Computing with Matrix Groups using Permutation Representations

Gene Cooperman[1] Larry Finkelstein[1] Michael Tselman[1]

{gene,laf,misha}@ccs.neu.edu
College of Computer Science
Northeastern University
Boston, MA 02115 / U.S.A.

Abstract

Permutation representations constructed from matrix groups defined over finite fields often have very high degree. New techniques are presented for performing effective computations with the resulting permutation group. These techniques are designed to work in an environment in which the degree of the permutation group is considered too large to permit the use of standard permutation algorithms for solving problems such as computing the order of the group and testing simplicity. The theory has been successfully tested on a representation of the sporadic simple group Ly, discovered by Lyons [10]. In [5], a permutation representation was constructed for Ly of degree $9,606,125$ on a conjugacy class of subgroups of order 3. Using this permutation representation and no specific knowledge of the group, we are able to apply our methods to construct a base of at most four points for the resulting permutation group, compute its order and verify simplicity. Monte Carlo algorithms for group membership presented in [2] are used to improve the performance of these algorithms.

1 Introduction

A well known technique for computing with subgroups of the general linear group $GL(n,q)$ is to construct a permutation representation for the group and then apply algorithms for computing with permutation groups. Previous efforts at constructing a permutation representation have relied on the action of the matrix group on the set of $(q^n-1)/(q-1)$ one dimensional subspaces of the underlying vector space V. Software based on ideas of Butler [4], Leon [9] and Sims [12] has been available for years, first in Cayley and later in GAP and MAGMA. However, the size of the permutation domain grows quickly with n beyond the limitations of modern technology, forcing us to look for other actions.

In [5], we have developed practical methods for constructing the permutation representation from the conjugacy action on a conjugacy class $\mathcal{O}$ of subgroups of prime order. The key idea is to assign to each subgroup of the conjugacy class a signature that can be efficiently computed and can be used as an index into a hash table $\mathcal{T}$ used to store the permutation domain. The signature is computed by taking a randomly chosen vector v and assigning to each subgroup, the image of v under a generator of the subgroup. This allows us to replace matrix-matrix multiplication by vector-matrix multiplication in the course of building $\mathcal{T}$, thereby saving a factor of n in the time to construct $\mathcal{T}$.

In this paper, we assume that a permutation representation has been constructed for G acting on $\mathcal{O}$ and present efficient methods for constructing a base for $G^{\mathcal{O}}$ and for testing simplicity of $G^{\mathcal{O}}$.

The key to the construction of a base for $G^{\mathcal{O}}$ is the fact that each point of $\mathcal{O}$ has group theoretic significance, namely a subgroup of G. However, when $|\mathcal{O}|$ is very large, space becomes an overriding concern. For example, in in the case of the Lyons sporadic simple group, Ly, $|\mathcal{O}|$ has size $9,606,125$ and so each permutation requires approximately 38MB of storage (assuming a four byte word per entry).

For permutation domains of the size required for Ly, it is difficult to store more than a handful of permutations, and so we have developed the theory for constructing a base in which permutations are never explicitly computed, but rather the hash table $\mathcal{T}$ is used to compute the action of a group element, stored as a matrix, on a point of the permutation domain. This increases the time required to perform this operation, in comparison to a simple array reference in the permutation domain, but allows us to store a significant number of group elements represented as matrices in the course of building a strong generating set. For example, in the matrix representation of Ly in $GL(111,5)$, each group element stored as a matrix requires only 12KB of storage (assuming one byte per matrix entry). Furthermore, the use of the signature described above reduces the complexity of computing the image of a point under the action of a group element in this hybrid setting. The details of the construction are presented in section 3.

Given a base in advance, there are numerous strategies available for computing a strong generating set for $G^{\mathcal{O}}$, and hence determining $|G^{\mathcal{O}}|$. One strategy is to use a heuristic procedure to obtain a candidate strong generating set $\mathcal{S}$ and then verify that strong generating set using either the Monte Carlo test described in [2] or the deterministic Brownie-Sims *Verify* test implemented in Magma. We omit discussion of these techniques in this paper, since they are already well described elsewhere.

In section 4 we give an elementary simplicity test for $G^{\mathcal{O}}$. This algorithm also exploits the fact that the points of the

[1]Partially supported by NSF Grant CCR-920449.

Int. Symp. on Symbolic and Algebraic Computation (ISSAC-95), ACM Press, 1995, pp. 259–264; copyright ACM

permutation domain have group theoretical significance.

2 Construction of a Permutation Representation

In [5] we described how to construct a permutation representation from the conjugation action on a conjugacy class of subgroups of prime order. The motivation for this approach is twofold.

- For many important examples, a permutation representation with close to minimal degree can be constructed in this manner.
- Each subgroup in the conjugacy class has an efficiently computable signature.

