Computing conjugacy class representatives in permutation groups

John J. Cannon, Derek F. Holt

Abstract

We describe a significantly improved algorithm for computing the conjugacy classes of a finite permutation group with trivial soluble radical. We rely on existing methods for groups that are almost simple, and we are concerned here only with the reduction of the general case to the almost simple case.

Dedicated to Charles Leedham-Green on the occasion of his 65th birthday

1 Introduction

We consider the problem of computing representatives of the conjugacy classes in a finite group G. Let N be the soluble radical (that is, the largest soluble normal subgroup) of G. If we can solve the conjugacy classes problem in G/N, then we can solve it in G by lifting the solution for G/N through elementary abelian layers of N regarded as modules for G over prime fields. This technique was first described in [5] for p-groups and in [8] for finite soluble groups, but it works equally well for general finite groups; see Section 8.8 of [6] for a more recent treatment. It is not a polynomial-time process, because it involves enumerating complete orbits in an action on the vectors of a vector space, but it has been observed to perform well in a wide range of practical examples. We shall therefore restrict our attention to groups with trivial soluble radical.

A group G is called *almost simple*, if there is a nonabelian simple group S with $S \leq G \leq \mathrm{Aut}(S)$. In this paper, we shall assume that methods are already available for computing conjugacy class representatives in finite almost simple groups. Current implementations generally use methods based on choosing random elements of the group for this purpose, as summarized in Section 4 of [4]. Note that the technique described in [1] of first finding representatives of classes of prime power order and their centralizers can often be applied to speed up calculations in larger groups.

More research needs to be carried out on the almost simple case, and we suspect that the ultimate method will involve identifying the isomorphism type of the group and then solving the problem by using detailed information about the specific simple group involved.

There are several other types of computations with finite groups in which the approach of reducing first to the case of groups with soluble radical and then to almost simple groups has been usefully employed. Examples are the calculation of maximal subgroups [3] and of automorphism groups [2].

In this article, we shall be concerned only with reducing the general problem for groups with trivial soluble radical to the case when the group G is almost simple. The methods described in [4] for this reduction perform well if there is a subgroup of G of very small index that decomposes nontrivially as a direct product, but when no such normal subgroup can be found, they can be prohibitively slow. This shortcoming has been observed already by Hulpke in [7], who proposed new methods based on calculating conjugacy classes in wreath products.

The approach that we shall describe in Section 3 also involves conjugacy classes in wreath products, but differs from [7] in that its treatment of classes inside and outside of the base group of the wreath product is uniform. In [7], more effort is devoted to the classes in the base group than is done here, and the performance for those classes might consequently be superior to ours, particularly with regard to memory usage. However, we hope that our uniform treatment of all of the classes will result in superior performance overall in difficult examples. Our implementation is limited in practice only by the size of the output; that is, by the number of conjugacy classes.

Our method relies on the facts that we can embed a finite group G with trivial soluble radical into a direct product of wreath products of almost simple groups by permutation groups, and that the conjugacy classes of elements in wreath products can be simply described. In Section 2, we state and prove the results that we need on conjugacy classes in wreath products.

We have implemented our algorithm in Magma for finite permutation groups, and we present some performance statistics, including comparisons with run-times for the existing methods based on [4], in Section 4. We are unfortunately not in the position to carry out useful experimental comparisons with the methods of [7], since the implementation described there is in GAP, and comparisons of run-times would not be meaningful.

There are a number of related problems that frequently arise in connection with calculations involving conjugacy classes. The centralizers of the class representatives are often required, and we may need the ability to identify rapidly the class representatives of given elements of G, and perhaps also to find the associated con-

jugating elements. For example, when calculating character tables of finite groups, information on the power maps of the classes is typically required, and this involves the identification of class representatives of specified elements of G.

We have implemented methods for solving each of these problems that work well in practice provided that we can solve the equivalent problems in almost simple groups. We shall comment briefly on these methods at the end of Section 3.

