The Brauer Tree of the Principal 19-Block of the Sporadic Simple Thompson Group

Gene Cooperman, Gerhard Hiss, Klaus Lux, and Jürgen Müller

CONTENTS

This paper completes the construction of the Brauer tree of the sporadic simple Thompson group in characteristic 19. Our main computational tool to arrive at this result is a new parallel implementation of the DirectCondense method.

1. INTRODUCTION AND RESULTS

Let Th be the sporadic simple Thompson group. In [Hiss and Lux 1989] the Brauer tree of the principal 19-block of Th has been determined up to two possibilities. In this note, we show which of them is the correct one, and we describe the new computational techniques that enabled us to decide between these two possibilities. We believe that the methods presented here will be powerful enough to solve even more difficult problems in the modular character theory of the sporadic groups.

As a general reference for the theory of blocks of cyclic defect, the interpretation of a Brauer tree and its planar embedding, see the introduction of [Hiss and Lux 1989]. The planar embedded Brauer tree of the principal 19-block of Th is given in Figure 1; it also appears in [Hiss and Lux 1989, p. 277, Case I]. Its nodes are labelled by the numbers of the corresponding ordinary irreducible characters, where we use the notation for the ordinary irreducible characters of Th as they are given in [Conway et al. 1985, p. 176] and in the GAP software [Schönert et al. 1994]. In Table 1 we list the ordinary irreducible characters of G lying in the principal 19-block, plus some additional information concerning them.

The degrees of the irreducible Brauer characters are given in Table 2. A Brauer character that

This paper is a contribution to the DFG research project "Algorithmic Number Theory and Algebra." Müller gratefully acknowledges financial support by DFG. Cooperman gratefully acknowledges the partial support by NSF Grant CCR-9509783.

1991 Mathematics Subject Classification: 20-04, 20C20, 20C34, 20C40.

χ	Degree	CC	19A
1	1	r	1
2	248	r	1
4	27 000	5	1
5	27 000	4	1
9	85 995	10	1
10	85 995	9	1
12	767 637	13	−1
13	767 637	12	−1
22	4 096 000	23	−1

χ	Degree	CC	19A
23	4 096 000	22	−1
25	4 881 384	r	−1
26	4 936 750	r	−1
29	6 696 000	30	1
30	6 696 000	29	1
35	21 326 760	36	1
36	21 326 760	35	1
43	76 271 625	r	1
44	77 376 000	r	1
48	190 373 976	r	−1

TABLE 1. Ordinary irreducible characters lying in the principal 19-block of Th. The column headed "CC" contains the entry "r" in rows corresponding to real-valued characters; otherwise it contains the number of the complex conjugate character. The last column contains the values of the characters on elements of class 19A. Characters that are connected on the Brauer tree must have unequal values on this class.

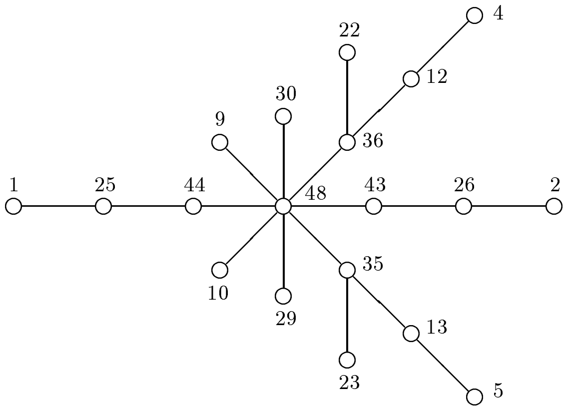

FIGURE 1. The Brauer Tree mod 19 of the sporadic simple Thompson group Th.

