CHEVIE—A SYSTEM FOR COMPUTING AND PROCESSING GENERIC CHARACTER TABLES

MEINOLF GECK, GERHARD HISS, FRANK LÜBECK, GUNTER MALLE, AND GÖTZ PFEIFFER

1. Introduction

CHEVIE is a package based on the computer algebra systems GAP [43] and Maple [10]. It consists of a library of data and programs for dealing with generic character tables of finite groups of Lie type and associated structures such as finite Weyl groups and Hecke algebras.

Preliminary versions of this system have already proved to be useful in various applications in constructive Galois theory, investigations on the subgroup structure and the modular character theory of finite groups of Lie type. One of the motivations for us was to provide an appropriate platform and environment for collecting known results, for working out new results in a systematic manner, and for performing experiments on substantial examples. Moreover, we found it most convenient to have electronic access to large tables previously available only in printed form. Unfortunately, working with such printed tables often is quite cumbersome due to occasional inaccuracies which are difficult to spot. (Note that, e. g., a printed version of the generic character table of $CSp_6(q)$, q odd, which was computed in [29] and is now part of CHEVIE, would require several hundred pages.) Thus, we also hope to provide a means for putting into print results of this type, with the risk of misprints minimized.

One of our main inspirations was the idea underlying the famous Cambridge Atlas of finite groups [11] which has become an important tool for investigating finite groups and their representations. While the Atlas contains the character tables of individual groups (like $GL_2(5)$) our objective is to give one generic character table for all groups in a series (like $GL_2(q)$, q any prime power). First examples of such generic tables were already computed by I. Schur (for $SL_2(q)$, see [44]) at the beginning of the century. J. A. Green [23] completely described the irreducible characters of the finite general linear groups. Explicit tables for other groups of small rank were determined by B. Srinivasan (for $Sp_4(q)$, see [47]) and B. Chang, R. Ree (for $G_2(q)$, see [9]). Using the description of finite groups of Lie type as fixed point sets of connected reductive groups under a Frobenius map, P. Deligne and G. Lusztig in [13] introduced new methods from algebraic geometry and ℓ-adic (intersection) cohomology in the study of irreducible representations of these groups. This enabled Lusztig finally to give a classification of the irreducible characters (see [31]) and a general procedure for constructing the irreducible characters in an algorithmic fashion—at least in principle, and up to scalar multiples (see [33] and the references there). Nevertheless, the explicit construction of a particular generic table still requires a huge amount of work and detailed knowledge.

The authors Frank Lübeck and Götz Pfeiffer acknowledge financial support by the Deutsche Forschungsgemeinschaft.

Finite Weyl groups, associated Hecke algebras and their representations play an important role in these theories. Certain specializations of Hecke algebras arise naturally as endomorphism rings of representations obtained by Harish-Chandra induction from Levi subgroups. However, results on representations of Hecke algebras also have an independent interest as well as applications in other areas of mathematics (e.g., in V. F. R. Jones´ construction of a two-variable polynomial invariant of knots and links, see [27]). Much of the first work on Hecke algebras was centered around the determination of the so-called generic degrees (see [2]). D. Kazhdan and G. Lusztig introduced the new concepts of left cells and their geometric interpretation in terms of intersection cohomology of Schubert varieties. Recently, general concepts in the representation theory of Hecke algebras have been developed, and explicit results on character tables and decomposition numbers were computed (see [22] and [18]).

We shall now give a brief sketch of the content and structure of CHEVIE, Version 2.1.

- Generic ordinary character tables for all series of finite groups of Lie type of small rank.
- Tables of Green polynomials (for exceptional and classical groups of low rank, and disconnected groups), and a program for calculating the Green polynomials for groups of type A.
- Programs for working with these tables (scalar products, tensor products, structure constants), constructing new tables and viewing respectively printing them in $\TeX$ format.
- Character tables of Hecke algebras for all types of rank at most 7, and recursive algorithms for computing them for types A_n and B_n. Tables and programs for the Weyl groups themselves are already available through GAP.
- Programs for computing Kazhdan–Lusztig polynomials, left cells and the corresponding representations for Weyl groups (effective for rank ≤ 4).

The next release will also contain the character table of the Iwahori-Hecke algebra of type E_8 (recently computed in [21]) and character tables for cyclotomic algebras associated to complex reflection groups, as well as programs for dealing with these. It should also be noted that some of the functions and features described in this article will only be available in that next release.

Our idea of using computer algebra systems such as Maple for performing symbolic calculations with the characters of a whole series of groups of Lie type dates back in 1986. We determined decomposition numbers for the groups $G_2(q)$ [25] and $SU_3(q)$ [17] with the assistance of Maple. In another direction, problems of constructive Galois theory motivated the computation of the unipotent characters of the Ree groups ${}^2F_4(q^2)$ [35] and also Green functions for disconnected groups [37]. These calculations involved the tedious work of typing existing tables into the computer, checking and correcting them. The amount of data became so large that we were forced to think about a systematic approach. At about 1990, a preliminary version of GAP was available. We used it as a platform for programs dealing with finite Weyl groups and Hecke algebras. (These are also available through GAP from Version 3, Release 1, on.) We then decided to combine the various tables, GAP and Maple procedures into a single system with well defined data structures: Version 1 of CHEVIE (unpublished).

CHEVIE is currently used by a PhD-student for the construction of the unipotent characters of $F_4(q)$. One of us (FL) is extending CHEVIE by programs which allow the automatic calculation of the head of a generic character table, and also of the Deligne–Lusztig characters $R_{T,1}$. One of the applications of this will be the construction of the unipotent characters of $E_6(q)$.

CHEVIE can be obtained via anonymous `ftp` through one of the `ftp`-servers

```
ftp.math.rwth-aachen.de (Internet number: 137.226.152.6)
ftp.iwr.uni-heidelberg.de (Internet number: 129.206.104.40)
```

`ftp` to one of the servers mentioned above, login as user `ftp` and give your full e-mail address as password. CHEVIE is in the directory `pub/chevie`. You will find the three files `README`, `chevie2r1V2_tar.Z` and `chevie2r1V3_tar.Z`. The `README` file contains detailed information on the installation of CHEVIE. For the use with Maple V Release 2 (or earlier) you need `chevie2r1V2_tar.Z`, and analogously for Maple V Release 3. In uncompressed form, CHEVIE requires 6 MB of disc space.

We close by inviting the users of CHEVIE to communicate their suggestions, criticisms and own contributions to the authors, as well as to inform us about results obtained using CHEVIE.

2. Finite Weyl groups

It is the purpose of this section to describe the setup for a computational approach to finite Weyl groups, their root systems, and the action of the groups on their roots. A good reference for the theoretical background is [26], Chapter 1. Let us briefly recall the main definitions.

Let V be a finite dimensional real vector space equipped with a positive definite symmetric bilinear form $(\ ,\ )$. For each non-zero vector $\alpha \in V$ the reflection s_α with root α is the isometry of V given by

$$s_\alpha(v) = v - 2\frac{(\alpha, v)}{(\alpha,\alpha)}\alpha \qquad \text{for all } v \in V.$$

A finite subset $\Phi \subseteq V$ is called a *root system* (in the sense of [26], 1.2) if the following conditions hold.

(R1) $0 \notin \Phi$ and Φ contains a basis for V.
(R2) If $\alpha, \beta \in \Phi$ then $s_\alpha(\beta) \in \Phi$.
(R3) If $\alpha \in \Phi$ and $\lambda \in \mathbb{R}$ such that $\lambda\alpha \in \Phi$ then $\lambda = \pm 1$.

We say that Φ is *crystallographic* if, moreover, the following condition holds.

(R4) $2(\alpha, \beta)/(\alpha, \alpha) \in \mathbb{Z}$ for all $\alpha, \beta \in \Phi$.

The dimension of V will also be called the *rank* of Φ.

The root system Φ will be called *reducible* if it can be decomposed into two proper subsets Φ_1, Φ_2 such that every root in Φ_1 is orthogonal to every root in Φ_2. In this case, both Φ_1 and Φ_2 form a root system in the respective subspaces of V which they span. Otherwise, Φ is called *irreducible.*

The *Weyl group* corresponding to the root system Φ is defined to be the subgroup $W = W(\Phi)$ of the orthogonal group of $(V, (\ ,\ ))$ generated by the reflections s_α, $\alpha \in \Phi$. (Note that we do not exclude the non-crystallographic types.) Condition (R2) implies that every $w \in W$ induces a permutation of the elements in Φ, and (R1) implies that the corresponding permutation representation of W on Φ is faithful. These groups, together with their actions on the root systems, are our basic object of interest.

2.1. **The Weyl record.** We are now looking for a suitable way to represent the above data on a computer, within the computer algebra system GAP [43]. The principal idea is to take a basis of V, and to express every root and every element of W by their coordinate vectors respectively their matrices relative to this basis. By (R1), we may assume that our basis consists of roots. Let us denote such a basis by $\Delta = \{\alpha_i \mid i \in I\} \subseteq \Phi$, where I is some finite index set with $|I| = \dim V$. It is a general fact that we can choose Δ such that every element $\alpha \in \Phi$ can be written in the form $\alpha = \sum_{i \in I} \lambda_i \alpha_i$ where either each $\lambda_i \geq 0$ or each $\lambda_i \leq 0$. In the first case, α will be called a *positive root*, otherwise α is a *negative root*. Thus, we have a partition of Φ into the subsets Φ^+ and Φ^- consisting of positive respectively negative roots. The elements in Δ will be called *simple roots*.

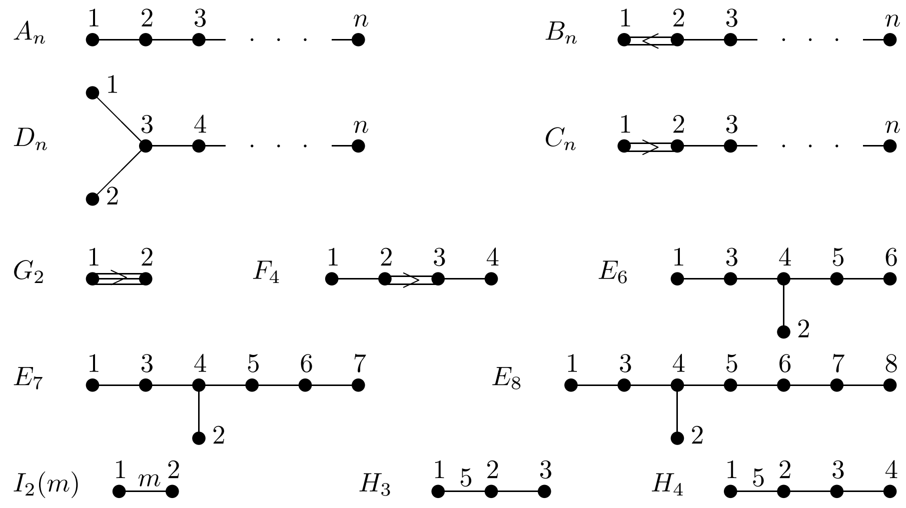
The reflections corresponding to the simple roots will be called *simple reflections*. For simplicity, we shall write $s_i := s_{\alpha_i}$ for $i \in I$. It is now important to note that given any element $\alpha \in \Phi$ there exist finitely many simple reflections $s_{i_1}, \ldots, s_{i_m}$ such that $s_{i_1} \ldots s_{i_m}(\alpha) \in \Delta$. This also implies that W is already generated by the set $S = \{s_i \mid i \in I\}$ of simple reflections.

In summary, both Φ and W can be reconstructed from Δ and the action of the elements of S on Δ. In order to describe this action, it is sufficient to know the following matrix

$$C := \left(2\frac{(\alpha_i, \alpha_j)}{(\alpha_i, \alpha_i)} \right)_{i,j \in I},$$

which we call the *Cartan matrix* of Φ. (It is unique up to simultaneous permutation of rows and columns.) Conversely, assume we are given a $d \times d$-matrix C which is the Cartan matrix of some root system Φ, with corresponding Weyl group W and a choice of simple roots Δ. Then we can recover the coordinate vectors of the roots in Φ and the matrices of the elements of W, such that the action of W on Φ is simply given by multiplying a matrix by a vector. The coordinate vectors corresponding to the simple roots are the standard d-tuples with $d-1$ zeros and one entry equal to 1.

Now let $I = \{1, \ldots, d\}$ and $C = (c_{ij})_{i,j \in I}$ be the Cartan matrix of an irreducible root system. This matrix can be conveniently encoded by the associated *Dynkin diagram* which is a graph with nodes labelled by the elements in I (which correspond to the elements in Δ) and edges given as follows: Let $i, j \in I$, $i \neq j$. If c_{ij} and c_{ji} are integers such that $|c_{ij}| \geq |c_{ji}|$ the vertices are connected by $|c_{ij}|$ lines, and if $|c_{ij}| > 1$ then we put an additional arrow on the lines pointing towards the node with label i. In all other cases, we simply put a single line equipped with the unique integer $p_{ij} \geq 1$ such that $c_{ij}c_{ji} = \cos^2(\pi/p_{ij})$. We now a give the complete list of these Dynkin diagrams together with a numbering of the nodes.



In GAP the Cartan matrix corresponding to one of the above irreducible root systems (with the specified labelling) is returned by the command `CartanMat` which takes as input a string giving the type (e.g., `"A"`, `"B"`, ..., `"I"`) and a positive integer giving the rank. For type $I_2(m)$, we give as a third argument the integer m. Given two Cartan matrices, their matrix direct sum (corresponding to the orthogonal direct sum of the root systems) is produced by using the command `DirectSumCartanMat`. These two functions return a matrix (i.e., a list of lists in GAP) with entries in $\mathbb{Z}$ or in a cyclotomic extension of the rationals.

The function `Weyl` takes the Cartan matrix of any finite root system (irreducible or not, and with any ordering of the simple roots) as input and produces a GAP record with components containing basic information about the corresponding root system Φ, the Weyl group W, and the action of W on Φ. These components contain, for example, the Cartan matrix itself, a name for the whole record, the rank, the number of positive roots, the permutations corresponding to all the roots, the degrees of the basic polynomial invariants, an operations record (as it should be in GAP), and several others. We describe only three of them in more detail.