2 Conjugacy classes in wreath products

This section is devoted to some theoretical results on conjugacy classes in wreath products. In general, for a subgroup H of a group G, we shall refer to the orbits of the conjugation action of H on the elements of G as the H-classes of G.

Let A be any group, let P be a permutation group acting on the set $\{1\,..\,d\}$, and let $W := A \wr P$. Then W is a semidirect product of A^d by P, and elements of W have the form (g, x) with $g \in P$, $x \in A^d$, and $x = (x_1, \ldots, x_d)$ with $x_i \in A$. In general, for $x \in A^d$, we denote the i-th component of x by x_i. The action of P on A^d in W is given by:

$$(g, 1)^{-1}(1, (x_1, \ldots, x_d))(g, 1) = (1, (x_{1^{g^{-1}}}, \ldots, x_{d^{g^{-1}}})).$$

Hence, for $g \in P$, $x, z \in A^d$, we have

$$(1, z)^{-1}(g, x)(1, z) = (g, y), \text{ where } y_i = z^{-1}_{i^{g^{-1}}} x_i z_i \text{ for } 1 \leq i \leq d. \tag{1}$$

Proposition 2.1. *With the above notation, let $x^{(1)}, \ldots, x^{(k)}$ be representatives of the conjugacy classes of A. Fix an element $g \in P$, and let $r_1, r_2, \ldots, r_s$ be representatives of the cycles of g in its action on $\{1\,..\,d\}$. Then*

$$R := \{\, (g, x) \mid x_i = 1 \text{ for } i \notin \{r_1, \ldots, r_s\},\ x_i \in \{x^{(1)}, \ldots, x^{(k)}\} \text{ for } i \in \{r_1, \ldots, r_s\} \,\}$$

is a set of representatives of the A^d-classes of elements of W of the form (g, x) for $x \in A^d$.

Proof. We show first that any element $(g, x) \in W$ is conjugate under A^d to an element in R. By using Equation 1, it is not hard to see that, for $x \in A^d$, we can find $z \in A^d$ such that $(1, z)^{-1}(g, x)(1, z) = (g, y)$, where $y_i = 1$ for $i \notin \{r_1, \ldots, r_s\}$. For example, for each r_j, we can choose $z_{r_j} = 1$, and then, for the remaining points in the orbit $r_j^{\langle g \rangle}$, choose $z_{\sigma(k)}$ (with $\sigma(k) = r_j^{g^k}$ and $k = 1, 2, \ldots$) to make $y_{\sigma(k)} = 1$.

Then, given $(g, y) \in W$ with $y_i = 1$ for $i \notin \{r_1, \ldots, r_s\}$, we can conjugate (g, y) by an element $(1, z)$ for which z_i is constant on each orbit of $\langle g \rangle$ to get an element of R.

Conversely, we need to show that no two elements of R are conjugate by an element of A^d. If $(1,z)^{-1}(g,x)(1,z) = (g,y)$ with $(g,x),(g,y) \in R$, then $x_i = y_i = 1$ for $i \notin \{r_1,\ldots,r_s\}$. It follows from Equation 1 that z_i is constant on the orbits of $\langle g \rangle$. But then Equation 1 implies that x_i and y_i are A-conjugate for each $i \in \{r_1,\ldots,r_s\}$ and so, by definition of R, $x = y$. □

The arguments in the final two paragraphs of the above proof yield the following slightly more general result.

Proposition 2.2. *Let $g \in P$ and $r_1,\ldots,r_s$ be as in Proposition 2.1, and let $S \leq A$. Then two elements $(g,x),(g,y) \in W$ with $x_i = y_i = 1$ for $i \notin \{r_1,\ldots,r_s\}$ are S^d-conjugate if and only if x_i and y_i are S-conjugate for each $i \in \{r_1,\ldots,r_s\}$.*

3 The reduction to almost simple groups

Let G be a finite group with trivial soluble radical. Then the socle M of G is isomorphic to a direct product of finite nonabelian simple groups. Let the simple factors of M be

$$S_{11}, S_{12}, \ldots, S_{1d_1}, S_{21}, \ldots, S_{2d_2}, \ldots, \ldots, S_{r1}, \ldots, S_{rd_r},$$

where S_{ij} and S_{kl} are conjugate in G if and only if $i = k$.