Char.	Degree
φ_1	1
φ_2	248
φ_4	27 000
φ_5	27 000
φ_9	85 995
φ_{10}	85 995
φ_{12}	740 637
φ_{13}	740 637
φ_{22}	4 096 000

Char.	Degree
φ_{23}	4 096 000
φ_{25}	4 881 383
φ_{26}	4 936 502
φ_{29}	6 696 000
φ_{30}	6 696 000
φ_{35}	16 490 123
φ_{36}	16 490 123
φ_{43}	71 335 123
φ_{44}	72 494 617

TABLE 2. Degrees of the irreducible Brauer characters.

corresponds to an edge of the tree connecting χ_i and χ_j, with $i < j$, is denoted by φ_i.

To obtain the result of this paper, we had to apply a new condensation technique to a module of dimension 976 841 775, the permutation module on the cosets of the third maximal subgroup of Th. The condensed module has dimension 1403 over $\mathbb{F}_{19}$ and can be analyzed with the MeatAxe [Parker 1984], giving the result. We have used the MeatAxe implementation in C by Michael Ringe [1994]. The details are given in Section 2. We remark that in order to arrive at only two possibilities for the Brauer tree in [Hiss and Lux 1989], we had to rule out several other possibilities using sophisticated techniques involving Green correspondence. We have checked the results of the condensation against these other possible trees. None of them is consistent with the condensation results.

The condensation method was originally conceived by Parker and Thackray [Thackray 1981], and is also described in [Ryba 1990] and [Lux et al. 1994]. The idea of the new DirectCondense technique goes back to Parker and Wilson, and a fuller discussion is given in [Cooperman and Tselman 1996]. In Section 3 we give details on the implementation and special techniques for doing the condensation in practice.

2. CONDENSATION

2.1. General Remarks

Let A be a finite-dimensional algebra over the field F and $e \in A$ be an idempotent. Let mod-A denote the category of finitely generated right A-modules. Then the exact functor

$$? \otimes_A Ae : \text{mod-}A \longrightarrow \text{mod-}eAe : M \longmapsto Me$$

is called the *condensation functor* with respect to e, and Me is called the *condensation* of M. Condensation gives us a tool to analyse the submodule structure of M by looking at Me instead. We will apply this for F a field of prime characteristic p, $A = FG$, where G is a finite group, K the *condensation subgroup* of G having order $|K|$ prime to p, and

$$e = e_K = |K|^{-1} \cdot \sum_{g \in K} g \in FG.$$

If M is an FG-module, then Me is the subset of M consisting of the elements left fixed by K. Let φ denote the Brauer character of M, which is extended arbitrarily to a class function $\tilde{\varphi}$ on G. Then we have

$$\dim_F(Me) = (\varphi|_K, 1_K)_K = (\tilde{\varphi}, 1_K^G)_G,$$

where $(\cdot,\cdot)_G$ denotes the scalar product for class functions. As each Brauer character can be written as a $\mathbb{Z}$-linear combination of ordinary characters restricted to the p-regular conjugacy classes, the computation of these scalar products can be carried out entirely in terms of ordinary characters. If the block under consideration is described by a Brauer tree, these linear combinations can directly be read off from the tree.

If P is an FG-permutation module on the finite set Ω, then the condensed module Pe can be described as follows. Let $\{O_i\}$ be the set of K-orbits on Ω, and $\bar{O}_i := \sum_{\omega \in O_i} \omega \in P$ be the orbit sums. Then $\{\bar{O}_i\}$ is an F-basis of Pe, and for $g \in G$ the action of $ege \in eFGe$ on Pe is given as

$$\bar{O}_i \cdot ege = \sum_j a_{ij}(g) \cdot |O_j|^{-1} \cdot \bar{O}_j,$$

where $a_{ij}(g) = \left|\{\omega \in O_i : \omega g \in O_j\}\right|$. Hence to find the action of ege, we are reduced to finding the K-orbits on Ω, their lengths, and the $a_{ij}(g)$.

