Computing the 2-modular characters of $\text{Fi}_{23}$

Max Neunhöffer

Lehrstuhl D für Mathematik
RWTH Aachen

Columbus 2005
Modular representations

\(G\) finite group, \(F\) field, \(p := \text{char}(F) \mid |G|\), \(V\) an \(F\)-vector space

- A modular representation of \(G\) on \(V\) is a group homomorphism \(\rho : G \rightarrow \text{GL}(V)\).
- \(G\) acts on \(V\) via \(\rho\) (\(V\) is an \(FG\)-module).
- \(\rho\) is called irreducible if there is no proper \(G\)-invariant subspace \(0 \neq U \subsetneq V\).

**Aim:** Classification of the irreducible modular representations of the sporadic simple groups.

**Example (\(Fi_{23}\) mod 2)**

\(G = Fi_{23}\) and \(p = 2\), \(|G| = 4.089.470.473.293.004.800\) (joint work with G. Hiß and F. Noeske).
Finding a composition series

Let $0 \neq U \subsetneq V$ be $G$-invariant.

- We get two new representations:
  \[
  \rho_U : G \to \text{GL}(U), \quad \rho_{V/U} : G \to \text{GL}(V/U).
  \]

- Iteration gives as „atoms“ $\rho_{S_i} : G \to \text{GL}(S_i)$ on the composition factors $S_i$ of $V$.

- Package \texttt{MEATAXE} [Parker, Thackray 1978,\ldots] computes a composition series automatically (\texttt{chop}).

Example ($Fi_{23}$ mod 2)

Permutation module $1^{Fi_{23}}_{2.Fi_{22}}$ of dimension 31.671 contains composition factors $1a, 782a, 1.494a, 3.588a, 19.940a$.  
about 4 days of CPU time in 8 GB main memory.
Construction of modular representations

How can we construct new representations from given ones?

**Theorem (Burnside-Brauer)**

\[ G \text{ simple}, \ V \text{ non-trivial irreducible } FG\text{-module}. \text{ For every irreducible } FG\text{-module } W \text{ there is an } m \in \mathbb{N} \text{ such that} \]

\[ W \text{ is a composition factor of } V^\otimes m. \]

OK, then we are done!? NO!

**Example \((Fi_{23} \mod 2)\)**

\[ \dim_F 19.940a \otimes 19.940a = 367.603.600 \]

One GF(2)-matrix \(\approx 18.403.938\) GB \(\approx 17.5\) PB.
Let $e = e^2 \in FG$ an idempotent. Consider $V = Ve \oplus V(1 - e)$.

### Definition (Schur functor)

\[
\mathcal{F} : \text{mod} - FG \to \text{mod} - eFGe \\
V \mapsto Ve \\
\phi \in \text{Hom}_{FG}(V, W) \mapsto \phi|Ve \in \text{Hom}_{eFGe}(Ve, We)
\]

- \(\mathcal{F}\) is exact.
- If \(V\) is an irreducible \(FG\)-Modul, then \(Ve\) is irreducible or \(Ve = 0\).

I.e. \(\mathcal{F}\) maps a composition series onto a composition series.

- If \(Ve \neq 0\) for all irreducible \(FG\)-modules \(V\), then \(eFGe\) and \(FG\) are Morita equivalent.
Let $K \leq G$ such that $p$ does not divide $|K|$. We choose

$$e := \frac{1}{|K|} \sum_{k \in K} k \in FK \leq FG.$$ 

**Task:** Given $g \in G$, determine action of $ege$ on $(V \otimes W)e$.

*Without* explicit computation of $V \otimes W$!

*Theorem (Lux, Wiegelmann 1997)*

*This can be done!*
Example \((\text{Fi}_{23} \mod 2)\)

- \(K \leq G, |K| = 3^9 = 19,683.\)
- \(eFGe\) and \(FG\) are Morita equivalent.
- \(\dim_F(19.940a \otimes 19.940a)e = 25.542.\)

One GF(2) matrix \(\approx 77,8\) MB.

About 1 week of CPU time to compute the operation of one element \(ege\) on \((19.940a \otimes 19.940a)e\).

But now we are done, aren’t we? **Unfortunately not.**
Remember: We investigate $Ve$ by giving matrices for generators of $eFGe$.

