, New York.

Holt, Derek F., Leedham-Green, Charles R., O'Brien, E.A., Rees, Sarah (1996), "Primitivity testing for matrix groups", *J. Algebra*.

Knuth, Donald E. (1973), *The Art of Computer Programming. Volume* 3*: Sorting and Searching*. Addison-Wesley, Massachusetts.

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

Liebeck, Martin W. (1985), "On the orders of maximal subgroups of the finite classical groups", *Proc. London Math. Soc.* (3), **50**, 426–446.

Neumann, Peter M., Praeger, Cheryl E. (1992), "A recognition algorithm for special linear groups", *Proc. London Math. Soc.* (3), **65**, 555–603.

Ringe, Michael (1992), *The C* MeatAxe. Lehrstuhl D für Mathematik, RWTH Aachen.

Schönert, Martin *et al.* (1994), GAP – *Groups, Algorithms and Programming*. Lehrstuhl D für Mathematik, RWTH Aachen.

Short, M.W. (1992), *The Primitive Soluble Permutation Groups of Degree less than* 256, Lecture Notes in Math., **1519**. Springer-Verlag, Berlin, Heidelberg, New York.

Sims, Charles C. (1970), "Computational methods in the study of permutation groups", *Computational problems in abstract algebra*, (Oxford, 1967), pp. 169–183. Pergamon Press, Oxford.
Sims, Charles C. (1971), "Determining the conjugacy classes of permutation groups", Garret Birkhoff and Marshall Hall, Jr. (Eds.), *Computers in algebra and number theory*, Proc. Amer. Math. Soc., **4**, (New York, 1970), pp. 191–195. Amer. Math. Soc., Providence, RI.