Available online at www.sciencedirect.com
ScienceDirect
Journal of Algebra 314 (2007) 75–96
JOURNAL OF Algebra
www.elsevier.com/locate/jalgebra

Enumerating big orbits and an application: B acting on the cosets of Fi_{23}

Jürgen Müller [a,*], Max Neunhöffer [a], Robert A. Wilson [b]

[a] Lehrstuhl D für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany
[b] School of Mathematical Sciences, Queen Mary, University of London, Mile End Road, London E1 4NS, United Kingdom

Received 20 July 2006
Available online 11 April 2007
Communicated by Eamonn O'Brien

Abstract

We describe a novel technique to handle big permutation domains for large groups. It is applied to the multiplicity-free action of the sporadic simple Baby Monster group on the cosets of its maximal subgroup Fi_{23}, to determine the character table of the associated endomorphism ring.
© 2007 Elsevier Inc. All rights reserved.

Keywords: Permutation groups; Orbit enumeration; Multiplicity-free action; Character tables; Sporadic simple Baby Monster group

1. Introduction

In recent years there has been increasing interest in dealing with large permutation representations, in particular of the sporadic finite simple groups. The aim of the present paper is to describe a novel technique to handle big permutation domains for large groups, and to give a substantial example application. The basic setup is as follows:

Let $G = \langle \mathcal{G} \rangle$ be a finite group acting from the right on a finite set X. For a given $x_1 \in X$ we want to enumerate the G-orbit $x_1G := \{x_1g \in X;\ g \in G\} \subseteq X$. This can be achieved efficiently with the well-known orbit-stabiliser algorithm given as Algorithm 1. As for its correctness recall that since only elements of G are applied, only points in x_1G are put into $\mathcal{D}$, and since x_1G is

* Corresponding author.
E-mail address: juergen.mueller@math.rwth-aachen.de (J. Müller).

0021-8693/$ – see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jalgebra.2007.03.038

finite, Algorithm 1 indeed terminates. After termination all generators of G have been applied to all points in $\mathcal{D}$, therefore $\mathcal{D}$ contains all points in the G-orbit x_1G exactly once. Note that here we do not need to know the group order $|G|$, nor whether G acts faithfully on X.

Algorithm 1 *(Orbit-stabiliser).*

```
Require: $G = \langle \mathcal{G} \rangle$ acting on $X$, $x_1 \in X$
  $\mathcal{D} \leftarrow [x_1]$   {collects the orbit}
  $\mathcal{T} \leftarrow [1_G]$   {collects a transversal}
  $\mathcal{S} \leftarrow [\,]$   {collects generators for the stabiliser}
  $i \leftarrow 1$
  while $i \leqslant \mathrm{Length}(\mathcal{D})$ do
    for $g \in \mathcal{G}$ do
      $x \leftarrow \mathcal{D}[i] \cdot g$
      if not ($x$ in $\mathcal{D}$) then
        append $x$ to $\mathcal{D}$
        append $\mathcal{T}[i] \cdot g$ to $\mathcal{T}$
      else
        $j \leftarrow \mathrm{Position}(\mathcal{D}, x)$   {$\mathcal{D}[j] = x$}
        append $\mathcal{T}[i] \cdot g \cdot \mathcal{T}[j]^{-1}$ to $\mathcal{S}$   {Schreier generator}
      end if
    end for
    $i \leftarrow i + 1$
  end while
  return ($\mathcal{D}, \mathcal{T}, \mathcal{S}$)   {orbit, transversal, stabiliser}
```

Moreover, $\mathcal{S}$ contains generators for the stabiliser $\mathrm{Stab}_G(x_1)$, as is implied by Schreier's Theorem, see e.g. [12, La.2.3.3], which we recall for convenience: If $\mathcal{T} = \{t_x \in G;\ x \in x_1G\} \subseteq G$ is a transversal for the G-orbit x_1G with respect to x_1, i.e. we have $x_1t_x = x$ for all $x \in x_1G$, and additionally assume $t_{x_1} = 1$, then the set $\mathcal{S} := \{tg \cdot (t_{x_1tg})^{-1} \in G;\ t \in \mathcal{T},\ g \in \mathcal{G}\} \subseteq G$ of *Schreier generators* generates $\mathrm{Stab}_G(x_1)$. Experience suggests that most of the Schreier generators typically turn out to be superfluous for generating $\mathrm{Stab}_G(x_1)$.

To perform Algorithm 1 we have to be able to keep all points in x_1G in the list $\mathcal{D}$ in main memory, and we have to be able to recognise whether a given point has already been stored. The necessary storing and recognising of points can of course be done using hashing techniques, such that we only need a nearly constant amount of time to look up a point, regardless of how many points have been stored. But if the G-orbit x_1G is too large to be stored completely in main memory, Algorithm 1 is no longer feasible. In this paper we present a novel technique allowing us to enumerate very big G-orbits being much too large in this sense; instead we assume that we know the group order $|G|$ and some additional information about G in advance.

In the first part, consisting of Sections 2–5, we discuss the ideas behind this technique and show how these lead to suitable generalisations of Algorithm 1. The basic idea of using a *helper subgroup U*, recalled in Section 2, was already considered by Richard Parker around 1995 (unpublished), and was independently made explicit in [14]. Based on practical experience, see e.g. [19,21], we were led to elaborate on this idea, and to use a whole chain of helper subgroups instead of a single one. To this end we first reconsider the basic idea in a more abstract context in Sections 2 and 3, and then allow for more than one helper subgroup in Section 4. The first part concludes with Section 5, where we briefly indicate how the situation needed to run these methods can be achieved in the most frequent case of linear actions.

The strategy described here has been implemented in GAP [8]. Altogether, the implementation of the various orbit enumeration algorithms and hashing techniques needs some 3000 lines of code and will be published soon in a GAP package ORB [20], including explicit input data for several examples, in particular the one considered below.

In the second part, consisting of Sections 6–9, we consider a particular application, which actually was part of the original motivation to develop the novel technique presented here, see [19]: the multiplicity-free action of the sporadic simple Baby Monster group B on the cosets of its maximal subgroup Fi_{23}, one of the sporadic simple Fischer groups.

Multiplicity-freeness of permutation actions, by way of the associated orbital graphs, is intimately related to the notions of distance-transitivity and distance-regularity, see [10], [5] as well as to spectra and the Ramanujan property, see [7], in algebraic graph theory. A lot of information is encoded in concise form in the character table of the endomorphism ring of the underlying permutation module; the necessary facts for this paper are recalled in Section 6.

The multiplicity-free actions of the sporadic simple groups have been classified in [3], and the associated character tables, including the one computed in this paper, have been collected from various sources in [4,17]. In particular, for the Baby Monster group B there are four multiplicity-free actions: on the cosets of $2.{}^2E_6(2).2$, of $2.{}^2E_6(2)$, of $2^{1+22}.Co_2$, and of Fi_{23}. The character tables for the former two actions have been determined in [9], while the character table for the third one has been computed in [18,19], also applying the computational techniques described here.

The aim of the second part now is to determine the character table for the fourth and largest multiplicity-free action of B, on the cosets of Fi_{23}, which has degree $\sim 10^{15}$. This action is particularly interesting, since not even the sizes of the associated Fi_{23}-orbits have been known before, and since it is related to the conjugation action of the sporadic simple Fischer–Griess Monster group $\mathbb{M}$ on its 6-transpositions, see [10].

In Section 7 we provide the infrastructure, consisting of helper subgroups and associated helper sets, to apply the strategy described in Section 4. In Section 8 a combination of the novel computational technique and a group theoretical analysis, using the action of $\mathbb{M}$ on its 6-transpositions, is applied to determine the Fi_{23}-orbits and the associated stabilisers, the result being given in Table 2. Finally, in Section 9 the character table of the associated endomorphism ring is computed, and given in Tables 7–10.

2. Archiving suborbits

The basic idea of the techniques described here is not to store single points in the G-orbit x_1G, but to archive the G-orbit in bigger chunks. To this end, we use a helper subgroup $U < G$: to enumerate x_1G we may as well enumerate the set of U-orbits contained in x_1G. Thus we want to be able to perform the following two tasks:

(1) Given a point $x \in X$, determine the size $|xU|$ and store appropriate pieces of the U-orbit xU, such that we can later perform (2).
(2) Given a point $x \in X$, decide whether or not x lies in one of the already stored U-orbits from (1).

This of course means that this should be done in a better way than just storing all points in xU separately. This is achieved using the following idea, see also [14]: let Y be another finite U-set and let $\bar{\ } : X \to Y$ be a homomorphism of U-sets, i.e. we have $\overline{xu} = \bar{x}u \in Y$ for all $x \in X$ and $u \in U$.

We then do the following preparations: after enumerating Y completely, using Algorithm 1, in every U-orbit in Y we arbitrarily choose a point and call it *U-minimal*. Furthermore, for each U-minimal point $y \in Y$ we store generators for the stabiliser $\mathrm{Stab}_U(y)$ together with its order, and for each point $y \in Y$ which is not U-minimal we store an element $u_y \in U$ such that $yu_y \in Y$ is the U-minimal point in the U-orbit yU. Here we have to assume that $\bar{\ }$ is efficiently computable, and that U and Y are small enough such that we can perform these preparations.

A point $x \in X$ is called *U-minimal* if $\bar{x} \in Y$ is U-minimal. Note that in a U-orbit $xU \subseteq X$ there may be more than one U-minimal point. More precisely, if $x \in X$ is U-minimal, the set of U-minimal points in xU is exactly $x\bar{S}$, where $\bar{S} := \mathrm{Stab}_U(\bar{x})$, because by definition $\bar{x}$ is the only U-minimal point in $\bar{x}U$ and $\bar{\ }$ is a homomorphism of U-sets.

Equipped with the above data, we now archive U-orbits $xU \subseteq X$ by only storing their U-minimal points. Given any point $x \in X$, we find a U-minimal point in xU by looking up $\bar{x} \in Y$: if $\bar{x}$ is U-minimal, then $x' := x$ is already U-minimal and we are done. Otherwise we have computed and stored an element $u_{\bar{x}} \in U$ such that $\bar{x}u_{\bar{x}}$ is U-minimal. But then $x' := xu_{\bar{x}} \in xU$ is U-minimal, because by $\bar{\ }$ it is mapped to $\overline{xu_{\bar{x}}} = \bar{x}u_{\bar{x}}$. The point x' is called the *U-minimalisation* of x.

Then to find the set $x'\bar{S}$ of all U-minimal points in $x'U$ we look up the stored generators for the stabiliser $\bar{S}$ and compute the set $x\bar{S}$ by an application of Algorithm 1.

Since $\bar{\ }$ is a homomorphism of U-sets we have $\mathrm{Stab}_U(x') = \mathrm{Stab}_{\bar{S}}(x')$, and thus once we know $|x'\bar{S}|$, we also know $|\mathrm{Stab}_{\bar{S}}(x')| = |\bar{S}|/|x'\bar{S}|$ and thus $|x'U| = |U|/|\mathrm{Stab}_U(x')|$. Therefore, both parts of task (1) are done.

If we are now given a point $x \in X$, we can decide whether we already know the U-orbit xU, by U-minimalising x and looking up its U-minimalisation x'. If we already know xU, then we have stored the U-minimal point x'. Otherwise, the U-orbit xU is new. Thus task (2) is done as well.