The first statement is partially justified by a theoretical result of Babai and Beals [1]. Specifically, if G is simple and $\nu(G)$ is the degree of the smallest permutation representation for G, then their result asserts that when G is simple, there is always a conjugacy class of subgroups of prime order whose size is bounded by $\nu(G)^c$, for some positive constant c. (W.M. Kantor has indicated to the authors that one can take the bound to be $c = 2$.) Babai and Beals use this result as the basis for a theory of computing with black box groups.

The second statement is justified by the use of the image of a random vector v as signature, as described in the introduction. It is valid to use v as a signature if and only if no non-identity element of G fixes v. In general, this is a hard condition to test, but if prior knowledge of the group structure is available, lower bounds can be established on the probability that a randomly chosen vector has this property. For example, in the case of Ly, this probability is at least $1 - 3.8 \times 10^{-23}$ [5]. Without knowledge of the group structure, it is still possible to give a lower bound of $1 - 1/q$ on the reliability of the enumeration, by employing a simple test based on the probability that a randomly chosen vector lies in the fixed point subspace of a non-identity element of G. This will be presented in a forthcoming paper [6].

3 Verification of a Base for a Conjugation Action

Let C be a conjugacy class of G consisting of elements of prime order p and let $\mathcal{O} = \{\langle x\rangle\colon\ x \in C\}$. We define the permutation representation:

$$\pi : G \mapsto \mathrm{Sym}(\mathcal{O})$$

by $\langle x\rangle^{\pi(g)} = \langle g^{-1}xg\rangle = \langle x\rangle^g$ for each $x \in C, g \in G$.

For simplicity, we refer to this permutation representation as the *conjugation action* of G on $\mathcal{O}$. The first result gives a description of the kernel of this action.

Lemma 3.1 *If K is the kernel of the action of G on $\mathcal{O}$, then either $K = \mathcal{C}_G(C)$, or $\mathcal{C}_G(C)$ is properly contained in K and $\langle C\rangle$ is an elementary abelian subgroup.*

Proof: By definition, $K = \{g \in G\colon\ \langle x\rangle^g = \langle x\rangle,\ \forall \langle x\rangle \in \mathcal{O}\}$. Clearly $\mathcal{C}_G(C) \subseteq K$. Therefore, if K commutes with C, then $K = \mathcal{C}_G(C)$. We may assume for the remainder of the proof, that $\mathcal{C}_G(C) \neq K$. By construction, K normalizes $\langle x\rangle$ for each $\langle x\rangle \in \mathcal{O}$. Further, x normalizes K, since $K \lhd G$. So,

$$[K, \langle x\rangle] \subseteq \langle x\rangle \cap K = 1 \text{ or } \langle x\rangle\ \ \forall \langle x\rangle \in \mathcal{O},$$

since x has prime order. But $K \neq \mathcal{C}_G(C)$ implies that $[K, \langle x\rangle] \neq 1$ for at least some $\langle x\rangle \in \mathcal{O}$. In this case, $[K, \langle x\rangle] = \langle x\rangle$. Therefore, $x \in K$, which implies that $C \subseteq K$ since C is a conjugacy class of G and $K \lhd G$. It now follows that $\langle C\rangle$ is elementary abelian. □

C generates an elementary abelian group precisely when C is in the kernel of the action on $\mathcal{O}$. Testing if C is in the kernel is easy since this condition is true if a single element of C acts trivially on $\mathcal{O}$. In the case where $\langle C\rangle$ is elementary abelian, it is more appropriate to consider the linear action of G on $\langle C\rangle$ rather than the permutation action and therefore we assume the following for the remainder of this section.

Assumption 3.2 *$\langle C\rangle$ is non-abelian.*

As a consequence of this assumption, we then have from Lemma 3.1 that $G/\mathcal{C}_G(C)$ acts faithfully on $\mathcal{O}$.

We now consider the task of finding a base for G acting on $\mathcal{O}$. The construction of a base requires two steps. The first is to find a set $B \subseteq C$ with the property that $\mathcal{C}_G(B) = \mathcal{C}_G(C)$. Note that B can be characterized as a base for the conjugation action of G on C. If $\mathcal{O}_B = \{\langle b\rangle : b \in B\}$, then $\mathcal{O}_B$ is "close" to being a base for $G^{\mathcal{O}}$. The second step is to extend $\mathcal{O}_B$ to a base for $G^{\mathcal{O}}$. Lemma 3.9 is the key to performing this extension.