`type`: This component contains a list describing the types of the various irreducible root systems involved in Φ. The function `PrintDynkinDiagram` applied to this component prints the corresponding Dynkin diagrams with the appropriate labelling of the nodes. If the original Cartan matrix (which may be irreducible or not) was not obtained through the function `CartanMat` then this is a convenient way to find out the actual type of the root system and the Weyl group under consideration.

`roots`: This component contains a list with the coordinate vectors of all the roots relative to the basis of simple roots. Assume there are d simple roots and N positive roots. Then `roots` is a list of total length $2N$. The ordering is such that the first d entries are the standard d-tuples $(1, 0, \ldots, 0), \ldots, (0, \ldots, 0, 1)$ (corresponding to the simple roots), followed by the remaining $N - d$ positive roots. (The N-th entry is the "highest" root of Φ.) The last N entries are the negative roots in Φ ordered correspondingly to the positive roots. All other functions dealing with roots refer to the ordering of the roots in this record component.

`permgens`: This component contains a list of permutations describing the action of the simple reflections in S on the roots Φ. The i-th entry in this list is a permutation of the set $\{1, \dots, 2N\}$ which gives exactly the effect of s_i on the roots Φ. Here, of course, we have to refer explicitly to the ordering of the roots determined before. Note also that, at this stage, we do not need to specify whether we act from the left or the right since we are dealing only with involutions.

All this information is not stored individually for all the types but is actually computed along the lines indicated above when the function `Weyl` is called.

The following example is the output taken from a session in GAP. We reproduce it here to give an impression of how things look like on the computer. For a detailed description of the program we refer to the CHEVIE manual [20].

```
gap> W:= Weyl(CartanMat("G", 2));
Weyl( [ [ 2, -1 ], [ -3, 2 ] ])
gap> W.type;
[ [ "G", [ 1, 2 ] ] ]
gap> PrintDynkinDiagram(last);
G2  1 > 2
gap> W.roots;
[ [1, 0], [0, 1], [1, 1], [1, 2], [1, 3], [2, 3],
  [-1, 0], [0, -1], [-1, -1], [-1, -2], [-1, -3], [-2, -3] ]
gap> W.permgens;
[ ( 1, 7)( 2, 3)( 5, 6)( 8, 9)(11,12),
  ( 1, 5)( 2, 8)( 3, 4)( 7,11)( 9,10) ]
```

2.2. **Working with elements in Weyl groups.** We now describe how actual computations with the elements of a finite Weyl group can be performed, along the approach set up above.

A basic tool in dealing with W is the length function. Given $w \in W$ it is possible to write $w = s_{i_1} \dots s_{i_m}$ where $i_1, \dots, i_m \in I$. If m is as small as possible we call the expression of w as a product of simple reflections a *reduced expression*, and the integer m the *length* of w, written $l(w)$. Clearly, $l(1) = 0$ and $l(s_i) = 1$ for all $i \in I$.

A user might be interested to think of the elements of W as such words in the simple reflections. In GAP, a word is simply represented as a list of integers corresponding to the simply roots, e.g., `[]` is the identity element, and `[1]`, `[2]`, etc. represent the reflection along the first, the second etc. simple root. For other purposes, it might be better to see the permutation of an element w on the root vectors. Since this is forced upon us by GAP we will assume throughout that W acts from the right on Φ, and write this action as α^w, for $\alpha \in \Phi$ and $w \in W$. The functions `WeylWordPerm` and `PermWeylWord` will do the conversion of one form into the other.

```
gap> W:= Weyl(CartanMat("D", 4));;
gap> PermWeylWord(W, [1, 3, 2, 1, 3]);
( 1,14,13, 2)( 3,17, 8,18)( 4,12)( 5,20, 6,15)( 7,10,11, 9)
(16,24)(19,22,23,21)
gap> WeylWordPerm(W, last);
[ 1, 3, 1, 2, 3 ]
```

We notice that the word we started with and the one that we ended up with, are not the same. But of course, they represent the same element of W. The reason for this difference is that the function `WeylWordPerm` always computes a reduced word which is the lexicographically smallest among all possible expressions of an element of W as a word in the simple reflections! The function `ReducedWeylWord` does the same but with an arbitrary word as input (and not with a permutation). In particular, the element used in the above example has length 5. Sometimes, it is not necessary to compute a reduced word for an element w and one is only interested in the length $l(w)$. The crucial point of working with the permutation representation of W on Φ is that the length of an element w can be computed very fast and effectively by looking at its effect on positive and negative roots. Since this will be of basic importance to us, we summarize the results as follows. (For a proof, see [26], 1.6).

(a) *The length of $w \in W$ is the number of $\alpha \in \Phi^+$ such that $\alpha^w \in \Phi^-$.*
(b) *For every simple reflection s_i we have $l(s_i w) = l(w) \pm 1$, and $l(s_i w) = l(w) + 1$ holds if and only if $\alpha_i^w \in \Phi^+$.*

Thus, if there are N positive roots and W is represented as a permutation group on $\{1, \dots, 2N\}$ such that the first N numbers correspond to the positive roots (as described in the previous section) then the length of an element $w \in W$ can be computed by counting the number of $i \in \{1, \dots, N\}$ such that $i^w > N$. This is what the function `WeylLengthPerm` does.

```
gap> LongestWeylWord(W);
[ 1, 2, 3, 1, 2, 3, 4, 3, 1, 2, 3, 4 ]
gap> PermWeylWord(W, last);
( 1,13)( 2,14)( 3,15)( 4,16)( 5,17)( 6,18)( 7,19)( 8,20)
( 9,21)(10,22)(11,23)(12,24)
gap> WeylLengthPerm(W, last);
12
```

These are the most basic functions available. All the other commands, described extensively in the CHEVIE manual, do build on these.

Another advantage of working with the permutation representation of W on Φ is that all the algorithms in GAP designed for general permutation groups can be applied, e.g., strong generating set, membership test, centralizer of elements, conjugacy test etc.

2.3. **Weyl subgroups.** We consider a subset $\tilde{\Phi} \subseteq \Phi$, and let $V' \subseteq V$ be the subspace of V spanned by $\tilde{\Phi}$. Then the subgroup $W' \subseteq W$ generated by the reflections corresponding to the roots in $\tilde{\Phi}$ also is a finite Weyl group. Its root system $\Phi' \subseteq V'$ is the set of all α^w for $\alpha \in \tilde{\Phi}$, $w \in W'$. A set of simple roots $\Delta' \subseteq \Phi'$, and hence a corresponding set of simple reflections $S' \subseteq W'$, can be computed algorithmically as follows (see [26], 8.2). For any $w \in W$ let $N(w) := \{\alpha \in \Phi^+ \mid \alpha^w \in \Phi^-\}$. Clearly, if $\alpha \in \Phi^+$ then $\alpha \in N(s_\alpha)$. In particular, we have that $\alpha \in N(s_\alpha) \cap \Phi'$ for all positive roots $\alpha \in \Phi'$. Then

$$\Delta' := \{\alpha \in \Phi' \mid N(s_\alpha) \cap \Phi' = \{\alpha\}\}$$

is a set of simple roots for Φ'. Note that our preparations in the previous section ensure that the set Φ' can simply be determined by a standard orbit algorithm, and the set $N(w)$ can indeed be computed efficiently using 2.2(a).

A record for such a Weyl subgroup is returned by the function `WeylSubgroup` which takes the original record for W and a list of indices for the roots as input. The record for the subgroup contains additional components the most important of which is `rootIndices` which gives the fusion of the root system Φ' into the original root system Φ.

```
gap> W:= Weyl(CartanMat("F", 4));
Weyl( [ [ 2, -1, 0, 0 ], [ -1, 2, -1, 0 ],
        [ 0, -2, 2, -1 ], [ 0, 0, -1, 2 ] ] )
gap> PrintDynkinDiagram(last.type);
F4  1 - 2 > 3 - 4
gap> W.roots[16];
[ 0, 1, 2, 2 ]
gap> h:= WeylSubgroup(W, [1, 2, 3, 16]);
Weyl( [ [ 2, -1, -1, 0 ], [ -1, 2, 0, -1 ],
        [ -1, 0, 2, 0 ], [ 0, -2, 0, 2 ] ] )
gap> PrintDynkinDiagram(last.type);
B4    4 < 2 - 1 - 3
gap> h.rootIndices{[1..4]};
[ 1, 2, 16, 3 ]
```

The last line tells us that the four simple roots in Δ' are, respectively, the 1., 2., 16., 3. root in Φ.

If $\tilde{\Phi} = \{\alpha_j \mid j \in J\}$ for some subset $J \subseteq I$ then W' is called a *parabolic subgroup* of W. In this case, we can take $\Delta' = \tilde{\Phi}$. Moreover, there is a distinguished set of (right) coset representatives of W' in W, defined as follows. Let

$$D := \{w \in W \mid l(s_j w) > l(w) \text{ for all } j \in J\}.$$

Then every element $w \in W$ can be written uniquely in the form $w = w'd$ where $w' \in W'$, $d \in D$ and $l(w) = l(w') + l(d)$. The elements in D are the unique elements of minimal length in the various right cosets of W' in W. (See [26], 1.10.) The set D can be computed recursively as follows. Let D_k be the set of elements in D of length $k \geq 0$. We start with $D_0 := \{1\}$. Now assume that D_k has already been computed, for some $k \geq 0$. Then D_{k+1} is the set of all products ws_i where $w \in D_k$, $i \in I$ and $l(s_j w s_i) > l(ws_i)$ for all $j \in J$. (Note that the condition on the length can be checked very quickly using 2.2(b).) The set D can be computed in GAP using the function `WeylRightCosetRepresentatives`.

2.4. **Character tables and induce/restrict matrices.** The ordinary complex character table of any finite Weyl group is computed in CHEVIE using the function `CharTable`. This command first checks whether or not W is irreducible. For each irreducible Weyl group a character table record is computed either using recursive formulas (for the classical types) or read into the system from a library file (for the exceptional types). Such a record is a usual character table record as defined in GAP, but with some additional components.

It is important to note that the conjugacy classes and the irreducible characters of a finite Weyl group each have a canonical labelling by certain combinatorial objects, and that such labellings are contained in a consistent way in the tables of CHEVIE. For the classes, these are partitions, pairs of partitions, or Carter's admissible diagrams [6]. For the characters, these are again partitions, pairs of partitions, or pairs of two integers (n, e) where n is the degree of the character and e is the smallest symmetric power of the natural reflection representation containing the given character as a constituent. This information is contained in the record components `classnames` and `irredinfo`. It is printed automatically when you display the character table in GAP. In the following example, we are dealing with the Weyl group of type A_3 where the classes and characters have a canonical labelling by the partitions of 4.

```
gap> W:= Weyl(CartanMat("A", 3));;
gap> ct:= CharTable(W);;
gap> DisplayCharTable(ct);
H(A3)
      2    3    2    3    .  2
      3    1    .    .    1  .
         1111  211   22   31  4
     2P 1111 1111 1111   31 22
     3P 1111  211   22 1111  4

1111        1   -1    1    1 -1
211         3   -1   -1    .  1
22          2    .    2   -1  .
31          3    1   -1    . -1
4           1    1    1    1  1
```

Moreover, the record component `classtext` actually contains representatives of the conjugacy classes given as words in the simple reflections. In the above example, this reads as follows.

```
gap> ct.classtext;
[ [  ], [ 1 ], [ 1, 3 ], [ 1, 2 ], [ 1, 2, 3 ] ]
```

In many applications it is useful to know the decomposition of the irreducible characters of W when we restrict them from W to a Weyl subgroup W' (defined as in Section 2.3). In order to apply the usual GAP functions for inducing and restricting characters and computing scalar products, we need to know the fusion map for the conjugacy classes of W' into those of W. This is done with the function `FusionConjugacyClasses`. The decomposition of induced characters into irreducibles then is a simple matter of combining some functions which already exist in GAP. There is an additional function, `InductionTable`, which performs this job. To illustrate this we give the following example in which we consider the Weyl group of type G_2 and a subgroup of type $A_1 \times A_1$.

```
gap> W:= Weyl(CartanMat("G", 2));
Weyl( [ [ 2, -1 ], [ -3, 2 ] ] )
gap> WeylSubgroup(W, [1, 4]);
Weyl( [ [ 2, 0 ], [ 0, 2 ] ] )
gap> PrintArray(InductionTable(last, W));
[ [ 0,  0,  0,  1 ],
  [ 1,  0,  0,  0 ],
  [ 0,  0,  1,  0 ],
  [ 0,  1,  0,  0 ],
  [ 0,  1,  1,  0 ],
  [ 1,  0,  0,  1 ] ]
```

The rows of this matrix correspond to the irreducible characters of W and the columns to those of W', as given by the command `CharTable`.

We shall have more to say about character table records in the section about Iwahori-Hecke algebras.

3. Iwahori-Hecke algebras

In this section we describe the basic functions dealing with the Iwahori-Hecke algebra associated with a finite Weyl group W. As before, we denote by $S = \{s_i \mid i \in I\}$ a set of simple reflections in W. We shall now make essential use of the fact that there is a nice way of writing down a presentation of W. For $s, t \in S$ let m_{st} be the order of the element $st \in W$. Then, clearly, the following relations hold:

$$s^2 = 1 \quad \text{and} \quad (st)^{m_{st}} = 1 \text{ for all } s, t \in S.$$

These actually form a set of defining relations for W. We can deform these relations so as to get defining relations for the Iwahori-Hecke algebra associated with W. This is done as follows.