2.2. The Thompson Group

Now we let $G := \mathrm{Th}$. For the necessary group theoretic information we refer the reader to [Conway et al. 1985]. We start our constructions with the irreducible representation D_{248} over $\mathbb{F}_2$ that is the 2-modular reduction of the irreducible ordinary representation of the same degree. Matrices for the action of two generators $A, B \in G$ on the module underlying D_{248} can be found in the group representation library [Wilson 1996a]. Here, A is a $2A$ element, B is a $3A$ element, and their product AB is a $19A$ element. This is a rationally rigid triple for G, see [Wilson 1996b; Pahlings 1990]. The following computations are carried out using D_{248} with the help of the MeatAxe and GAP.

We now let $P_1 := AB$, $P_2 := AB^2$, and $H := \langle H_1, H_2 \rangle$, where

$$H_1 := (P_1P_2P_1P_2^2P_1P_2)^2P_2,$$
$$H_2 := (H_1^{18}P_1^{-3}AP_1^3)^{18}.$$

Furthermore, we let $K := \langle K_1, K_2 \rangle$, where $K_1 := H_1$, $K_2 := Q_2Q_1Q_2^{-1}$, and

$$Q_1 := H_1H_2H_1H_2^2(H_1H_2)^2(H_1H_2^2)^2,$$
$$Q_2 := (H_2H_1)^2H_1^3(H_2H_1)^3.$$

We are going to choose K as our condensation subgroup. Hence we have to show that $|K|$ is not divisible by 19, and we have to find the scalar products $(\chi, 1_K^G)_G$ for all irreducible ordinary characters of G.

We find that H_1 is of order 36, and that both H_1, H_2 centralize H_1^{18}. Hence $H \leq C_G(H_1^{18}) \cong 2_+^{1+8} \cdot A_9$. We will show that H equals that centralizer. First we collect a few elements in H generating a normal subgroup $O \leq H$ of order 2^9. Then it turns out that $D_{248}|_H$ has two different constituents of dimension 8, $8a$ and $8b$ say, where $8b$ restricts irreducibly to K, while $8a$ has a fixed space of dimension 1. Now the vector fixed by K

yields an orbit of length 120 under the action of H, yielding a permutation representation P_{120} of H. This permutation group turns out to be of order $181\,440 = |A_9|$. As O is contained in the kernel of P_{120}, we conclude that $H = C_G(H_1^{18})$.

By construction, we know that K is contained in a subgroup $2_+^{1+8}.L_2(8){:}3$ of H, which is of index 120 in H. Using the action of $D_{248}|_K$ on an orbit of a suitable vector, we find that $|K| \geq 774\,144$, hence we have $K \cong 2_+^{1+8}.L_2(8){:}3$.

To compute the scalar products $(\chi, 1_K^G)_G$, we first observe that $(\chi, 1_K^G)_G = (\chi_H, 1_K^H)$ holds. The character tables of G and H are accessible in GAP, and it turns out that the fusion map of the conjugacy classes of H into those of G is uniquely determined up to table automorphisms of G. Furthermore, as the normal subgroup O of H is contained in K, it is enough to find the fusion map from $L_2(8){:}3$ to A_9, which is induced by the fusion map from K to H. Hence we choose a factor fusion map from the character table of H to that of A_9. Having fixed such a map, the admissible table automorphisms of A_9 are those that leave the chosen fusion map invariant. It now turns out that there are exactly two possible fusion maps from $L_2(8){:}3$ to A_9, and these are not conjugate under the action of the admissible table automorphisms. A look at the classes of A_9 that are the images of the classes of elements of order 36 under the factor fusion map shows, that these classes in turn are in the image of exactly one of the candidate fusion maps from $L_2(8){:}3$ to A_9. Now the scalar products $(\chi, 1_K^G)_G$ for all irreducible ordinary characters of G can be computed. They are given in Table 3.

The module we are going to condense is the FG-permutation module P on the cosets of H, where $F := \mathbb{F}_{19}$. It is found as the 19-modular reduction of the corresponding $\mathbb{Z}G$-permutation module. The latter in turn is found as the action of G on the orbit of a nontrivial vector in the module underlying D_{248} that is fixed by H. Using the MeatAxe, such a vector is found to exist and to be uniquely determined. The orbit is of length $976\,841\,775$, which is the index of H in G. From Table 3, where the scalar products $(\chi, 1_H^G)_G$ are given, we compute $\dim_F(Pe) = 1403$.

