PEAKWORD CONDENSATION AND SUBMODULE LATTICES: AN APPLICATION OF THE MEAT-AXE

KLAUS LUX, JÜRGEN MÜLLER, MICHAEL RINGE

Abstract. We describe a new condensation method for computing the submodule lattice of a module for a finite dimensional algebra over a finite field, which exploits the idea of condensation and extends it to the case of primitive idempotents. The method has been implemented in the new C version of the Meat-Axe developed at Aachen, and we give several examples which have been analysed with our method.

Introduction

In this note we describe a new method for computing the submodule lattice of a module for a finite dimensional algebra over a finite field. We felt the need for such an algorithm while working on the problem of determining decomposition numbers for finite sporadic simple groups. Our plan was to analyze permutation modules for these groups along the lines of [14]. One of the major tasks was to acquire a detailed knowledge of the submodule structure of these modules. Thus we asked ourselves whether it might be possible to use the idea of condensation, see [14], to compute submodule lattices. Since the dimension of a module is reduced considerably under condensation, it is much easier to analyze the condensed module than the original one. This approach is completely different from the one described in [15], which is based on the computation of endomorphism rings.

We make use of a theorem of D. Benson and J. Conway, which shows how to find the whole submodule lattice given enough information about the incidence structure of the set of join-irreducible elements, see Section 1. In fact, we are able to prove a version of the Benson-Conway Theorem requiring weaker assumptions. No matrix operations are involved in applying the Benson-Conway Theorem. We will show that the set of all join-irreducible elements of the submodule lattice can be put into bijection with sets of submodules of a set of condensed modules, see Section 2. This is the point where condensation comes into play.

If V is an A-module for some algebra A and e is an idempotent, then the connections between the submodule structures of the condensed module Ve, which is an eAe-module, and the A-module V are well-known. This dates back at least to [6]. In theory, it would be best to take e as a sum of pairwise orthogonal idempotents such that there is a bijection between the summands and the set of isomorphism types of irreducible A-modules. But it turns out that it is far more practical to use a set of not necessarily orthogonal primitive idempotents, one for each irreducible A-module. We will show how to exploit Meat-Axe techniques to compute the action of eAe on Ve, provided the primitive idempotent e is suitably chosen. The choice we make is described in Section 3.

1991 *Mathematics Subject Classification.* 06C05, 15-04, 16G10, 20C40.

In Section 4 we present an algorithmic description of our method showing which matrix operations are involved. The reader who is familiar with the MEAT-AXE, see [12], will readily recognize that all the steps can be done by standard MEAT-AXE programs. Our method has been implemented by one of us (Ringe) in the new C version of the MEAT-AXE, see [13], developed at Aachen. Of course, the C MEAT-AXE and the lattice package are available on request, our e-mail addresses are given at the end of this note. The programs are also contained in the shared library of the Aachen computer algebra system GAP, see [17].

Our aim in designing the programs was to automize the process as much as possible. Given a module V, i. e., a finite set of matrices acting on their standard row vector space, the first step of our method is to compute a list of all constituents of the given module and their multiplicities. This has always been *the* standard task for the MEAT-AXE; now there is a single program which performs this task. Our method then provides an efficient way to use the MEAT-AXE to obtain a complete overview on the module structure. Next, the programs compute a list of all join-irreducible submodules of V and the incidences between them. This information is used to compute a list of all submodules, the incidences between them, and the isomorphism types of the irreducible subquotients. Furthermore the radical and socle series as lists of submodules and a lattice-direct decomposition, if one exists, are computed. Further details of the implementation are given in Section 5.

We conclude this note with a few examples which have been examined using our method, see Section 6. We hope these are suitable to show that it is possible to use our method to obtain valuable information in several areas of computational representation theory.

Throughout the following text let F be a field and A be a finite-dimensional algebra over F. Every module considered in the sequel will be a unital right module of finite dimension unless something else is explicitly stated. For the theory developed in Sections 1, 2, and 3 no finiteness condition on F is necessary. As soon as we are going to describe the algorithm in detail, i. e., from Section 4 on, we will assume F to be finite.

ACKNOWLEDGEMENTS

We wish to thank Lehrstuhl D für Mathematik at the RWTH Aachen for providing the necessary computer environment without which we would not have been able to develop and test our ideas. We also want to thank the Deutsche Forschungsgemeinschaft for financial support granted in the framework of the joint research project 'Algorithmic Number Theory and Algebra'.

1. THE BENSON-CONWAY THEOREM

In this section we collect the necessary facts from lattice theory. Since we are interested in submodule lattices, they are stated in terms of submodules of a given module. The notions from lattice theory, which are used here without further explanation, can be found in [1].

1.1. **Notation:** Let V be an A-module and S be a constituent of V. Then the set of submodules of V defines a modular lattice of finite length $\mathcal{M} = \mathcal{M}(V)$ whose partial ordering $\leq$ is given by the natural set theoretic inclusion. $\mathcal{L}_S$ denotes the set of all submodules $W \leq V$ such that $W/\mathrm{Rad}(W) \cong S$ as A-modules. In the

sequel the elements of $\mathcal{L}_S$ will be called *local* submodules with respect to S. The union of the $\mathcal{L}_S$ for all the constituents of V is denoted by $\mathcal{L}$. Hence this is the set of all join-irreducible elements of $\mathcal{M}$. Similarly, $\mathcal{L}_{S\oplus S}$ denotes the set of all submodules $W \leq V$ such that $W/\mathrm{Rad}(W) \cong S \oplus S$ as A-modules. This is exactly the set of submodules which can be written as the sum of two different elements of $\mathcal{L}_S$. Finally, $\mathcal{M}_S$ denotes the set of all submodules $W \leq V$ such that $W/\mathrm{Rad}(W)$ has S as its only constituent. $\mathcal{M}_S$ becomes a lattice by restriction of the partial ordering.

We are now going to state the Benson-Conway Theorem, which shows how to rebuild a given modular lattice from the incidence structure of the subset $\mathcal{L}$. The notions used here are taken from [2].