Let R be any commutative ring with a fixed choice of elements $q_s \in R$ for $s \in S$ such that $q_s = q_t$ whenever $s, t \in S$ are conjugate in W. We then have a corresponding *Iwahori-Hecke algebra* $H = H_R(\mathbf{q})$ over R with parameters $\mathbf{q} = (q_s \mid s \in S)$. It is defined as the associative R-algebra with $1 = T_1$ generated by elements T_s, $s \in S$, subject to the following relations.

$$\begin{array}{rcll} T_s^2 & = & q_s T_1 + (q_s - 1) T_s & \text{for } s \in S \\ T_s T_t T_s \cdots & = & T_t T_s T_t \cdots & \text{for } s, t \in S, \text{ and } m_{st} \text{ factors on each side.} \end{array}$$

It is known that, if $w = s_1 \cdots s_m = s_1' \cdots s_m'$ are two reduced expressions of $w \in W$ as products of simple reflections $s_i, s_i' \in S$ then the corresponding products of the generators T_s will give the same element of H, which we may therefore denote by T_w. Then the elements $\{T_w \mid w \in W\}$ actually form a free R-basis of H.

There is a universal choice for R and $\mathbf{q}$: Let u_s, $s \in S$, be indeterminates over $\mathbb{Q}$ such that $u_s = u_t$ whenever $s, t \in S$ are conjugate in W, and let $A_0 = \mathbb{Z}[u_s \mid s \in S]$ be the corresponding polynomial ring. Then $H_0 := H_{A_0}(\mathbf{u})$ is called the *generic Iwahori-Hecke algebra* associated with W. If R and $\mathbf{q}$ are given as above then $H_R(\mathbf{q})$ can be obtained by specialization from H_0: There is a unique ring homomorphism $f : A_0 \to R$ such that $f(u_s) = q_s$ for all $s \in S$. Then we can view R as an A_0-module via f and we can identify $H_R(\mathbf{q}) = R \otimes_{A_0} H_0$.

If we choose $R = \mathbb{Q}$ and $q_s = 1$ for all $s \in S$, then the above construction yields nothing but the group algebra of W over $\mathbb{Q}$. In this sense, the algebra H_0 is the generic deformation of the group algebra of W.

3.1. **Algebra elements and representations.**

Assume we are given a Weyl record of W as described in Section 2. Assume, moreover, that we have defined in GAP elements $\{q_s \mid s \in S\}$ in some commutative ring R. Then the command `Hecke`, applied to that record and that list of elements, adds to the record an additional component `T`. This component is a function which returns, for an element $w \in W$, the corresponding basis element T_w in H. These elements can be added, multiplied etc. with the usual operations `+`, `-`, `*` etc. The list of ring elements $(q_s \mid s \in S)$ is stored in the record component `parameter`. Note that, in the original Weyl record, this list consists of 1's.

As a GAP object, any such element is a record with two components `basrep` and `operations`. The first one is a list of pairs (a_w, w) where a_w is the coefficient of T_w in the given element. The second contains the functions implementing addition, multiplication etc.

Let us give an example in which we perform some computations in the generic Iwahori-Hecke algebra H_0 associated with the Weyl group W of type A_3.

```
gap> W:= Weyl(CartanMat("A", 3));
Weyl( [ [ 2, -1, 0 ], [ -1, 2, -1 ], [ 0, -1, 2 ] ] )
gap> W.parameter;
[ 1, 1, 1 ]
gap> u:= Indeterminate(Rationals);;  u.name:= "u";;
gap> Hecke(W, u);  W.parameter;
[ u, u, u ]
gap> T:= W.T;;  T([1])^2;
(u)*T([ ])+(u - 1)*T([ 1 ])
gap> last.basrep;
[ [ u, [  ] ], [ u - 1, [ 1 ] ] ]
gap> T(LongestWeylWord(W))^2 = T([1, 2, 3])^4;
true
```

In this way, we can work with arbitrary elements of the Iwahori-Hecke algebra H. We point out explicitly that such computations can be carried out for H defined over any commutative ground ring which can be constructed in GAP, and any choice of parameters.

One of the main applications that we originally had in mind was the possibility of working with representations of Iwahori-Hecke algebras. Assume we are given any set of matrices $A_s \in R^{n\times n}$ ($s \in S$). The fact that H is given by generators and defining relations immediately implies that there is a (unique) representation $\rho : H \to R^{n\times n}$ such that $\rho(T_s) = A_s$ for all $s \in S$, if and only if the matrices A_s satisfy the same relations as those for the generators T_s of H.

A general approach for the construction of representations is in terms of W-graphs, see [28], p.165. Any such W-graph carries a representation of H. Note that, for these purposes, it is necessary to assume that the parameters of H are squares of some elements of the ground ring. The simplest example, the standard W-graph defined in [28], Example 6.2, yields a "deformation" of the natural reflection representation of W. This can be produced in CHEVIE as follows.

```
gap> W:= Weyl(CartanMat("B", 2));
Weyl( [ [ 2, -2 ], [ -1, 2 ] ] )
gap> v:= Indeterminate(Rationals);;  v.name:= "v";;
gap> Hecke(W, [v^2, v^2]);
gap> ref:= HeckeReflectionRepresentation(W);
[ [ [ -v^0, 0*v^0 ], [ -v^2, v^2 ] ],
  [ [ v^2, -2*v^0 ], [ 0*v^0, -v^0 ] ] ]
gap> WeylConjugacyClasses(W);
[ [  ], [ 1 ], [ 1, 2, 1, 2 ], [ 2 ], [ 1, 2 ] ]
gap> CharHeckeRepresentation(ref, last);
[ 2*v^0, v^2 - 1, -2*v^4, v^2 - 1, 0*v^0 ]
```

By the last command, we have also computed the character values of the natural representation on elements T_w where w runs over a set of representatives of minimal length of the conjugacy classes of W.

Another possibility to construct W-graphs is by using the Kazhdan-Lusztig theory of left cells (see [28]). There are some programs in CHEVIE for computing Kazhdan-Lusztig polynomials, left cells and the corresponding representations.

We continue the above example where W is of type B_2.

```
gap> cells:= LeftCells(W);;
gap> List(last, i-> Length(i[1]));
[ 1, 3, 1, 3 ]
gap> cells[2];
[ [ [ 1 ], [ 2, 1 ], [ 1, 2, 1 ] ],
  [ [ 0 ], [ 1, 0 ], [ 0, 1, 0 ] ] ]
gap> LeftCellRepresentation(W, last, v);
[ [ [  -v^0,      v,  0*v^0 ],
    [ 0*v^0,    v^2,  0*v^0 ],
    [ 0*v^0,      v,   -v^0 ] ],
  [ [   v^2, 0*v^0,  0*v^0 ],
    [     v,  -v^0,      v ],
    [ 0*v^0, 0*v^0,    v^2 ] ] ]
gap> CheckHeckeDefiningRelations(W, last);
true
```

Thus, we have computed all left cells of W. There are four such cells, containing 1, 3, 1, 3 elements, respectively. The result of `LeftCells` is a list of pairs (C, μ) where C is the set of elements in a left cell of W, and μ is the corresponding lower triangular matrix of coefficients $\mu(x, y)$, for $x, y \in C$ (see [28], Definition 1.2). Then the corresponding left cell representation is returned in the form of two matrices. The last command checks that these matrices indeed satisfy the defining relations for H and, hence, define a representation of H.

One should note, however, some limitations of these programs. They are designed to deal with many elements for one fixed group. For this purpose, they compute, for example, explicitly a complete list of all elements of W. Thus, it is reasonable to apply these programs to groups of rank ≤ 4. For groups of bigger rank, one should use a special purpose program like the one provided by F. DuCloux [16].

Finally, we give a simple example of how various little programs can be combined to check some properties of left cells. The following sequence of GAP and CHEVIE commands checks that, for all irreducible crystallographic Weyl groups of rank 2, 3, 4, the values $\mu(x, y)$, for x, y in any left cell of W, are $0, 1$.

```
gap> list:= [["A", 2], ["B", 2], ["G", 2], ["A", 3], ["B", 3],
>           ["A", 4], ["B", 4], ["D", 4], ["F", 4]];;
gap> for tr in list do
>      W:= Weyl(CartanMat(tr[1], tr[2]));
>      cells:= LeftCells(W);
>      Print("(type, rank) = ", tr, ",  ");
>      Print("No. of cells = ", Length(cells), ":  ");
>      #  display set of mu values.
>      Print(Set(Flat(List(cells, i-> i[2]))), "\n");
>   od;
```

The result of this is as follows.

```
(type, rank) = [ "A", 2 ],  No. of cells = 4:  [ 0, 1 ]
(type, rank) = [ "B", 2 ],  No. of cells = 4:  [ 0, 1 ]
(type, rank) = [ "G", 2 ],  No. of cells = 4:  [ 0, 1 ]
(type, rank) = [ "A", 3 ],  No. of cells = 10:  [ 0, 1 ]
(type, rank) = [ "B", 3 ],  No. of cells = 14:  [ 0, 1 ]
(type, rank) = [ "A", 4 ],  No. of cells = 26:  [ 0, 1 ]
```

```
(type, rank) = [ "B", 4 ],  No. of cells = 50:  [ 0, 1 ]
(type, rank) = [ "D", 4 ],  No. of cells = 36:  [ 0, 1 ]
(type, rank) = [ "F", 4 ],  No. of cells = 72:  [ 0, 1 ]
gap> time;
15970760
```

The time given is in 1/1000 seconds, hence the whole computation took roughly 4 1/2 hours (on a HP 9000/725) most of which was spent, of course, on W of type F_4.

3.2. **Minimal length representatives in conjugacy classes.** We can use the GAP functions for Weyl groups to verify a theorem about the conjugacy classes of a Weyl group. In fact, these programs have been used to prove this theorem for crystallographic Weyl groups of exceptional type in [22]. Using the Weyl group of type H_4 as an example, we show here that the theorem also holds for the non-crystallographic types. We thereby also wish to demonstrate how the existing functions from CHEVIE can be combined into little programs or sequences of commands which perform the necessary tasks.

Let $w, w' \in W$ and $s \in S$. Then either $l(sws) = l(w)$ or $l(sws) - l(w) = \pm 2$. We write $w \xrightarrow{s} w'$, or simply $w \to w'$, if $w' = sws$ and $l(w') \leq l(w)$. The reflexive transitive closure of this relation on W is also denoted by $\to$. Moreover we write $w \sim w'$, if there exists an $x \in W$ such that $w' = xwx^{-1}$, $l(w) = l(w')$, and $l(xw) = l(x) + l(w)$ or $l(wx^{-1}) = l(w) + l(x^{-1})$. Again the transitive closure of this relation on W is also denoted by $\sim$. For each conjugacy class C of W denote by $C_{\min}$ the set of elements of minimal length in C. Then the following holds (see [22], Theorem 1.1, for the crystallographic case).

(a) *For each $w \in C$ there exists an element $w' \in C_{\min}$ such that $w \to w'$.*
(b) *For all $w, w' \in C_{\min}$ we have $w \sim w'$.*

Given a particular Weyl group W, we can use the GAP programs for Weyl groups to verify these statements for W. In order to verify the two statements we have to perform several steps.

(1) For every conjugacy class C of W we have to construct the set $C_{\min}$ of elements of minimal length.
(2) For each element $w \in W$ we have to find a sequence $w = w_0 \xrightarrow{s_1} w_1 \to \cdots \xrightarrow{s_r} w_r$ where w_r has minimal length in the conjugacy class C of w in W.
(3) We have to find a sufficient number of instances $w \sim w'$ via some $x \in W$ for $w, w' \in C_{\min}$ such that the transitive closure of these establishes part (b) of the theorem.

We will achieve all of this by a slightly different strategy.

Let $S_1 \subseteq S$, let $W_1 = \langle S_1 \rangle$ be the corresponding parabolic subgroup of W and let D_1 be a subset of the set of distinguished right coset representatives of W_1 in W such that $W \to W_1 D_1$ (i.e. for each $w \in W$ there exist elements $w_1 \in W_1$, $d_1 \in D_1$ such that $w \to w_1 d_1$). Moreover let $S_2 \subseteq S_1$, $W_2 = \langle S_2 \rangle$, and let D_2 be the set of distinguished right coset representatives of W_2 in W_1. Then $W \to W_2 D$ where

$$D = \{d_2 d_1 | d_2 \in D_2, d_1 \in D_1, \text{ and } l(d_2 d_1 s) > l(d_2 d_1) \text{ for all } s \in S_2\}.$$

Note that if $S_1 = S$ then D is the set of distinguished double coset representatives of W_2 in W.

A tool to produce coset representatives according to the preceding remark is given by the function `SelectedCosets`. It takes the Weyl group W, two sets I and J such that $S_1 = \{s_i | i \in I\}$ and $S_2 = \{s_i | i \in J\}$ (in particular, $J \subseteq I$), and a list of coset representatives as its arguments and returns the new list D of coset representatives.

```
SelectedCosets:= function(W, I, J, cos)
   local c, d, x, w, new;

   new:= [];
   for d in WeylRightCosetRepresentatives(W, I, J) do
      w:= PermWeylWord(W, d);
      for c in cos do
         x:= w * c;
         if ForAll(J, j-> j/x <= W.N) then
            Add(new, x);
   fi; od; od;
   return new;
end;
```

If this function is applied along a chain of parabolic subgroups of W we end up with a (relatively small) set $L \subseteq W$ such that $W \to L$.

The Weyl group W of type H_4 may serve as an example.

```
gap> h4:= Weyl(CartanMat("H", 4));;
gap> cos:= [ () ];;
gap> cos:= SelectedCosets(h4, [1..4], [1..3], cos);;
gap> cos:= SelectedCosets(h4, [1..3], [1,2], cos);;
gap> cos:= SelectedCosets(h4, [1,2], [2], cos);;
gap  cos:= SelectedCosets(h4, [2], [], cos);;
gap> Length(cos);
376
```

That means, instead of $|W| = 14400$ elements, we only have to consider 376 elements.