Let $A, B \in G$ denote the elements introduced above. In Section 3 we will describe the method of computation of both the action of A, B on the orbit of the seed vector and the matrices M_{eAe} and M_{eBe} giving the action of eAe and eBe on Pe. We consider the subalgebra

$$\mathcal{K} := \langle eAe, eBe \rangle \leq eFGe.$$

χ	pb	H	K	χ	pb	H	K	χ	pb	H	K	χ	pb	H	K
1	$\star$	1	1	13	$\star$	.	2	25	$\star$	2	15	37		1	43
2	$\star$	.	.	14		.	.	26	$\star$	.	8	38		1	43
3		.	.	15		.	.	27		.	6	39		2	62
4	$\star$	.	.	16		.	1	28		.	6	40		2	70
5	$\star$	.	.	17		.	.	29	$\star$	.	6	41		1	73
6		.	.	18		.	.	30	$\star$	.	6	42		1	86
7		1	3	19		1	9	31		.	15	43	$\star$	.	95
8		1	2	20		.	2	32		2	21	44	$\star$	2	111
9	$\star$	.	.	21		1	10	33		2	31	45		.	100
10	$\star$	.	.	22	$\star$	.	5	34		.	25	46		1	119
11		1	3	23	$\star$	.	5	35	$\star$	.	27	47		1	147
12	$\star$	.	2	24		1	8	36	$\star$	.	27	48	$\star$	1	236

TABLE 3. Two-permutation characters of G. A star in the second column indicates a character in the principal block of FG. The columns headed H and K give the scalar products $(\chi, 1_H^G)_G$ and $(\chi, 1_K^G)_G$, respectively.

In fact there is no reason why $\mathcal{K}$ should not be equal to $eFGe$, but we do not know whether this is the case. Using the MeatAxe, with the two matrices M_{eAe} and M_{eBe}, we find the $\mathcal{K}$-constituents of Pe. Their dimensions and their multiplicities are given in Table 4.

Dim.	Mlp.	Dim.	Mlp.	Dim.	Mlp.
$1a$	3	$10a$	1	$62a$	2
$2a$	1	$14a$	4	$70a$	2
$3a$	1	$20a$	1	$73a$	1
$3b$	1	$20b$	1	$86a$	1
$6a$	1	$21a$	2	$87a$	1
$6b$	1	$31a$	2	$97a$	3
$8a$	1	$43a$	1	$119a$	1
$9a$	1	$43b$	1	$147a$	1

TABLE 4. Results of condensation.

2.3. Proof of the Result

We have to decide between two possible Brauer trees for the principal 19-block of G. The first possibility, Case I, is the tree given in Figure 1, which will turn out to be the correct one. The second possibility, Case II, is given in Figure 2.

The degrees of the irreducible Brauer characters φ_i in the principal block and the dimensions d_i of the corresponding condensed modules for Case I are given in Table 5. They can be computed from the Brauer tree and Table 3. For example, in Case I, $\varphi_{44} = \chi_{44} - \chi_{25} + \chi_1$ on 19-regular elements of G. Hence $d_{44} = 111 - 15 + 1 = 97$. In Case II, φ_{35}, φ_{36} have degree 20 586 123 and φ_{44} has degree 64 302 617, and we have $d_{35} = d_{36} = 25$ and $d_{44} = 87$. All other degrees and dimensions remain unchanged.

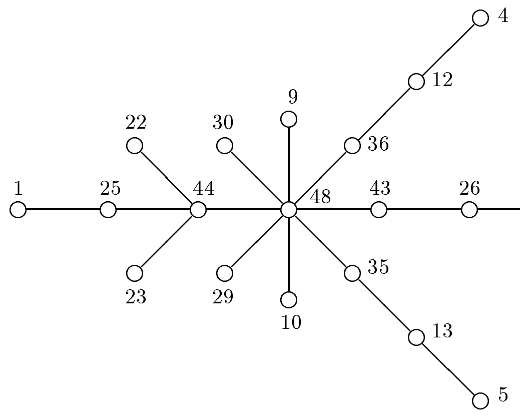

FIGURE 2. The second candidate (Case II) for the Brauer Tree mod 19 of Th.