The proper setting for constructing B is either the natural embedding of G in the full matrix ring $\mathrm{Mat}(n, q)$ or else the homomorphic image of G as a subgroup of $\mathrm{Sym}(\mathcal{O})$. Because the arguments are similar, we adopt notation that emphasizes the general nature of the proof. We define two algebraic objects H and $\bar{G} = G/K$. Either $H = \mathrm{Mat}(n, q)$ and K=1, so that $G = \bar{G}$, or else $H = \mathrm{Sym}(\mathcal{O})$ and $K = \mathcal{C}_G(C)$. Note that G acts by conjugation as a group of automorphisms of $\mathrm{Mat}(n, q)$ and thus familiar group theory definitions such as $\mathcal{C}_H(B)$, $B \subseteq C$ and $\mathcal{N}_G(M)$ for M a G-invariant subalgebra of $\mathrm{Mat}(n, q)$ make sense. Also, we assume in the case $H = \mathrm{Mat}(n, q)$ that whenever we conjugate by an element $h \in H$, that h is a unit in H, i.e. that $h \in GL(n, q)$.

Lemma 3.3 *If $\bar{B}^h$ commutes with $\mathcal{C}_H(\bar{B})$ for $\bar{B} \subseteq \bar{G}, h \in H$, then h normalizes $\mathcal{C}_H(\bar{B})$.*

Proof: Note that $\bar{B}^h$ commutes with $\mathcal{C}_H(\bar{B})$ implies that $\mathcal{C}_H(\bar{B}) \subseteq \mathcal{C}_H(\bar{B}^h) = \mathcal{C}_H(\bar{B})^h$. But $|\mathcal{C}_H(\bar{B})| = |\mathcal{C}_H(\bar{B})^h|$ and so $\mathcal{C}_H(\bar{B}) = \mathcal{C}_H(\bar{B})^h$. □

Theorem 3.4 *Assume a generating set S for G and an element $b \in C$ are known. One can find a set, $B \subseteq C$, of at most $\log_2 |G|$ elements satisfying $\mathcal{C}_{\bar{G}}(\bar{B}) = \mathcal{C}_{\bar{G}}(\bar{C})$. Let t be an upper bound on the time to compute $\mathcal{C}_H(\bar{B}')$ for $\bar{B}' \subseteq B$, and let k be an upper bound on the time to compute if an element of H commutes with $\mathcal{C}_H(\bar{B}')$. Then the time to compute B is $O(|B||S|k + |B|t)$.*

Proof: The algorithm can be simply described as follows.

Starting with $B = \{b\}, b \in C$
Compute $\mathcal{C}_H(\bar{B})$
While there exists $(b, g) \in B \times S$ untested
  If $\bar{b}^{\bar{g}}$ does not commute with $\mathcal{C}_H(\bar{B})$
    Add b^g to B
    Compute $\mathcal{C}_H(\bar{B})$

It is claimed that when there are no more untested pairs, $\mathcal{C}_{\bar{G}}(\bar{B}) = \mathcal{C}_{\bar{G}}(\bar{C})$. Since $\langle B\rangle \subseteq G$ and the elements of B form a non-redundant generating set for $\langle B\rangle$, $|B| \leq \log_2 |G|$.

The proof divides into two cases. In the first case, assume that $\mathcal{N}_{\bar{G}}(\mathcal{C}_H(\bar{B}))$ is proper in $\bar{G}$. Then there exists a $g \in S$, for which $\bar{g} \notin \mathcal{N}_{\bar{G}}(\mathcal{C}_H(\bar{B}))$. By Lemma 3.3, $\bar{B}^{\bar{g}}$ and $\mathcal{C}_H(\bar{B})$ do not commute. So, there is a $b \in B$ such that $\bar{b}^{\bar{g}}$ does not commute with $\mathcal{C}_H(\bar{B})$, and our algorithm finds a new pair (b, g) and augments B.

In the second case, $\mathcal{N}_{\bar{G}}(\mathcal{C}_H(\bar{B})) = \bar{G}$. This implies that for all $\bar{g} \in \bar{G}$, $\mathcal{C}_H(\bar{B}^{\bar{g}}) = \mathcal{C}_H(\bar{B})^{\bar{g}} = \mathcal{C}_H(\bar{B})$. Hence $\bar{B}^{\bar{g}}$ commutes with $\mathcal{C}_H(\bar{B})$. But $\bar{C} = \bar{b}^{\bar{G}}$, and so $\bar{C}$ commutes with $\mathcal{C}_H(\bar{B})$. In particular, $\mathcal{C}_{\bar{G}}(\bar{B}) \subseteq \mathcal{C}_{\bar{G}}(\bar{C})$. Since the reverse inclusion is clearly true, we may infer that $\mathcal{C}_{\bar{G}}(\bar{B}) = \mathcal{C}_{\bar{G}}(\bar{C})$ as claimed.

Given the defined parameters k and t, the timing follows directly from the algorithm. □

When $H = \mathrm{Sym}(\mathcal{O})$, so that $\bar{G} = G/\mathcal{C}_G(C)$, it may be the case that $\mathcal{C}_{\bar{G}}(\bar{C})$ is non-trivial. The next results shows that when this happens, $G^{\mathcal{O}}$ is imprimitive and this leads to a permutation representation of smaller degree.