Suppose that the function `TestCyclicShifts` returns `true` for an element $w \in W$ if $l(w) = l(w')$ for all $w' \in W$ with $w \to w'$ and returns an element y such that $w \to y$ and $l(y) < l(w)$ otherwise, then the following lines of code produce a list `min` of elements such that $W \to$ `min` and every element in `min` has minimal length with respect to the relation $\to$.

```
gap> min:= [];
gap> for w in cos do
>       y:= TestCyclicShifts(h4, w);
>       if y = true then
>           AddSet(min, w);
>       else
>          if not y in min and not y in cos then
>             Add(cos, y);
>    fi; fi; od;
gap> Length(min);
112
```

With the aid of the GAP function `ConjugacyClasses` for permutation groups we split that list into conjugacy classes and look at the lengths of the elements in one class.

```
gap> cc:= ConjugacyClasses(h4);;
gap> new:= List(cc, x-> []);;
gap> for m in min do
>       Add(new[Position(cc, ConjugacyClass(h4, m))], m);
>    od;
gap> List(new[2], x-> WeylLengthPerm(h4, x));
[ 18, 18, 18, 18, 18, 18 ]
```

All elements of `min` belonging to the second conjugacy class of W have the same length! This turns out to be the case for all 34 conjugacy classes of W. So we have shown that all elements which are minimal with respect to the relation $\to$ are in fact of minimal length in their conjugacy class. Hence (a) holds for W of type H_4.

In order to verify (b) we define a function `CyclicShifts` which produces for any element $w \in W$ the orbit of w under the equivalence relation which is generated by $w \xrightarrow{s} w'$ for some $s \in S$ with $l(w) = l(w')$. It is easy to see that in such a case $w \sim w'$.

```
CyclicShifts:= function(W, w)
   local orbit, x, y, i, s, n, N;

   s:= W.permgens;  N:= W.N;  n:= W.dim;
   orbit:= [w];
   for x in orbit do
      for i in [1..n] do
         y:= s[i] * x;
         if (i^x <= N and i/y > N) or (i^x > N and i/y <= N) then
            y:= y * s[i];
            if not y in orbit then
               Add(orbit, y);
   fi; fi; od; od;
   return orbit;
end;
```

In most cases this test is sufficient in order to establish the relation $\sim$ for a class of elements in the list `new`.

```
gap> rest:= [];;
gap> for n in new do
>       if not IsSubset(CyclicShifts(h4, n[1]), n) then
>          Add(rest, n);
>    fi; od;
gap> Length(rest);
3
```

We are left with three sets of elements, where the relation $\sim$ still has to be established. Application of the function `WeylWordPerm` to the lists of elements in the list `rest` reveals that these are the sets

$$\{s_1, s_2, s_3, s_4\},$$
$$\{s_1s_3, s_1s_4, s_2s_4\}, \text{ and}$$
$$\{s_2s_3, s_3s_2, s_3s_4, s_4s_3\}.$$

Now it is easy to see from the defining relations of W that, for instance the generators s_4 and s_3 are conjugate via the element s_3s_4 in such a way that $s_4 \sim s_3$. The other cases follow by a similar argument. Hence the theorem is verified for the Weyl group of type H_4.

Similarly, the case of the Weyl group of type H_3 (in fact, of any particular finite Weyl group) can be treated. Note that checking (a) and (b) for the dihedral groups of type $I_2(m)$ is an easy exercise. In combination with the results in [22] we now see that (a) and (b) hold for any finite Weyl group.

3.3. Character tables of Iwahori-Hecke algebras.

In the previous section we have defined two binary relations $\longrightarrow$ and $\sim$ on W. These allow us to define the analogue of the ordinary character table of W for the corresponding Iwahori-Hecke algebra H. For this purpose, we fix the following choice of a ground ring and for the parameters of H. Let $A = \mathbb{Z}[v_s \mid s \in S]$ be a polynomial ring as in Section 3 and H be the Iwahori-Hecke algebra over A corresponding to W and with parameters $\mathbf{v} = (v_s^2 \mid s \in S)$. Let K be the field of fractions of A and H_K the algebra obtained by scalar extension from A to K. Then it is known that H_K is semisimple and split (see [19]). This algebra plays the analogous role as the group algebra of W over $\mathbb{Q}$. Its representations and characters will be called ordinary.

Let $\mathrm{Irr}(H_K)$ be the set of irreducible characters of H_K, and $\{C\}$ be the set of conjugacy classes of W. For each class C we choose an element w_C of minimal possible length in C. Then the matrix of values

$$\chi(T_{w_C}), \qquad \text{for } \chi \text{ an irreducible character of } H_K \text{ and } C \text{ a class of } W,$$

is called the *generic character table* of H_K.

On one hand, the relation $\sim$ on W implies that this matrix is in fact independent of the choice of representatives of minimal length. On the other hand, if $w \in W$ is any element then there exist elements $f_{w,C} \in K$ such that

$$\chi(T_w) = \sum_C f_{w,C}\ \chi(T_{w_C}) \qquad \text{for all } \chi \in \mathrm{Irr}(H_K),$$

where the sum is over all conjugacy classes C of W. Now the relation $\longrightarrow$ implies that the elements $f_{w,C}$ actually lie in A and that they can be computed in an algorithmic way. (See [22] for details.) They will be called the *class polynomials* of W.

The character tables of the Iwahori-Hecke algebras of all types are explicitly known. For type $I_2(m)$, they are easily constructed from [12], Theorem (67.14). For type H_3, the W-graphs and explicit matrix representations are given in [30]. The table for H_4 has been computed by means of the W-graphs in [1]. The tables for the remaining exceptional types are computed in [19], [18], [21], algorithms for computing the tables for type A_n and B_n are contained in [42, 24], [40, 41], for type D_n see [24].

For each type, there is a function (e.g., `HeckeCharTableG2`) which takes as input a list of parameters and returns the character table record of the corresponding generic Iwahori-Hecke algebra H_K where the indeterminates have been specialized to the prescribed values. (For type $I_2(m)$ there is a second argument specifying the value of m.) The user has to take care of the right choice of parameters: For type $I_2(m)$, G_2, H_3, H_4, E_7, E_8, the argument must be a list of square roots of the parameters for the Iwahori-Hecke algebra, and for type $I_2(m)$, H_3, H_4 one cannot work over the rationals but over a suitable cyclotomic extension (e.g., the field containing the 5-th roots of unity for type H_3, H_4). As already explained in Section 2.4 the character table records contain additional components which give canonical labellings for the classes and characters as well as explicit representatives of minimal length in the conjugacy classes of W.

In the following example we show how the class polynomials are actually used to compute the character values on basis elements T_w where $w \in W$ is not necessarily of minimal length in its conjugacy class. We consider the Weyl group of type G_2.

```
gap> W:= Weyl(CartanMat("G", 2));
Weyl( [ [ 2, -1 ], [ -3, 2 ] ] )
gap> v:= Indeterminate(Rationals);;  v.name:= "v";;
gap> Hecke(W, [v^2, v^6]);
gap> cc:= WeylConjugacyClasses(W);;
gap> elts:= List(Elements(W), i-> WeylWordPerm(W, i));
[ [  ], [ 2, 1, 2, 1, 2 ], [ 2 ], [ 2, 1, 2, 1 ], [ 2, 1 ],
  [ 2, 1, 2 ], [ 1 ], [ 1, 2, 1, 2, 1, 2 ], [ 1, 2, 1, 2, 1 ],
  [ 1, 2 ], [ 1, 2, 1, 2 ], [ 1, 2, 1 ] ]
gap> cp:= List(last, i-> WeylClassPolynomials(W, cc, i));
[ [ v^0,   0,   0,         0,       0,   0 ],
  [   0,   0, v^8, v^8 - v^6, v^6 - 1,   0 ],
  [   0,   0, v^0,         0,       0,   0 ],
  [   0,   0,   0,         0,     v^0,   0 ],
  [   0,   0,   0,       v^0,       0,   0 ],
  [   0, v^6,   0,   v^6 - 1,       0,   0 ],
  [   0, v^0,   0,         0,       0,   0 ],
  [   0,   0,   0,         0,       0, v^0 ],
  [   0, v^8,   0, v^8 - v^2, v^2 - 1,   0 ],
  [   0,   0,   0,       v^0,       0,   0 ],
  [   0,   0,   0,         0,     v^0,   0 ],
  [   0,   0, v^2,   v^2 - 1,       0,   0 ] ]
```

The following commands will compute the character table of the corresponding Iwahori-Hecke algebra H with parameters (v^2, v^6), and then the values of the irreducible characters on all basis elements T_w, $w \in W$.

```
gap> ct:= HeckeCharTableG2([v, v^3]);
CharTable( "H(G2)" )
gap> ct.irreducibles * TransposedMat(cp * v^0);
```

We don't print it here; please try it yourself!

4. Generic character tables

In this section we shortly describe the theoretical background underlying the definition of a generic character table of a series of finite groups of Lie type.

4.1. **Generic finite reductive groups.**

A general reference for finite reductive groups and their representation theory are the books by Carter [8] and by Digne and Michel [15].

The concept of a *complete root datum* was introduced by Broué and Malle in [3]. Here, we follow Broué, Malle and Michel [5], where a complete root datum is called *generic finite reductive group.* A *series of finite reductive groups* is the collection of all finite reductive groups associated to a *generic finite reductive group.*

A complete root datum or a generic finite reductive group is a pair $\mathbb{G} = (\Gamma, W\phi)$, where Γ is a root datum, W its Weyl group and ϕ an automorphism of finite order of Γ. The group W as well as ϕ are contained in the common overgroup $\mathrm{Aut}(\Gamma)$ of automorphisms of Γ, so that $W\phi$ simply means a coset of W in $\mathrm{Aut}(\Gamma)$. A *root datum*

Γ is a quadruple $\Gamma = (X, \Phi, Y, \Phi^\vee)$, where X and Y are free $\mathbb{Z}$-modules of the same finite rank endowed with a non-degenerate pairing $X \times Y \to \mathbb{Z}$, written as $(x, y) \mapsto \langle x, y \rangle$. Furthermore, Φ and $\Phi^\vee$ are finite subsets of X and Y, respectively, and there is a bijection $\alpha \mapsto \alpha^\vee$ between Φ and $\Phi^\vee$ such that $\langle \alpha, \alpha^\vee \rangle = 2$, $\beta - \langle \beta, \alpha^\vee \rangle \alpha \in \Phi$ and $\beta^\vee - \langle \alpha, \beta^\vee \rangle \alpha^\vee \in \Phi^\vee$ for all $\alpha, \beta \in \Phi$. An *automorphism* ϕ *of* Γ is an automorphism of finite order of Y such that $\Phi^\vee$ is invariant under ϕ and Φ is invariant under the adjoint automorphism of ϕ on X.

For $\alpha \in \Phi$ let s_α be the automorphism of X defined by $s_\alpha(x) = x - \langle x, \alpha^\vee \rangle \alpha$ and $s_\alpha^\vee$ the automorphism of Y defined by $s_\alpha^\vee(y) = y - \langle \alpha, y \rangle \alpha^\vee$. Then $s_\alpha^\vee$ is adjoint to s_α with respect to the given pairing. The *Weyl group* W of the root datum Γ is defined as the subgroup of the automorphism group of Y generated by the $s_\alpha^\vee$ for $\alpha \in \Phi$. In particular, W is a subgroup of $\mathrm{Aut}(\Gamma)$.

Let V be the real vector space spanned by $\Phi^\vee$ in $Y \otimes_{\mathbb{Z}} \mathbb{R}$. Then W, considered as a group of automorphisms of $Y \otimes_{\mathbb{Z}} \mathbb{R}$ (by linear extension) leaves V invariant. We can endow V with a W-invariant positive definite symmetric bilinear form such that $\Phi^\vee$ is a crystallographic root system in the sense of Section 2, and W is its Weyl group. For details we refer to [46], 9.1.8.

Given a root datum Γ and a prime number p, there exists a pair $(\mathbf{G}, \mathbf{T})$, where $\mathbf{G}$ is a connected reductive group defined over $\bar{\mathbb{F}}_p$, the algebraic closure of the prime field $\mathbb{F}_p$, and $\mathbf{T}$ is a maximal torus of $\mathbf{G}$ such that Γ is the root datum associated to the pair $(\mathbf{G}, \mathbf{T})$. The algebraic group $\mathbf{G}$ is uniquely determined by Γ and p, up to isomorphism. Given a complete root datum $\mathbb{G} = (\Gamma, W\phi)$, a prime number p, a power q of p and an element ϕ' in the coset $W\phi$, there is a Frobenius morphism $F : \mathbf{G} \to \mathbf{G}$, such that $\mathbf{T}$ is F-stable and F induces on Y the homomorphism $q \cdot \phi'$. The associated finite reductive group, denoted by $\mathbf{G}^F$ or $\mathbb{G}(q)$ is uniquely determined by $\mathbb{G}$ and q, up to isomorphism. The collection of all groups $\mathbb{G}(q)$, $q = p^f$, $f \in \mathbb{N}$, p a prime number, is called the *series of finite reductive groups* (or finite groups of Lie type) associated to $\mathbb{G}$.

From a complete root datum $\mathbb{G}$ we get another complete root datum, called the *dual* of $\mathbb{G}$ and denoted by $\mathbb{G}^*$, by changing the roles of X and Y (and so of Φ and $\Phi^\vee$) and changing ϕ to its adjoint. We denote the associated connected reductive groups by $\mathbf{G}^*$. The Frobenius morphism of $\mathbf{G}^*$ corresponding to a power q of p is denoted by F^*, and $\mathbb{G}^*(q)$ is called the *dual group* of $\mathbb{G}(q)$.

Example 4.1. Let $X = Y = \mathbb{Z}^2$, the pairing being the standard inner product on X. Let $\Phi := \{(2, -1), (-1, 2), (1, 1), (-2, 1), (1, -2), (-1, -1)\}$ and $\Phi^\vee := \{(1, 0), (0, 1), (1, 1), (-1, 0), (0, -1), (-1, -1)\}$. If we map the i-th element of Φ onto the i-th element of $\Phi^\vee$ we get a root datum Γ which describes the reductive groups $SL_3(\bar{\mathbb{F}}_p)$. This root datum together with the identity map on Y or its negative define complete root data. They describe the series $\{SL_3(q)\}$ or $\{SU_3(q)\}$, respectively. The dual group of $SL_3(q)$ is $PGL_3(q)$.