1.2. **Definition:** A subset $\mathcal{D}$ of $\mathcal{L}$ is called a *dotted-line*, if it contains at least three elements and is maximal subject to the following property: For all pairs X, Y of different elements of $\mathcal{D}$ the equality $X + Y = \sum \mathcal{D}$ holds.

1.3. **Remark:** If X, Y are elements of the dotted-line $\mathcal{D}$, then neither of X, Y is contained in the other. Furthermore, there exists a constituent S such that $X, Y \in \mathcal{L}_S$. This means that $\sum \mathcal{D} \in \mathcal{L}_{S\oplus S}$. Conversely, for each submodule $W \in \mathcal{L}_{S\oplus S}$ there exists a dotted-line $\mathcal{D}$, such that $W = \sum \mathcal{D}$. Finally, there is a bijection from the elements of $\mathcal{D}$ to the set of maximal submodules of $\sum \mathcal{D}$. If F is finite, we therefore have $|\mathcal{D}| = |F|^{[E_S:F]} + 1$, where E_S denotes the endomorphism ring of the A-module S and $|\cdot|$ denotes the cardinality of a set.

1.4. **Definition:** Let $\mathcal{M}(\mathcal{L})$ be the set of all subsets $\mathcal{X}$ of $\mathcal{L}$ with the following properties:
a) For each $X \in \mathcal{X}$, $Y \in \mathcal{L}$, $Y \leq X$ we have $Y \in \mathcal{X}$.
b) For each dotted-line $\mathcal{D}$ and two different submodules $X, Y \in \mathcal{D} \cap \mathcal{X}$ we have $\mathcal{D} \subseteq \mathcal{X}$.

1.5. **Benson-Conway Theorem:** $\mathcal{M}(\mathcal{L})$ is a lattice with respect to the natural partial ordering on the set of all subsets of $\mathcal{L}$, and the mapping $\tau : \mathcal{M} \to \mathcal{M}(\mathcal{L}) : X \mapsto \{Y \in \mathcal{L}; Y \leq X\}$ is an isomorphism of lattices. Its inverse is given by $\tau^{-1} : \mathcal{M}(\mathcal{L}) \to \mathcal{M} : \mathcal{X} \mapsto \sum \mathcal{X}$.

Proof: See [2], Main Theorem.

1.6. **Corollary:** Let $\mathcal{M}(\mathcal{L}_S)$ be the subset of $\mathcal{M}(\mathcal{L})$ consisting of elements of $\mathcal{L}_S$. Then the mapping $\sigma : \mathcal{M}_S \to \mathcal{M}(\mathcal{L}_S) : X \mapsto \{Y \in \mathcal{L}_S; Y \leq X\}$ is an isomorphism of lattices. Its inverse is given by $\sigma^{-1} : \mathcal{M}(\mathcal{L}_S) \to \mathcal{M}_S : \mathcal{X} \mapsto \sum \mathcal{X}$.

As it turns out, not all of the sometimes numerous dotted-lines which can be found in the lattice have to be used to find all submodules. The next theorem shows how to choose a suitable subset of dotted-lines which still suffices to reconstruct the whole submodule lattice, thereby reducing the number of dotted-lines which have to be computed explicitly for a given module and shortening the computations to find $\mathcal{M}(\mathcal{L})$.

1.7. **Theorem:** For all constituents S and all submodules $Z \in \mathcal{L}_{S\oplus S}$ let a dotted-line $\mathcal{D}_Z$ be given, such that $\sum \mathcal{D}_Z = Z$. Let Δ be the set of these dotted-lines. Then $\mathcal{M}(\mathcal{L})$ is the set of all subsets $\mathcal{X}$ of $\mathcal{L}$ which fulfill the following conditions:
a) For each $X \in \mathcal{X}$, $Y \in \mathcal{L}$, $Y \leq X$ we have $Y \in \mathcal{X}$.
b) For each dotted-line $\mathcal{D} \in \Delta$ and two different submodules $X, Y \in \mathcal{D} \cap \mathcal{X}$ we have $\mathcal{D} \subseteq \mathcal{X}$.

Proof: The existence of such a dotted-line $\mathcal{D}_Z$ for each $Z \in \mathcal{L}_{S\oplus S}$ is assured by Remark 1.3. Now let $\mathcal{X}$ be a subset of $\mathcal{L}$ fulfilling the conditions given in the statement. We show $\mathcal{X} \in \mathcal{M}(\mathcal{L})$ by induction on the composition length of $\sum \mathcal{X}$, the assertion being true if $\sum \mathcal{X}$ has length one or two.

Let $X_1, X_2 \in \mathcal{X}$ such that $X_1 + X_2 = X \in \mathcal{L}_{S\oplus S}$. Then we have $\mathrm{Rad}(X) = \mathrm{Rad}(X_1) + \mathrm{Rad}(X_2)$. Let $\mathcal{X}_{\mathrm{Rad}} := \{Y \in \mathcal{X}; Y \leq \mathrm{Rad}(X)\}$. Then $\sum \mathcal{X}_{\mathrm{Rad}} = \mathrm{Rad}(X)$ and $\mathcal{X}_{\mathrm{Rad}}$ also fulfills the conditions of the theorem. Hence we have $\mathcal{X}_{\mathrm{Rad}} \in \mathcal{M}(\mathcal{L})$, by induction.

Now we consider the maximal submodules $Y_1 := X_1 + \mathrm{Rad}(X)$ and $Y_2 := X_2 + \mathrm{Rad}(X)$ of X. Again, the sets $\mathcal{X}_1 := \{Y \in \mathcal{X}; Y \leq Y_1\}$ and $\mathcal{X}_2 := \{Y \in \mathcal{X}; Y \leq Y_2\}$ fulfill the conditions of the theorem and we have $\sum \mathcal{X}_1 = Y_1$, $\sum \mathcal{X}_2 = Y_2$, and $\mathcal{X}_1, \mathcal{X}_2 \in \mathcal{M}(\mathcal{L})$, by induction.

Now let $Z_1, Z_2 \in \mathcal{D}_X \in \Delta$ such that $Z_1 \leq Y_1$ and $Z_2 \leq Y_2$. It follows that $Z_1, Z_2 \in \mathcal{X}$, hence $\mathcal{D}_X \subseteq \mathcal{X}$.