Char.	Degree	d_i	Char.	Degree	d_i
φ_1	1	1	φ_{23}	4 096 000	5
φ_2	248	.	φ_{25}	4 881 383	14
φ_4	27 000	.	φ_{26}	4 936 502	8
φ_5	27 000	.	φ_{29}	6 696 000	6
φ_9	85 995	.	φ_{30}	6 696 000	6
φ_{10}	85 995	.	φ_{35}	16 490 123	20
φ_{12}	740 637	2	φ_{36}	16 490 123	20
φ_{13}	740 637	2	φ_{43}	71 335 123	87
φ_{22}	4 096 000	5	φ_{44}	72 494 617	97

TABLE 5. Degrees of the irreducible Brauer characters (Case I).

By Tables 3 and 5, the $eFGe$-module Pe has unique composition factors of degrees 147 and 119, respectively. These correspond to the two defect 0 characters χ_{46} and χ_{47}. All other $eFGe$-composition factors of Pe have a smaller dimension, in fact at most 97. It now follows from Table 4, that the irreducible $eFGe$-modules of dimensions 147 and 119 restrict irreducibly to $\mathcal{K}$. Finally, Table 4 shows that there is an $eFGe$-composition factor of Pe of dimension at least 97, different from the ones of dimensions 147 and 119. This implies that Case I is correct.

3. COMPUTATIONS

The key idea of the condensation method applied here is due to Parker and Wilson (private communication). It consists in using the interpretation of the abstract set Ω, where G acts on Ω as a set of vectors in a space. This gives us a compact and efficient way to compute the K-orbits and the action of group elements $g \in G$. A fuller discussion of this method is contained in [Cooperman and Tselman 1996]. The program from that description was used as the base, but performance enhancements, modifications for robustness and ease of use, and checkpointing facilities had to be added to make it work for the case of $G = \mathrm{Th}$ in its 248-dimensional representation D_{248} over $\mathbb{F}_2$, where the set Ω consists of 976 841 775 vectors and $K \cong 2_+^{1+8}.L_2(8):3$ has order 774 144. This section concentrates on these implementation issues.

3.1. The Algorithm

The vectors in $\mathbb{F}_2^{248}$ are stored with eight entries per byte. So, a vector requires 31 bytes. Since the machines having been used have 64-bit words, the vectors are in fact stored in 32 bytes each. Matrix-vector multiplication over $\mathbb{F}_2$ is implemented using logical bit operations. A lookup table of all linear combinations of each set of four adjacent rows under $\mathbb{F}_2$ is kept for each matrix. Such a lookup table was first used by Arlazarov et al. [Aho et al. 1975; Arlazarov et al. 1970, p. 245], and was popularized by Parker [1984] in his software for the MeatAxe.

The natural algorithm would have been to maintain a hash table for all the 976 841 775 vectors in Ω. Each vector would be stored in the table, along with an index indicating which K-orbit it comes from. At 32 bytes per vector, plus hash table overhead, this would imply the use of more than 32 gigabytes, which would be far from possible at our site. The solution is to store only $1/m$ of the vectors, where $m = 64$ in our case, thus allowing the entire computation to proceed within the available virtual memory. This is accomplished by applying a second hash-like function, and by saving in the hash table only those vectors for which the function returns zero modulo m. There are a few small orbits that do not have such vectors. In the case of such orbits, all of the vectors are stored in the hash table. Since K has order 774 144, it is always feasible to place all vectors of a single orbit into a temporary hash table of size less than 32 megabytes.