We now turn to the question of what we gain using this idea: to enumerate X completely using Algorithm 1, all points in X have to be stored. In contrast, to enumerate X as described above, for each U-orbit in Y we pick its U-minimal point, $y \in Y$ say, and only store the points in $\{x \in X;\ \bar{x} = y\} \subseteq X$, i.e. the points in the fibre of $\bar{\ }$ over y. Since only the U-orbits yU being in the image of $\bar{\ }$ are needed, we may assume that $\bar{\ } : X \to Y$ is surjective. Since $\bar{\ }$ maps U-orbits in X to U-orbits in Y we have

$$\bigl|\{x \in X;\ \bar{x} = y\}\bigr| = \sum_{xU \in X/U,\, \overline{xU} = yU} \bigl|\mathrm{Stab}_U(y)\bigr| / \bigl|\mathrm{Stab}_U(x)\bigr|.$$

Hence the number of U-minimal points in X to be stored is

$$\begin{aligned} N_X &:= \sum_{yU \in Y/U} \bigl|\{x \in X;\ \bar{x} = y\}\bigr| \\ &= \sum_{y \in Y} 1/|yU| \cdot \bigl|\{x \in X;\ \bar{x} = y\}\bigr| \end{aligned}$$

$$= 1/|U| \cdot \sum_{y \in Y} \left|\mathrm{Stab}_U(y)\right| \cdot \left|\{x \in X;\ \bar{x} = y\}\right|$$

$$= 1/|U| \cdot \sum_{y \in Y} \sum_{xU \in X/U,\, \overline{xU} = yU} \left|\mathrm{Stab}_U(y)\right|^2 / \left|\mathrm{Stab}_U(x)\right|.$$

We have $N_X \geqslant 1/|U| \cdot \sum_{y \in Y} |\{x \in X;\ \bar{x} = y\}| = |X|/|U|$, with equality if and only if $|\mathrm{Stab}_U(y)| = 1$ for all $y \in Y$. Thus the saving factor is $|X|/N_X \leqslant |U|$, where equality is achieved if and only if Y entirely consists of regular U-orbits.

Letting ν_Y be the number of U-orbits in Y, and $\lambda_Y := |Y|/\nu_Y$ be the average length of the U-orbits in Y, we have

$$|X|/N_X = \lambda_Y \cdot \frac{1/|Y| \cdot \sum_{y \in Y} |\{x \in X;\ \bar{x} = y\}|}{1/\nu_Y \cdot \sum_{yU \in Y/U} |\{x \in X;\ \bar{x} = y\}|}.$$

The fraction on the right-hand side can be understood as a quotient of average cardinalities of fibres, where in the numerator we average over Y, while in the denominator we average over the U-orbits in Y. Actually, for the common cases discussed in Section 5, where X and Y are linear structures and the homomorphism $\bar{\ } : X \to Y$ of U-sets is derived from a linear map, the fibres $\{x \in X;\ \bar{x} = y\} \subseteq X$ all have one and the same cardinality, which hence equals $|X|/|Y|$. Thus in this case we indeed get a saving factor of $|X|/N_X = \lambda_Y$. In general, the numerator of course always equals $|X|/|Y|$, but in practice the denominator does not seem to be under good control.

Some numerical data are given in Table 4 below: e.g. letting X be the subset of the Fi_{23}-orbit $X_{23}^{\pi} \subseteq M_4$ enumerated as described at the end of Section 8, we have $|X| = 281\,092\,626\,984\,960 \sim 2.8 \cdot 10^{14}$, and for its image $Y \subseteq M_3$ we have $|Y| = 4\,397\,288\,393\,040 \sim 4.4 \cdot 10^{12}$ and $\nu_Y = 471$, hence $\lambda_Y \sim 9.3 \cdot 10^9$, where $|U| = 47\,377\,612\,800 \sim 4.7 \cdot 10^{10}$. Hence we have $|X|/|Y| \sim 64$, while it turns out that $1/\nu_Y \cdot \sum_{yU \in Y/U} |\{x \in X;\ \bar{x} = y\}| \sim 3038$, yielding a saving factor, compared to λ_Y, of only $|X|/N_X \sim 196\,455\,480 \sim 2 \cdot 10^8$.

Recall that the price we pay for this saving is that we need structural information about G, to build up the additional infrastructure with U and $\bar{\ } : X \to Y$, and to be able to compute stabiliser orders efficiently.

3. Orbit enumeration by suborbits

The algorithm presented in this section is the heart of the whole method. For the enumeration of an orbit $x_1 G$ it outperforms a standard orbit algorithm like Algorithm 1, because it can save up to a factor of $\sim |U|$ in space usage under good conditions. It is also used in a crucial way in the generalisation of the trick from Section 2 to a chain of helper subgroups that is described in Section 4.

We first describe how U-orbits are archived in the slightly more abstract situation in this section, then we present Algorithm 2 and explain all the procedures called in it, before we proceed to define a certain transversal to use Schreier's Theorem and then prove termination and correctness.

We keep the notation from Section 2, that is $U < G$ and $\bar{\ } : X \to Y$ is a homomorphism of U-sets, we assume that we have chosen a U-minimal point in each U-orbit in Y and again a point $x \in X$ is called U-minimal, if $\bar{x}$ is the chosen U-minimal point in $\bar{x}U$.

Now we can perform the following tasks, which are an abstraction of what was described in Section 2, allowing us to formulate Algorithm 2:

(a) For every $x \in X$, find $u \in U$ such that xu is U-minimal.
(b) For every U-minimal point $x \in X$, find generators for $\bar{S} := \mathrm{Stab}_U(\bar{x})$ and the order $|\bar{S}|$.

In the sequel let $\mathsf{Minimaliser}_U(x)$ be the result of a procedure returning an element $u \in U$ as in (a), where we assume that $\mathsf{Minimaliser}_U(x) = 1_U$ whenever x already is U-minimal. Moreover, let $\mathsf{BarStabiliser}_U(x)$ be the result of a procedure returning $|\bar{S}|$ and generators for $\bar{S}$ as in (b). Having (a) and (b) at hand, we can devise procedures StoreSuborbit and LookupSuborbit performing tasks (1) and (2) exactly as described in Section 2:

Information on the U-orbits is collected in a database $\mathcal{D}$. If $x \in x_1tU$ is U-minimal, where $t \in G$, then $\mathsf{StoreSuborbit}(\mathcal{D}, x, t)$ invokes $\mathsf{BarStabiliser}_U(x)$, enumerates the orbit $x\bar{S}$ using Algorithm 1 thereby determining $|xU|$ exactly as described in Section 2. Then it stores the set $x\bar{S}$ of U-minimal points $x' \in xU$ in the database $\mathcal{D}$ together with $|xU|$. Hence this allows us to keep track of the total number $\mathsf{Size}(\mathcal{D})$ of points in all U-orbits already stored in the database $\mathcal{D}$. In addition, an element $t \in G$ with $x_1tU = xU$ representing the U-orbit is stored as a word in the generators of G. This is used below to define a right transversal of $\mathrm{Stab}_G(x_1)$ in G.

The procedure $\mathsf{LookupSuborbit}(\mathcal{D}, x)$, where $x \in X$ is U-minimal, returns either true or false, depending on whether xU is already stored in $\mathcal{D}$ or not. This is just done by looking up x itself, exactly as in Section 2. If x is already stored, we also have access to a representative $t \in G$ with $x_1tU = xU$ stored above.

Note that for both procedures (1) and (2) task (a) was crucial to first reach a U-minimal x at all. Also, as in Section 2, we have to be able to compute orders of any subgroup $\langle \mathcal{S} \rangle \leqslant G$ generated by some subset $\mathcal{S} \subseteq G$, usually by using a relatively small permutation representation for G. Note that the ability to compute subgroup orders also facilitates membership testing for $\langle \mathcal{S} \rangle$. Moreover, to save memory, all group elements of G which arise are stored as words in the given generators $\mathcal{G}$ and $\mathcal{U}$.

Algorithm 2 *(Orbit-stabiliser by suborbits).*

```
Require: $G = \langle \mathcal{G} \rangle$ acting on $X$, $U = \langle \mathcal{U} \rangle \leqslant G$, $x_1 \in X$ $U$-minimal, $0 \leqslant f \leqslant 1$
  $\mathcal{D} \leftarrow$ empty database of $U$-orbits
  StoreSuborbit($\mathcal{D}, x_1, 1_G$)
  $\mathcal{R} \leftarrow [1_G]$
  $\mathcal{S} \leftarrow [\,]$   {collects generators for the stabiliser}
  $p \leftarrow 1$
  loop
    $i \leftarrow 1$
    while $i \leqslant$ Length($\mathcal{R}$) do
      $r \leftarrow \mathcal{R}[i]$
      for $g \in \mathcal{G}$ do
        $u \leftarrow$ Minimaliser$_U(x_1rg)$
        $l \leftarrow$ LookupSuborbit($\mathcal{D}, x_1rgu$)
        if $l =$ false then
          StoreSuborbit($\mathcal{D}, x_1rgu, rg$)   {with determining its size}
          append $rg$ to $\mathcal{R}$
        end if
        if $l =$ true or $p > 1$ then
          $s \leftarrow$ SchreierGenerator($\mathcal{D}, x_1 r, g$)
          if $s \notin \langle\mathcal{S}\rangle$ then
            append $s$ to $\mathcal{S}$
          end if
        end if
        if Size($\mathcal{D}$) $\cdot |\langle\mathcal{S}\rangle| \geqslant f \cdot |G|$ then
          return ($\mathcal{D}, \mathcal{S}$)   {database, stabiliser}
        end if
      end for
      $i \leftarrow i + 1$
    end while
    $p \leftarrow p + 1$
    $\mathcal{R}_0 \leftarrow \mathcal{R}$
    $\mathcal{R} \leftarrow [\,]$
    for $t$ in $\mathcal{R}_0$ do
      for $u \in \mathcal{U}$ do
        append $tu$ to $\mathcal{R}$
      end for
    end for
  end loop
```

We now proceed to prove termination and correctness of Algorithm 2. To use Schreier's Theorem from the introduction, we have to define a right transversal of $\mathrm{Stab}_G(x_1)$ in G. As this would be too big to be kept in memory completely, we define the transversal by means of an algorithm that, given $x \in x_1 G$, produces an element $t_x \in G$ with $x_1 t_x = x$. Remember that for every U-orbit xU in our database we have stored an element $t \in G$ such that $xU = x_1 t U$, and by U-minimalisation we can find an element $u \in U$ with $x_1 t u$ being U-minimal.

Given $x \in x_1 G$, we let $v := \mathsf{Minimaliser}_U(x)$ and then look up xv in the database finding $t \in G$ such that $xvU = xU = x_1 t U$.

With $u := \mathsf{Minimaliser}_U(x_1 t)$ we have that xv and $x_1 t u$ are both U-minimal and lie in the same U-orbit, thus there is an $s \in \overline{S} := \mathrm{Stab}_U(\overline{x_1 t u})$ with $x_1 t u s = xv$. To compute and uniquely define s we perform Algorithm 1 with the stored and thus fixed generators of $\overline{S}$ and set s to be the first element found with the above property. We then define $t_x := tusv^{-1}$. Note that this uniquely defines t_x using our stored data.