Finally, let $L \leq X$ be an arbitrary local submodule. Then there is a maximal submodule M of X such that $L \leq M$. Letting $\mathcal{X}_M := \{Y \in \mathcal{X}; Y \leq M\}$, we conclude that $\sum \mathcal{X}_M = M$ and $\mathcal{X}_M \in \mathcal{M}(\mathcal{L})$, also by induction. This shows $L \in \mathcal{X}$ and completes the proof.

We conclude this section with a remark, which shows how to use Theorem 1.5 to decompose the submodule lattice of V as a direct sum of lattices, thereby reducing the computation of the whole submodule lattice to the computation of the submodule lattices of the direct summands.

1.8. **Remark:** A subset $\mathcal{L}_1$ of $\mathcal{L}$ is called a *Benson-Conway block*, if $\mathcal{L}_1$ and its complement $\mathcal{L}_2 := \mathcal{L} \setminus \mathcal{L}_1$ are elements of $\mathcal{M}(\mathcal{L})$. Then the submodules which correspond to $\mathcal{L}_1$ and $\mathcal{L}_2$ via Theorem 1.5 are direct summands of V, and each submodule U of V has a unique representation $U = U_1 \oplus U_2$ where $U_1 \leq \sum \mathcal{L}_1$ and $U_2 \leq \sum \mathcal{L}_2$.

2. Condensation with Primitive Idempotents

In Section 1 we have shown how to use the subsets $\mathcal{M}_S$ and $\mathcal{L}_S$ of $\mathcal{M}$ to find the whole submodule lattice of V. The first aim of this section is, given an S-primitive idempotent $e \in A$, i. e., a primitive idempotent, such that $(eA)/\mathrm{Rad}(eA) \cong S$ as A-modules, to obtain a canonical correspondence from the subset $\mathcal{M}_S(V)$ of A-submodules of V to the set $\mathcal{M}(Ve)$ of all eAe-submodules of Ve.

2.1. **Lemma:** Let $e \in A$ be an arbitrary idempotent.
a) Let W be an A-submodule of V. Then We is an eAe-submodule of Ve and we have $(V/W)e \cong Ve/We$ as eAe-modules. Conversely, let $\tilde{W}$ be an eAe-submodule of Ve. Then $W := \tilde{W} \cdot A$ is an A-submodule of V, such that $We = \tilde{W}$.
b) If S is an irreducible A-module, then either $Se = \{0\}$ or Se is an irreducible eAe-module.

Proof: See [6].

2.2. **Lemma:** Let $e \in A$ be an S-primitive idempotent and $\tilde{W}$ be an eAe-module. Furthermore let $W := \tilde{W} \otimes_{eAe} eA$. Then $W/\mathrm{Rad}(W)$ has S as its only constituent.

Proof: Let T be an irreducible A-module. Since eA is an eAe-A-bimodule, using the Adjointness Theorem [3], Theorem 2.19, and the eAe-module isomorphism $\mathrm{Hom}_A(eA, T) \cong Te$ gives $\mathrm{Hom}_{eAe}(\tilde{V}, Te) \cong \mathrm{Hom}_A(\tilde{V} \otimes_{eAe} eA, T)$ as additive groups.

2.3. **Theorem:** Let $e \in A$ be an S-primitive idempotent. Then the mapping $\kappa : \mathcal{M}_S(V) \to \mathcal{M}(Ve) : W \mapsto We$ is an isomorphism of lattices. Its inverse is given by $\kappa^{-1} : \mathcal{M}(Ve) \to \mathcal{M}_S(V) : \tilde{W} \mapsto \tilde{W} \cdot A$.

Proof: Clearly $\tilde{W} \cdot A$ is an epimorphic image of $\tilde{W} \otimes_{eAe} eA$. So using Lemmas 2.1 and 2.2, κ and κ^{-1} are well-defined and $\kappa \circ \kappa^{-1} = \mathrm{id}$. Finally κ is clearly injective, since for all $W_1, W_2 \in \mathcal{M}_S(V)$, $W_1 \neq W_2$ either $(W_1 + W_2)/W_1$ or $(W_1 + W_2)/W_2$ has S as a constituent.

The second aim of this section is to give a characterization of the submodules $\mathcal{L}_S(V)$ of V and $\mathcal{L}(Ve)$ of Ve. For an irreducible A-module S let again E_S denote its endomorphism ring, which is a finite skew field extension of F. Let E_S^* denote its multiplicative group.

2.4. **Theorem:** Let $e \in A$ be an idempotent, such that $(eA)/\mathrm{Rad}(eA)$ has S as its only constituent.
a) $[E_S : F]$ divides $\dim_F(Se)$. Equality holds, if and only if e is S-primitive.
b) If e is S-primitive, then E_S^* acts transitively on $Se \setminus \{0\}$.

Proof: a) Let $e = \sum_{i=1}^r e_i$ be an orthogonal decomposition of e into S-primitive idempotents. Then we have the following isomorphisms of F-vector spaces:

$$Se \cong \mathrm{Hom}_A(eA, S) \cong \oplus_{i=1}^r \mathrm{Hom}_A(e_i A, S) \cong \oplus_{i=1}^r E_S.$$

Hence $\dim_F(Se) = r \cdot [E_S : F]$.
b) Se can be regarded as a one-dimensional E_S-vector space.

2.5. **Theorem:** Let $e \in A$ be an S-primitive idempotent.
a) $\mathcal{L}(Ve)$ is the set of eAe-submodules of Ve which are of the form $v \cdot eAe$ for some $v \in Ve \setminus \{0\}$.
b) $\mathcal{L}_S(V)$ is the set of A-submodules of V which are of the form $v \cdot A$ for some $v \in Ve \setminus \{0\}$.