For $1 \leq i \leq r$, let S_i be a group isomorphic to the S_{ij}; let P_i be the transitive permutation group of degree d_i induced by the conjugation action of G on the socle factors $\{S_{i1},\ldots,S_{id_i}\}$; and let $A_i := N_G(S_{i1})/C_G(S_{i1})$. So A_i is an almost simple group with (up to isomorphism) $S_i \leq A_i \leq \mathrm{Aut}(S_i)$.

The groups S_{ij} with associated normalizers and centralizers and the permutation groups P_i can be computed rapidly using existing functionality for permutation groups for examples within the scope of the algorithms described in this paper. We need also to find a permutation group isomorphic to A_i, which we shall also denote by A_i, together with an explicit epimorphism $\psi_i : N_G(S_{i1}) \to A_i$ with kernel $C_G(S_{i1})$. In many examples, the action of $N_G(S_{i1})$ on the orbits of $C_G(S_{i1})$ conveniently provides such a representation. If not, then we use some other easily defined action, such as the action by multiplication on the cosets of the subgroup generated by $C_G(S_{i1})$ and a suitable Sylow subgroup of $N_G(S_{i1})$.

As described in Section 3.2 of [2] and also in Section 3.2 of [3], we can use the maps ψ_i to compute homomorphisms $\rho_i : G \to W_i := A_i \wr P_i$. We can then combine the ρ_i to yield a monomorphism $\rho : G \to W := W_1 \times \cdots \times W_r$. Note that ρ induces an isomorphism from M to the socle of W.

We start the conjugacy class computation by recursively computing the classes of G/M. Generally, G/M is a moderately small group, so this presents no difficulty. We take each class representative $\hat{g}M$ of G/M in turn, let $\hat{g}$ be an inverse image in G, and let $g = \rho(\hat{g})$. We shall compute the class representatives of G that map onto $\hat{g}M$ in G/M by computing the class representatives of $\mathrm{Im}(\rho)$ of the form gx, for $x \in \mathrm{Soc}(W)$, and then taking their inverse images under ρ to get the representatives in G.

We compute the class representatives of $\mathrm{Im}(\rho)$ of the required form gx in two steps. In the first step, we find a set R of $\mathrm{Soc}(W)$-class representatives of the form gx. In the second step, we obtain $\mathrm{Im}(\rho)$-class representatives by taking orbit representatives of the action of $C := \rho(\hat{C})$ on R, where $\hat{C}$ is the inverse image in G of $C_{G/M}(\hat{g}M)$.

Before beginning the first step, we compute conjugacy class representatives of each of the almost simple groups A_i (recall that we are assuming in this paper that we can solve the problem for almost simple groups). In fact, we need the S_i-class representatives in A_i, where $S_i = \mathrm{Soc}(A_i)$, but since A_i/S_i is small, these are easily found from the A_i-class representatives and their centralizers in A_i. We choose these representatives in such a way that two representatives y, z for which yS_i and zS_i are conjugate in A_i/S_i satisfy $yS_i = zS_i$.

Throughout our technical description, we shall refer to the following illustrative example. Let $\mathrm{Im}(\rho) = (\mathrm{Sym}(5) \wr \mathrm{Alt}(4)) \times (\mathrm{Sym}(5) \wr \mathrm{Sym}(2))$. So $\mathrm{Soc}(W)$ has six factors, all isomorphic to $\mathrm{Alt}(5)$, which are permuted with orbits of length 4 and 2 under the action of $\mathrm{Im}(\rho)$. The lists of representatives of the $\mathrm{Alt}(5)$-classes of $\mathrm{Sym}(5)$ are

$$X_e := [\,1,\ (1,2)(3,4),\ (1,2,3),\ (1,2,3,4,5),\ (1,2,3,5,4)\,] \quad \text{and}$$

$$X_o := [\,(1,2),\ (1,2)(3,4,5),\ (1,2,3,4)\,],$$

which map onto $\mathrm{Alt}(5)$ and $(1,2)\mathrm{Alt}(5)$, respectively.