This definition has two important consequences: firstly because the stored representative for the very first stored U-orbit $x_1 U$ is the identity, we have $t_{x_1} = 1_G$. Secondly, if t is the stored representative for a U-orbit $x_1 t U$ then $t_{x_1 t} = t$ and $t_{x_1 t u} = tu$ for $u := \mathsf{Minimaliser}_U(x_1 t)$.

Now we explain what the procedure SchreierGenerator in Algorithm 2 does to compute generators of $\mathrm{Stab}_G(x_1)$: during the execution of Algorithm 2 we constantly apply a generator $g \in \mathcal{G}$ to some point $x_1 r$, where $r = tw$ with t being the stored representative of the U-orbit $x_1 t U$, and w being some element of U that comes from the last two **for** loops in the main loop. Then we try to look up the U-orbit $x_1 t w g U$.

In such a situation, $x_1 t w g U$ either is a newly found U-orbit, in which case it is stored with twg as its representative, or it is already known. If in the latter case we have $w = 1$, which happens in the first iteration of the outer loop, the Schreier generator $t_{x_1 t} g t_{x_1 t g}^{-1}$ is trivial, because

t is the stored representative for x_1tU and tg is the one for x_1tgU. Therefore Algorithm 2 does not calculate a Schreier generator in that case.

In all other cases x_1twgU is then known as a stored U-orbit $x_1t'U$. The procedure call SchreierGenerator$(\mathcal{D}, x_1t, g)$ then returns $t_{tw}gt_{twg}^{-1}$ by calculating the two transversal elements as described above from stored data.

We now address the question of correctness: Algorithm 2 by construction only stores U-orbits that are contained in x_1G, thus at any time Size$(\mathcal{D}) \leqslant |x_1G|$. Moreover, in $\mathcal{S}$ only elements of the stabiliser $\mathrm{Stab}_G(x_1)$ are collected, thus at any time $|\langle\mathcal{S}\rangle|$ is a divisor of $|\mathrm{Stab}_G(x_1)|$.

Let first $f := 1$. In the **while** loop we first apply the generators $\mathcal{G}$ of G to representatives of known U-orbits. At the end of the outer **loop** the generators $\mathcal{U}$ of U are then applied to these representatives, such that in the next iteration of **loop** new points in the same U-orbits are used. Thus the algorithm will eventually apply all generators of G to all points in all enumerated U-orbits and thus will eventually find all U-orbits. Similarly, all Schreier generators will eventually be found, which by Schreier's Theorem implies $\langle\mathcal{S}\rangle = \mathrm{Stab}_G(x_1)$. Since $|x_1G| \cdot |\mathrm{Stab}_G(x_1)| = |G|$, this implies that Algorithm 2 terminates, and returns a database $\mathcal{D}$ containing all U-orbits in x_1G, as well as generators for $\mathrm{Stab}_G(x_1)$.

The above analysis shows that Algorithm 2 also terminates for any $0 \leqslant f < 1$, and returns part of x_1G and a subgroup $\langle\mathcal{S}\rangle \leqslant \mathrm{Stab}_G(x_1)$. The idea behind this is as follows: as soon as we have Size$(\mathcal{D}) \cdot |\langle\mathcal{S}\rangle| > |G|/2$, we conclude that indeed $\langle\mathcal{S}\rangle = \mathrm{Stab}_G(x_1)$, and in particular we know the size $|x_1G|$. Hence if we specify $f > 1/2$, then Algorithm 2 only computes the fraction f of the whole G-orbit x_1G, which is often enough for applications, see Section 8.

The above correctness proof shows that in the worst case the running time of Algorithm 2 is no better than the running time of Algorithm 1. Still, in practice a rather small subset of Schreier generators suffices to generate the full stabiliser $\mathrm{Stab}_G(x_1)$, hence typically $\mathrm{Stab}_G(x_1)$ is already reached after a small fraction of the whole computation. Moreover, the counter p typically assumes only very small values, in particular if we enumerate only part of the orbit by specifying $f < 1$; see also Table 4. Hence in practice the computation is dominated by enumerating U-orbits, which is done by applying the elements of $\mathcal{G}$ only to the stored U-orbit representatives, instead of applying them to all elements of x_1G. Thus if the infrastructure is set up optimally we are able to obtain a time saving factor of $\sim |U|$ as well.

4. Iterating orbit enumeration by suborbits

To archive U-orbits we had to assume that U is small enough such that enumeration of the U-orbits in the helper U-set can be done by Algorithm 1. For large groups G this tends to imply that U is too small to be helpful. Now the idea is to use a larger helper subgroup $U < V < G$, together with a helper V-set, to enumerate a G-orbit by V-orbits using Algorithm 2, where in turn orbit enumeration in the helper V-set is done by U-orbits, for some small helper subgroup $U < V$. This is done in a way that we can iterate it to use a chain of subgroups totally ordered by inclusion.

Recall that to perform an orbit enumeration by U-orbits we need a definition of U-minimality and we need to be able to do tasks (a) and (b) from Section 3, that is we need procedures Minimaliser$_U$ and BarStabiliser$_U$. We now present the setup for building this infrastructure for V, using the same infrastructure already in place for U.

Let X be a finite G-set, let Z be a finite V-set, and let Y be a finite U-set, together with a homomorphism of V-sets $\tilde{}: X \to Z$ and a homomorphism of U-sets $\bar{}: Z \to Y$. By abuse of notation we denote the composition of $\tilde{}$ and $\bar{}$, mapping X to Y, also by $\bar{}$: it is a homomorphism of U-sets. We can now use the definition of U-minimality for both the group V acting on Z and the group G acting on X.

In a precomputation we first calculate a transversal $\mathcal{L}$ for the left cosets of U in V, that is a subset $\mathcal{L} \subseteq V$ of size $|\mathcal{L}| = [V:U]$ such that $V = \bigcup_{t \in \mathcal{L}} tU$, where we assume the index $[V:U]$ to be small enough such that this is feasible, and that $1_V \in \mathcal{L}$.

Then we enumerate all of Z by U-orbits. Note that when the U-infrastructure is set up optimally, this saves a factor of $\sim |U|$ in space usage. In every V-orbit of Z we arbitrarily choose one U-minimal point z and call it *V-minimal*. We run the V-orbit by U-orbit enumeration of that V-orbit with starting point z using Algorithm 2, such that we get as an additional result the order and generators for $\mathrm{Stab}_V(z)$, which we store together with z. Note that during this calculation we store every U-minimal point in zV.

Further, for every U-minimal point $w \in zU$, $w \neq z$, we store a word in the generators of $\mathrm{Stab}_U(\bar{z}) = \mathrm{Stab}_U(\overline{w})$ mapping w to z. For every U-minimal point $w \in zV \setminus zU$ we compute and store the number of an element of $\mathcal{L}$ mapping w into the U-orbit zU. Note that this is possible, because for every point $w \in zV$ there is an element of V mapping it to z and thus an element of $\mathcal{L}$ mapping it into zU.

We now define similarly to the above a point $x \in X$ to be *V-minimal* if $\widetilde{x} \in Z$ is V-minimal. With these preparations we can now perform the procedures $\mathsf{Minimaliser}_V$ for all points in X, and $\mathsf{BarStabiliser}_V$ for V-minimal points in X in the following way:

Given any $x \in X$, we first use $\mathsf{Minimaliser}_U$ to find a U-minimal point $w := xu \in X$ for some $u \in U$. Thus by definition $\widetilde{w}$ is U-minimal as well, because it is mapped by $\bar{}$ to $\overline{w}$. Therefore, $\widetilde{w}$ was stored during our precomputation. Let $z \in Z$ be the chosen V-minimal point in $\widetilde{w}V$.

There are three cases: firstly, if $\widetilde{w} = z$, then we are done, returning $v := u$, since w is V-minimal by definition. Secondly, if $\widetilde{w} \in zU$, $\widetilde{w} \neq z$, then since both z and $\widetilde{w}$ are U-minimal, we have a stored element $s \in \mathrm{Stab}_U(\bar{z}) = \mathrm{Stab}_U(\overline{w}) \leqslant U$ such that $\widetilde{w}s = z$ and we can return $v := us$. If $\widetilde{w} \notin zU$ we have stored an element $t \in \mathcal{L}$ such that $\widetilde{w}t \in zU$, thus letting $u' := \mathsf{Minimaliser}_U(wt)$, the above cases finally give us an element $v := utu's$ such that $xutu's$ is V-minimal. In all three cases, we have found an element $v \in V$ such that xv is V-minimal thereby finding $\mathsf{Minimaliser}_V(x)$.

If $x \in X$ is V-minimal we have that $\widetilde{x}$ is the V-minimal point in $\widetilde{x}V$ and thus we have stored the order and generators for $\mathrm{Stab}_V(\widetilde{x})$ during our precomputation using Algorithm 2. Therefore we can easily provide a procedure $\mathsf{BarStabiliser}_V$.

The definition of V-minimality for points in X together with the procedures $\mathsf{Minimaliser}_V$ and $\mathsf{BarStabiliser}_V$ now fulfil exactly tasks (a) and (b) from Section 3 with Z in place of Y and $\widetilde{}$ in place of $\bar{}$ and V in place of U. Thus we can iterate the saving trick in this way and enumerate G-orbits by V-orbits.

Note that in practice the above-mentioned precomputations can all be done on the fly whenever a point $x \in X$ is encountered which is mapped by $\widetilde{}$ to an as yet unknown V-orbit $\widetilde{x}V \subseteq Z$. Moreover, to compute a transversal $\mathcal{L}$ for the left cosets of U in V, we can just use a transitive V-set a point stabiliser of which is contained in U and enumerate it by U-orbits.

Finally, this can be iterated as follows: let $U_1 < U_2 < \cdots < U_k < U_{k+1} := G$ be a chain of helper subgroups, together with U_i-sets Y_i and homomorphisms $\pi_i : Y_{i+1} \to Y_i$ of U_i-sets, for $1 \leqslant i \leqslant k$, where we let $Y_{k+1} := X$. Then we are able to enumerate a G-orbit in X by U_k-orbits using Algorithm 2. To do so, for $k \geqslant i \geqslant 2$ in turn U_i-orbits in Y_i are enumerated by U_{i-1}-orbits, also using Algorithm 2. Finally U_1-orbits in Y_1 are enumerated using Algorithm 1.

5. Common case: Linear actions

In this section we describe concrete cases in which the above methods can be used, together with ways to find suitable helper sets and subgroups. These techniques have already been applied successfully in the single helper subgroup case to various substantial examples, see for example [14,19,21].

5.1. *Action on vectors*

Let X be a finite-dimensional FG-module, where F is a finite field and FG is the group algebra of G over F. Then in particular X can be considered as a G-set. Let $U < G$ be a subgroup such that there is an FU-submodule $0 < X' < X|_U$. Then the natural map $\overline{} : X \to X/X' =: Y$ to the quotient FU-module Y is a homomorphism of FU-modules, and thus is a homomorphism of U-sets.

The quotient FU-module Y has to fulfil several conditions in order to be of practical use: on the one hand, the F-dimension of Y has to be small enough such that all its U-orbits can be enumerated in the precomputation and such that we can store the necessary information for U-minimalisation. On the other hand, the F-dimension of Y has to be big enough such that the average size of the U-orbits in Y is as big as possible.