Proof: a) Let $\tilde{W} := v \cdot eAe$. Choose a direct sum decomposition of $\tilde{W}/\mathrm{Rad}(\tilde{W})$ into irreducible summands and take the corresponding decomposition of $v + \mathrm{Rad}(\tilde{W})$. Then the assertion follows immediately using Theorem 2.4.
b) Let $W \in \mathcal{L}_S(V)$. Take some $v \in We \setminus \mathrm{Rad}(W)e$. Suppose $vA \leq \mathrm{Rad}(W)$. Then we have $v \in vAe \leq \mathrm{Rad}(W)e$, a contradiction. Now let $W := v \cdot A$ for some $v \in Ve \setminus \{0\}$. Then $We \in \mathcal{L}(Ve)$ and $W = We \cdot A \in \mathcal{L}_S(V)$ by Theorem 2.3.

We conclude this section by a short digression giving a generalisation of a theorem due to R. Dilworth, see [4]. For a right A-module V, let V^* denote its dual, which is a left A-module. Then the Dilworth Theorem states that the sets $\mathcal{L}(V)$ and $\mathcal{L}(V^*)$ have the same cardinality $|\mathcal{L}(V)| = |\mathcal{L}(V^*)|$.

2.6. **Theorem:** Let V be an A-module and V^* be its dual. Furthermore, let S be a constituent of V with dual module S^*. Then $|\mathcal{L}_S(V)| = |\mathcal{L}_{S^*}(V^*)|$.

Proof: Let $e \in A$ be an S-primitive idempotent. Then we have $(Ve)^* \cong eV^*$ as left A-modules. By Corollary 1.6 we have $|\mathcal{L}_S(V)| = |\mathcal{L}(Ve)|$ and $|\mathcal{L}_{S^*}(V^*)| = |\mathcal{L}(eV^*)|$. Finally, by the Dilworth Theorem, we have $|\mathcal{L}(Ve)| = |\mathcal{L}((Ve)^*)| = |\mathcal{L}(eV^*)|$.

3. Peakwords

It is clear that it would be very difficult to compute primitive idempotents explicitly. But this is in fact not necessary. It is sufficient to know the action of the idempotent on V. We are now going to define a certain projection on V, which later will be shown to be the action of a corresponding primitive idempotent.

For an A-module V let $\varrho_V : A \to \mathrm{End}_F(V) : a \mapsto a_V$ be the corresponding representation. For $a \in A$ there exists $N \in \mathbb{N}$ such that

$$\{0\} \leq \mathrm{Ker}(a_V) < \mathrm{Ker}(a_V^2) < \ldots < \mathrm{Ker}(a_V^N) = \mathrm{Ker}(a_V^{N+1}).$$

This gives rise to the Fitting decomposition $V = \mathrm{Ker}(a_V^N) \oplus \mathrm{Im}(a_V^N)$ as vector spaces. So the Fitting projection from V onto $\mathrm{Ker}(a_V^N)$ with respect to the complement $\mathrm{Im}(a_V^N)$ is well-defined in $\mathrm{End}_F(V)$.

3.1. **Theorem:** Let V be a faithful A-module and $a \in A$. Then there exists an element $e \in A$ which induces the Fitting projection on V with respect to a. Furthermore, e is uniquely determined and an idempotent.

Proof: Let N be as above, $K := \mathrm{Ker}(a_V^N)$, and $I := \mathrm{Im}(a_V^N)$. Furthermore, let $m_K, m_I \in F[t]$ denote the minimum polynomials of a_V on K and I. Then $m_K = t^N$ and there are $P, Q \in F[t]$ such that $1 = P \cdot m_I + Q \cdot m_K \in F[t]$. Now define $e := P(a) \cdot m_I(a) \in A$.

The main aim of this section is to show that the idempotent e corresponding to the Fitting projection with respect to an element $a \in A$ is indeed a primitive idempotent if a is suitably chosen as a so-called peakword, see Definition 3.4.

3.2. **Theorem:** Let V be a faithful A-module, $a \in A$, and $e \in A$ the corresponding idempotent. Let S be a constituent of V and $S = \mathrm{Ker}(a_S^n) \oplus \mathrm{Im}(a_S^n)$ its Fitting decomposition with respect to a. Then we have $Se = \mathrm{Ker}(a_S^n)$.

Proof: As in the proof of Theorem 3.1 above, let $m_k, m_i \in F[t]$ denote the minimum polynomials of a_S on $k := \mathrm{Ker}(a_S^n)$ and $i := \mathrm{Im}(a_S^n)$. We have $m_k = t^n$. Since S is a subquotient of V, m_k divides m_K and m_i divides m_I. Hence

$$1 = P \cdot \frac{m_I}{m_i} \cdot m_i + Q \cdot \frac{m_K}{m_k} \cdot m_k \in F[t].$$

Now $\mathrm{Ker}(a_S^n) = S(P(a) \cdot \frac{m_I}{m_i}(a) \cdot m_i(a)) = S(P(a) \cdot m_I(a)) = Se$.

3.3. **Theorem:** Let E_S be the endomorphism ring of the constituent S of V and $a \in A$.
a) $[E_S : F]$ divides $\dim_F(\mathrm{Ker}(a_S))$.
b) If $[E_S : F] = \dim_F(\mathrm{Ker}(a_S))$, then E_S^* acts transitively on $\mathrm{Ker}(a_S) \setminus \{0\}$.

Proof: Analogously to the proof of Theorem 2.4.

3.4. **Definition:** Let V be a faithful A-module and S be a constituent of V. An element $a \in A$ is called an *S-peakword*, if the following conditions are fulfilled:
a) $\mathrm{Ker}(a_T) = \{0\}$ for all constituents T of V which are not isomorphic to S.
b) $\dim_F(\mathrm{Ker}(a_S^2)) = [E_S : F]$.

3.5. **Remark:** As is seen from Theorem 2.4 the second condition in the Definition above means, that

$$\{0\} \neq \mathrm{Ker}(a_S) = \mathrm{Ker}(a_S^2) = \mathrm{Ker}(a_S^3) = \dots .$$

Hence the Fitting-decomposition of S with respect to a_S is given by $S = \mathrm{Ker}(a_S) \oplus \mathrm{Im}(a_S)$; so $n = 1$ in the notation of Theorem 3.2.

3.6. **Theorem:** The idempotent $e \in A$ corresponding to an S-peakword $a \in A$ is S-primitive.