In general, we have $g = ((g_1, x_1), \ldots, (g_r, x_r)) \in \mathrm{Im}(\rho)$ with $(g_i, x_i) \in W_i$, $g_i \in P_i$, and $x_i \in A_i^{d_i}$. By Proposition 2.1, there exist (and we can compute) elements $w_i \in A_i^{d_i} \leq W_i$ such that $(g_i, x_i)^{w_i} = (g_i, y_i)$, where the only nontrivial components of y_i lie in orbit representatives of the cycles of g_i on $\{1\,..\,d_i\}$, and these components lie in a set of conjugacy class representatives of A_i. So $g^w = ((g_1, y_1), \ldots, (g_r, y_r))$, with $w = w_1 w_2 \cdots w_r$.

In our example, we choose $g = ((g_1, x_1), (g_2, x_2))$, with $g_1 = (1,2)(3,4) \in \mathrm{Alt}(4)$, $x_1 = ((1,2,3), (3,4,5), (1,2)(3,4), 1) \in \mathrm{Sym}(5)^4$, $g_2 = (1,2) \in \mathrm{Sym}(2)$, $x_2 = ((1,2), (3,5,4)) \in \mathrm{Sym}(5)^2$. Then, with $w_1 = (1, (3,5,4), 1, 1)$, $w_2 = ((4,5), (3,5))$, and $w = w_1 w_2$, we find that $g^w = ((g_1, y_1), \ldots, (g_r, y_r))$ with $y_1 = ((1,2,3,4,5), 1, (1,2)(3,4), 1)$ and $y_2 = ((1,2)(3,4,5), 1)$. (Here we are identifying $\mathrm{Sym}(5)^4$ and $\mathrm{Sym}(5)^2$ with their images under the their natural embeddings into W_1 and W_2.)

To simplify the following description, we shall assume that $w = 1$. In practice, we can perform our calculations with g^w in place of g, and conjugate our class representatives by w^{-1} before computing their inverse images under ρ. In our example, we have $w \in \mathrm{Soc}(W) = \rho(M)$, so we can simply replace g by g^w; but in general we may not even have $w \in \rho(G)$, so we need eventually to conjugate back by w^{-1}.

By Proposition 2.2, two elements (g_i, y_i), (g_i, y_i') with the property defined above are conjugate under $\mathrm{Soc}(W_i) \cong S_i^{d_i}$ if and only if each of the corresponding nontrivial components of y_i and y_i' are conjugate under S_i. Let us write $y_i = y_{i1}y_{i2}\cdots y_{is_i}$, where g_i has s_i cycles, and each y_{ij} has all of its components trivial, except possibly for the representative of the j-th cycle of g_i. The required $\mathrm{Soc}(W)$-conjugacy class representatives in $g\mathrm{Soc}(W)$ have the form $((g_1, y_1'), \ldots, (g_r, y_r'))$, where $y_i' = y_{i1}'y_{i2}'\cdots y_{is_i}'$, and the nontrivial component of each y_{ij}' is one of the pre-computed S_i-class representatives of A_i with $y_{ij}S_i = y_{ij}'S_i$. From these pre-computed lists of S_i-class representatives of A_i, we can list the possibilities for each y_{ij}', and thereby obtain a complete set R of $\mathrm{Soc}(W)$-class representatives of elements of the form $g\mathrm{Soc}(W)$.

In our example, we have $y_{11}' = (h, 1, 1, 1)$ with $h \in X_e$, $y_{12}' = (1, 1, h, 1)$ with $h \in X_e$, and $y_{21}' = (h, 1)$ with $h \in X_o$.