The definition of the Suzuki and Ree groups is slightly more complicated. Let us content ourselves with the remark that the Suzuki groups $Sz(q^2) = {}^2B_2(q^2)$, $q^2 = 2^{2n+1}$, and the Ree groups ${}^2G_2(q^2)$, $q^2 = 3^{2n+1}$, respectively ${}^2F_4(q^2)$, $q^2 = 2^{2n+1}$, each form a series of finite reductive groups.

4.2. **Semisimple class types.**

We fix a complete root datum $\mathbb{G}$ and use the notations $\mathbf{G}$, $\mathbb{G}(q)$ as above. Let $s_1, s_2 \in \mathbb{G}(q)$ be semisimple elements. We say that s_1 and s_2 have the same *semisimple class type*, if and only if their centralizers in the algebraic group $\mathbf{G}$ are conjugate by an element of $\mathbb{G}(q)$. It is clear that the semisimple elements of one given semisimple class type form a union of conjugacy classes of $\mathbb{G}(q)$.

The semisimple class types of $\mathbb{G}(q)$ for all prime powers q can be parameterized in terms of the complete root datum, i.e., independently of q. More precisely, there exists a finite set $\mathcal{C}$ of labels c of the form $c = (\Psi, W', \sigma)$, where $\Psi \subseteq \Phi$ is a closed subset, $W' \leq W$ and $\sigma \in W$. (A subset $\Psi \subseteq \Phi$ is called closed, if $\mathbb{Z}\Psi \cap \Phi = \Psi$.) This classification of centralizers of semisimple elements in the special case of simply-connected groups $\mathbf{G}$ was first obtained by Carter [7] and Deriziotis [14]. We do not want to go into further details as to which triples actually occur as labels. Let us just add a few comments. Given a prime p, and thus a connected reductive group $\mathbf{G}$, the first two components of the label c specify a closed subgroup $\mathbf{C}$ of $\mathbf{G}$. This group is generated by $\mathbf{T}$, the root subgroups of $\mathbf{G}$ with respect to $\mathbf{T}$ corresponding to Ψ and the elements of $N_{\mathbf{G}}(\mathbf{T})$ mapping to W'. We have an exact sequence

$$1 \to W_\Psi \to W' \to \mathbf{C}/\mathbf{C}^\circ \to 1,$$

where W_Ψ denotes the Weyl group of Ψ, and $\mathbf{C}^\circ$ the connected component of $\mathbf{C}$. (The *Weyl group* of Ψ is the subgroup of W generated by the $s_\alpha^\vee$ for $\alpha \in \Psi$.) Distinct pairs (Ψ, W') give rise to distinct $\mathbf{G}$-conjugacy classes of closed subgroups of $\mathbf{G}$. Moreover, if we choose a power q of p, and thus a Frobenius morphism F of $\mathbf{G}$, the $\mathbb{G}(q)$-conjugacy classes of the F-stable elements in the $\mathbf{G}$-class of $\mathbf{C}$ are distinguished by the third component σ of the label (Ψ, W', σ). For each semisimple class type in $\mathbb{G}(q)$ there is a unique label c in $\mathcal{C}$ such that for each semisimple element s in this class type, $\mathbf{C} = C_{\mathbf{G}}(s)$ is in the $\mathbb{G}(q)$-conjugacy determined by c.

There exists a positive integer m, and for each $i \in \mathbb{Z}$ and each $c \in \mathcal{C}$, a polynomial $n_{c,i} \in \mathbb{Q}[x]$ with the following property. If q is a prime power with $q \equiv i (\mathrm{mod}\ m)$, and a semisimple class type of $\mathbb{G}(q)$ corresponds to $c \in \mathcal{C}$, then the number of $\mathbb{G}(q)$-conjugacy classes forming this class type is equal to $n_{c,i}(q)$.

Example 4.2. We continue with the notation of Example 4.1. Let p be a prime and c be the triple $(\{(2,-1),(-2,1)\}, \{1, s_{(2,-1)}\}, 1)$. Then c determines the subgroup $\mathbf{C}$ of the following elements in $SL_3(\bar{\mathbb{F}}_p)$:

$$\begin{pmatrix} \begin{matrix} & \\ & A \end{matrix} & \begin{matrix} 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 \end{matrix} & \det(A)^{-1} \end{pmatrix}, \quad A \in GL_2(\bar{\mathbb{F}}_p).$$

Now let q be a power of p. The elements of $SL_3(q)$ which have $\mathbf{C}$ as centralizer are exactly the diagonal matrices $\mathrm{diag}(a, a, a^{-2})$ with $a \in \mathbb{F}_q^\times$, $a^3 \neq 1$. Let $\tilde{\zeta}$ be a generating element of $\mathbb{F}_q^\times$. Then we can parameterize the above elements as follows:

$$\mathrm{diag}(\tilde{\zeta}^i, \tilde{\zeta}^i, \tilde{\zeta}^{-2i}) \ \text{ with } \begin{cases} 1 \leq i \leq q-1,\ \frac{q-1}{3} \nmid i, \text{ if } q \equiv 1 (\mathrm{mod}\ 3) \\ 1 \leq i \leq q-1,\ (q-1) \nmid i, \text{ otherwise} \end{cases}$$

It is not difficult to see that different elements in this set are not conjugate in $SL_3(q)$. So the number of conjugacy classes in the semisimple class type corresponding to the $\mathbb{G}(q)$-conjugacy class of c is $q-4$, if $q \equiv 1 (\mathrm{mod}\ 3)$, and $q-2$ for all other q. Thus $m = 3$, $n_{c,1}(x) = x - 4$, and $n_{c,2}(x) = n_{c,3}(x) = x - 2$.

4.3. **Lusztig series types of irreducible characters.** We recall that we have a partition of the set of irreducible (complex) characters of $\mathbb{G}(q)$ into *rational Lusztig series* $\mathcal{E}(\mathbb{G}(q), (s)_{\mathbb{G}^*(q)})$. Here $(s)_{\mathbb{G}^*(q)}$ runs through the conjugacy classes of semisimple elements s in the dual group $\mathbb{G}^*(q)$. (This is explained, e.g., in [15], 13.16, 13.17, 14.41.)

We say that two rational Lusztig series $\mathcal{E}(\mathbb{G}(q),(s_1)_{\mathbb{G}^*(q)})$ and $\mathcal{E}(\mathbb{G}(q),(s_2)_{\mathbb{G}^*(q)})$ have the same *Lusztig series type* if and only if s_1 and s_2 have the same semisimple class type in the dual group $\mathbb{G}^*(q)$.

The characters in the rational Lusztig series $\mathcal{E}(\mathbb{G}(q),(1)_{\mathbb{G}^*(q)})$ are called *unipotent characters*. Lusztig has shown that these can be parameterized in terms of $\mathbb{G}$, independently of q.

4.4. **Unipotent conjugacy classes and Lusztig series.** Fix $\mathbb{G}$ and let $c \in \mathcal{C}$. Then there exists a positive integer m, and for each $i \in \mathbb{Z}$ a finite set $\mathcal{U}_{c,i}$ of "labels", such that the following holds. If p is a prime and q a power of p with $q \equiv i (\bmod\ m)$, then the set of unipotent conjugacy classes of $\mathbf{C}^F$ is in bijection with $\mathcal{U}_{c,i}$. Here, $\mathbf{C}$ is an F-stable subgroup of $\mathbf{G}$, determined by c, and $\mathbf{G}$ is the algebraic group determined by $\mathbb{G}$ and p.

Let $c^* \in \mathcal{C}^*$, where $\mathcal{C}^*$, of course denotes the set of labels for the semisimple class types of the dual group $\mathbb{G}^*$. There exists a finite set Λ_{c^*} of "labels" with the following property. For a semisimple element $s \in \mathbb{G}^*(q)$ in the semisimple class type c^*, the irreducible characters in the Lusztig series $\mathcal{E}(\mathbb{G}(q),(s)_{\mathbb{G}^*(q)})$ are in bijection with Λ_{c^*}. This follows from Lusztig's Jordan decomposition of characters [32]. In the special case that the centralizer $\mathbf{C}^*$ of s in $\mathbf{G}^*$ is connected, the set Λ_{c^*} may be taken to be the set of labels of the unipotent characters of the complete root datum of $\mathbf{C}^*$.

4.5. **Generic character tables.** For a fixed complete root datum $\mathbb{G}$ let m_0 be a positive integer which is divisible by all the numbers m appearing in 4.2 and 4.4, and let $1 \leq k \leq m_0$. A *generic character table* associated to $\mathbb{G}$ and k describes the values of the irreducible characters of all groups $\mathbb{G}(q)$ with $q \equiv k (\bmod\ m_0)$. More precisely it consists of the following data:

(a) Columns labelled by pairs (c, u) with $c \in \mathcal{C}$ and $u \in \mathcal{U}_{c,k}$ such that $n_{c,k} \neq 0$.
(b) For each c appearing in (a) a parameterization of the semisimple conjugacy classes in $\mathbb{G}(q)$ having a representative with centralizer $\mathbf{C}$ determined by c (they form a semisimple class type). This parameterization can be given in a uniform way for all relevant prime powers q.
(c) Rows labelled by pairs (c^*, λ^*) with $c^* \in \mathcal{C}^*$ and $\lambda \in \Lambda_{c^*}$ and $n_{c^*,k} \neq 0$.
(d) For each c^* appearing in (c) a parameterization of the semisimple conjugacy classes in $\mathbb{G}^*(q)$ having a representative with centralizer $\mathbf{C}^*$ determined by c^*.
(e) The entries: for each column labelled by (c, u) and row labelled by (c^*, λ^*) there is an entry which describes the character values for the characters labelled by λ^* in $\mathcal{E}(\mathbb{G}(q),(s)_{\mathbb{G}^*(q)})$ where $s \in \mathbb{G}^*(q)$ is in the class type with label c^*, on all elements of $\mathbb{G}(q)$ of the form tu' where t is a semisimple element of type c, and u' is a unipotent element of $C_{\mathbf{G}}(t)^F$ lying in a conjugacy class with label u (see 4.4). The values are described in terms of the parameters used in (b) and (d).

In CHEVIE we realize (b) as follows. We parameterize those elements in the center of a representative $\mathbf{C}$ as above whose centralizer in $\mathbf{G}$ is exactly $\mathbf{C}$. In this way we do not have unique representatives for the classes in the semisimple class type corresponding to c, but each of these classes is represented by the same number of elements. (The same holds for (d).)

To a column of the generic character table one can associate the set of conjugacy classes of the elements tu' as in (e). Such a set of conjugacy classes is called a *class* *type*. Similarly we call the set of characters corresponding to a row of the generic table a *character type*.

We remark that it may happen that a conjugacy class lies in more than one class type, and similarly for character types. For example, for $a, b \in \mathbb{F}_q^\times$, the two elements

$$\begin{pmatrix} a & 1 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & b \end{pmatrix}, \qquad \begin{pmatrix} b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 1 \\ 0 & 0 & 0 & a \end{pmatrix}$$

are conjugate in $GL_4(q)$, their semisimple parts have the same centralizer, but if $a \neq b$, the unipotent parts are not conjugate in this centralizer.

In the next section we will indicate how a generic character table is implemented in CHEVIE.

4.6. **Remarks.** In this paper we describe a version of CHEVIE which contains a library of known generic character tables and programs which support computations with these tables. In a future version of CHEVIE there will also be programs which *compute* (at least parts of) generic character tables, starting just from a given complete root datum. These programs are developed by the third named author. A paper which explains the methods used in these programs is in preparation. It will contain much more details about the concepts introduced in this section.

5. Examples

In this section we show a sample session of the Maple part of CHEVIE, dealing with generic character tables. We give examples for a lot of the CHEVIE commands and explain how they work. The screen output in the examples is sometimes slightly edited or omitted to limit this section to reasonable length. For the same reason some commands are only mentioned but not shown in the session.

5.1. **Some basic commands.** If CHEVIE is installed on your system, you can invoke the Maple-part by typing `cheviem` in a UNIX-shell. This calls Maple and loads the CHEVIE-programs. The welcome message tells you to try the command `?CHEVIE`. This shows you a list of all available CHEVIE commands as well as some general information about CHEVIE. Similarly you get online help for each single CHEVIE command.

Now let us stick to the example of the last section, namely the groups $SL_3(q)$. The number m_0 mentioned in 4.5 can be chosen to be equal to 3 in this case, hence the generic character tables for these groups depend on the congruences of q modulo 3. The CHEVIE library of generic character tables contains two tables for SL_3, they have the names `SL3.1` and `SL3.n1`; the first is for $q \equiv 1 (\text{mod } 3)$ and the second for the other cases. Let us look at the first one. It can be read into the CHEVIE session with the command `GenCharTab('SL3.1');` (don't forget the `'`). The command `GenCharTab();`, i.e., without arguments, shows a list of all tables available, in particular we can easily find those corresponding to $SL_3(q)$.

With `Status('SL3.1');` we can see the size of this table, i.e., the number of class and character types as well as the order of $\mathbb{G}(q)$.

We get information about the class types and the corresponding centralizers with the command `PrintInfoClass`.

```
> PrintInfoClass(‘SL3.1‘);
clt   Information
=====================================================
1   [ , [1, 0], [A_2, [1, 1, 1]]]
2   [ , [1, 1], [A_2, [2, 1]]]
3   [ , [1, 2], [A_2, [3]]]
4   [ , [1, 3], [A_2, [3]]]
5   [ , [1, 4], [A_2, [3]]]
6   [ , [2, 0], [A_1, [1, 1]]]
7   [ , [2, 1], [A_1, [2]]]
8   [ , [3, 0], [A_0, [1]]]
9   [ , [4, 0], [A_0, [1]]]
10   [ , [5, 0], [A_0, [1]]]
```

The rows of this information table correspond to the columns of the generic character table. Each row is a list with a certain number of entries (three in the above example), which may again be lists. For technical reasons, the first entry is empty. The third entry contains information about the labels of the class types (4.2 and 4.4).