We thus have to find an appropriate helper subgroup U together with a good quotient fulfilling these conditions simultaneously. For example, we might guess a subgroup U, and try to find a suitable FU-submodule X' by using the algorithms to compute submodule lattices described in [16], available in the MeatAxe [24].

Note that a possible pitfall is that the zero vector in Y is necessarily U-minimal, hence all points in X' are U-minimal as well. Thus, given $x_1 \in X$, all points in $x_1 G \cap X'$ have to be stored, which means that for these points we do not save anything. A possible remedy is to choose $X' < X$ such that $x_1 G \cap X' = \emptyset$, but this poses a further condition for the quotient to be good, which cannot always be fulfilled.

Now we proceed as follows: first we choose helper subgroups $U < V < G$. Then we try to find an FV-submodule $0 < X'' < X|_V$, and subsequently we try to find an FU-submodule $0 < X'/X'' < (X/X'')|_U$, which amounts to looking for an FU-submodule $X' < X|_U$ which contains X''. We then let $Z := X/X''$ and $Y := X/X'$. The natural maps $\widetilde{} : X \to Z$ and $\overline{} : X \to Y$ are then homomorphisms of FV-modules and FU-modules, respectively, and $\overline{}$ factors through $\widetilde{}$ as required. Of course this procedure can be iterated for more than two helper subgroups to get a whole chain of submodules.

5.2. *Projective action*

In the situation of Section 5.1 we can also use projective action, i.e. the natural action on the set $\mathbb{P}(X)$ of one-dimensional F-subspaces of X. The action on $\mathbb{P}(X)$ is usually implemented by choosing an F-basis for X, and storing one-dimensional subspaces as normalised vectors, i.e. vectors in which the first non-zero entry is equal to 1; note that this choice of representative depends on the chosen F-basis. The action of a group element, given by a representing matrix, is then vector–matrix multiplication, followed by multiplying with a scalar to re-normalise vectors.

Given an FU-submodule $X' < X|_U$, the natural map $\overline{} : X \to X/X' =: Y$ induces a map from $\mathbb{P}(X) \to \mathbb{P}(Y) \dot\cup \{0\}$, where all one-dimensional F-subspaces of X' are mapped to the zero-space $\{0\} \leqslant Y$. Since $0 \in Y$ is fixed under the action of U, this again is a homomorphism of U-sets.

In practice, if we have $\dim_F(X) = d$ and $\dim_F(X') = e$, we may choose an F-basis $(b_1, b_2, \dots, b_d)$ of X such that $(b_{d-e+1}, b_{d-e+2}, \dots, b_d)$ is an F-basis for X'. Writing the vectors in X with respect to this F-basis, and writing the vectors in Y with respect to the truncated F-basis $(b_1 + X', b_2 + X', \dots, b_{d-e} + X')$, the natural map $\bar{\ }$ is just taking the first $d - e$ components. Note that using these F-bases we do not have to re-normalise vectors after applying the natural map.

5.3. *Action on d-dimensional subspaces*

Similar to the projective action case, for any $1 < d \leqslant \dim_F(X)$ we get a natural homomorphism of U-sets from the set of d-dimensional F-subspaces of X to the set of F-subspaces of Y of dimension at most d.

After choosing an F-basis for X, the d-dimensional F-subspaces of X are described by matrices of full rank d in full echelon form. Hence the action of a group element, given by a representing matrix, on such a d-dimensional F-subspace is matrix–matrix multiplication, followed by computing the full echelon form of the resulting matrix. In practice, we choose F-bases as described in Section 5.2.

Note that typically the set of F-subspaces of Y of dimension at most d, where we assume $\dim_F(Y) > d$, is too large to be enumerated completely. Thus in practice we only consider the F-subspaces of dimension exactly d in Y, and treat the F-subspaces of X being mapped by $\bar{\ }$ to F-subspaces of dimension less than d as "zero vectors." But since for the latter we do not save anything, the saving factor might become too small. A possible remedy is to consider various quotients X/X', X/X'', X/X''', …, and to treat only those F-subspaces of X as "zero vectors" which by all associated natural maps are mapped to F-subspaces of dimension less than d. For an application of this idea see [19, Section III.15.2] and [21].

6. Endomorphism rings and their character tables

We recall the necessary facts about permutation modules and their endomorphism rings; as general references see e.g. [1,19,29].

Let G be a finite group, let $H \leqslant G$ and let $n := [G : H]$. Let $X \neq \emptyset$ be a transitive G-set such that $\mathrm{Stab}_G(x_1) = H$, for some $x_1 \in X$, and let $X = \dot{\bigcup}_{i=1}^{r} X_i$, where the $X_i \subseteq X$ are the H-orbits. The number $r \in \mathbb{N}$ is called the *rank* of X. For all $1 \leqslant i \leqslant r$ we choose $x_i \in X_i$ and $g_i \in G$ such that $x_1 g_i = x_i$, where we assume $g_1 = 1$ and $X_1 = \{x_1\}$, and we let $H_i := \mathrm{Stab}_H(x_i) \leqslant H$ and $k_i := |X_i| = |H|/|H_i|$.

For $1 \leqslant i \leqslant r$, the orbits $\Gamma_i := (x_1 g, x_i g)G \subseteq X \times X$ of the diagonal action of G on $X \times X$ are called *orbitals*; hence we have $|\Gamma_i| = |G|/|H_i| = nk_i$. Let $1 \leqslant i^* \leqslant r$ be defined by $\Gamma_{i^*} = (x_i, x_1)G$, then X_{i^*} is called the H-orbit *paired* to X_i; note that we have $k_{i^*} = k_i$. Let the ith *orbital graph* be the simple directed graph with vertex set X and edge set Γ_i, and let $A_i = [a_{i,x,y}] \in \{0,1\}^{n \times n}$, with row index $x \in X$ and column index $y \in X$, be its adjacency matrix, i.e. we have $a_{i,x,y} = 1$ if and only if $(x, y) \in \Gamma_i$.

Let $\mathbb{Z}X$ be the associated permutation $\mathbb{Z}G$-module, and let $E := \mathrm{End}_{\mathbb{Z}G}(\mathbb{Z}X)$ be its endomorphism ring, i.e. the set of all $\mathbb{Z}$-linear maps $\mathbb{Z}X \to \mathbb{Z}X$ commuting with the action of G. By [25], see also [13, Chapter II.12], the set $\{A_i;\ 1 \leqslant i \leqslant r\} \subseteq E$ is a $\mathbb{Z}$-basis for E, called the *Schur basis*, and it can also be considered as a $\mathbb{C}$-basis for $E_{\mathbb{C}} := E \otimes_{\mathbb{Z}} \mathbb{C} \cong \mathrm{End}_{\mathbb{C}G}(\mathbb{C}X)$, which is a split semisimple $\mathbb{C}$-algebra. Moreover, E is commutative if and only if the permutation character

$1_H^G \in \mathbb{Z}\,\mathrm{Irr}_{\mathbb{C}}(G)$ associated with the G-set X is *multiplicity-free*, i.e. all the constituents of 1_H^G occur with multiplicity 1, where $\mathrm{Irr}_{\mathbb{C}}(G)$ denotes the set of irreducible $\mathbb{C}$-valued characters of G.

From now on suppose E is commutative. Then letting $\mathrm{Irr}_{\mathbb{C}}(E)$ be the set of irreducible $\mathbb{C}$-valued characters of $E_{\mathbb{C}}$, we have $|\mathrm{Irr}_{\mathbb{C}}(E)| = r$, and $\lambda(A_1) = 1$ for all $\lambda \in \mathrm{Irr}_{\mathbb{C}}(E)$. The *character table* of E is defined as the matrix $\Phi_E := [\lambda(A_i)] \in \mathbb{C}^{r\times r}$, with row index $\lambda \in \mathrm{Irr}_{\mathbb{C}}(E)$ and column index $1 \leqslant i \leqslant r$. Hence in particular Φ_E is invertible. Moreover, there is a natural bijection, called the *Fitting correspondence*, between the irreducible characters of $E_{\mathbb{C}}$ and the constituents of 1_H^G; the Fitting correspondent of $\lambda \in \mathrm{Irr}_{\mathbb{C}}(E)$ is denoted by $\chi_\lambda \in \mathrm{Irr}_{\mathbb{C}}(G)$. In particular, we have $1/\chi_\lambda(1) = (1/n) \cdot \sum_{i=1}^{r} \|\lambda(A_i)\|^2/k_i$, where $\|\cdot\|$ denotes the complex absolute value; thus degrees of Fitting correspondents are easily computed from Φ_E.

For $1 \leqslant i \leqslant r$ let $P_i = [p_{h,i,j}] \in \mathbb{Z}^{r\times r}$, with row index $1 \leqslant h \leqslant r$ and column index $1 \leqslant j \leqslant r$, be the representing matrix of A_i for its right regular action on E, with respect to the Schur basis, i.e. we have $A_h A_i = \sum_{j=1}^{r} p_{h,i,j} A_j$. Hence the map $E \to \mathbb{Z}^{r\times r} : A_i \mapsto P_i$, for $1 \leqslant i \leqslant r$, is a faithful representation of E. The matrices P_i are called *collapsed adjacency matrices* or *intersection matrices*, since their entries are given by $p_{h,i,j} = |X_h \cap X_{i^*} g_j| \in \mathbb{N}_0$.

In particular, the first row and the first column of P_i are given as $p_{1,i,j} = \delta_{i,j}$ and $p_{h,i,1} = k_h \cdot \delta_{h,i^*}$, where $\delta_{\cdot,\cdot} \in \{0,1\}$ denotes the Kronecker function, and the column sums of P_i are for all j identically given as $\sum_{h=1}^{r} p_{h,i,j} = \sum_{h=1}^{r} |X_h \cap X_{i^*} g_j| = k_i$. Moreover, we have $k_j \cdot |X_h \cap X_{i^*} g_j| = k_h \cdot |X_j \cap X_i g_h|$, implying the identity $p_{h,i,j} = |X_j \cap X_i g_h| \cdot k_h/k_j = p_{j,i^*,h} \cdot k_h/k_j$. Thus from $\sum_{j=1}^{r} |X_j \cap X_i g_h| = k_i$, depending on h we get the weighted row sums of P_i as $\sum_{j=1}^{r} k_j p_{h,i,j} = k_h k_i$.

The character table of E and the intersection matrices are related as follows: if Φ_E is given, the P_i are easily computed using the formula $P_i = \Phi_E^{\mathrm{tr}} \cdot \mathrm{diag}[\lambda(A_i);\ \lambda \in \mathrm{Irr}_{\mathbb{C}}(E)] \cdot \Phi_E^{-\mathrm{tr}}$, where $\mathrm{diag}[\,\cdot\,] \in \mathbb{C}^{r\times r}$ denotes the diagonal matrix having the indicated entries. Conversely, if the P_i are given, the set $\{[\lambda(A_i);\ 1 \leqslant i \leqslant r] \in \mathbb{C}^r;\ \lambda \in \mathrm{Irr}_{\mathbb{C}}(E)\}$, consisting of the rows of Φ_E to be computed, is characterised as the unique $\mathbb{C}$-basis of $\mathbb{C}^r$ consisting of simultaneous eigenvectors for all the matrices $P_i^{\mathrm{tr}} \in \mathbb{C}^{r\times r}$, for $1 \leqslant i \leqslant r$, and having 1 as their first entry.