Proof: First of all let T be an arbitrary irreducible A-module and $f \in A$ a T-primitive idempotent. Then $Vf \neq \{0\}$, hence T is a constituent of V. Now using Theorems 3.2 and 3.3, we have $\dim_F(Te) = 0$ for all irreducible A-modules T which are not isomorphic to S, and $\dim_F(Se) = [E_S : F]$, hence e is S-primitive by Theorem 2.4.

3.7. **Corollary:** From the regularity condition on a given in Definition 3.4 follows that $(eA)/\mathrm{Rad}(eA)$ has S as its only constituent. Provided the regularity condition on a is fulfilled, then the dimension condition is also fulfilled if and only if E_S^* acts transitively on $\mathrm{Ker}(a_S) \setminus \{0\}$.

4. The Algorithm

In this section we write down the method as a list of steps and show which matrix operations are involved. Details concerning the implementation of several of the steps will be given in Section 5. From now on we assume the field F to be finite.

Let $\mathcal{S}$ denote the standard basis of $V := F^{1\times n}$ and $m(\mathcal{B}, \alpha, \mathcal{C}) \in F^{n\times n}$ denote the matrix of the F-linear mapping α corresponding to the bases $\mathcal{B}$ and $\mathcal{C}$, the mapping being written on the right hand side. Let the matrix algebra A be given as a finite set of algebra generators $\{m(\mathcal{S}, a_i, \mathcal{S})\} \subseteq F^{n\times n}$. Then V is the canonical A-module.

Algorithm:

- Input: $\{m(\mathcal{S}, a_i, \mathcal{S})\} \subseteq F^{n\times n}$.
- For every constituent S of V compute its multiplicity k and a set of matrices $\{m(\mathcal{S}, (a_i)_S, \mathcal{S})\} \subseteq F^{n_S\times n_S}$, see 5.1.
- For each constituent S do:
  - Repeat:
    - Find an element $w \in A$ such that $\mathrm{Ker}(w_T) = \{0\}$ for all constituents $T \ncong S$, and $l := \mathrm{Ker}(w_S) = \mathrm{Ker}(w_S^2)$.
  - Until E_S^* acts transitively on $\mathrm{Ker}(w_S)\setminus\{0\}$. Hence w is an S-peakword, see 5.2.
  - Compute N such that $\ker(w^N) = \ker(w^{N+1})$ and let $\mathcal{C}$ be a basis for $\ker(w^N)$ given by $m(\mathcal{C}, \mathcal{S}) \in F^{lk\times n}$, see 5.3.

- Choose a linear independent subset $\mathcal{B} \subseteq V$, such that $V = \mathrm{im}(w^N) \oplus < \mathcal{B} >$. Let π be the corresponding projection of V onto $< \mathcal{B} >$ and $m(\mathcal{S}, \pi, \mathcal{B}) \in F^{n \times lk}$ be its matrix. Furthermore, let $\tilde{\pi} := \pi|_{\ker(w^N)}$.
- Compute $m(\mathcal{C}, \tilde{\pi}, \mathcal{B}) \in F^{lk \times lk}$.
- Compute $M := m(\mathcal{C}, \tilde{\pi}, \mathcal{B})^{-1} \cdot m(\mathcal{C}, \mathcal{S}) = m(\mathcal{B}, \tilde{\pi}^{-1}, \mathcal{S}) \in F^{lk \times n}$.
- For all i do:
  - Compute $A_i := M \cdot m(\mathcal{S}, a_i, \mathcal{S}) \cdot m(\mathcal{S}, \pi, \mathcal{B}) \in F^{lk \times lk}$. If $e \in A$ is the idempotent corresponding to w, then $e\pi = \pi$, and hence

$$\begin{aligned} A_i &= m(\mathcal{B}, \tilde{\pi}^{-1} a_i \pi, \mathcal{B}) \\ &= m(\mathcal{B}, \tilde{\pi}^{-1}, \mathcal{C}) \cdot m(\mathcal{C}, e a_i e, \mathcal{C}) \cdot m(\mathcal{C}, \tilde{\pi}, \mathcal{B}) \\ &\sim m(\mathcal{C}, e a_i e, \mathcal{C}). \end{aligned}$$

  - Compute generating vectors $m(\mathcal{V}, \mathcal{B}) \in F^{s \times lk}$ for the cyclic $< \{A_i\} >$-submodules of $F^{1 \times lk}$. Warning: $< \{A_i\} >$ is a subalgebra of eAe, but not necessarily equal to eAe, see 5.4.
  - Compute $\mathcal{R}_S := m(\mathcal{V}, \mathcal{B}) \cdot M = m(\mathcal{V}, \tilde{\pi}^{-1}, \mathcal{S}) \in F^{s \times n}$. Hence this matrix contains a set of generating vectors for the S-local submodules of V, see 5.5.
  - Delete redundant vectors in $\mathcal{R}_S$ and compute all dotted-lines consisting of S-local submodules, see 5.6.
- Form the union of all the $\mathcal{R}_S$ and compute the inclusions among the local submodules of V, see 5.7.
- Compute $\mathcal{M}(\mathcal{L})$ and the isomorphism types of all irreducible subquotients, see 5.8.

5. Implementation

The algorithm described in the preceding section has been implemented as a set of computer programs written in C. This package is a part of the new C version of the Meat-Axe, see [13], which has been developed from R. Parker's original FORTRAN programs, see [12], at Aachen. The programs run on several Unix workstations, i. e., DEC-Station 3000 and 5000 series, DEC Alpha, HP 9000/730, Sun 3, and Sun 4, and on the IBM PC under both Linux and MS-DOS.

Many functions as the finite field arithmetic and basic matrix operations were already available in the Meat-Axe and could be used in the lattice package with no or only slight changes. Actually, they constitute the first two parts of a four layer architecture:

- Low level functions for finite field arithmetics, elementary row operations on matrices, and file input and output.
- More complex procedures like matrix multiplication, null space and spin-up. Many of these operations correspond to Meat-Axe programs, so the only change was to convert the standalone programs into functions which can be called from other programs.
- Executable programs which perform one major step of the algorithm, e. g., calculation of composition series.
- A shell script which invokes all the programs automatically.