For example in the row beginning with 7 the last entry means that the semisimple part of an element in the class type corresponding to the seventh column has Dynkin diagram of type A_1 and the unipotent part corresponds to the class with Jordan block of dimension 2 in this centralizer. The second entry of the information list is a list with two components. The first of these numbers the $\mathbb{G}(q)$-classes of centralizers of semisimple elements and the second the unipotent classes in these centralizers.

Now let us look at the character values. We have two commands, one which prints the values on the screen and one which produces a LaTeX file with the table (or parts of it) in a pretty printed form.

```
> PrintVal(‘SL3.1‘,9,7); # or PrintVal(‘SL3.1‘); for the whole table

clt   Value of character type    9    on class type clt
7   GEWZ1^(aA*nN-2*aA*mM)+GEWZ1^(aA*mM-2*aA*nN)+GEWZ1^(aA*nN+aA*mM)

> PrintToTeX(‘SL3.1‘,9,7); # PrintToTeX(‘SL3.1‘); for the whole table
```

The latter command produces a LaTeX file, and applies `latex` and a previewer to it, if available on your system. The result looks as follows.

$SL_3(q)$	$C_7(a)$
$\chi_9(n,m)$	$\zeta_1^{an-2am}+\zeta_1^{am-2an}+\zeta_1^{an+am}$

$\zeta_1 := \exp(\frac{2\pi\sqrt{-1}}{q-1})$

The expression `GEWZ1` (or ζ_1 in pretty printed form) is called a *generic root of unity.* You can find an explanation of the CHEVIE names for generic roots of unity by typing `?GEW`.

Furthermore, the pretty printed table tells you that the conjugacy classes in the class type of the seventh column are distinguished by a parameter a and the characters in the ninth character type are distinguished by parameters n and m (we use the names `aA, nN, mM` in the CHEVIE session to minimize an accidental use of these variables for other purposes).

More detailed information about these parameters (this is 4.5(b) and (d)) can be obtained with the commands `PrintClassParam` and `PrintCharParam`.

```
> PrintClassParam(`SL3.1`,7);
clt   Parameters   Exceptions
=====================================================
7   [aA = 1 .. q-1]   [[aA, 1/3*q-1/3]]
> NrClasses(g,7); # number of classes in class type 7
                                    q - 4
```

This means that the character values for all classes in this class type can be obtained by specializing the parameter a (or `aA`) to the integers from 1 to $q-1$ except those which are multiples of $(q-1)/3$ (recall that we are in the case $q \equiv 1 (\mathrm{mod}\ 3)$). More generally, the *Exceptions* list consists of expressions of the form [x,y], and this means that the parameters must satisfy $y \nmid x$. (For $c \in \mathcal{C}$ as in 4.5 we parameterize under *Parameters* the elements of the center of $\mathbf{C}^F$ and under *Exceptions* we describe those elements whose centralizer in $\mathbf{G}$ is properly bigger than $\mathbf{C}$; compare with Example 4.2.)

If we want to do some calculations with this generic table, we need a way to tell the programs that $(q-1)/3$ is an integer. This is done by defining for each table two functions `setCongruence.nr` and `unsetCongruence.nr` where `nr` is an internal group number stored in position `[-2,0]` of the table. In our example they are as follows.

```
> nr:=`SL3.1`[-2,0];
                                 nr := A2204
> print(setCongruence.nr);
proc() global qQ,q; qQ := 'qQ'; q := 3*qQ+1; NULL end
> print(unsetCongruence.nr);
proc() global q,qQ; q := 'q'; qQ := 1/3*q-1/3; NULL end
```

The first function is called before and the second after any calculations with the table. They cause the substitution of `q` by `3*qQ + 1` during these computations; the programs assume that `qQ` is an integer.

Finally we mention that some general information about a table in the CHEVIE library, for example references to the literature, can be obtained with `PrintInfoTab`.

5.2. **Orthogonality relations.** As an example for computations with a generic character table we explain the computation of some scalar products. We do this with the (small) table for the groups $SL_2(q)$ with even q.

```
> GenCharTab(`SL2.0`);
    # output omitted
> PrintCharParam(`SL2.0`,3);
cht   Parameters   Exceptions
=====================================================
3   [nN = 1 .. q-1]   [[nN, q-1]]
> NrChars(`SL2.0`,3);
                                  1/2 q - 1

> Scalar(`SL2.0`,[3],`SL2.0`,[3]);
Possible Exceptions:    {[nN1+nN2, q-1], [nN1-nN2, q-1]}
Scalar_SL2.0,SL2.0(3,3)=
                                     0
> Norm(`SL2.0`,[3]);
```

```
Possible Exceptions:     {[2*nN, q-1]}
Norm_SL2.0(3)=
                                             1
```

Here the characters $\chi_3(n)$ of the third character type are distinguished by a parameter n, which must be an integer not divisible by $q-1$. The function `Scalar`, called as above, does the following. It first substitutes the parameter n in the values for the third character type by n_1 and n_2, respectively, and substitutes q with the corresponding `setCongruence.nr` function. Then it computes the scalar product via the usual formula

$$\langle\chi_3(n_1),\chi_3(n_2)\rangle = \frac{1}{|SL_2(q)|}\sum_{g\in SL_2(q)}\chi_3(n_1)(g)\chi_3(n_2)(g^{-1}).$$

But how can this sum be computed? For doing this, the CHEVIE tables contain two additional types of information we have not yet mentioned. The first is the number of elements in each conjugacy class or, equivalently, the centralizer orders. They only depend on the class type and can be read off the table with `CentOrd`. This reduces the above sum to a sum over the conjugacy classes. The second is a set of Maple procedures which describe for each class type how to sum over all classes in this type. For example the classes in the forth class type are distinguished by a parameter a.

```
> PrintClassParam(‘SL2.0‘,4);
clt   Parameters   Exceptions
=======================================================
4   [aA = 1 .. q+1]   [[aA, q+1]]
> NrClasses(‘SL2.0‘,4);
                                          1/2 q
> nr:=‘SL2.0‘[-2,0];
                                      nr := A1202
> print(Klassen.nr.Summe.4);
proc(tt) local s1; s1 := nesum(tt,aA = 1 .. q); linkomb(1/2,s1) end
```

This summation procedure means that character values $f(a)$ on the classes of this class type can be added up by computing $\frac{1}{2}\sum_{a=1}^{q} f(a)$. So this procedure contains implicitly again the information we get with `PrintClassParam` and also how many different parameters correspond to the same class.

During these computations an expression like $\sum_{i=0}^{q-2} x^i$ is substituted by $(x^{q-1}-1)/(x-1)$ if CHEVIE cannot simplify the subexpression x to 1, and it is substituted by $q-1$ in the other case. In the first case the procedure also generates information, printed as a *Possible Exception*, which describes the condition for $x=1$. If for example $x=\zeta_1^k$ with ζ_1 a primitive $(q-1)$-th root of unity we have $x=1$ if and only if k is divisible by $q-1$; the exception is printed as `[kK, q-1]`.

In our example `Scalar(‘SL2.0‘,[3],‘SL2.0‘,[3]);` we got 0 as the *generic* result. But for two characters from the third character type the scalar product is 0 only if we take two different characters from this type. Otherwise the scalar product is 1 and so there must be some information about *Possible Exceptions.* And we really get such exceptions which tell us that the calculations are valid only if for fixed n_1 the parameter n_2 or $-n_2$ is not congruent to n_1 modulo $q-1$. These are exactly the two cases where n_1 and n_2 correspond to the same character in this type.

We can compute the scalar product of a character with itself by using the command `Norm` as demonstrated.

Similarly it is possible to compute second orthogonality relations and class multiplication coefficients. Here procedures which sum over all characters in a character type are used. These functions are called `Ortho2Norm, Ortho2Scalar` and `ClassMult`.

5.3. **A tensor product in $SL_3(q)$.** We return to the groups $SL_3(q)$, but now with $q \not\equiv 1 (\text{mod } 3)$. As an application of CHEVIE we demonstrate how to decompose a tensor product of two characters generically. We load the corresponding generic character table with `GenCharTab('SL3.n1');`.

The degrees of the irreducible characters of $SL_3(q)$ can be obtained with the command `CharDeg`.

Our aim is to decompose the tensor product of the irreducible character χ_2 of degree q^2+q with itself (use `NrChars` to see that this character type contains just one character).

Instead of typing `'SL3.n1'` we now just type `g`. This shortcut is always possible for the last table loaded with `GenCharTab`.

We first produce a copy `h` of our table `g` on which the tensor product is to be stored, then calculate the tensor product $\chi_2 \cdot \chi_2$ and store it on `h`.

```
> Copy(g,h,[],[]);
> Tensor(g,2,g,2,h);
```

Next we compute generic scalar products of all the characters on `g` with the character on `h`.

```
> Scalar(g,h);
Scalar_SL3.n1,h(1,1)=
                                        1
Scalar_SL3.n1,h(2,1)=
                                        2
Scalar_SL3.n1,h(3,1)=
                                        2
Possible Exceptions:     {[2*nN1, q-1]}
Scalar_SL3.n1,h(4,1)=
                                        0
Possible Exceptions:     {[2*nN1, q-1]}
Scalar_SL3.n1,h(5,1)=
                                        0
Possible Exceptions:     {[mM1-nN1, q-1], [mM1, q-1],
[nN1+mM1, q-1], [-nN1+2*mM1, q-1], [mM1-2*nN1, q-1], [nN1, q-1]}
Scalar_SL3.n1,h(6,1)=
                                        0
Possible Exceptions:     {[nN1, q-1]}
Scalar_SL3.n1,h(7,1)=
                                        0
Possible Exceptions:     {[nN1*q, q^2+q+1], [nN1+nN1*q, q^2+q+1]}
Scalar_SL3.n1,h(8,1)=
                                        0
```

The sum of the degrees of the unipotent constituents of χ_2^2 equals $2q^3 + 2q^2 + 2q + 1$, which is smaller than $\chi_2^2(1) = q^4 + 2q^3 + q^2$. Hence χ_2^2 must have some non-unipotent constituents.

Generically, the inner product of χ_2^2 with the non-unipotent characters equals 0. In order to find the non-unipotent constituents, we specify the character parameters to the values indicated by the *Possible Exceptions* message and calculate inner products again.

We look at the character parameters and their allowed ranges.

```
> PrintCharParam(g,[4..8]);
cht   Parameters   Exceptions
=======================================================
4   [nN = 1 .. q-1]   [[nN, q-1]]
5   [nN = 1 .. q-1]   [[nN, q-1]]
6   [nN = 1 .. q-1, mM = 1 .. q-1]   [[nN-mM, q-1], [nN, q-1],
    [mM, q-1]]
7   [nN = 1 .. q^2-1]   [[nN, q+1]]
8   [nN = 1 .. q^2+q+1]   [[nN, q^2+q+1]]
```

We see, for example, that the parameter n of the fourth character type may take any value between 1 and $q-1$, which is not divisible by $q-1$. The inner product of $\chi_4(n)$ with χ_2^2 is 0 except possibly if $q-1$ divides $2n$. The only possible exception here is $n=(q-1)/2$, which only can occur if q is odd.

We therefore distinguish the cases of even and odd q. Let us demonstrate the calculations in the case that q is even. Then we only have to re-calculate the scalar products of χ_2^2 with $\chi_6(n,m)$ and $\chi_7(n)$. We first have to tell the programs about these restriction on the congruence class of q. The built in procedures which set or unset the congruence classes are as follows.

```
> nr := g[-2,0];
                                    nr := A2202
> print(setCongruence.nr);
proc() global qQ,q; qQ := 'qQ'; q := qQ; NULL end
> print(unsetCongruence.nr);
proc() global q,qQ; q := 'q'; qQ := q; NULL end
```

In other words, they do nothing (but renaming the parameters). This is reasonable since in general there are no restrictions on the congruence class of q modulo 2. We therefore have to write new procedures (if we do not know how to do this, we can use those of $\mathrm{SL}_2(q)$).

```
> setCongruenceA2202 :=
> proc() global qQ,q; qQ := 'qQ'; q := 2*qQ; NULL end:
> unsetCongruenceA2202 :=
> proc() global q,qQ; q := 'q'; qQ := 1/2*q; NULL end:
```

The specializations are best performed on a copy of the original generic character table `g`.

```
> Copy( g, new );
> SpecCharParam( new, 6, mM = -nN + (q - 1)*xX );
Substituted
      {mM = -nN+(q-1)*xX}
in character type 6.
> Scalar( new, 6 , h );
Possible Exceptions:     {[3*nN1, q-1], [2*nN1, q-1]}
Scalar_new,h(6,1)=
                                                    1
```

Since q is even and not congruent to 1 modulo 3, the possible exceptions `[3*nN1, q-1]`, `[2*nN1, q-1]` for the parameter n_1 printed by the program do not occur in the range of n_1. Substituting the parameter n of χ_6 by a number which is congruent to $2m$ (or m by a number congruent to $2n$) modulo $q-1$ again gives exactly the same character as the one just considered, so we do not have to repeat the calculation.

All remaining exceptions lead to parameter combinations which do not give irreducible characters (see the output of `PrintCharParam(g)` above).

```
> Copy( g, new );
> SpecCharParam( new, 7, nN = (q - 1)*xX );
Substituted
      {nN = (q-1)*xX}
in character type 7.
> Scalar( new, 7 , h );
Scalar_new,h(7,1)=
                                       1
```

We get the result

$$\chi_2^2 = \chi_1 + 2\chi_2 + 2\chi_3 + \frac{1}{2}\sum_{n=1}^{q-2}\chi_6(n, q-1-n) + \frac{1}{2}\sum_{n=1}^{q}\chi_7(n(q-1)).$$

The factors 1/2 above are due to the fact that each irreducible character in the range of the sums occurs twice.