We move on now to the second part of the process. The group C, in its action by conjugation, permutes the $\mathrm{Soc}(W)$-classes that map onto g, and so it induces a corresponding action on the set R of $\mathrm{Soc}(W)$-class representatives. (Note, however, that since we are now performing our calculations with g^w in place of g, we must replace C by C^w before proceeding.) The class representatives of $\mathrm{Im}(\rho)$ that map onto g consist of representatives of the orbits of this action of C on R.

We shall now simplify our notation by eliminating the first subscript in the elements y_{ij}'. Let $\tilde{g} := ((g_1, 1), \ldots, (g_r, 1))$. Let $B_1, B_2, \ldots, B_s$ be the list of socle factors of W that we have chosen as orbit representatives of the cycles of g. So these are the socle factors in which nontrivial components of the y_i' may occur. For $1 \le k \le s$, let $U_k := \{z_{k1}, \ldots, z_{ku_k}\} \subset B_k$ be the set of allowed elements y_{ij}' for that socle factor. So we have

$$R = \{\, \tilde{g} z_{1j_1} z_{2j_2} \cdots z_{sj_s} \mid \text{ with } 1 \le j_k \le u_k \text{ for } 1 \le k \le s \,\}.$$

In our example, $s = 3$, and $U_1 = \{\, (h, 1, 1, 1) \mid h \in X_e \,\}$, $U_2 = \{\, (1, 1, h, 1) \mid h \in X_e \,\}$, and $U_3 = \{\, (h, 1) \mid h \in X_o \,\}$. So $|R| = |U_1 \times U_2 \times U_3| = 75$.

So there is an obvious bijection between R and $U := U_1 \times \cdots \times U_s$. We shall compute our action of C on U, and the orbit representatives will then correspond to the required class representatives.

Let c be a generator of C. Then, for an element $\tilde{g} z_{1j_1} z_{2j_2} \cdots z_{sj_s} \in R$, $(\tilde{g} z_{1j_1} z_{2j_2} \cdots z_{sj_s})^c$ is conjugate under $\mathrm{Soc}(W)$ to another element $\tilde{g} z_{1j_1'} z_{2j_2'} \cdots z_{sj_s'} \in R$, which we need to

identify. Now c permutes the orbits of $\langle g \rangle$ on the socle factors of W, and $B_1, \ldots, B_s$ are representatives of these orbits. So we can write B_{k^c} for the representative of the orbit of B_k^c. Putting $l = k^c$, it is not hard to see that the element $z_{lj'_l}$ in the above expression depends only on z_{kj_k}. Furthermore, we have

$$(\tilde{g} z_{1j_1} z_{2j_2} \cdots z_{sj_s})^c = (\tilde{g} z_{1j'_1} z_{2j'_2} \cdots z_{sj'_s})^{v_{1^c} v_{2^c} \cdots v_{s^c}},$$

where $v_k \in \mathrm{Soc}(W)$ has all of its nontrivial components in those socle factors that lie in the cycle of g containing B_k, and v_{k^c} depends only on z_{kj_k}.

For each $z_{kj_k} \in U_k$, we can carry out conjugacy tests under the appropriate subgroups of $\mathrm{Soc}(W)$ to identify the elements $z_{lj'_l}$ and also the corresponding conjugating elements v_l, where $l = k^c$. The method of calculating the v_k is described in the proof of Proposition 2.1. This involves calculating one conjugating element within the simple group B_k together with a few group element multiplications, so the problem is once again reduced to solving the same problem within an almost simple group.

So our action of C on U is induced from an action of C on $U_1 \cup \cdots \cup U_s$ in which the sets $U_1, \ldots, U_s$ are blocks. Although the set U can be unpleasantly large, this observation simplifies the computation of the action considerably. We do, however, require orbit representatives of C on U, and so we still need to compute this action explicitly, which can result in the use of significant amounts of memory in larger examples. A referee commented that we could in many cases reduce the memory requirements by computing the fusion of conjugacy classes under the action of C in two stages: firstly under the subgroup of C that normalizes all of the socle factors, and then secondly under the whole of C. We have not yet experimented with this possibility.