7. *B* acting on the cosets of *Fi*$_{23}$

We are now ready to consider the promised example. The group theoretical and representation theoretic data concerning the groups involved is available in [6]. Computations with characters and with permutation and matrix representations are done with GAP [8] and the MeatAxe [24], in particular we make use of the algorithms to compute submodule lattices described in [16]. We only indicate the major steps; for more technical details we refer to [19], where we have already reported on these computations.

From now on let $G := B$ be the sporadic simple Baby Monster group, and let $H := Fi_{23}$ be the sporadic simple Fischer group, which is a maximal subgroup of G. Then the permutation character 1_H^G has degree $1\,015\,970\,529\,280\,000 \sim 10^{15}$, and by [3] it is multiplicity-free of rank $r = 23$, its constituents have pairwise distinct degrees, and hence in particular are $\mathbb{Q}$-valued. We consider the action of G on the set of right cosets of H, the ultimate aim being to determine the character table of the associated endomorphism ring; recall that not even the sizes of the H-orbits have been known before.

First we construct an $\mathbb{F}_2 G$-module, containing an H-invariant but not G-invariant vector, placing ourselves into the situation described in Section 5.1: let $4370a$ be the absolutely irreducible $\mathbb{F}_2 G$-module of $\mathbb{F}_2$-dimension 4370; by [11] this is the smallest faithful representation of G over fields of characteristic 2. Representing matrices for standard generators, in the sense of [26], have been constructed in [27] and are available in [28], where also words in the standard generators giving standard generators for H are available. It turns out that $4370a|_H$ has absolutely irreducible constituents $782a$ and $3588a$, the notation as usual indicating $\mathbb{F}_2$-dimensions. Thus $4370a$ does not serve our purposes, and we proceed as follows:

Since the unique absolutely irreducible ordinary representation of G of degree 4371 has 2-modular constituents $4370a$ and $1a$, where the latter denotes the trivial $\mathbb{F}_2G$-module, by Thompson's Theorem, see [13, Corollary I.17.5], there is a uniserial $\mathbb{F}_2G$-module M having descending composition series $(1a, 4370a)$. Since $4371|_H$ has absolutely irreducible ordinary constituents having degrees 1, 782 and 3588, we conclude by Zassenhaus's Theorem, see [13, Corollary I.17.3], that $M|_H \cong 1a \oplus 782a \oplus 3588a$ as $\mathbb{F}_2H$-modules. Hence we let $0 \neq x_1 \in M$ be the non-trivial H-invariant vector, which is not G-invariant, and thus its G-orbit $X := x_1G \subseteq M$ is isomorphic as a G-set to the set of right cosets of H.

To construct the $\mathbb{F}_2G$-module M explicitly, we consider the cohomology group $\mathrm{Ext}^1_{\mathbb{F}_2G}(1a, 4370a) \cong H^1_{\mathbb{F}_2}(G, 4370a) := Z^1_{\mathbb{F}_2}(G, 4370a)/B^1_{\mathbb{F}_2}(G, 4370a)$, where the latter are the groups of 1-cocycles and 1-coboundaries of G with values in $4370a$, respectively, see [2, Chapter 3.4]. As we already know that there is a non-split extension of $1a$ with $4370a$, we conclude by [2, Corollary 2.5.4] that $H^1_{\mathbb{F}_2}(G, 4370a) \neq \{0\}$. By an application of the probabilistic technique to compute upper bounds on dimensions of group 1-cohomology described in [15], we find $\dim_{\mathbb{F}_2}(H^1_{\mathbb{F}_2}(G, 4370a)) \leqslant 1$, hence we have equality, and thus the probabilistic technique indeed yields a genuine non-trivial 1-cocycle in $Z^1_{\mathbb{F}_2}(G, 4370a) \setminus B^1_{\mathbb{F}_2}(G, 4370a)$. Using the interpretation in [2, Proposition 3.7.2] any such 1-cocycle describes the matrix entries for a non-split extension M of $1a$ with $4370a$.

Note that to store a point in M we need $\lceil 4371/8 \rceil = 547$ Bytes, hence to store all of X needs $555\,735\,879\,516\,160\,000 \sim 5.6 \cdot 10^{17}$ Bytes. Hence we are indeed tempted to apply the strategy described in Section 4. We choose the following chain of subgroups, see Table 1:

$$G = B > H = Fi_{23} > U_3 := S_8(2) > U_2 := 2^{10}:A_8 > U_1 := A_7.$$

Words in the standard generators for H giving non-standard generators for the maximal subgroup $S_8(2)$ are available in [28]. We derive a suitable small faithful permutation representation of $S_8(2)$, and by a random search we find standard generators for $S_8(2)$. The subgroup $2^{10}:A_8 < S_8(2)$ again is maximal, and since the unique transitive permutation representation of $S_8(2)$ on 2295 points also is available in terms of standard generators in [28], Algorithm 1 yields generators for $2^{10}:A_8$. By a random search we find generators for a complement A_8 of the normal subgroup $2^{10} \lhd 2^{10}:A_8$, and finally generators for $A_7 < A_8$.

As described in Section 5.1, we specify a chain of smaller and smaller quotients M_i of M: first let $M_5 := M$ and $M_4 := 782a$ and let $\pi = \pi_4$ be the natural projection of $M|_H$ onto its direct

Table 1
The subgroup chain

i	U_i	$\lvert U_i \rvert$	$[U_i : U_{i-1}]$	$\dim_{\mathbb{F}_2}(M_i)$
5	B	4 154 781 481 226 426 191 177 580 544 000 000	$\sim 10^{15}$	4371
4	Fi_{23}	4 089 470 473 293 004 800	86 316 516	782
3	$S_8(2)$	47 377 612 800	2295	42
2	$2^{10}:A_8$	20 643 840	8192	31
1	A_7	2520	2520	18

summand isomorphic to M_4. We find that $M_4|_{U_3}$ has a uniquely determined quotient module M_3 being isomorphic to a uniserial module with descending composition series $(16a, 26a)$. Moreover, we similarly find that $M_3|_{U_2}$ has a uniquely determined submodule of $\mathbb{F}_2$-dimension 11. The quotient module M_2 with respect to this submodule has Loewy series $(1a, 4a, 6a \oplus 6a, 14a)$. Finally, $M_2|_{U_1}$ turns out to have a uniquely determined quotient module $M_1 \cong 4a \oplus 14a$. The associated homomorphisms $\pi_i : M_{i+1} \to M_i$, for $1 \leqslant i \leqslant 3$, are just the natural maps.

8. The Fi_{23}-orbits

Keeping the notation of Section 6, the next task is to determine the partition $X = \dot{\bigcup}_{i=1,\dots,23} X_i \subseteq M$ of X into the H-orbits $X_i = x_i H$ by finding suitable representatives $x_i \in X$; note that we do not even know the sizes $k_i = |X_i|$ in advance. To do this, we do not describe the X_i directly, but instead find the H-orbits $X_i^\pi = x_i^\pi H \subseteq M_4$. These in turn are enumerated using the strategy described in Section 4, applied to the group H and the chain of helper subgroups $U_3 > U_2 > U_1$. The final result is given in Table 2, where the H-orbits X_i are sorted according to their size k_i.

If we are given some $x_i \in X$, to enumerate $x_i^\pi H$ we run Algorithm 2 with some parameter $1/2 < f < 1$; some numerical data on how this behaves in practice is given in Table 4 at the end of this section. This ensures that we find $\widetilde{H}_i := \mathrm{Stab}_H(x_i^\pi) \leqslant H$. Then we compute $x_i \widetilde{H}_i \subseteq X_i$ by Algorithm 1. Thus we obtain $H_i := \mathrm{Stab}_H(x_i) \leqslant \widetilde{H}_i$, and we have $[H_i : \widetilde{H}_i] = |x_i \widetilde{H}_i|$ as well as $k_i = [H : H_i]$. For group theoretical computations, such as the determination of subgroup orders,

Table 2
H-orbits in X

i	k_i	$\|H_i\|$	H_i	$\widetilde{H}_i$	$[\widetilde{H}_i : H_i]$
1	1	$\sim 4.1 \cdot 10^{18}$	Fi_{23}		
2	412 896	9 904 359 628 800	$O_8^+(3):2_2$		
3	86 316 516	47 377 612 800	$S_8(2)$	Fi_{23}	86 316 516
4	195 747 435	20 891 566 080	$2^{11}.M_{23}$	Fi_{23}	195 747 435
5	8 537 488 128	479 001 600	S_{12}		
6	23 478 092 352	174 182 400	$O_8^+(2)$		
7	33 816 182 400	120 932 352	$[3^9].[2^{10}].S_3$	$[3^9].[2^{10}].3^2.2$	3
8	113 778 447 552	35 942 400	$2 \times {}^2F_4(2)'$	$2.Fi_{22}$	3 592 512
9	160 533 964 800	25 474 176	$S_3 \times G_2(3)$	$S_3 \times O_7(3)$	1080
10	504 245 392 560	8 110 080	$2^{10}.M_{11}$	$2^{11}.M_{11}$	2
11	1 044 084 577 536	3 916 800	$S_4(4):4$		
12	1 152 560 897 280	3 548 160	$(2 \times 2.M_{22}).2$		
13	1 584 771 233 760	2 580 480	$2^7.A_8$		
14	5 282 570 779 200	774 144	$2^7.U_3(3)$	$2^7.U_3(3).2$	2
15	7 888 639 030 272	518 400	$(A_6 \times A_6):2^2$		
16	12 678 169 870 080	322 560	$2^2.L_3(4).2^2$		
17	21 514 470 082 560	190 080	$2 \times M_{12}$		
18	43 028 940 165 120	95 040	M_{12}		
19	50 712 679 480 320	80 640	$2.L_3(4).2_2$		
20	133 120 783 635 840	30 720	$2^4.2^4.A_5.2$		
21	190 172 548 051 200	21 504	$2^6:L_3(2):2$		
22	262 954 634 342 400	15 552	$3^4.2^{1+4}.S_3$		
23	283 991 005 089 792	14 400	$(A_5 \times A_5):2^2$		

we use the smallest faithful permutation representation of H on 31 671 points, being available in [28].

Hence we have to find suitable representatives $x_i \in X$ for the H-orbits X_i. Beginning with $x_1 \in X$, we apply a few random elements of G, and for the points $x \in X$ thus obtained we enumerate $x^{\pi}H$. This random search yields 14 of the H-orbits, namely those for $i \in \{1, 7, 11, 13, \ldots, 23\}$, being underlined in Table 2. These H-orbits of course tend to be the large ones, and summing up the associated orbit sizes k_i, and dividing by $|X|$, we obtain a fraction of $\sim 499/500$. Hence it seems rather improbable to find further H-orbits using such a random search. As the small H-orbits for $i \in \{2, \ldots, 6, 8, 9, 10, 12\}$ are missing, we are tempted to look for large candidate subgroups of H instead which might occur as stabilisers H_i.

Now the Schur double cover $2.G := 2.B$ of the Baby Monster group is a subgroup of the sporadic simple Fischer–Griess Monster group $\mathbb{M}$. More precisely, it is the involution centraliser $2.G = C_{\mathbb{M}}(a)$ of an element a in the $2A$-conjugacy class in $\mathbb{M}$, where a is a 6-*transposition*, since the product of a with any of its conjugates has order at most 6.