With exactly the same methods we find for odd q

$$\begin{aligned}\chi_2^2 &= \chi_1 + 2\chi_2 + 2\chi_3 + \chi_5((q-1)/2)\\ &+\tfrac{1}{2}\left(\sum_{n=1}^{(q-3)/2}\chi_6(n, q-1-n) + \sum_{n=(q+1)/2}^{q-2}\chi_6(n, q-1-n)\right)\\ &+\tfrac{1}{2}\left(\sum_{n=1}^{(q-1)/2}\chi_7(n(q-1)) + \sum_{n=(q+3)/2}^{q}\chi_7(n(q-1))\right).\end{aligned}$$

We conclude this paragraph with a curious fact first noticed (by us) when playing around with CHEVIE. Let $G = SL_2(q)$, with even q. Then the square of the Steinberg character χ of G is exactly the sum of all irreducible characters of G, each occuring with multiplicity 1. We do not know of any other finite group G having an irreducible character with this property.

5.4. **Applications of multiplication constants.** We conclude by giving two examples of applications of the generic calculation of structure constants. Let G be a finite group, C_1, C_2, C_3 three conjugacy classes of G. Then the structure constant m_{C_1,C_2,C_3} counts the number of triples $(\sigma_1, \sigma_2, \sigma_3)$ of elements with $\sigma_i \in C_i$ and $\sigma_1\sigma_2\sigma_3 = 1$. The knowledge of structure constants is important for example if one wants to prove particular types of generation of G, say by a rigid class triple in the sense of constructive Galois theory, or by a Hurwitz triple.

It is well known that there exists an easy character theoretic formula for the evaluation of m_{C_1,C_2,C_3}. This is implemented in CHEVIE for generic character tables. We first look at groups $\mathbb{G}$ of type ${}^2G_2$. Using the commands `PrintInfoTab`, which shows among other things the references to the literature, and `PrintInfoClass` we can find out that the classes we are interested in are lying in class types 8, 2 and 12.

```
> GenCharTab(`2G2`);
```

```
  # output omitted
> x:=ClassMult(g,8,2,12);
                                              2           1/2
                                    x := [q  + 1 - q 3    , {}]
> factor(x[1]/CentOrd(g,12));
                                                 1
```

The preceding calculation shows that the normalized structure constant of $G = {}^2G_2(q^2)$ for the class triple (C_8, C_2, C_{12}) equals 1. If one can prove moreover that any triple $(\sigma_1, \sigma_2, \sigma_3)$ as above even generates G, then this class triple is rigid in the sense of constructive Galois theory. The latter was verified in [34], Theorem 2.1 (our class triple corresponds to the class vector $\mathbf{C}_2$ in loc. cit.).

Continuing the above example, we calculate:

```
> x:=ClassMult(g,8,3,12)[1]/CentOrd(g,12);
                                               4    2           1/2
                                          1/2 q  - q  + 1/2 q 3
                                  x := ------------------------
                                                2          1/2
                                               q  + 1 - q 3
> x:=ClassMult(g,8,3,13)[1]/CentOrd(g,13);
                                               4       2
                                          1/2 q  + 1/2 q
                                       x := ---------------
                                                   2
                                                  q  + 1

> x:=ClassMult(g,8,3,14)[1]/CentOrd(g,14);
                                               4    2           1/2
                                          1/2 q  - q  - 1/2 q 3
                                  x := ------------------------
                                                2          1/2
                                               q  + 1 + q 3
```

Here the class C_8 contains involutions and the class C_3 contains elements of order 3. Moreover, writing $q^2 = 3^{2n+1}$, if $n \equiv 2, 3 \pmod 6$, C_{12} represents a class of rational elements of order 7, if $n \equiv 1, 4 \pmod 6$, C_{13} represents such a class, and if $n \equiv 0, 5 \pmod 6$, C_{14} represents a class of 7-elements. Thus for every congruence of $n > 0$, the above yields a $(2, 3, 7)$-structure constant different from 0. A look at the maximal subgroups of G shows that in fact among these $(2, 3, 7)$-triples there must be generating ones, proving that G is a Hurwitz group for $n > 0$ (see [36], Proposition 5, for the details).

6. Cyclotomic Algebras and Generic Blocks

The procedures and data files described in this section are available from CHEVIE Version 3.0 on.

6.1. Generic Blocks.

Let $\mathbb{G}(q)$ be a finite group of Lie type in characteristic p and ℓ a prime different from p. Then the distribution of the ordinary unipotent characters of $\mathbb{G}(q)$ into ℓ-blocks can conveniently be described by a generalization of Harish-Chandra series, see [5]. We give a short description of this connection. The order of the generic finite reductive group $\mathbb{G}$ may be written as a polynomial in x, whose irreducible factors are either equal to x or cyclotomic polynomials $\Phi_d(x)$:

$$|\mathbb{G}| = x^N \prod_d \Phi_d(x)^{a(d)} .$$

A generic subtorus $\mathbb{T}$ of $\mathbb{G}$ is called a *Φ_d-torus* if its generic order is a power of $\Phi_d(x)$. A Levi subgroup $\mathbb{L}$ of $\mathbb{G}$ is called *d-split* if it is the centralizer of a Φ_d-torus. Let $\mathcal{E}(\mathbb{G})$ be the set of (labels of) generic unipotent characters of $\mathbb{G}$ as introduced in Section 4.3. Then $\boldsymbol{\gamma} \in \mathcal{E}(\mathbb{G})$ is called *d-cuspidal* if $|\mathbb{G}|/(|Z(\mathbb{G})|\boldsymbol{\gamma}(1))$ is not divisible by $\Phi_d(x)$ as a polynomial in x, where $\boldsymbol{\gamma}(1)$ denotes the generic degree of the generic unipotent character $\boldsymbol{\gamma}$. A pair $(\mathbb{L}, \boldsymbol{\lambda})$ consisting of a d-split Levi subgroup $\mathbb{L}$ of $\mathbb{G}$ and a d-cuspidal unipotent character $\boldsymbol{\gamma} \in \mathcal{E}(\mathbb{L})$ is called a *d-cuspidal pair*.

Now suppose that ℓ divides $\Phi_d(q)$ but no $\Phi_e(q)$ for other cyclotomic polynomials $\Phi_e(x)$ occurring in the order polynomial of $\mathbb{G}$. By the results of [5], 5.24, there is a natural bijection between the unipotent ℓ-blocks of $\mathbb{G}(q)$ and the d-cuspidal pairs $(\mathbb{L}, \boldsymbol{\lambda})$ of $\mathbb{G}$ up to W-conjugation. Moreover, the unipotent characters in the ℓ-block parameterized by $(\mathbb{L}, \boldsymbol{\lambda})$ are in bijection with the irreducible characters of the cyclotomic Weyl group $W_{\mathbb{G}}(\mathbb{L}, \boldsymbol{\lambda})$, and this bijection maps induction in $W_{\mathbb{G}}(\mathbb{L}, \boldsymbol{\lambda})$ to the $R_{\mathbb{L}}^{\mathbb{G}}$-functor between the corresponding sets of generic unipotent characters (see [5], 3.2). The set of constituents $\mathcal{E}(\mathbb{G}, (\mathbb{L}, \boldsymbol{\lambda}))$ of $R_{\mathbb{L}}^{\mathbb{G}}(\boldsymbol{\lambda})$, for $(\mathbb{L}, \boldsymbol{\lambda})$ d-cuspidal, is called a d-Harish-Chandra series. Hence in fact ℓ-blocks (restricted to unipotent characters) coincide with d-Harish-Chandra series.

Conjecturally this should be explained by cyclotomic algebras $\mathcal{H}(W_{\mathbb{G}}(\mathbb{L}, \boldsymbol{\lambda}))$ as introduced in [4].

6.2. **Cyclotomic Weyl Groups and Cyclotomic Algebras.** The cyclotomic Weyl groups $W_{\mathbb{G}}(\mathbb{L}, \boldsymbol{\lambda})$ occurring in the description of generic blocks of generic finite reductive groups all are complex reflection groups. A special case is given by the real reflection groups or finite Coxeter groups which occur in the cases $d = 1, 2$. The CHEVIE-system also implements properties of these complex reflection groups as well as of their associated cyclotomic algebras. As for the real reflection groups, this is contained in the GAP-part of CHEVIE. The basic data type in connection with cyclotomic algebras is the character table record. It contains a generic character table of the cyclotomic algebra and additional data like, for example, Schur elements (see [18] for a definition). They are accessed via functions `HeckeCharTable`W, where W is the name of a primitive complex reflection group according to the notation of Shephard and Todd [45]. These take as input a list of parameters, and return the character table record of the corresponding generic cyclotomic algebra $\mathcal{H}(W)$ where the indeterminates have been specialized to the prescribed values. As in the case of Iwahori-Hecke algebras, the user has to take care of the right choice of parameters: In most cases the arguments to `HeckeCharTable`W must be a list of suitable roots of the parameters for the cyclotomic algebra, and these have to be defined over suitable cyclotomic extensions of the rationals. As for a number of types the complete table is not yet known, the function then just returns the ordinary character table of W. Unfortunately, the theory of cyclotomic algebras is not as nice as in the special case of real reflection groups, so not all procedures for Weyl groups and Iwahori-Hecke algebras make sense for these new data types. In particular, at present there is no well defined concept of character table for a generic cyclotomic algebra, since there are several possible choices for the basis elements.

The use of this data type is best illustrated on an example. Let $\mathbb{G}$ be a simple group of type E_8. We want to study the unipotent ℓ-blocks for primes $\ell > 2$ dividing $\Phi_8(q)$. From [5], Table 3, we find that the principal block B_0 belongs to the 8-cuspidal pair $(\mathbb{L}_0, 1)$, where $\mathbb{L}_0$ is a torus of generic order $\Phi_8(x)^2$ and 1 is the unique unipotent character of this torus. In the notation introduced above, the unipotent part of B_0 is just $\mathcal{E}(\mathbb{G}, (\mathbb{L}_0, 1))$. Further, by [5], Table 1, there exist 6 more blocks $B_1, \dots, B_6$ of positive defect parameterized by the 8-cuspidal characters of the Levi subgroup $\mathbb{L}_1 = \Phi_8 \cdot {}^2D_4$, while all other unipotent characters lie in blocks of defect zero.

The cyclotomic Weyl group $W_{\mathbb{G}}(\mathbb{L}_0, 1)$ for the principal block B_0 is the primitive complex reflection group G_9 (in the notation of [45]). The character table of the corresponding cyclotomic algebra $\mathcal{H}$ is contained in the CHEVIE-system. The corresponding specialization parameters for $\mathcal{H}$ in this situation are found in [39], Table 5.3:

```
gap> v:= X(Cyclotomics); v.name:="v";
gap> B0:= HeckeCharTableG9([1,v,v^2,v^3,1,v^4]);;
```

Here v denotes a square root of q. We may now determine the degrees of the unipotent characters in B_0. This is done by dividing the index of $\mathbb{L}_0$ in $\mathbb{G}$ by the Schur elements of $\mathcal{H}$. Calculations with polynomials are best performed in Maple, so we first print out the Schur elements and enter them into Maple.

```
gap> B0.schurelements;
```

$\vdots$

Now leave `chevieg` and start `cheviem`:

```
> v:=sqrt(q);
> schurelements:= ... # input the Schur elements from GAP
> ind:= (q^2-1)*(q^8-1)*(q^12-1)*(q^14-1)*(q^18-1)*(q^20-1)
       *(q^24-1)*(q^30-1)/(q^4+1)^2;   # the p-prime part of the index
> for i to 32 do degrees[i]:= simplify(ind/schurelements[i]); od;

                                                     120
                                  degrees[1] := q

                              4     2          12     10     8     6     4     2
   degrees[2] := - 1/3 (q   - q   + 1) (q    + q    + q   + q   + q   + q   + 1)

         12     6          16     12     8     4          12     10     6     2
       (q    + q   + 1) (q    + q    + q   + q   + 1) (q    - q    + q   - q   + 1)

          24     18     12     6         32
        (q    + q    + q    + q   + 1) q
```

$\vdots$

Via the degrees, we may identify the characters in B_0 with the notation of Lusztig. According to Section 6.1, the decomposition of $R_{\mathbb{L}_0}^{\mathbb{G}}(1)$ is now given by:

```
> for i to 32 do RLG[i]:=
  sign(lcoeff(degrees[i]))*simplify(subs(q=RootOf(X^4+1),degrees[i]));
  od;

                                     RLG[1] := 1
```

```
                                RLG[2] := -1
```

⋮

Similarly, the character degrees in the other non-trivial unipotent ℓ-blocks $B_1, \ldots, B_6$ with cyclotomic Weyl group Z_8 may be obtained using the generic procedure `HeckeCharTableCyclic` where the parameters in this case can be found in [4], Table 8.1.