In our example, we have $C = \langle\, \mathrm{Soc}(W), c_1, c_2, c_3, c_4, c_5 \,\rangle$, where

$$\begin{aligned}
c_1 &= (\,(1,2)(3,4),\ (1,1,1,1)\,) \in W_1,\\
c_2 &= (\,(1,3)(2,4),\ (1,1,1,1)\,) \in W_1,\\
c_3 &= (\,1,\ ((1,2),(1,2),1,1)\,) \in W_1,\\
c_4 &= (\,1,\ ((1,2),(1,2))\,) \in W_2,\\
c_5 &= (\,(1,2),\ ((1,2),1)\,) \in W_2.
\end{aligned}$$

Clearly c_1, c_2 and c_3 fix all points in U_3, and c_4 and c_5 fix all points in $U_1 \cup U_2$.

Now, for an element $x \in W_1$ of the form $(\,(1,2)(3,4),\ (h,1,1,1)\,)$ with $h \in X_e$, we have $x^{c_1} = (\,(1,2)(3,4),\ (1,h,1,1)\,)$, which is conjugate by $(1,h^{-1},1,1)$ to $(\,(1,2)(3,4),\ (h,1,1,1)\,)$. So c_1 fixes all elements of U_1 and, for the conjugating elements v_1, we can choose the elements $(1,h,1,1)$. Similarly, c_1 fixes all elements of U_2, so c_1 induces the identity on U. We find that c_2 interchanges B_1 and B_2, and also interchanges corresponding elements of U_1 and U_2, whereas c_3 fixes B_1

and B_2, but interchanges the elements of U_1 and of U_2 that correspond to the class representatives $(1,2,3,4,5)$ and $(1,2,3,5,4) \in X_e$.

So, if we denote the elements of U_1, U_2, and U_3 by $\{\alpha_1, \beta_1, \gamma_1, \delta_1, \epsilon_1\}$, $\{\alpha_2, \beta_2, \gamma_2, \delta_2, \epsilon_2\}$, and $\{\alpha_3, \beta_3, \gamma_3\}$, respectively, then the permutations induced on $U_1 \cup U_2 \cup U_3$ by c_2 and c_3 are respectively $(\alpha_1, \alpha_2)(\beta_1, \beta_2)(\gamma_1, \gamma_2)(\delta_1, \delta_2)(\epsilon_1, \epsilon_2)$ and $(\delta_1, \epsilon_1)(\delta_2, \epsilon_2)$.

Similar calculations show that c_4 and c_5 induce the identity on U_3, although the conjugating elements v_3 need to be computed and stored. For example

$$\begin{aligned} (\,(1,2),\,((1,2,3,4),1)\,)^{c_5} &= \\ (\,(1,2),\,(1,(1,2))\,)\,(\,(1,2),\,((1,2,3,4),1)\,)\,(\,(1,2),\,((1,2),1)\,) &= \\ (\,(1,2),\,((1,2),(1,3,4)\,)) &= \\ (\,(1,2),\,((1,2,3,4),1)\,)^{v_3} & \end{aligned}$$

with $v_3 = ((1,4,3),1)$.

So only c_2 and c_3 induce nontrivial actions on U, and we find that there are 30 orbits of C on U and hence 30 $\mathrm{Im}(\rho)$-conjugacy classes in $g\mathrm{Soc}(W)$.

In general, if we require the centralizers of the class representatives, then we can calculate these at the same time as the representatives themselves with little extra cost. For an element $g = ((g_1, y_1), \ldots, (g_r, y_r)) \in W$ in which each y_i has at most one nonzero component in each orbit of g_i, the centralizer in $\mathrm{Soc}(W)$ of g is generated by elements whose components are constant on each orbit of g_i, and lie in the centralizer of the nontrivial component of y_i (if any) on that orbit. So we can calculate the centralizers in $\mathrm{Soc}(W)$ of the class representatives $g' = ((g_1, y_1'), \ldots, (g_r, y_r')) \in R$.