Corollary 3.5 *If $H = \mathrm{Sym}(\mathcal{O})$ and $\mathcal{C}_H(\bar{C})$ is non-trivial, then the set of orbits of $\mathcal{C}_H(\bar{C})$ forms a proper block system for $\bar{G}$.*

Proof: Since $\bar{G}$ acts transitively on $\mathcal{O}$ and $\mathcal{C}_H(\bar{C})$ is normalized by $\bar{G}$, the set of orbits of $\mathcal{C}_H(\bar{C})$ forms a block system for $\bar{G}$. In order to show that the block system is proper, it suffices to show that $\mathcal{C}_H(\bar{C})$ acts intransitively on $\mathcal{O}$. By Assumption 3.2, $\langle C \rangle$ is non-abelian and so $\langle \bar{C} \rangle$ is a non-trivial subgroup of $\bar{G}$ by Lemma 3.1. Since C is a conjugacy class of G, $C \cap \mathcal{C}_G(C) = \emptyset$. This implies that each $x \in C$ acts non-trivially on $\mathcal{O}$ but does have fixed points, for example, $\langle x \rangle$. In particular, each element of $\bar{C}$ has fixed points on $\mathcal{O}$. It then follows that $\mathcal{C}_H(\bar{C})$ acts intransitively as required. □

This result shows how to construct a non-trivial system of imprimitivity for $\bar{G}$ acting on $\mathcal{O}$ when $H = \mathrm{Sym}(\mathcal{O})$ and $\mathcal{C}_H(\bar{C})$ is non-trivial. In this case, we can construct a permutation representation for G acting on the set of orbits of $\mathcal{C}_H(\bar{C})$ which will have smaller degree than the one for G acting on $\mathcal{O}$ and we work with this one instead. Thus for the remainder of the section, we assume that the following assumption holds.

Assumption 3.6 *If $H = \mathrm{Sym}(\mathcal{O})$, then $\mathcal{C}_H(\bar{C}) = 1$.*

Under Assumption 3.6, if $H = \mathrm{Sym}(\mathcal{O})$ and B is chosen by the algorithm in Theorem 3.4, then $\bar{B}$ satisfies $\mathcal{C}_H(\bar{B}) = \mathcal{C}_H(\bar{C}) = 1$. Since $B \subseteq C$, $\mathcal{C}_G(C) \subseteq \mathcal{C}_G(B)$, and this implies that $\mathcal{C}_G(B) = \mathcal{C}_G(C)$. We can then extract from Theorem 3.4 the following criteria.

Corollary 3.7 *B is a base for G^C if $[\bar{B}^{\bar{S}}, \mathcal{C}_H(\bar{B})] = 1$.*

Remark The most expensive part of the algorithm used to construct B is computing the subgroup $\mathcal{C}_H(\bar{B})$, either when $H = \mathrm{Mat}(n, q)$ or when $H = \mathrm{Sym}(\mathcal{O})$. In the case where $H = \mathrm{Mat}(n, q)$, this can be accomplished by solving a system of $|B|n^2$ linear equations in n^2 variables, based on the matrix equations $AX = XA$ where $A \in B$ and X represents an arbitrary element of $\mathcal{C}_H(A)$. In general, the time to solve such a system is $O(|B|rn^4 \log q)$ where r is the rank of the system with $n \leq r \leq n^2$. A special case that is important in practice is when $\langle C \rangle$ acts irreducibly on the underlying vector space. In this case, $\mathcal{C}_H(C)$ is a field $GF(q^r)$ where $r|n$ and G can be viewed as an absolutely irreducible subgroup of $GL(n/r, q^r)$. An algorithm for computing $\mathcal{C}_H(C)$ in this case is described in [7].

The next lemma sometimes provides a faster way to verify that B is a base for G^C. The proof is clear.

Lemma 3.8 *If $\langle B \rangle$ is transitive in its conjugate action on C, then B is a base.*

The previous results allow us to to obtain a base B for the conjugation action of G on C. By Lemma 3.1 and Assumption 3.2, $G/\mathcal{C}_G(C)$ has a faithful, transitive representation on $\mathcal{O}$. The following result is the start point for extending $\mathcal{O}_B = \{\langle b \rangle : b \in B\}$ to a base for $G^{\mathcal{O}}$.