Next we want to determine the decomposition of $R^{\mathbb{G}}_{\mathbb{L}_1}(\boldsymbol{\mu})$ for those unipotent characters $\boldsymbol{\mu}$ of the 8-split Levi subgroup $\mathbb{L}_1$ which are not 8-cuspidal. These characters lie in the ℓ-block b_0 of $\mathbb{L}_1$ parameterized by the 8-cuspidal pair $(\mathbb{L}_0, 1)$ in $\mathbb{L}_1$. As above, we first identify the characters in $\mathcal{E}(\mathbb{L}_1, (\mathbb{L}_0, 1))$ with irreducible characters of the cyclotomic Weyl group $W_{\mathbb{L}_1}(\mathbb{L}_0, 1) \cong Z_4$, using `chevieg`:

```
gap> q:= X(Rationals); q.name:="q";
gap> b0:= HeckeCharTableCyclic(4,[1,q^2,q^4,q^6]);;
gap> DisplayCharTable(b0);
H(Z(4))
           2  2   2    2    2
             1a  4a   2a   4b

phi_{1,0}     1   1    1    1
phi_{1,1}     1 q^2  q^4  q^6
phi_{1,2}     1 q^4  q^8 q^12
phi_{1,3}     1 q^6 q^12 q^18

gap> b0.schurelements;
[ -1 + q^(-2) + q^(-4) - q^(-8) - q^(-10) + q^(-12),
  q^2 - 2 + 2*q^(-4) - q^(-6), -q^6 + 2*q^4 - 2 + q^(-2),
  q^12 - q^10 - q^8 + q^4 + q^2 - 1 ]
```

and `cheviem`:

```
> ind:=(q^2-1)*(q^4-1)*(q^6-1);
> for i to 4 do degrees[i]:= simplify(ind/schurelements[i]); od;

                                             12
                             degrees[1] := - q

                                        6    4    2
                        degrees[2] := q  (q  + q  + 1)

                                          2    4    2
                       degrees[3] := - q  (q  + q  + 1)

                               degrees[4] := 1
```

Let $\boldsymbol{\mu} \in \mathcal{E}(\mathbb{L}_1, (\mathbb{L}_0, 1))$ denote the unipotent character of $\mathbb{L}_1$ of generic degree $x^6(x^4 + x^2 + 1)$. Then according to [5], 3.2, the decomposition of $R^{\mathbb{G}}_{\mathbb{L}_1}(\boldsymbol{\mu})$ is given, up to sign, by the decomposition of the induced of the second irreducible character of $W_{\mathbb{L}_1}(\mathbb{L}_0, 1)$ to $W_{\mathbb{G}}(\mathbb{L}_0, 1)$. We first indicate how to find the fusion.

```
gap> WG:= HeckeCharTableG9(1);;
gap> WL1:= HeckeCharTableCyclic(4,1);;
gap> SubgroupFusions(WL1,WG);
[ [ 1, 13, 2, 13 ], [ 1, 9, 3, 8 ], [ 1, 11, 3, 12 ], [ 1, 8, 3, 9 ],
  [ 1, 12, 3, 11 ], [ 1, 7, 2, 6 ], [ 1, 6, 2, 7 ], [ 1, 10, 2, 10 ] ]
gap> WL1.reflclasses;
[ 2 ]
gap> WG.reflclasses;
[ 8, 4 ]
```

The character table of the cyclotomic Weyl group itself is obtained with the parameter 1. Possible fusions are determined by the GAP-command `SubgroupFusions`. The list of classes of generating reflections of the cyclotomic Weyl group is stored in the record-component `reflclasses`. Since generating reflections have to fuse to generating reflections, the fourth of the above possible fusions is the right one. Now one can proceed as in Section 2.4 to find the decomposition of the induced of the second character of $W_{\mathbb{L}_1}(\mathbb{L}_0, 1)$. The necessary signs to adjust the multiplicities obtained were already computed above:

```
gap> tr:= Induced(WL1, WG, [WL1.irreducibles[2]], [ 1, 8, 3, 9 ]);
[ [ 48, 0, -8, 0, 0, 0, 0, 8*E(4), -8*E(4), 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ] ]
gap> dec:= Decompose(WG, tr);
[ [ 0 ], [ 0 ], [ 1 ], [ 1 ], [ 0 ], [ 0 ], [ 0 ], [ 0 ], [ 1 ], [ 1 ],
  [ 0 ], [ 0 ], [ 0 ], [ 0 ], [ 1 ], [ 1 ], [ 1 ], [ 1 ], [ 0 ], [ 0 ],
  [ 1 ], [ 1 ], [ 1 ], [ 1 ], [ 0 ], [ 0 ], [ 1 ], [ 1 ], [ 1 ], [ 1 ],
  [ 1 ], [ 1 ] ]
gap> q:=v^2;
gap> ind:= (q^2-1)*(q^8-1)*(q^12-1)*(q^14-1)*(q^18-1)*(q^20-1)
        *(q^24-1)*(q^30-1)/(q^4+1)^2;   #  the p-prime part of the index
gap> degrees:= List(B0.schurelements, i -> ind/i);;
gap> rlg:= List([1..32],
            i -> dec[i] * SignInt(LeadingCoefficient(degrees[i])));
[ [ 0 ], [ 0 ], [ -1 ], [ -1 ], [ 0 ], [ 0 ], [ 0 ], [ 0 ], [ -1 ],
  [ -1 ], [ 0 ], [ 0 ], [ 0 ], [ 0 ], [ -1 ], [ -1 ], [ -1 ], [ 1 ],
  [ 0 ], [ 0 ], [ 1 ], [ 1 ], [ -1 ], [ 1 ], [ 0 ], [ 0 ], [ 1 ],
  [ 1 ], [ -1 ], [ -1 ], [ 1 ], [ 1 ] ]
```

This shows in particular that $R^{\mathbb{G}}_{\mathbb{L}_1}(\boldsymbol{\mu})$ is multiplicity-free.

6.3. **Fake Degrees and Principal Series Degrees.** The `chevieg`-package also contains a routine for the computation of fake degrees of a complex reflection group given by its character table. Let's look at the example of the group G_8:

```
gap> x:= X(Cyclotomics); x.name:="x";
gap> W:= HeckeCharTableG8(1);;
gap> FakeDegrees(W, W.reflchar, x);
[ x^0, x^6, x^12, x^18, x^5 + x, x^8 + x^4, x^11 + x^7, x^11 + x^7,
  x^14 + x^10, x^17 + x^13, x^10 + x^6 + x^2, x^12 + x^8 + x^4,
  x^14 + x^10 + x^6, x^16 + x^12 + x^8, x^15 + x^11 + x^7 + x^3,
  x^13 + 2*x^9 + x^5 ]
```

The reflection character of W is stored in the component `W.reflchar`.

Conjecturally, many complex reflection groups possess a set of associated unipotent degrees sharing many properties with the degrees of unipotent characters of generic finite reductive groups (see for example [38]). Those degrees lying in the principal 1-series can be computed directly from the Schur elements with the function `PrincipalSeriesDegrees`:

```
gap> v:= X(Cyclotomics); v.name:="v";
gap> G8:= HeckeCharTableG8(v);;
gap> pdegrees:= PrincipalSeriesDegrees(G8, v^2);
[ v^0, (-1/12)*v^36 + (-1/4*E(4))*v^34 + (1/4)*v^32 + (1/12)*v^28 +
    (-1/4*E(4))*v^26 + (1/2)*v^24 + (1/4*E(4))*v^22 + (1/12)*v^20 +
    (1/4)*v^16 + (1/4*E(4))*v^14 + (-1/12)*v^12, (1/4)*v^36 +
    (-1/4)*v^34 + (1/4)*v^32 + (1/4)*v^28 + (-1/4)*v^26 + (1/2)*v^24
    + (-1/4)*v^22 + (1/4)*v^20 + (1/4)*v^16 + (-1/4)*v^14 + (1/4)*v^12,
.
.
.
```

Here v is a square root of q.

Acknowledgements. We wish to thank Jean Michel for much help in developing a proper conceptual framework for the original Weyl group programs, and for many extremely useful contributions of new programs, in particular those on Weyl subgroups.

References

[1] Alvis, D., Lusztig, G.: The representations and generic degrees of the Hecke algebra of type H_4. J. reine angew. Math. **336**, 201–212 (1982); Correction: ibid. **449**, 217–218 (1994)

[2] Benson, C.T., Curtis, C.W.: On the degrees and rationality of certain characters of finite Chevalley groups. Trans. Amer. Math. Soc. **165**, 251–273 (1972)

[3] Broué, M., Malle, G.: Théorèmes de Sylow génériques pour les groupes réductifs sur les corps finis. Math. Ann. **292**, 241–262 (1992)

[4] ______: Zyklotomische Heckealgebren. In: Représentations unipotentes génériques et blocs des groupes réductifs finis, Astérisque, vol. 212, Société Mathématique de France, pp. 119–189 (1993)

[5] Broué, M., Malle, G., Michel, J.: Generic blocks of finite reductive groups. In: Représentations unipotentes génériques et blocs des groupes réductifs finis, Astérisque, vol. 212, Société Mathématique de France, pp. 7–92 (1993)

[6] Carter, R.W.: Conjugacy classes in the Weyl group. Compositio Math. **25**, 1–59 (1972)

[7] ______: Centralizers of semisimple elements in finite groups of Lie type. Proc. London Math. Soc. **37**, 491–507 (1978)

[8] ______: Finite groups of Lie type: Conjugacy classes and complex characters. New York: Wiley 1985

[9] Chang, B., Ree, R.: The characters of $G_2(q)$. Symposia Mathematica **XIII**, 395–413 (1974)

[10] Char, B.W., Geddes, K.O., Gonnet, G.H., Leong, B.L., Monagan, M.B., Watt, S.M.: Maple V, Language Reference Manual. Springer 1991

[11] Conway, J.H., Curtis, R.T., Norton, S.P., Parker, R.A., Wilson, R.A.: Atlas of Finite Groups. London: Oxford University Press 1984

[12] Curtis, C.W., Reiner, I.: Methods of representation theory, vol. I, II. New York: Wiley 1981/1987

[13] Deligne, P., Lusztig, G.: Representations of reductive groups over finite fields. Annals of Math. **103**, 103–161 (1976)

[14] Deriziotis, D.I.: Conjugacy classes and centralizers of semisimple elements in finite groups of Lie type. Vorlesungen aus dem Fachbereich Mathematik der Universität Essen, Heft 11, Germany (1984)

[15] Digne, F., Michel, J.: Representations of finite groups of Lie type. Cambridge: Cambridge Univ. Press 1991

[16] DuCloux, F.: Coxeter Version 1.0. Université de Lyon, France (1991)

[17] Geck, M.: Eine Anwendung von MAPLE in der Darstellungstheorie der unitären Gruppen. Diplomarbeit, Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany (1988)

[18] ———: Beiträge zur Darstellungstheorie von Iwahori–Hecke Algebren. Habilitationsschrift, Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany (1993)

[19] ———: On the character values of Iwahori-Hecke algebras of exceptional type. Proc. London Math. Soc. **68**, 51–76 (1994)

[20] Geck, M., Hiss, G., Lübeck, F., Malle, G., Pfeiffer, G.: CHEVIE – Generic Character Tables of Finite Groups of Lie Type, Hecke Algebras and Weyl Groups. Preprint 93-62, Interdisziplinäres Zentrum für wissenschaftliches Rechnen der Universität Heidelberg, Germany (1993)

[21] Geck, M., Michel, J.: On "good" elements in the conjugacy classes of finite Coxeter groups and their eigenvalues on the irreducible representations of Iwahori-Hecke algebras. *Submitted to:* Proc. London Math. Soc.

[22] Geck, M., Pfeiffer, G.: On the irreducible characters of Hecke algebras. Adv. in Math. **102**, 79–94 (1993)

[23] Green, J.A.: The characters of the finite general linear groups. Trans. Amer. Math. Soc. **80**, 402–447 (1955)

[24] Halverson, T., Ram, A.: Murnaghan-Nakayama rules for characters of Iwahori-Hecke algebras of classical type. Preprint (1994)

[25] Hiss, G.: On the decomposition numbers of $G_2(q)$. J. Algebra **120**, 339–360 (1989)

[26] Humphreys, J.E.: Reflections groups and Coxeter groups. Cambridge studies in advanced mathematics, vol. 29, Cambridge Univ. Press 1990

[27] Jones, V.F.R.: Hecke algebra representations of braid groups and link polynomials. Annals of Math. **126**, 335–388 (1987)

[28] Kazhdan, D., Lusztig, G.: Representations of Coxeter groups and Hecke algebras. Invent. Math. **53**, 165–184 (1979)

[29] Lübeck, F.: Charaktertafeln für die Gruppen $\mathrm{CSp}_6(q)$ mit ungeradem q und $\mathrm{Sp}_6(q)$ mit geradem q. Dissertation, Universität Heidelberg, Germany (1993)

[30] Lusztig, G.: On a theorem of Benson and Curtis. J. Algebra **71**, 490–498 (1981)

[31] ———: Characters of reductive groups over a finite field. Annals of Mathematical Studies, vol. 107, Princeton University Press 1985

[32] ———: On the representations of reductive groups with disconnected centre. In: Orbites Unipotentes et Représentationes, I. Groupes finis et Algèbres de Hecke, Astérisque vol. 168, Société Mathématique de France, pp. 157–166 (1988)

[33] ———: Remarks on computing irreducible characters. J. Amer. Math. Soc. **5**, 971–986 (1992)

[34] Malle, G.: Exceptional groups of Lie type as Galois groups. J. reine angew. Math. **392**, 70–109 (1988)

[35] ———: Die unipotenten Charaktere von ${}^2F_4(q^2)$. Comm. Algebra **18**, 2361–2381 (1990)

[36] ———: Hurwitz groups and $G_2(q)$. Canad. Math. Bull. **33**, 349–357 (1990)

[37] ———: Generalized Deligne–Lusztig characters. J. Algebra **159**, 64–97 (1993)

[38] ———: Unipotente Grade imprimitiver komplexer Spiegelungsgruppen. J. Algebra, to appear (1994)

[39] ———: Degrés relatifs des algèbres cyclotomiques associées aux groupes de réflexions complexes de dimension deux. Submitted (1994)

[40] Pfeiffer, G.: Young characters on Coxeter basis elements of Iwahori–Hecke algebras and a Murnaghan–Nakayama formula. J. Algebra **168**, 525–535 (1994)

[41] ———: Character values of Iwahori-Hecke algebras of type B. To appear (1994)

[42] Ram, A.: A Frobenius formula for the characters of the Hecke algebras. Invent. Math. **106**, 461–488 (1991)

[43] Schönert, M., et al.: GAP – Groups, Algorithms, and Programming. Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany, fourth ed., (1994)

[44] Schur, I.: Untersuchungen über die Darstellung der endlichen Gruppen durch gebrochene lineare Substitutionen. J. reine angew. Math. **132**, 85–137 (1907)

[45] Shephard, G.C., Todd, J.A.: Finite unitary reflection groups. Canad. J. Math. **6**, 274–304 (1954)

[46] Springer, T.A.: Linear algebraic groups. Boston: Birkhäuser 1981

[47] Srinivasan, B.: The characters of the finite symplectic group $Sp(4,q)$. Trans. Amer. Math. Soc. **131**, 488–525 (1968)

M.G.: Lehrstuhl D für Mathematik, RWTH Aachen, D–52062 Aachen, Germany

G.H.,F.L.,G.M.: IWR der Universität Heidelberg, Im Neuenheimer Feld 368, D–69120 Heidelberg, Germany

G.P.: Department of Mathematics, University of St. Andrews, North Haugh, St. Andrews, Fife KY16 9SS, Scotland