This decision to store only some of the vectors affects another part of the natural algorithm. At several stages in the algorithm, one must determine if a new vector, ωg, has previously been encountered, and if it has been encountered, what is the index of the orbit to which it belongs. The new solution requires one to do a local search of vectors in the orbit $\omega g K$ until one either finds a vector whose second hash value is 0 modulo m, or else until all vectors of the orbit $\omega g K$ have been found. At this point, one has found a vector ω' in the same orbit as ωg. Either ω' is contained in the main hash table, and one can look up the index of the orbit, or one adds ω' to a queue of representatives of new orbits to be explored. Note that one is able to determine a vector that should be contained in the hash table even without having access to the hash table. This is an important point in Section 3.2, where the distributed version of the algorithm is described.

The other complication of the implementation was the decision whether to first store the vectors in the hash table, and then compute the matrix entries a_{ij} (see Section 2.1) or else to compute both at once. For efficiency reasons, it was decided to compute the two at the same time. Thus, when an image vector ωg is computed, one next determines whether a vector from the corresponding orbit is in the hash table. If this is the case, one increments the appropriate a_{ij} entry. But if it was not part of a known orbit, when does one increment an appropriate a_{ij}? Further, one must be careful that the number of images ωg in the queue representing new orbits does not grow too large. This is especially worrisome at the beginning, when many points of new orbits are found, and for an unknown orbit there may be a large number of member vectors waiting in the queue. Ensuring proper accounting and memory restrictions is what leads to complications. The same code is then used to condense the second and further matrices, maintaining the orbit data structures found during the condensation of the first matrix.

3.2. Parallelization

Parallelization was carried out through a master-slave architecture. The code was much simplified by the use of the STAR/MPI software [Cooperman 1995], a transaction-oriented parallel LISP. The computation used GCL Common LISP. This parallel tool is based on the MPICH implementation of MPI. At Northeastern University, a small "homegrown" subset of MPI has been implemented by one of us (G. C.) and R. Kyzas as a teaching tool, which replaced MPICH in part of the computations. This software helped to handle such concurrency issues as when two distinct slaves were exploring the same orbit with two distinct orbit representatives from the queue of new orbits.

3.3. Timing

The computations were done using eight 175 MHz Alpha 3000/300 workstations at Northeastern University. While the seven slave workstations had 64 megabytes and 300 megabytes of virtual memory, the master had 192 megabytes and 1 gigabyte of virtual memory, timing tests indicate that a matrix-vector multiplication costs about 23 μs. The computation took about one month on eight workstations. The calculation was roughly divided into 10 days to determine the orbits, 10 days for the condensation of the first generator with respect to those orbits, and 10 days for the second generator. The first two parts of the computation were mixed together, as described in Section 3.1, although the majority of the first 10 days was still spent in building new orbits and the majority of the second 10 days in computing the condensation of the first generator.

There are two constraints on the speed of the computation: the number of slaves, or equivalently, the CPU speed of the slaves, and the size of semiconductor memory on the master. Determination of the orbits during the first 10 days was primarily constrained by the available CPU power. This is because the number of matrix-vector multiplications in the computation of the points of an orbit O_i is roughly proportional to $|O_i|k$, where k is the size of the generating set for K. Yet, $|O_i|/m$ points are needed to be stored on the master. Thus, the slave carries out km matrix-vector multiplications for each hash access on the master.

The second phase is constrained both by CPU speed and by available memory on the master. The condensation of a generator with respect to those orbits requires approximately $|O_i|m$ matrix-vector multiplications on average, while $|O_i|$ points need to be accessed on the master from the hash array. Hence the slave carries out m matrix-vector multiplications for each hash access on the master. In addition, in the second phase, the hash table is almost full and one must also update the matrix associated with the condensation. Thus, there are still more demands on memory.

The fastest overall performance was determined by balancing these competing demands of memory and CPU time. It was found empirically that it was advantageous to choose m such that the hash array was larger than RAM, and to accept a certain paging penalty. The memory demands for the hash array were proportional, of course, to $|\Omega|/m$. In our example, m was chosen as 64, and the hash array occupied 610 Megabytes, including hash overhead.