Lemma 3.9 *Assume that B is a base for G^C. Let $\Gamma = \cup_{\langle b \rangle \in \mathcal{O}_B} \langle b \rangle^{\langle B \rangle} \subseteq \mathcal{O}$ and let L be the point stabilizer $G_{\mathcal{O}_B}$ of $\mathcal{O}_B$. Then $L/\mathcal{C}_G(C)$ acts faithfully on Γ and the action of an element of L on Γ is uniquely determined from its action on each $\langle b \rangle \in \mathcal{O}_B$. In particular, this shows that $|L/\mathcal{C}_G(C)| \leq (p-1)^{|\mathcal{O}_B|}$ and $L/\mathcal{C}_G(C)$ is abelian.*

Proof: To prove that $L/\mathcal{C}_G(C)$ acts faithfully on Γ, it suffices to show that $G_\Gamma = \mathcal{C}_G(C)$. Let $K = G_\Gamma$. Since B leaves Γ invariant, it follows that K is normalized by $\langle B \rangle$. On the other hand, K normalizes each subgroup of $\mathcal{O}_B$, and thus

$$[K, \langle b \rangle] \subseteq K \cap \langle b \rangle, \ \forall \langle b \rangle \in \mathcal{O}_B.$$

Since $\langle b \rangle \cong Z_p$ for some prime p, either K commutes with B, or else there exists $\langle b \rangle \in \mathcal{O}_B$ with $[K, \langle b \rangle] = \langle b \rangle$. If the latter occurs, then $b \in K$, and so b normalizes, and hence centralizes each $\langle b' \rangle \in \mathcal{O}_B$. This implies that $b \in \mathcal{C}_G(B) = \mathcal{C}_G(C)$, which contradicts Assumption 3.2. Thus K commutes with B and so $K = \mathcal{C}_G(C)$ as required.

Now, let $h \in L$. The action of h on $\langle b \rangle \in \mathcal{O}_B$ is determined by a rule that specifies the value of $b^h \in \langle b \rangle$, for each $\langle b \rangle \in \mathcal{O}_B$. We must show that this rule uniquely determines the action of h on Γ. Let $\langle y \rangle$ be a point of Γ. Then there exists $b_0 \in B$ and $b = b_1 b_2 \cdots b_k \in \langle B \rangle$ with each $b_i \in B$, such that $\langle y \rangle = \langle b_0 \rangle^b$. But then $\langle y^h \rangle = \langle {b_0}^{bh} \rangle = \langle {b_0}^{hb'} \rangle = \langle {b_0}^{b'} \rangle$ where $b' = {b_1}^h {b_2}^h \cdots {b_k}^h$.

Since each element of $L/\mathcal{C}_G(C)$ is uniquely determined by its action on the points of $\mathcal{O}_B$, it follows that $|L/\mathcal{C}_G(C)| \leq (p-1)^{|\mathcal{O}_B|}$. To show that $L/\mathcal{C}_G(C)$ is abelian, it suffices to prove that the commutator $[x, y]$ of arbitrary elements $x, y \in L$ acts trivially on each $\langle b \rangle \in \mathcal{O}_B$. But this follows from the fact that x and y normalize $\langle b \rangle$ and $Aut(\langle b \rangle)$ is abelian. □

Lemma 3.9 and the next lemma allow us to deduce an upper bound on the size of a base for $G^{\mathcal{O}}$.

Lemma 3.10 (from [2], Lemma 5.1) *Let $G \subseteq \mathrm{Sym}(\Omega)$ act transitively on Ω. Let b be the size of a minimal base for G and m the minimal size of support for a non-identity element of G. Then $bm \geq |\Omega|$.*

Corollary 3.11 *G has a base of size at most $|\mathcal{O}_B|(1 + \log_2(p-1))$. Consequently, the minimal support of an element of G acting on $\mathcal{O}$ is at least $|\mathcal{O}|/|\mathcal{O}_B|(1+\log_2(p-1))$. In particular, if $p = 3$, then G has a base of size at most $2|\mathcal{O}_B|$ and the minimal support of an element is at least $|\mathcal{O}|/(2|\mathcal{O}_B|)$.*

Lemma 3.9 and Corollary 3.11 give us the tools to extend the base $\mathcal{O}_B$ into a base for the action of G on $\mathcal{O}$. Lemma 3.9 provides $(p-1)^{|\mathcal{O}_B|}$ candidates for elements of L. Let $\mathcal{H}$ be the set of $(p-1)^{|\mathcal{O}_B|} - 1$ rules that specify the possible action of a non-identity element of L on the $|\mathcal{O}_B|$ subgroups $\langle b \rangle \in \mathcal{O}_B$. For each $h \in \mathcal{H}$, the rules prescribed by h allows one to compute $\langle y \rangle^h$ without knowing in advance that there exists a corresponding element of L (or even of $GL(n, q)$).

The following code describes a randomized algorithm for extending $\mathcal{O}_B$ to a base $\mathcal{O}_{B'}$ of $G^{\mathcal{O}}$. The algorithm is *Las Vegas* in the sense that the only uncertainty is how long the procedure takes, but the answer returned is always correct.