5.1. **Chop:** The first step is to calculate a composition series of the given module. This job, called chopping, was usually done 'by hand' via using several separate

MEAT-AXE programs. Even though the procedure is relatively simple, it may become very tedious in practice. Now there is one single program which performs this task.

The action of the generators on the module must be supplied to the program as a set of matrices in MEAT-AXE format. During the chopping, the module is recursively split into a submodule and a quotient using the standard MEAT-AXE methods. Internally, the result of this procedure is stored as a binary tree. The root corresponds to the given module, internal nodes are reducible modules and the leaves are the constituents.

In order to split a given module or to prove its irreducibility the algorithm needs an element of the algebra with a non-trivial but low-dimensional kernel. Such elements are searched for by taking linear combinations of certain products of the generators and examining their nullity. Each linear combination calculated in this way may be identified with a finite sequence of coefficients. Concatenating these numbers and treating them as digits mod q, where q is the field order, yields a canonical number which is used to identify the element. In order to avoid scalar multiples only those numbers which have a '1' as first digit in their base q representation are considered.

An example may make this clearer. Let a, b and c be the generators of the algebra. Algebra elements are then computed as linear combinations of 1, a, b, c, a^2, ab, ac, ba, b^2, bc, ca, cb, c^2, a^3, a^2b, a^2c, aba, ... If the field is GF(3), say, the first elements are:

Number		Algebra element
decimal	in base 3	
1	1	1
3	10	a
4	11	$a+1$
5	12	$a+2$
9	100	b
10	101	$b+1$
⋮	⋮	⋮
469	122101	$ab+2a^2+2c+b+1$
⋮	⋮	⋮

This numbering scheme is the same as used by the MEAT-AXE program ZSM.

The main loop of the CHOP program simply calculates algebra elements 1, 2, 3, ..., and examines their kernels. If an element with non-trivial kernel is found, the program tries to split the module; one vector of the kernel is taken and the smallest submodule containing this vector is calculated. This may be a proper submodule, and in this case the action of the generators on the submodule as well as on the quotient are calculated and the same procedure is applied recursively to both submodule and quotient.

Splitting the module again and again the program will eventually arrive at the constituents. Norton's criterion of irreducibility, see [12], is used to recognize this situation. However, this requires an algebra element with a small kernel to be found, because each vector in the kernel must be examined to see whether it generates the whole module. What 'small' means exactly, depends on both the dimension and the ground field. For this reason a 'maximal admissible nullity' m is maintained

by the program. Initially m is set to 3, but the user may choose a different value at run-time. Any algebra element a with a kernel of a dimension not exceeding m is used for the Norton test. If no algebra elements with small nullity appear m is incremented up to a maximal value of $m = 10$. This strategy is rather simple, but seems to be flexible enough in most cases.

Algebra elements with trivial kernel are useless for the algorithm, so an attempt is made to avoid unnecessary computation of such elements. Once an element is known to have a trivial kernel on a given module M, the program will mark it as invertible and ignore it for all constituents of M. Internally this is done by keeping at each node a list of invertible algebra elements identified by their number. The CHOP algorithm can then trace back through the parent nodes and find out which numbers, i. e., elements, may be ignored.

5.2. **Peakwords:** The next step is to calculate a peakword for each of the constituents. This is done by searching through the elements of the algebra using the same ordering as in the CHOP program.

If a constituent S is irreducible but not absolutely irreducible, the nullity of any element in the algebra will be a multiple of $[E_S : F]$, where F is the ground field and E_S the splitting field of S, see Theorem 3.3. This situation is recognized by calculating the greatest common divisor of all nullities which occur during the search. If the number of nullities is sufficiently high, 50 say, the calculated greatest common divisor is very likely to coincide with $[E_S : F]$.

To check this, we make use of Corollary 3.7. To decide for two vectors $v_1, v_2 \in \mathrm{Ker}(w_S)$, whether there is an endomorphism of S which maps v_1 to v_2, we use the standard basis algorithm, see [12]. Since S is irreducible, this results in a standard basis for S and therefore in standard matrices for the action of the generators on S. Now there is an endomorphism mapping v_1 to v_2 if and only if these standard matrices corresponding to the standard bases afforded by v_1 and v_2 are equal.

5.3. **Condensation:** For each constituent, the peakword given by its canonical number is evaluated as a matrix acting on V and then repeatedly raised to higher powers until the nullity stabilizes. The stable nullity equals the multiplicity k of the constituent times the degree $[E_S : F]$ of the splitting field. Having a power w^N of the peakword with stable nullity, the condensation onto its kernel, i. e., strictly speaking, the projection of V onto $V/\mathrm{im}(w^N)$, is determined in the same way as it is also in the MEAT-AXE program ZQT. This is the technical reason for introducing the subset called $\mathcal{B}$ in the Algorithm given Section 4. The matrix M is computed to assure that the canonical projection of $\ker(w^N)$ onto $V/\mathrm{im}(w^N)$ becomes the identity.

5.4. **Compute Cyclic Submodules:** For each of the condensed modules a representative of each cyclic submodule is found. This is done by spinning up each vector, up to scalar multiples, and maintaining a list of different cyclic submodules. As the dimension of the module grows, the number of vectors to spin up quickly becomes very large. This poses an upper limit on the dimension of condensed modules, i. e., on the multiplicity of the constituents.

A second limit concerns the number of cyclic submodules. Usually there are many fewer cyclic submodules than one-dimensional subspaces. Sometimes, however, it may happen that the peakword found in 5.2 is 'bad', in the sense that the condensed generators commute or may even be zero. In such a case one finds a

large number of cyclic submodules and the subsequent steps will probably take too much time. For this reason, the peakword program has an option to exclude one or more specified peakwords from the search. So, if the peakword turns out to be 'bad', the user can try another one.