Let $Z := Z(2.G) = \langle a \rangle$ and let $H' < 2.G$ be a subgroup isomorphic to the Fischer group Fi_{23}, hence we have $H \cong (H' \times Z)/Z$. By [23] we have $H' = C_{\mathbb{M}}(a, b)$, where $\langle a, b \rangle \cong S_3$, where in turn b also is a 6-transposition and ab belongs to the $3A$-conjugacy class in $\mathbb{M}$. Given $g \in 2.G$ we have $H' \cap H'^g = C_{\mathbb{M}}(\langle a, b \rangle, \langle a, c \rangle) = C_{\mathbb{M}}(a, b, c)$, where $c = a^g$ also is a 6-transposition and $\langle a, c \rangle \cong S_3$. Since $N_{2.G}(H') = \langle a \rangle \times H'$, we may assume that $H'^g \neq H'$, and thus $\langle a, b \rangle \cap \langle a, c \rangle = \langle a \rangle$.

To deduce the corresponding information in G itself, we need to quotient by the subgroup Z, i.e. we have to determine $((H' \times Z) \cap (H'^g \times Z))/Z$. Since $(H' \times Z) \cap (H'^g \times Z) = (H' \cap (H'^g \times Z)) \times Z$, there are two cases: in the *split* case we have $(H' \cap H'^g) \times Z = (H' \times Z) \cap (H'^g \times Z)$, while in the *non-split* case we have $(H' \cap H'^g) \times Z \lhd (H' \times Z) \cap (H'^g \times Z)$, a normal subgroup of index 2. Thus we are in the non-split case if and only if

$$C_{\mathbb{M}}(a, b, c) = H' \cap H'^g < H' \cap \bigl(H'^g \times Z\bigr) = C_{\mathbb{M}}(a, b) \cap \bigl(C_{\mathbb{M}}(a, c) \times \langle a \rangle\bigr).$$

This in turn is the case if and only if there is $x \in N_{\mathbb{M}}(\langle a, b, c \rangle)$ such that $a^x = a$, $b^x = b$ and $c^x = c^a$.

We use the table of centralisers of subgroups of $\mathbb{M}$ given in [22, Table 1] to look for suitable subgroups being generated by triples (a, b, c) of 6-transpositions, such that $\langle a, b \rangle \cong \langle a, c \rangle \cong S_3$, and both ab and ac belong to the $3A$-conjugacy class in $\mathbb{M}$. The subgroups leaping to mind are listed in Table 3; the fourth column indicates whether the split "$+$" or the non-split "$-$" case occurs, and in the fifth column the corresponding row of Table 2 is given.

For example, the subgroup generated might be isomorphic to S_4, where $a = (1, 2)$ and $b = (2, 3)$, while $c = (1, 4)$ or $c = (2, 4)$. There are two such subgroups: one has centraliser $S_8(2)$ and normaliser $S_4 \times S_8(2)$ in $\mathbb{M}$, while the other has centraliser $2^{11}.M_{23}$ and normaliser $S_4 \times 2^{11}.M_{23}$.

In the first case the involutions in S_4 are 6-transpositions, since they centralise elements of order 17, but the centraliser in $\mathbb{M}$ of the $2B$-conjugacy class is isomorphic to $2^{1+24}.Co_1$, thus has no such elements. It follows from [23, Table 3] that there is a conjugacy class of subgroups isomorphic to S_4, being generated by a triple (a, b, c) where bc also is a 6-transposition. This obviously is a split case, proving row $i = 3$. In the second case, considering the conjugacy class fusion from $S_4 \times 2^{11}.M_{23}$ to $\mathbb{M}$ shows that the transpositions in S_4 indeed are 6-transpositions. This also is a split case, proving row $i = 4$.

For the other cases we proceed similarly. To check conjugacy class fusions we use the character table library of GAP, even though in many cases they are well known or easy to see. As

Table 3
Centralizers of certain subgroups of $\mathbb{M}$

$\langle a,b,c\rangle$	$C_{\mathbb{M}}(\langle a,b,c\rangle)$	$N_{\mathbb{M}}(\langle a,b,c\rangle)$	Split	i
$3^2:2$	$O_8^+(3)$	$((3^2:2)\times O_8^+(3)).S_4$	$-$	2
S_4	$S_8(2)$	$S_4\times S_8(2)$	$+$	3
S_4	$2^{11}.M_{23}$	$S_4\times 2^{11}.M_{23}$	$+$	4
A_5	A_{12}	$(A_5\times A_{12}):2$	$-$	5
$2S_4$	$2\times {}^2F_4(2)'$	$(2S_4\times {}^2F_4(2)').2$	$+$	8
$3^{1+2}:2^2$	$G_2(3)$	$(3^{1+2}:2^2\times G_2(3)).2$	$-$	9
$4^2:S_3$	$2^{10}.M_{11}$	$(4^2:S_3\times 2^{10}.M_{11}).2$	$+$	10
$2\times S_5$	$2.M_{22}$	$(S_5\times 2.M_{22}).2$	$-$	12
$L_2(7)$	$S_4(4).2$	$(L_2(7)\times S_4(4).2).2$	$-$	11
$L_2(11)$	M_{12}	$(L_2(11)\times M_{12}).2$	$+/-$	17, 18

it turns out, we rediscover rows $i=11$ as well as $i=17$ and $i=18$, which have already been found by the random search. Moreover, we remark that the existence of stabilisers as in rows $i=2$ and $i=3$ has also been stated in [10, p. 3422].

At this stage we have just a single orbit left to find, and the number of points left is 23 478 092 352. Hence the last stabiliser has order 174 182 400, which strongly hints at $O_8^+(2)$ as indicated in row $i=6$.

It remains to find representatives $x_i\in X$, for $i\in\{2,\dots,6,8,9,10,12\}$, and to prove row $i=6$. Given generators for the associated stabiliser H_i, we compute the subspace $\mathrm{Fix}_M(H_i)<M$ consisting of the H_i-invariant vectors, and for each $x\in\mathrm{Fix}_M(H_i)\setminus\{0,x_1\}$ we proceed as follows: we compute a few elements $y\in xG\subseteq M$, and check whether $y^\pi\in M_4$ is a point in an H-orbit encountered earlier. If we succeed in proving $y^\pi\in X_j^\pi$, for some j, then Algorithm 2 also yields an element $h\in H$ such that $y^\pi h=x_j^\pi$. It is then checked whether $yh=x_j$ holds, which proves that $y\in X$ and hence $x\in X$. It is easy then to compute the associated subgroups $\widetilde{H}_i$, and we remark that it turns out that $X_i^\pi=\{0\}\subseteq M_4$ for $i\in\{3,4\}$.

Hence we are left with actually finding generators for the various H_i: words in the standard generators of H giving generators of the maximal subgroups $H_3=S_8(2)$, and $H_4=2^{11}.M_{23}$, and $H_5=S_{12}$ are available in [28]. Moreover, we have $H_2=O_8^+(3):2_2<O_8^+(3):S_3$, and $H_8=2\times{}^2F_4(2)'<2.Fi_{22}$, as well as $H_9=S_3\times G_2(3)<S_3\times O_7(3)$, and $H_{10}=2^{10}.M_{11}<2^{11}.M_{23}$, and $H_{12}=(2\times 2.M_{22}).2<2^2.U_6(2).2$, where the overgroups again are maximal subgroups of H, hence generators for these H_i are easy to find as well. Note that for $i=9$ there are two conjugacy classes of subgroups of $S_3\times O_7(3)$ isomorphic to $S_3\times G_2(3)$ only one of which yields a suitable vector $x_9\in X$.

For the candidate $H_6=O_8^+(2)$ there are three conjugacy classes of maximal subgroups of H containing a subgroup isomorphic to $O_8^+(2)$, namely $S_8(2)$, and $O_8^+(3):S_3$, and $2.Fi_{22}$. Again it is easy to find generators for the relevant subgroups isomorphic to $O_8^+(2)$. Indeed it turns out that a subgroup $O_8^+(2)<S_8(2)$ yields a suitable vector $x_6\in X$, thus proving row $i=6$.

We conclude this section by presenting some numerical data on the enumeration of the H-orbits $X_i^\pi=x_i^\pi H\subseteq M_4$, for $i\notin\{1,3,4\}$, with respect to the helper subgroup U_3 and the map $\pi_3:M_4\to M_3$. This has been done using a slight modification of Algorithm 2, where we have specified $f=1$, but the break condition has been $p=2$, i.e. the generators of U_3 are never applied to U_3-orbit representatives. Moreover, motivated by the analysis at the end of Section 2, for $i\notin\{2,8,9\}$ all points $x\in X_i^\pi$ such that $|\mathrm{Stab}_{U_3}(x^{\pi_3})|>10^5$ are ignored and their U_3-orbits

Table 4
Statistics for H-orbits in X^{π}

i	$\widetilde{k}_i$	$\lvert\mathcal{X}\rvert$	$\widetilde{k}_i/\lvert\mathcal{X}\rvert$	U_3-orbits	$N_{\mathcal{X}}$	$\lvert\mathcal{X}\rvert/N_{\mathcal{X}}$
23	283 991 005 089 792	281 173 991 454 720	0.99	8105	1 433 928	196 086 547
22	262 954 634 342 400	260 326 657 382 400	0.99	6977	1 198 807	217 154 769
21	190 172 548 051 200	188 272 393 804 800	0.99	5271	1 263 408	149 019 472
20	133 120 783 635 840	131 793 266 626 560	0.99	3916	621 625	212 014 102
19	50 712 679 480 320	49 702 192 081 920	0.98	1899	228 710	217 315 342
18	43 028 940 165 120	42 170 681 548 800	0.98	1485	438 005	96 278 995
17	21 514 470 082 560	21 085 044 664 320	0.98	770	198 485	106 229 914
16	12 678 169 870 080	12 300 050 810 880	0.97	524	138 605	88 741 753
15	7 888 639 030 272	7 659 885 219 840	0.97	490	78 695	97 336 364
14	2 641 285 389 600	2 562 503 731 200	0.97	154	69 664	36 783 758
13	1 584 771 233 760	1 490 058 823 680	0.94	131	96 244	15 482 095
12	1 152 560 897 280	1 083 499 683 840	0.94	101	20 861	51 939 009
11	1 044 084 577 536	1 015 328 563 200	0.97	100	18 941	53 604 802
10	252 122 696 280	223 859 220 480	0.88	33	8864	25 254 875
6	23 478 092 352	21 311 994 512	#0.90	24	409 886	51 994
7	11 272 060 800	10 158 220 800	*0.88	8	193 554	52 482
5	8 537 488 128	7 262 008 320	0.85	11	966	7 517 606
9	148 642 560	135 080 640	*0.90	5	17 794	7591
2	412 896	366 792	*0.88	2	122	3006
8	31 671	31 416	#0.90	2	13 064	2

simply are not stored. Thus we enumerate a certain subset $\mathcal{X} \subseteq X_i^{\pi}$, which still consists of U_3-orbits. For the H-orbits whose percentage is marked with a * we increased the stabiliser limit for storing to $3 \cdot 10^{10}$, and for those marked with a # we imposed no limit at all.