Set $\mathcal{O}_{B'} \leftarrow \mathcal{O}_B$, $\mathcal{H}' \leftarrow \mathcal{H}$
While $\mathcal{H}' \neq \emptyset$
  Let $\langle y \rangle \in \Gamma$ be randomly chosen
  For each $h \in \mathcal{H}'$
    If $\langle y \rangle^h \neq \langle y \rangle$
      Set $\mathcal{H}' \leftarrow \mathcal{H}' \setminus \{h\}$
      If $\langle y \rangle \notin \mathcal{O}_{B'}$
        Append $\langle y \rangle$ to $\mathcal{O}_{B'}$

Corollary 3.11 shows that for a fixed element of $\mathcal{H}$, the probability of a random $\langle y \rangle \in \Gamma$ being moved by h is at least $1/|\mathcal{O}_B|(1+\log_2(p-1))$. Thus with constant probability, after we have examined $|\mathcal{O}_B|(1+\log_2(p-1))$ randomly chosen points, we find a $\langle y \rangle$ moved by h. In particular, with constant probability, the procedure described above terminates after we have examined $2^{|\mathcal{O}_B|}|\mathcal{O}_B|(1+\log_2(p-1))$ randomly chosen elements of $\mathcal{O}$. This works fairly well when $|\mathcal{O}_B|$ is small. Further work is required to make it efficient when $|\mathcal{O}_B|$ is "large".

4 Verification of Simplicity

We continue with the notation of section 3. Thus $C = \{x^g : g \in G\}$ and $\mathcal{O} = \{\langle x^g \rangle : g \in G\}$. In this section, we present an elementary test for simplicity of the permutation representation of G on $\mathcal{O}$. We operate under Assumption 3.2, so that C is not in the kernel of the permutation representation. Thus, by Lemma 3.1, $\bar{G} = G/\mathcal{C}_G(C)$ acts faithfully on $\mathcal{O}$. We assume that we know a base and strong generating set for G, and hence the order of G. We also assume that G is primitive. Otherwise, G has a permutation representation of smaller degree and we use that one instead.

Observe that $\{\langle xz \rangle \in \mathcal{O} : z \in \mathcal{C}_G(C)\}$ is a block of $\mathcal{O}$ containing $\langle x \rangle$. By our assumption of primitivity, this block must be trivial. Hence if $[x, g] \in \mathcal{C}_G(C)$, then $[x, g] = 1$. In particular, for each $x \in C$, $\mathcal{C}_{\bar{G}}(\bar{x}) = \bar{\mathcal{C}}_G(x)$. As a consequence of this fact, we may assume that $\mathcal{C}_G(C) = 1$ and that $\bar{G} = G$.

Proposition 4.1 *If K is a regular normal subgroup of G, then K is an elementary abelian r-subgroup for some prime r.*

Proof: The stabilizer in G of the point $\langle x \rangle \in \mathcal{O}$ is $\mathcal{N}_G(\langle x \rangle)$. Since K is a regular normal subgroup of G, it follows that $\mathcal{N}_K(\langle x \rangle) = 1$. In particular, $\mathcal{C}_K(x) = 1$. An equivalent formulation is that K admits a fixed point free automorphism x of order p. By a theorem of Thompson, ([8], Kapitel V, Haupsatz 8.14) K is nilpotent. But $G^{\mathcal{O}}$ is primitive, and so K must be an elementary abelian r subgroup for some prime r. □

Proposition 4.2 *Let $N = \langle C \rangle$. Then one of the following two cases occurs:*

1. *G has a regular normal subgroup K, and $N = \langle x \rangle K$.*
2. *G has a unique minimal normal subgroup H and $N = \langle x \rangle H$.*

Proof: By the previous result, if K is a regular normal subgroup of G, then K is an elementary abelian r subgroup for some prime r. But then $G = KN_G(\langle x \rangle)$ and (1) follows. To prove (2), we may assume that G does not have a regular normal subgroup. In this case, G has a unique minimal normal subgroup H, which must be transitive. Thus if $g \in G$, then $g = g_1 h$ where $g_1 \in N_G(\langle x \rangle)$ and $h \in H$. Thus

$$(\langle x \rangle)^{g_1 h} = (\langle x \rangle)^h \subseteq \langle x \rangle H,$$

which implies that $N \subseteq \langle x \rangle H$. On the other hand, since N contains both $\langle x \rangle$ and H, equality follows. □

It is fairly straightforward to apply this proposition to test the simplicity of G. First compute $N = \langle C \rangle$. If $|N| < |G|$, then G is not simple. So, assume that $G = N$. Case (1) can only arise if $|G|$ involves exactly two distinct primes. But then G is solvable by Burnside's theorem. Thus if $|G|$ involves more than 2 primes, then we can assume that $N = H\langle x \rangle$ where H is the unique minimal normal subgroup of G.

At this point, simplicity can be verified by showing that $x \in H$, or equivalently that $x \in [N, N]$. An alternative way to verify that $x \in H$ is to check if x is conjugate in G to one of its distinct powers.