5.5. **Uncondense:** The representatives of cyclic submodules found in 5.4 are then 'uncondensed', i. e., multiplied by the matrix M, to obtain generators under spin-up for the local submodules of V. Since the algebra generated by the condensed matrix generators is not necessarily equal to the condensed algebra, one cannot expect to find a one-to-one relation between cyclic submodules of the condensed module and local submodules of V. In practice this means that different cyclic submodules can uncondense to the same local submodule in the sense that more than one of the uncondensed vectors can spin up to one and the same local submodule. At the end of this step we have a list of local submodules together with the mapping of cyclic submodules of the condensed module to local submodules of V. The latter yields an equivalence relation among the cyclic submodules, where the equivalence classes are labelled by the local submodules.

5.6. **Dotted-lines:** To find the dotted-lines, we have to know which pairs of local submodules have the same sum. This could be done in the large module by spinning up pairs of generating vectors and comparing the resulting submodules. However, since dotted-lines never contain local submodules belonging to different constituents, see the remarks after Definition 1.2, it is possible to do the calculation in the condensed module, which is much faster due to its smaller dimension, see Theorem 2.3. The only difference is, that instead of pairs of vectors one has to consider pairs of equivalence classes of vectors in the sense explained in 5.5. The amount of computation necessary is further reduced by using Theorem 1.7. For a given pair of different S-local submodules, it is sufficient to find one dotted-line containing them, instead all such dotted-lines, of which there can, in fact, be more than one. Internally, a dotted-line is represented as a bit string, where a non-zero entry at position i means that the i^{th} local submodule belongs to the dotted-line.

5.7. **Incidences:** The next step is to compute all incidences between the local submodules. This information is stored in an incidence matrix, which contains a non-zero entry at position (i, k) if the i^{th} local submodule is contained in k^{th} local submodule. Internally, the incidence matrix is represented as an array of bit strings.

One way of computing the incidences would be to compare directly the local submodules which have been obtained in the preceding step. However, since it is in general not possible to hold all local subspaces simultaneously in memory or in an external file, we would have to spin up the local submodules each time they are needed, a very time-consuming task for large modules.

For this reason, a different strategy is used. In order to check whether the S-local submodule X contains the T-local submodule Y for another constituent T of V, it is sufficient to consider the condensation $\tilde{X}$ with respect to the T-peakword and to check whether the cyclic submodules belonging to Y in the sense of 5.5 are contained in $\tilde{X}$. The submodule X is condensed in the same way as is described in 5.3. As the dimensions of the condensed modules are usually much smaller than the dimension of the given module, one can hold all modules in memory at the same time.

5.8. **Compute All Submodules:** The knowledge of all incidences and dotted-lines of local submodules is sufficient to calculate the complete submodule lattice. In this final step no matrix operations are involved. Instead the program works with bit strings representing the incidences and dotted-lines. A submodule is uniquely determined by the local submodules it contains, see Theorem 1.5, and can therefore also be represented as a bit string. As Theorem 1.5 states, submodules correspond to sets of local submodules which fulfill conditions which are easily checked on the bit string level.

As an option, the program allows the user to compute 'blocks' of the incidence matrix, where a block is defined as a connected component of the incidence graph with the additional condition, that all local submodules with respect to one and the same constituent are contained in the same block. Then all blocks are closed under the forming of dotted-lines, see Remark 1.3, and under forming subsets of local submodules according to Theorem 1.5. Hence these blocks are Benson-Conway blocks in the sense of Remark 1.8 and correspond to uniquely determined direct summands of V. Each submodule of V is then given as a direct sum of submodules lying in different blocks.

Submodules are calculated generation by generation, the first generation consisting of all submodules generated by one local submodule. In the n^{th} generation all submodules generated by a submodule of generation $n-1$ plus one local submodule are calculated. If no more submodules appear, the algorithm terminates. All results, i. e.,

- a list of all local submodules with their dimensions,
- the incidence matrix of local submodules,
- a list of dotted-lines,
- a list of all submodules giving for each submodule the local submodules it contains and its maximal submodules, and
- the elements of the radical series,

are written to a text file.

Optionally, the program computes only the submodule lattices of the direct summands of V corresponding to the blocks described above.

6. Examples

6.1. **Left cell representations.** In the modular representation theory of finite groups of Lie type the notion of Hecke algebras has become very important in recent times. It might be possible that the examination of the left cell representations of Hecke algebras using a computer may lead to results which are also of theoretical interest. As references for Hecke algebras and left cell representations take, e. g., [10] and [18]. As a reference for the notation used here, see [5].

Below, the submodule structures of p-modular reductions of the left cell representation 72_1 of the Hecke algebra of type F_4 over some prime fields $GF(p)$ are given. As it turns out, the submodule structure is already determined by giving the decomposition of the p-modular reduced module into irreducible summands and their socle series. The different constituents are denoted by their degrees and by lower case letters, if necessary.

$e = 2$	$GF(5)$	$1 \oplus 4 \oplus 12 \oplus 16 \oplus 16 \oplus 5 \oplus \begin{bmatrix} 2b \\ 5 \\ 2a \end{bmatrix} \oplus \begin{bmatrix} 2a \\ 5 \\ 2b \end{bmatrix}$
$e = 3$	$GF(7)$	$6 \oplus 9a \oplus 9b \oplus 12 \oplus \begin{bmatrix} 1a \\ 1b \oplus 1c \\ 1d \end{bmatrix} \oplus \begin{bmatrix} 4a \\ 4b \oplus 4c \\ 4d \end{bmatrix} \oplus \begin{bmatrix} 4a \\ 4b \oplus 4c \\ 4d \end{bmatrix}$
$e = 4$	$GF(17)$	$6 \oplus 9a \oplus 9b \oplus 16 \oplus 16 \oplus 4b \oplus \begin{bmatrix} 4c \\ 4b \\ 4a \end{bmatrix}$
$e = 6$	$GF(13)$	$4 \oplus 6 \oplus 16 \oplus 16 \oplus \begin{bmatrix} 1b \\ 5a \oplus 5b \\ 1a \end{bmatrix} \oplus \begin{bmatrix} 2a \\ 5a \\ 2d \end{bmatrix} \oplus \begin{bmatrix} 2c \\ 5b \\ 2b \end{bmatrix}$
$e = 8$	$GF(17)$	$4 \oplus 6 \oplus 9a \oplus 9b \oplus 12 \oplus \begin{bmatrix} 8b \\ 8a \end{bmatrix} \oplus \begin{bmatrix} 8b \\ 8a \end{bmatrix}$
$e = 12$	$GF(73)$	$4 \oplus 9a \oplus 9b \oplus 12 \oplus 16 \oplus 16 \oplus \begin{bmatrix} 3b \\ 3a \end{bmatrix}$