In Table 4 we have compiled the following data: the H-orbits X_i^{π} are sorted according to their size $\widetilde{k}_i := |X_i^{\pi}| = [H : \widetilde{H}_i]$, we give the cardinality $|\mathcal{X}|$ of the subsets $\mathcal{X} \subseteq X_i^{\pi}$ actually enumerated, which fraction of whole H-orbit X_i^{π} this is, the number of U_3-orbits in $\mathcal{X}$, the number $N_{\mathcal{X}}$ of U_3-minimal points in $\mathcal{X}$, and the saving factor $N_{\mathcal{X}}/|\mathcal{X}|$. The fractions $|\mathcal{X}|/\widetilde{k}_i$ being very close to 1 shows that indeed the generators of the helper subgroup have to be applied to orbit representatives only at the very end of an orbit enumeration.

To store a point in M_4 we need $\lceil 782/8 \rceil = 98$ Bytes, thus to store all of $X^{\pi} \subseteq M_4$ still needs $99\,565\,111\,869\,440\,000 \sim 10^{17}$ Bytes. To enumerate X^{π} applying the strategy described in Section 4 and the slight modification given above, using the ORB package, needs $\sim 1.1 \cdot 10^9$ Bytes of memory space, and ~ 4800 s ~ 80 min of CPU time on a 3.2 GHz Pentium IV processor, where both figures include the time and space required to enumerate and store the appropriate portions of the helper sets M_3, M_2 and M_1.

9. The character table

The final task is now to compute the intersection matrix $P_2 = [p_{h,2,j}] \in \mathbb{Z}^{23\times 23}$ for the smallest non-trivial H-orbit X_2, which has size $k_2 = 412\,896$, and since it is the only H-orbit having this size it is self-paired. We have

$$p_{h,2,j} = |X_2 g_h \cap X_j| \cdot k_h / k_j,$$

hence the task is to enumerate all of $X_2 g_h$ explicitly, successively for every $2 \leqslant h \leqslant 23$, and to determine which H-orbits X_j (where $1 \leqslant j \leqslant 23$) the various points $x \in X_2 g_h$ belong to; recall that we are done for $h = 1$.

As we have not enumerated the H-orbits X_j directly, but the H-orbits X_j^π instead, the membership test is done by checking whether $x^\pi \in X_j^\pi$ holds, whenever $j \notin \{1, 3, 4\}$; the cases $j \in \{3, 4\}$ will be commented on below, while $j = 1$ only occurs for $i = 2$ and checking whether $x = x_1$ is easy anyway.

In turn, as we have enumerated only parts of the X_j^π explicitly, we have to check a few points in $x^\pi H$ for membership. This only allows us to prove membership, but not to disprove it. Hence we let j vary, and in a first run we test a very few points in $x^\pi H$, at most 5 say, for membership in X_j^π. If x^π cannot be proven to belong to a particular H-orbit, we start a second run where we test some more points in $x^\pi H$, at most 1000 say. Now this is done for all $x \in X_2 g_h$, and it turns out that after the second run only a very few points have not been proven to belong to a particular H-orbit, in particular including those which belong to X_3 or X_4.

Hence we have found lower bounds for the matrix entries $p_{h,2,j} \in \mathbb{N}_0$. Now we have $\sum_{j=1}^{23} p_{h,2,j} k_j = k_2 k_h$, and moreover $p_{h,2,j} = p_{j,2,h} \cdot k_j / k_h$, which is an integrality condition, and in particular implies that $p_{h,2,j} = 0$ if and only if $p_{h,2,i} = 0$. It turns out that these conditions are sufficient to find all the matrix entries $p_{h,2,j}$. The resulting intersection matrix P_2 is shown in Tables 5–6.

Finally, it turns out that all the row eigenspaces of the matrix $P_2^{\mathrm{tr}} \in \mathbb{Q}^{23 \times 23}$ are already 1-dimensional, hence normalising the eigenvectors to have 1 as their first entry yields the character table Φ_E, which together with the degrees of the Fitting correspondents is shown in Tables 7–10.

Table 5
Intersection matrix P_2

i	k_i	1	2	3	4	5	6	7	8	9	10
1	1	.	1	.	.	.	.	.	.	.	.
2	412 896	412 896	2	136	.	.	1	4	.	.	.
3	86 316 516	.	28 431	.	.	462	1	.	.	.	.
4	195 747 435	.	.	.	.	.	135	.	.	.	.
5	8 537 488 128	.	.	45 696	.	.	.	3888	.	.	1056
6	23 478 092 352	.	56 862	272	16 192	.	136	.	.	.	.
7	33 816 182 400	.	327 600	.	.	15 400	.	8	.	364	.
8	113 778 447 552	.	.	.	.	.	.	.	3200	1134	.
9	160 533 964 800	.	.	.	.	.	.	1728	1600	728	.
10	504 245 392 560	.	.	.	.	62 370	.	.	.	.	.
11	1 044 084 577 536	.	.	.	.	.	12 096	.	.	.	.
12	1 152 560 897 280	.	.	.	129 536	.	.	.	.	.	1760
13	1 584 771 233 760	.	.	275 400	8096	.	16 335	78 732	.	.	33 440
14	5 282 570 779 200	.	.	.	.	.	16 200	.	.	2106	.
15	7 888 639 030 272	.	.	91 392	.	924	79 296	23 328	.	.	37 312
16	12 678 169 870 080	.	.	.	.	178 200	.	.	.	37 908	.
17	21 514 470 082 560	.	.	.	.	.	.	139 968	12 480	.	101 376
18	43 028 940 165 120	.	.	.	.	.	.	.	24 960	58 968	.
19	50 712 679 480 320	.	.	.	259 072	124 740	.	.	.	.	2112
20	133 120 783 635 840	.	.	.	.	.	226 800	157 464	.	.	135 168
21	190 172 548 051 200	.	.	.	.	.	16 200	.	280 800	75 816	10 560
22	262 954 634 342 400	.	.	.	.	30 800	33 600	7776	.	235 872	.
23	283 991 005 089 792	.	.	.	.	.	12 096	.	89 856	.	90 112

Table 6
Intersection matrix P_2, continued

i	11	12	13	14	15	16	17	18	19	20	21	22	23
1	.	.	.	.	.	.	.	.	.	.	.	.	.
2	.	.	.	.	.	.	.	.	.	.	.	.	.
3	.	.	15	.	1	.	.	.	.	.	.	.	.
4	.	22	1	.	.	.	.	.	1	.	.	.	.
5	.	.	.	.	1	120	.	.	21	.	.	1	.
6	272	.	242	72	236	.	.	.	.	40	2	3	1
7	.	.	1680	.	100	.	220	.	.	40	.	1	.
8	.	.	.	.	.	.	66	66	.	.	168	.	36
9	.	.	.	64	.	480	.	220	.	.	64	144	.
10	.	770	10 640	.	2385	.	2376	.	21	512	28	.	160
11	1360	1232	.	.	36	112	.	1980	700	.	672	486	176
12	1360	.	.	4320	1575	1400	.	.	211	496	128	567	600
13	.	.	30	2376	.	9632	.	396	3420	40	30	945	175
14	.	19 800	7920	128	1350	.	6270	990	2370	2560	844	1512	3300
15	272	10 780	.	2016	626	15 120	792	3696	12 866	480	1008	4596	2546
16	1360	15 400	77 056	.	24 300	240	29 700	396	420	13 056	3088	1350	5400
17	.	.	.	25 536	2160	50 400	440	6996	28 560	1792	3136	13 824	6360
18	81 600	.	10 752	8064	20 160	1344	13 992	21 032	3360	24 064	30 016	11 232	14 760
19	34 000	9284	109 440	22 752	82 710	1680	67 320	3960	5542	41 664	16 016	9828	24 110
20	.	57 288	3360	64 512	8100	137 088	11 088	74 448	109 368	23 672	38 976	76 707	45 600
21	122 400	21 120	3600	30 384	24 300	46 320	27 720	132 660	60 060	55 680	108 608	81 972	64 800
22	122 400	129 360	156 800	75 264	153 200	28 000	168 960	68 640	50 960	151 520	113 344	81 640	118 600
23	47 872	147 840	31 360	177 408	91 656	120 960	83 952	97 416	135 016	97 280	96 768	128 088	126 272

Table 7
The character table

i	$\chi_\lambda(1)$	1	2	3	4	5	6	7
1	1	1	412 896	86 316 516	195 747 435	8 537 488 128	23 478 092 352	33 816 182 400
2	4371	1	−137 632	18 115 812	−10 472 085	−1 159 411 968	1 449 264 960	3 757 353 600
3	96 255	1	82 016	8 890 596	5 701 995	457 037 568	327 742 272	1 297 296 000
4	9 458 750	1	41 888	3 232 548	−43 605	123 026 688	57 841 344	314 160 000
5	63 532 485	1	−32 032	2 275 812	414 315	−77 223 168	−2 312 640	179 625 600
6	347 643 114	1	10 208	704 484	1 589 355	10 679 040	46 398 528	−9 609 600
7	356 054 375	1	−17 248	900 900	−1 508 949	−20 097 792	43 902 144	32 672 640
8	4 221 380 670	1	−3232	324 324	103 275	−2 453 760	15 121 728	−12 297 600
9	4 275 362 520	1	14 816	725 796	−43 605	16 743 168	−7 316 928	31 920 000
10	9 287 037 474	1	6896	132 516	699 435	736 128	11 096 352	4 502 400
11	13 508 418 144	1	−11 632	475 812	111 915	−9 283 968	−491 040	17 673 600
12	108 348 770 530	1	7328	246 564	−43 605	3 421 440	1 729 728	4 502 400
13	309 720 864 375	1	−1120	89 892	−181 845	−172 800	3 172 032	−3 638 400
14	635 966 233 056	1	3408	69 284	147 755	295 040	2 450 528	−169 600
15	1 095 935 366 250	1	−4576	126 756	2475	−1 324 800	−949 824	1 061 760
16	6 145 833 622 500	1	2864	51 876	−26 325	316 800	−507 744	309 120
17	6 619 124 890 560	1	1088	39 204	25 515	138 240	−300 672	−1 065 600
18	12 927 978 301 875	1	−2128	19 620	−40 149	67 968	706 464	186 240
19	38 348 970 335 820	1	−1232	15 524	37 675	19 840	−69 472	−233 600
20	89 626 740 328 125	1	944	1188	15 147	−79 488	61 344	63 360
21	211 069 033 500 000	1	560	1188	−12 501	−51 840	12 960	−68 736
22	284 415 522 641 250	1	−16	−5724	8235	17 280	50 976	78 720
23	364 635 285 437 500	1	−400	−1116	−5589	26 496	−71 136	−7296