5 Application to Lyon's Group

In this section, we assemble the pieces that relate to the construction of the Lyons group. This has the useful side effect of testing the methods described in the previous sections on a large example of independent interest. Initially, we were given two matrices, $\alpha, \beta \in GL(111, 5)$, where $|\alpha| = 2$ and $|\beta| = 5$. These matrices were developed by Robert Wilson and provided to us by Klaus Lux. We assume that we have constructed the hash table $\mathcal{T}$ used to store $\mathcal{O}$ but do not know the exact structure of G. The goal is to apply the techniques developed in sections 3 and 4.

The first step in constructing a base using the theory developed in section 3 is to construct a subset $B \subseteq C$ with the property that $\mathcal{C}_G(B) = \mathcal{C}_G(C)$. In executing the algorithm described in Theorem 3.4, if it is suspected that $\langle C \rangle$ acts absolutely irreducibly on V, so that $\mathcal{C}_G(C) \subseteq \mathcal{C}_{GL(n,q)}(C) = Z(GL(n, q))$, then it suffices to find a subset $B \subseteq C$ so that $\langle B \rangle$ acts absolutely irreducibly as well. One can use the following heuristic.

Initialize B to 3 randomly chosen elements of C
While $\langle B \rangle$ does not act absolutely irreducibly on V
  Add a randomly chosen element of C to B

The test for absolute irreducibility can be performed using Parker's Meataxe [11] as implemented in GAP. In the case at hand, this procedure terminates after one iteration, which yields a set B containing three elements satisfying $\mathcal{C}_G(B) = \mathcal{C}_G(C)$.

The next step is extending $\mathcal{O}_B$ to a base for $G^{\mathcal{O}}$. This is done through application of the procedure described in section 3. In particular, this procedure finds only one additional base point, thus yielding a base $\mathcal{O}_{B'}$ with precisely four points.

There are numerous strategies available for computing a strong generating set for $G^{\mathcal{O}}$ given a base in advance. Because we suspect that $G^{\mathcal{O}}$ is isomorphic to the Lyons group, one strategy is to use a heuristic procedure to obtain a candidate strong generating set $\mathcal{S}$ and then use either the Monte Carlo strong generating test described in [2] or the

deterministic Brownie-Sims *Verify* routine as implemented in Magma. The knowledge of a small base in advance speeds up both procedures.

In describing the heuristic procedure for computing $\mathcal{S}$, we denote the base points by $\langle b_i \rangle$, $i = 1, \ldots, 4$, with $\langle b_1 \rangle$ understood to be the root of the tree $\mathcal{T}_1$ used to store $\mathcal{O}$. The construction of $\mathcal{T}_i$ uses the method described in [5] and employs the same vectors for computing signatures.

Initialize $\mathcal{S} \leftarrow S$ and $\Sigma_1 \leftarrow S$
For $i = 2$ to 4 do
  Let Σ_i be c_i randomly chosen Schreier generators computed using Σ_{i-1} and $\mathcal{T}_{i-1}$
  Compute $\mathcal{T}_i$ with root $\langle b_i \rangle$ and generating set Σ_i
  Append Σ_i to $\mathcal{S}$

In each case, the Schreier generators stored in Σ_i are computed from a randomly chosen element of $\mathcal{T}_i$ and Σ_i. In this sense, the Schreier generator may be characterized as randomly chosen from amongst all possible Schreier generators. In general though, one can not show that with constant probability, such an element enlarges the current point stabilizer subgroup generated by Σ_i. This is why the procedure is heuristic. The parameters c_2, c_3, c_4 are inputs to the procedure. In the application at hand, c_2 and c_3 were set to 100 and c_4 was set to 1, since Σ_4 contained only the identity element. The sizes of $\mathcal{T}_2$ and $\mathcal{T}_3$ are respectively 7,185,024 and 750. It should be further noted that the total storage required for all the data structures $\mathcal{T}_i$ is slightly more than the storage needed for 3 permutations acting on $9,606,125$ points. The storage required for the Schreier generators is negligible in comparison, which is why we have the luxury of choosing 100 Schreier generators for each point stabilizer subgroup.

The next step is to apply to $\mathcal{S}$ one of the strong generating tests referred to earlier. In either test, one necessarily has to sift an element of the group through the data structures $\mathcal{T}_i$, $i = 1, 2, 3$. In the matrix domain, we do not have to perform any matrix multiplications. This follows from the existence of the vector $v = v_{init}$ that was used for constructing signatures. So, for example, if we are attempting to sift $g \in G_{\{\langle b_1, \ldots, b_{i-1} \rangle\}}$ through $\mathcal{T}_i$, then the algorithm described in [5] will hash into the table $\mathcal{T}_i$ using the two signature vectors $v^{g^{-1} b_i g}$ and $v^{g^{-1} {b_i}^2 g}$. The result is a word w_i such that $w_i \in G_{\{\langle b_1, b_2, \ldots, b_{i-1} \rangle\}}$ and $g w_i^{-1}$ fixes $\langle b_i \rangle$. In this way, we see that sifting at each level only involves vector-matrix computations. The total complexity is related to the total depths of the trees $\mathcal{T}_i$, $i = 1, 2, 3$. The depths of the trees $\mathcal{T}_1$ and $\mathcal{T}_2$ are 5 and 3 respectively, while the 750 elements of $\mathcal{T}_3$ can be stored directly. At any rate, in sifting, we save a factor of 111, or n in general.