6.2. **The automorphism group of the 4-dimensional cube.** The following example is mentioned in [11], where it is used to construct a binary group code of type $(32, 17, 8)$, which has been discovered by Y. Cheng and N. Sloane. Let $G \cong C_2 \wr S_4$ be the automorphism group of the 4-dimensional cube, acting on the 32 edges of the cube. The 2-modular reduction of this permutation representation has two constituents $1a$ and $2a$ with multiplicity 12 and 10, respectively. Note that this is not the best case since the dimensions of the constituents are small in comparison to the dimension of the module. This module is found to have 70 local submodules, 128 dotted-lines, and 373 submodules, hence we are not going to print the submodule lattice in any more detail here. The programs need 36 seconds of CPU-time on a DEC-Station 5300 to chop the module and to compute its local submodules, the incidences between them, and the dotted-lines. Finally, 104 seconds are spent to find all the submodules using Theorem 1.5, where no matrix operations are involved.

6.3. **The structure of a projective indecomposable module.** We will give the complete submodule lattice of the projective cover P of the trivial module for the sporadic simple Rudvalis group in characteristic 5. This is a module of dimension 41875 over $GF(5)$. Note that it is not possible to examine P directly on todays computers. But using the condensation technique which is described in [14], it is possible to construct a module M of dimension 348 over $GF(5)$, such that there is an isomorphism of the submodule lattices of M and P which respects the isomorphism types of irreducible subquotients, see [7]. M has constituents $1a$, $1b$, $9a$, $24a$, $30a$, and $238a$ with multiplicities 4, 4, 2, 1, 2, and 1, respectively. These constituents correspond to irreducible modules for the Rudvalis group which have degrees 1, 273, 783, 2219, 3380, and 30234, respectively.

The submodule lattice contains 18 local submodules, one dotted-line, and 60 submodules. It is depicted in the diagram below, where the elements of the socle series are denoted by circles and the local submodules are denoted by bold boxes or circles. Each submodule gets a number when it occurs during the computation of all submodules, these numbers are also depicted. The diagram has also been

generated by one of the programs in the submodule lattice package. The programs need 1516 seconds of CPU-time on a DEC-Station 5300 to chop M and to compute its submodule lattice.

6.4. **The structure of another projective indecomposable module.** Finally, let P denote the projective cover of the trivial module for the sporadic simple Mathieu group M_{12} in characteristic 2. This example has also been considered in [16]. P is a module of dimension 704 over $GF(2)$ and has constituents $1a$, $10a$, and $44a$ with multiplicities 16, 16, and 12, respectively. P is found to have 1559 local submodules and 78807 dotted-lines. The programs need 81812 seconds, i. e., 22 hours, 43 minutes, and 32 seconds, of CPU-time on a HP 9000/730 to chop P and to compute its local submodules, the incidences between them, and the dotted-lines.

References

[1] M. Aigner: Combinatorial Theory, Springer, 1976.
[2] D. Benson, J. Conway: Diagrams for Modular Lattices, J. Pure Appl. Alg., 37, 1985.
[3] C. Curtis, I. Reiner: Methods of Representation Theory I, Wiley, 1981.
[4] R. Dilworth: Proof of a Conjecture on Finite Modular Lattices, Ann. Math., 60, 1954.
[5] M. Geck, K. Lux: The Decomposition Numbers of the Hecke Algebra of Type F_4, Man. Math., 70, 1991.
[6] J. Green: Polynomial Representations of GL_n, Lecture Notes in Mathematics, 830, Springer, 1980.
[7] G. Hiss, J. Müller: The 5-modular Irreducible Characters of the Sporadic Simple Rudvalis Group and its Double Cover, in preparation.
[8] B. Huppert: Endliche Gruppen I, Springer, 1983.
[9] M. Isaacs: Character Theory of Finite Groups, Academic Press, 1976.
[10] D. Kazhdan, G. Lusztig: Representations of Coxeter Groups and Hecke Algebras, Invent. Math., 53, 1979.
[11] G. Michler: Ring Theoretical and Computational Methods in Group Representation Theory, Vorlesungen aus dem Fachbereich Mathematik der Universität Essen, vol. 18.
[12] R. Parker: The Computer Calculation of Modular Characters, in: M. Atkinson (Ed.): Computational Group Theory, 1984.
[13] M. Ringe: The C Meat-Axe, Manual, RWTH Aachen, 1993.
[14] A. Ryba: Condensation Programs and their Application to the Decomposition of Modular Representations, J. Symb. Comp., 1990.
[15] G. Schneider: Oral communication.
[16] G. Schneider: The Structure of the Projective Indecomposable Modules of the Mathieu Group M_{12} in Characteristic 2 and 3, Arch. Math., 60, 1993.
[17] M. Schönert: (ed.) GAP-3.3, Manual, RWTH Aachen, 1993.
[18] J. Shi: The Kazhdan-Lusztig Cells in certain Affine Weyl Groups, Lecture Notes in Mathematics, 1179, Springer, 1980.

(Jürgen Müller) Lehrstuhl D für Mathematik, RWTH Aachen, Templergraben 64, D-52062 Aachen, Germany

(Klaus Lux) Lehrstuhl D für Mathematik, RWTH Aachen, Templergraben 64, D-52062 Aachen, Germany

(Michael Ringe) Institut für Theoretische Physik, Lehrstuhl E, RWTH Aachen, Templergraben 64, D-52062 Aachen, Germany

E-mail address, Jürgen Müller: `jmueller@tiffy.math.rwth-aachen.de`

E-mail address, Klaus Lux: `klux@tiffy.math.rwth-aachen.de`

E-mail address, Michael Ringe: `mringe@thphys.physik.rwth-aachen.de`