REFERENCES

[Aho et al. 1975] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, MA, 1975.

[Arlazarov et al. 1970] V. L. Arlazarov, E. A. Dinic, M. A. Kronrod, and I. A. Faradžev, "The economical construction of the transitive closure of an oriented graph", *Dokl. Akad. Nauk SSSR* **194** (1970), 487–488. In Russian. Translation in *Soviet Math. Dokl.* **11**:5 (1970), 1209–1210.

[Conway et al. 1985] J. H. Conway, R. T. Curtis, S. P. Norton, R. A. Parker, and R. A. Wilson, *Atlas of finite groups*, Oxford University Press, Oxford, 1985.

[Cooperman 1995] G. Cooperman, "STAR/MPI: Binding a parallel library to interactive symbolic algebra systems", pp. 126–132 in *ISSAC 95: Proceedings of the International Symposium on Symbolic and*

Algebraic Computation (Montreal, 1995), edited by A. H. M. Levelt, ACM Press, New York, 1995.

[Cooperman and Tselman 1996] G. Cooperman and M. Tselman, "New sequential and parallel algorithms for generating high dimension Hecke algebras using the condensation technique", pp. 155–160 in *ISSAC 96: Proceedings of the International Symposium on Symbolic and Algebraic Computation* (Zürich, 1996), edited by Y. N. Lakshman, ACM Press, New York, 1996.

[Hiss and Lux 1989] G. Hiss and K. Lux, *Brauer trees of sporadic groups*, Oxford Science Publications, Oxford Univ. Press, New York, 1989.

[Lux et al. 1994] K. Lux, J. Müller, and M. Ringe, "Peakword condensation and submodule lattices: an application of the MEAT-AXE", *J. Symbolic Comput.* **17**:6 (1994), 529–544.

[Pahlings 1990] H. Pahlings, "Realizing finite groups as Galois groups", pp. 137–152 in *Darstellungstheorietage* (Sion, 1989), Bayreuth. Math. Schr. **33**, Universität Bayreuth, 1990.

[Parker 1984] R. A. Parker, "The computer calculation of modular characters (the Meat-Axe)", pp. 267–274 in *Computational group theory* (Durham, 1982), edited by M. D. Atkinson, Academic Press, London, 1984.

[Ringe 1994] M. Ringe, "The C-MeatAxe", software manual, Lehrstuhl D für Mathematik, RWTH Aachen, Germany, 1994. See http://www-gap.dcs.st-and.ac.uk/~gap/Share/meataxe.html.

[Ryba 1990] A. J. E. Ryba, "Computer condensation of modular representations", *J. Symbolic Comput.* **9**:5-6 (1990), 591–600.

[Schönert et al. 1994] M. Schönert et al., *GAP: Groups, algorithms, and programming*, Lehrstuhl D für Mathematik, RWTH Aachen, 1994. See ftp://dimacs.rutgers.edu/pub/ or http://www-gap.dcs.st-and.ac.uk/~gap.

[Thackray 1981] J. G. Thackray, *Modular representations of finite groups*, Ph.D. thesis, Cambridge University, 1981.

[Wilson 1996a] R. A. Wilson, "Atlas of finite group representations", online database, University of Birmingham, UK, 1996. See http://www.mat.bham.ac.uk/atlas/.

[Wilson 1996b] R. A. Wilson, "Standard generators for sporadic simple groups", *J. Algebra* **184**:2 (1996), 505–515.

Gene Cooperman, College of Computer Science, Northeastern University, Boston, MA 02115, USA (gene@ccs.neu.edu)

Gerhard Hiss, IWR der Universität Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany (hiss@euterpe.iwr.uni-heidelberg.de)

Klaus Lux, L.: Lehrstuhl D für Mathematik, RWTH Aachen, Templergraben 64, 52062 Aachen, Germany (klux@math.rwth-aachen.de)

Jürgen Müller, IWR der Universität Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany (jmueller@euterpe.iwr.uni-heidelberg.de)

Received February 20, 1997; accepted April 28, 1997