Having verified the strong generating set, and hence the order of $G^{\mathcal{O}}$, we now wish to verify simplicity. We use the method described in section 4. In the case where $|G^{\mathcal{O}}| = |Ly|$, primitivity on $\mathcal{O}$ can be checked from the size of the orbits of the point stabilizer, since any block is a union of a subset of these orbits and the size of a block divides $|\mathcal{O}|$. Proving that $\langle C \rangle^{\mathcal{O}} = G^{\mathcal{O}}$ can be reduced to verifying that $\langle B \rangle^{\mathcal{O}} = G^{\mathcal{O}}$. The easiest way to do this, again working within our space constraints, is to start from scratch, but this time with generating set B instead of $\{\alpha, \beta\}$ and constructing the permutation representation on $\langle b_1 \rangle^{\langle B \rangle} \subseteq \mathcal{O}$. This can be done more efficiently than in the initial computation of a strong generating set for $G^{\mathcal{O}}$, since we know in advance, $|G^{\mathcal{O}}|$ and $|\mathcal{T}|_i, i = 1, 2, 3$. Finally, we "suspect" that each $x \in C$ is inverted by 1/2 of the elements in $G^{\mathcal{O}}_{\langle x \rangle}$, and so finding an element that confirms this fact is easy. This certifies the simplicity of the permutation group.

6 Acknowledgments

The authors gratefully acknowledge discussions with the following people: Alexander Hulpke, Bill Kantor, Wolfgang Lempken, Steve Linton, Gene Luks, Klaus Lux, Gerhard Michler, Reiner Staszewski and Michael Weller. The authors especially acknowledge that the matrix generators for the Lyons group, for which these techniques were tested, were developed by Robert Wilson and provided to us by Klaus Lux. Finally, we thank Bill Kantor for originally suggesting the "testbed" of the Lyons group.

7 Bibliography

References

[1] L. Babai and R. Beals, "Las Vegas Algorithms for Matrix Groups", *Proc. 24th IEEE FOCS* (1993), pp. 427–436.

[2] L. Babai, G. Cooperman, L. Finkelstein and Á. Seress, "Nearly Linear Time Algorithms for Permutation Groups with a Small Base", *Proc. of the 1991 International Symposium on Symbolic and Algebraic Computation* (ISSAC '91), Bonn, pp. 200–209, July, 1991.

[3] R. Beals, "Computing blocks of imprimitivity for small-base groups in nearly linear time," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **11**, 1993, pp. 17–26, L. Finkelstein and W.M. Kantor (eds.), AMS, Providence, RI. **11**, 1993, pp. 17–26, L. Finkelstein and W.M. Kantor (eds.), AMS, Providence, RI.

[4] G. Butler, "The Schreier Algorithm for Matrix Groups", SYMSAC '76 (Proc. 1976 ACM Symp.on Symbolic and Algebraic Computation. Yorktown Heights, 1976) R.D. Jenks (ed.). ACM. New York. pp 167-170.

[5] G. Cooperman, L. Finkelstein, B. York and M. Tselman, "Constructing Permutation Representations for Large Matrix Groups", *Proc. of International Symposium on Symbolic and Algebraic Computation (ISSAC '94)*, ACM Press, pp. 134–138.

[6] G. Cooperman, L. Finkelstein, M. Tselman and B. York, "Constructing Permutation Representations for Matrix Groups", preprint.

[7] D. F. Holt and S. Rees "Testing modules for irreducibility", manuscript.

[8] B. Huppert, "Endliche Gruppen I", Springer Verlag, New York, 1967.

[9] J. Leon, "On an Algorithm for Finding a Base and Strong Generating Set for a Group Given by a Set of Generating Permutations", *Math. Comp.* **35** (1980), pp. 941–974.

[10] R. Lyons, "Evidence for a New Finite Simple Group", Journal of Algebra, **20** (1972), 540–569.

[11] R. Parker, 'The computer calculation of modular characters. (The Meat-Axe)', in M. Atkinson (ed.), *Computational Group Theory*, Academic Press, London, 267-74, 1984.

[12] C.C. Sims, "Computation with Permutation Groups", in *Proc. Second Symposium on Symbolic and Algebraic Manipulation*, edited by S.R. Petrick, ACM Press, New York, 1971, pp. 23–28.

[13] C.C. Sims, "The Existence and Uniqueness of Lyons Group", Proceeding of the Gainseville Conference, (1972), pp 138–141.