Table 8
The character table, continued

i	8	9	10	11	12	13
1	113 778 447 552	160 533 964 800	504 245 392 560	1 044 084 577 536	1 152 560 897 280	1 584 771 233 760
2	1 404 672 192	−5 945 702 400	39 426 594 480	−21 483 221 760	−4 743 048 960	−110 868 769 440
3	−1 788 671 808	−511 948 800	12 027 702 960	−9 527 341 824	6 966 984 960	30 484 602 720
4	183 218 112	258 508 800	1 991 288 880	1 252 323 072	−1 021 697 280	4 906 012 320
5	−32 332 608	35 481 600	1 084 693 680	550 851 840	−432 034 560	−2 400 567 840
6	57 081 024	−167 270 400	224 426 160	533 820 672	271 607 040	−9 741 600
7	−21 155 904	63 866 880	185 985 072	−186 810 624	778 242 816	−259 829 856
8	−15 494 976	74 188 800	87 499 440	−219 034 368	−142 145 280	29 121 120
9	14 841 792	4 147 200	110 118 960	−61 012 224	62 588 160	198 033 120
10	−38 864 448	20 044 800	−21 727 440	115 105 536	171 953 280	32 315 760
11	7 584 192	−18 662 400	32 946 480	−61 205 760	−22 584 960	−74 323 440
12	−11 866 176	−6 912 000	5 609 520	−1 790 208	−28 857 600	−1 265 760
13	6 934 464	−6 912 000	12 798 000	19 554 048	−7 568 640	3 745 440
14	6 681 024	5 913 600	−1 900 240	−8 656 128	8 992 640	−2 385 200
15	−254 016	1 935 360	−841 680	6 983 424	3 168 000	10 755 360
16	1 197 504	691 200	−2 857 680	2 467 584	−777 600	−4 879 440
17	−1 498 176	−460 800	2 430 000	−1 928 448	3 732 480	−3 810 240
18	−627 264	−414 720	−2 332 368	−1 292 544	−307 584	−943 056
19	−576	76 800	−292 560	472 832	−1 668 480	588 720
20	36 288	−709 632	−452 304	−850 176	134 784	854 064
21	−129 600	248 832	73 008	200 448	−335 232	518 832
22	−46 656	138 240	114 480	532 224	−293 760	−481 680
23	119 232	−82 944	86 832	−352 512	508 032	−42 768

Table 9
The character table, continued

i	14	15	16	17	18
1	5 282 570 779 200	7 888 639 030 272	12 678 169 870 080	21 514 470 082 560	43 028 940 165 120
2	65 216 923 200	−292 171 815 936	573 908 924 160	−796 832 225 280	531 221 483 520
3	28 447 848 000	58 091 185 152	118 446 831 360	158 430 504 960	−222 361 251 840
4	−3 514 104 000	3 727 696 896	12 802 648 320	10 166 446 080	20 332 892 160
5	1 235 995 200	−300 174 336	4 718 165 760	−4 534 548 480	−8 511 713 280
6	916 660 800	2 067 158 016	−1 656 357 120	−679 311 360	1 892 782 080
7	−2 109 032 640	−1 909 619 712	−643 458 816	1 675 634 688	1 177 473 024
8	499 867 200	−274 627 584	−544 631 040	−5 806 080	592 220 160
9	197 640 000	−366 363 648	5 218 560	−75 479 040	−452 874 240
10	217 339 200	−118 153 728	122 446 080	−322 237 440	661 893 120
11	−10 756 800	200 600 064	−34 179 840	269 982 720	836 075 520
12	−80 222 400	35 030 016	−96 802 560	−145 152 000	−11 612 160
13	−43 200	−48 356 352	−17 729 280	18 524 160	−16 035 840
14	−15 211 200	36 246 016	7 220 480	−39 797 760	−41 656 320
15	2 721 600	1 741 824	−31 921 920	−5 806 080	−58 060 800
16	5 417 280	−5 515 776	518 400	14 515 200	11 612 160
17	648 000	5 308 416	933 120	14 100 480	−9 953 280
18	2 928 960	787 968	6 269 184	7 216 128	−6 967 296
19	−1 924 800	−2 025 984	4 348 160	−1 582 080	5 468 160
20	938 304	−1 866 240	518 400	−746 496	−1 658 880
21	−720 576	898 560	1 237 248	−1 410 048	995 328
22	25 920	−262 656	−1 416 960	2 903 040	−1 658 880
23	191 808	290 304	−311 040	−1 741 824	995 328

Table 10
The character table, continued

i	19	20	21	22	23
1	50 712 679 480 320	133 120 783 635 840	190 172 548 051 200	262 954 634 342 400	283 991 005 089 792
2	1 460 859 079 680	−2 739 110 774 400	−782 603 078 400	3 246 353 510 400	−1 168 687 263 744
3	239 651 343 360	190 079 809 920	−857 327 328 000	28 598 169 600	218 194 808 832
4	7 936 220 160	8 210 885 760	47 791 814 400	−25 333 862 400	−90 188 550 144
5	−1 053 803 520	12 753 417 600	10 828 857 600	−17 953 689 600	3 908 653 056
6	3 994 721 280	−5 895 711 360	1 568 160 000	−10 005 811 200	6 838 013 952
7	3 238 050 816	−155 675 520	−44 478 720	−6 826 659 840	4 981 616 640
8	722 856 960	813 214 080	−13 996 800	−1 025 740 800	−578 285 568
9	−1 233 239 040	−1 778 474 880	666 144 000	148 377 600	2 518 290 432
10	−489 991 680	959 091 840	−1 020 988 800	174 182 400	−479 582 208
11	−664 312 320	−183 254 400	−1 004 918 400	593 510 400	125 024 256
12	83 082 240	268 168 320	−170 553 600	212 889 600	−59 609 088
13	−61 793 280	98 133 120	−116 640 000	190 771 200	−74 649 600
14	−22 725 120	16 717 440	9 264 000	80 076 800	−41 576 448
15	36 449 280	−18 264 960	41 644 800	94 187 520	−83 349 504
16	15 137 280	9 797 760	−15 085 440	−21 934 080	−10 450 944
17	−9 953 280	−18 195 840	27 993 600	27 648 000	−35 831 808
18	−2 225 664	−16 744 320	22 654 080	−8 663 040	−276 480
19	−919 040	−1 537 920	−7 036 800	−17 100 800	23 365 632
20	2 198 016	3 825 792	6 065 280	−4 534 272	−3 815 424
21	−1 893 888	−4 053 888	−1 316 736	−2 764 800	8 570 880
22	−69 120	−3 058 560	51 840	6 082 560	−2 709 504
23	705 024	4 572 288	−1 026 432	−700 416	−3 151 872

References

[1] E. Bannai, T. Ito, Algebraic Combinatorics I: Association Schemes, Benjamin, 1984, MR0882540 (87m:05001).

[2] D. Benson, Representations and Cohomology I, Cambridge Stud. Adv. Math., vol. 30, Cambridge Univ. Press, 1998, MR1644252 (99f:20001a).

[3] T. Breuer, K. Lux, The multiplicity-free permutation characters of the sporadic simple groups and their automorphism groups, Comm. Algebra 24 (1996) 2293–2316, MR1390375 (97c:20020).

[4] T. Breuer, J. Müller, Character tables of endomorphism rings of multiplicity-free permutation modules of the sporadic simple groups and their cyclic and bicyclic extensions, http://www.math.rwth-aachen.de/~Juergen.Mueller/mferctbl/mferctbl.html, 2005.

[5] A. Brouwer, A. Cohen, A. Neumaier, Distance-Regular Graphs, Ergeb. Math. Grenzgeb. (3), vol. 18, Springer, 1989, MR1002568 (90e:05001).

[6] J. Conway, R. Curtis, S. Norton, R. Parker, R. Wilson, Atlas of Finite Groups, Oxford Univ. Press, 1985, MR0827219 (88g:20025).

[7] G. Davidoff, P. Sarnak, A. Valette, Elementary Number Theory, Group Theory, and Ramanujan Graphs, London Math. Soc. Stud. Texts, vol. 55, Cambridge Univ. Press, 2003, MR1989434 (2004f:11001).

[8] The GAP Group, GAP—Groups, Algorithms, and Programming, Version 4.4, http://www.gap-system.org, 2005.

[9] D. Higman, A monomial character of Fischer's baby monster, in: Proc. of the Conference on Finite Groups, Utah, 1975, Academic Press, 1976, pp. 277–283, MR0409627 (53 #13379).

[10] A. Ivanov, S. Linton, K. Lux, J. Saxl, L. Soicher, Distance-transitive representations of the sporadic groups, Comm. Algebra 23 (1995) 3379–3427, MR1335306 (96g:20019).

[11] C. Jansen, The minimal degrees of faithful representations of the sporadic simple groups and their covering groups, LMS J. Comput. Math. 8 (2005) 122–144, MR2153793.

[12] D. Johnson, Presentations of Groups, London Math. Soc. Stud. Texts, vol. 15, Cambridge Univ. Press, 1997, MR1472735 (98e:20001).

[13] P. Landrock, Finite Group Algebras and Their Modules, London Math. Soc. Lecture Note Ser., vol. 84, Cambridge Univ. Press, 1983, MR0737910 (85h:20002).

[14] F. Lübeck, M. Neunhöffer, Enumerating large orbits and direct condensation, Experiment. Math. 10 (2) (2001) 197–205, MR1837671 (2002m:20028).
[15] K. Lux, Algorithmic methods in modular representation theory, Habilitationsschrift, RWTH Aachen, 1997.
[16] K. Lux, J. Müller, M. Ringe, Peakword condensation and submodule lattices: An application of the MeatAxe, J. Symbolic Comput. 17 (6) (1994) 529–544, MR1300352 (95h:68091).
[17] J. Müller, On the multiplicity-free actions of the sporadic simple groups, preprint, http://www.math.rwth-aachen.de/~Juergen.Mueller/, 2007.
[18] J. Müller, On the action of the sporadic simple Baby Monster group on the cosets of $2^{1+22}.Co_2$, preprint, http://www.math.rwth-aachen.de/~Juergen.Mueller/, 2006.
[19] J. Müller, On endomorphism rings and character tables, Habilitationsschrift, RWTH Aachen, 2003.
[20] J. Müller, M. Neunhöffer, F. Noeske, GAP-4 package ORB, http://www.math.rwth-aachen.de/~Max.Neunhoeffer/Computer/Software/Gap/orb.html, 2006.
[21] J. Müller, M. Neunhöffer, F. Röhr, R. Wilson, Completing the Brauer trees for the sporadic simple Lyons group, LMS J. Comput. Math. 5 (2002) 18–33, MR1916920 (2003e:20015).
[22] S. Norton, Anatomy of the Monster I, in: The Atlas of Finite Groups: Ten Years On, Birmingham, 1995, in: London Math. Soc. Lecture Note Ser., vol. 249, Cambridge Univ. Press, 1998, pp. 198–214, MR1647423 (99i:20021).
[23] S. Norton, The uniqueness of the Fischer–Griess Monster, in: Finite Groups—Coming of Age, Montreal, 1982, in: Contemp. Math., vol. 45, Amer. Math. Soc., 1985, pp. 271–285, MR0822242 (87b:20025).
[24] M. Ringe, The C-MeatAxe2.4, RWTH Aachen, 2003.
[25] I. Schur, Zur Theorie der einfach transitiven Permutationsgruppen, Sitzungsberichte der Preußischen Akademie der Wissenschaften, 1933, pp. 598–623.
[26] R. Wilson, Standard generators for sporadic simple groups, J. Algebra 184 (1996) 505–515, MR1409225 (98e:20025).
[27] R. Wilson, A new construction of the Baby Monster and its applications, Bull. London Math. Soc. 25 (5) (1993) 431–437, MR1233405 (94k:20027).
[28] R. Wilson, R. Parker, S. Nickerson, J. Bray, Atlas of finite group representations, http://brauer.maths.qmul.ac.uk/Atlas/, 2005.
[29] P. Zieschang, An Algebraic Approach to Association Schemes, Lecture Notes in Math., vol. 1628, Springer, 1996, MR1439253 (98h:05185).