Elements of the centralizer of g' lying outside of $\mathrm{Soc}(W)$ arise from elements of the stabilizer of g' under the action of C described above. We can compute generators of this stabilizer at the same time as we calculate the orbit containing g'. Each such generator is equal modulo $\mathrm{Soc}(W)$ to an element in $C_{\mathrm{Im}(\rho)}(g')$. This element can be found by making use of the conjugating elements $v_k \in \mathrm{Soc}(W)$ that were computed along with the action of C; we omit the details.

For an arbitrary element of G, we can locate the representative of the conjugacy class of that element and, if required, an associated conjugating element as follows. Working first in G/M, we conjugate the given element to $\hat{g}$, where $\hat{g}M$ is one of the class representatives of G/M. Now let $\rho(\hat{g}) = g = ((g_1, x_1), \ldots, (g_r, x_r)) \in W$. Then we can locate the $\mathrm{Soc}(W)$-class representative $g' \in R$ of $g \in \mathrm{Im}(\rho)$ in the same way as described above for finding such representatives when calculating the action of C on R. Finally, we can use the data structures used to compute the orbits of the action of C on R, including the conjugating elements $v_k \in \mathrm{Soc}(W)$, to locate the orbit representative g'' of g' under this action. Then the inverse image of g'' under ρ is the required class representative of the given element in G. Of course, if we

wish to execute such calculations retrospectively, then we need to store the set R together with the required data pertaining to the action of C on R.

The size of the set R and its associated memory requirements are the principal obstacles that arise when applying this algorithm to groups with large numbers of socle factors. We observe in examples that the size of R has a high correlation with the number of conjugacy classes being computed, which accounts for our remark in the introduction that our implementation is limited in practice only by the size of the output.

The largest such set R generally occurs when each $g_i = 1$ and each $x_i \in \mathrm{Soc}(W_i)$, in which case $|R| = \prod_{i=1}^{r} c_i^{d_i}$, where c_i is the number of conjugacy classes of S_i.

4 Some performance statistics

In the table, we record some performance statistics for our implementation in Magma of the conjugacy classes algorithm for groups with trivial soluble radical. In each of these examples, the time for computing the classes of the almost simple groups involved is negligible.

In all but the final three examples, the group W listed in the first column is equal to $\mathrm{Aut}(\mathrm{Soc}(W))$, and we computed the classes for one representative of each conjugacy class of subgroups G of W with $\mathrm{Soc}(W) = \mathrm{Soc}(G) \leq G \leq W$. The second column shows the minimum and maximum of the number of conjugacy classes of these groups G, and the third column shows the largest size of the set R (as defined in the description of the algorithm) that arises for these groups. The final two columns list the minimum and maximum running times in seconds for the existing Magma implementation and for our new implementation of the conjugacy classes calculation for these groups. In most cases, the entry for the existing implementation is blank. This indicates that, for at least one of the groups G involved, the time was greater than two hours. The running times for the different subgroups G arising for a fixed W were observed to be dependent principally, but not entirely, on the number of classes computed; that is, on the size of the output. The computations were run on a 2GHz Pentium 4 under Linux.

The final three examples in the table are larger, and we carried out the computations for only a small randomly chosen sample of the subgroups G with $\mathrm{Soc}(W) = \mathrm{Soc}(G) \leq G \leq W$. More precisely, for each possible order of the groups G, we randomly selected up to three conjugacy classes of subgroups of that order, and used representatives of those selected classes for the calculation. Some of these examples had significant memory requirements; for example the computations with subgroups of $\mathrm{Sym}(5) \wr H$ with $H \leq \mathrm{Sym}(9)$ required about 2.5 Gigabytes of RAM.

W	classes	max $\lvert R\rvert$	Old times	New times
$S_5 \wr S_3$	50–343	125	0.01–2.2	0.04–0.3
$S_5 \wr S_4$	118 –2401	625		0.1–1.4
$S_5 \wr S_5$	118 –16807	3125		0.3–7
$A_5 \wr S_6$	690 –15625	15625		1.2–3.3
$A_5 \wr S_8$	2890 –390625	390625		87–284
$(S_5 \wr S_2) \times (\mathrm{PGL}(2,7) \wr S_2)$	366–3969	900	0.02–593	0.1–1.9
$(S_5 \wr S_3) \times (\mathrm{PGL}(2,7) \wr S_3)$	2689–250047	27000		2.5–99
$\mathrm{P\Gamma L}(2,9) \wr S_2$	28–169	49	0.01–0.8	0.1–0.6
$\mathrm{P\Gamma L}(2,9) \wr S_3$	92–2197	343		0.1–6.5
$\mathrm{P\Gamma L}(2,9) \wr S_4$	225–28561	2401		0.3–79
$M_{22}.2 \wr S_2$	90–441	144	0.1–11	0.2–0.7
$M_{22}.2 \wr S_3$	484–9261	1728		0.5–4.1
$M_{22}.2 \wr S_4$	2106–194481	20736		3.0–64
$S_{10} \wr S_2$	324–1764	576	0.02–940	0.2–0.8
$S_{10} \wr S_3$	3956–74088	13824		1.5–13
$S_{10} \wr S_4$	22350–3111696	331776		28–1750
$\mathrm{PGL}(2,47) \wr S_2$	377–2401	676	0.3–34	0.3–2.8
$\mathrm{PGL}(2,47) \wr S_3$	3900–117649	17576		2.7–30
$\mathrm{P\Gamma U}(4,3) \wr S_2$	205–3721	400		1.4–205
$\mathrm{P\Gamma U}(4,3) \wr S_3$	1460–226981	8000		4.1 – 721
$\mathrm{PGL}(2,47) \wr S_4$	33930–1623076	456976		89–1866
$(S_5 \wr A_4) \times (\mathrm{PGL}(2,7) \wr A_4)$	57150–2126250	810000		281–2682
$S_5 \wr H$, $H \leq S_9$	30486–13452117	1953125		2519–91545

References

[1] G. Butler. An inductive schema for computing conjugacy classes in permutation groups. *Math. Comp.*, 62:363–383, 1994.

[2] J.J. Cannon and D.F. Holt. Automorphism group computation and isomorphism testing in finite groups. *J. Symbolic Comput.*, 35:241–267, 2003.

[3] J.J. Cannon and D.F. Holt. Computing maximal subgroups of finite groups. *J. Symbolic Comput.*, 37:589–609, 2004.

[4] J.J. Cannon and B. Souvignier. On the computation of conjugacy classes in permutation groups. In W. Küchlin, editor, *Proceedings of International Symposium on Symbolic and Algebraic Computation, Maui, July 21–23, 1997*, pages 392–399. Association for Computing Machinery, 1997.

[5] V. Felsch and J. Neubüser. An algorithm for the computation of conjugacy classes and centralizers in p-groups. In Edward W. Ng, editor, *Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Comput. Sci.*, pages 452–465, Berlin, Heidelberg, New York, 1979. (Marseille, 1979), Springer-Verlag.

[6] D.F. Holt, B. Eick and E.A. O'Brien. *Handbook of Computational Group Theory* Chapman & Hall/CRC, 2005.

[7] A. Hulpke. Conjugacy classes in finite permutation groups via homomorphic images. *Math. Comp.*, 69 no. 232:1633–1651, 2004.

[8] M. Mecky and J. Neubüser. Some remarks on the computation of conjugacy classes of soluble groups. *Bull. Austral. Math. Soc.*, 40:281–292, 1989.

Addresses:

School of Mathematics and Statistics
University of Sydney
NSW 2006
Australia
e-mail: john@maths.usyd.edu.au

Mathematics Institute
University of Warwick
Coventry CV4 7AL
Great Britain
e-mail: dfh@maths.warwick.ac.uk