Question (The Generation Problem)

How can $eFGe$ be generated by “a few” elements?

If $\mathcal{E} \subseteq FG$ with $\langle \mathcal{E} \rangle = FG$. Then $\langle e\mathcal{E}e \rangle = eFGe$ does not follow!

- Let $C := \langle e\mathcal{E}e \rangle \leq eFGe$.
  Instead of $Ve$ we consider the $C$-module $Ve|_C$.
- Contrary to $Ve$ we can not directly conclude things from $Ve|_C$ to $V$. 
Let $K \trianglelefteq N \leq G$.

**Theorem (F. Noeske, 2005)**

If $\mathcal{T}$ is a set of **double coset representatives** of $N\backslash G/N$ and $\mathcal{N}$ a set of **generators** of $N$, then we have

$$eFGe = \langle e\mathcal{N}e, e\mathcal{T}e \rangle.$$ 

**Example ($Fi_{23}$ mod 2)**

- $N$ the 7th maximal subgroup, $[G : N] = 1.252.451.200$
- $|\mathcal{T}| = 36$ and $|\mathcal{N}| = 3$, i.e. 38 generators for $eFGe$.
- Computation of $\mathcal{T}$: see second half of this talk.
Brauer Characters

Let $G_{p'}$ be the set of $p$-regular elements of $G$.

- To each modular representation on $V$ we can assign a class function $\beta_V$ on $G_{p'}$ (Brauer character).

**Aim**

**Determine the irreducible Brauer characters $\varphi_1, \ldots, \varphi_\ell$.**

- $\beta_V$ is a $\mathbb{Z}_{\geq 0}$-linear combination of the $\varphi_1, \ldots, \varphi_\ell$. This decomposition corresponds to the decomposition of $V$ into its composition factors.
- $\beta_V \otimes W(g) = \beta_V(g) \cdot \beta_W(g)$ for all $g \in G_{p'}$.
- We know explicitly Brauer characters $\vartheta_1, \ldots, \vartheta_\ell$ such that

$$\langle \vartheta_1, \ldots, \vartheta_\ell \rangle_{\mathbb{Z}} = \langle \varphi_1, \ldots, \varphi_\ell \rangle_{\mathbb{Z}}.$$
Equations in Brauer characters

**Question:** How to determine \( \varphi_1, \ldots, \varphi_\ell \) from the \( \vartheta_1, \ldots, \vartheta_\ell \)?

- We have

\[
\vartheta_i = \sum_{j=1}^{\ell} a_{ij} \varphi_j, \quad a_{ij} \in \mathbb{Z}_{\geq 0}.
\]

- Because of \( \langle \vartheta_1, \ldots, \vartheta_\ell \rangle_{\mathbb{Z}} = \langle \varphi_1, \ldots, \varphi_\ell \rangle_{\mathbb{Z}} \) it follows that

\[
\beta_{V \otimes W} = \sum_{i=1}^{\ell} t_i \vartheta_i, \quad t_i \in \mathbb{Z} \quad \text{(GAP)}
\]

\[
= \sum_{j=1}^{\ell} s_j \varphi_j, \quad s_j \in \mathbb{Z}_{\geq 0} \quad \text{(condens. & MEATAXE)}
\]
The Result

Solution of

System of Equations

\[ \sum_{i=1}^{\ell} a_{ij} t_i = s_j, \quad j = 1, \ldots, \ell \]

with GAP gives the 2-modular characters.

Example (Degrees in the principal block of \(Fi_{23} \mod 2\))

<table>
<thead>
<tr>
<th></th>
<th>1,</th>
<th>782,</th>
<th>1.494,</th>
<th>3.588,</th>
<th>19.940,</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>57.408,</td>
<td>94.588,</td>
<td>94.588,</td>
<td>79.442,</td>
<td>583.440,</td>
</tr>
<tr>
<td></td>
<td>1.951.872,</td>
<td>724.776,</td>
<td>979.132,</td>
<td>1.997.872,</td>
<td>1.997.872,</td>
</tr>
</tbody>
</table>
Problem

$G := \text{Fi}_{23} = \langle a, b \rangle$ with $|G| = 4.089.470.473.293.004.800,$

$N = \langle n_1, n_2, n_3 \rangle \leq G$ with $|N| = 3.265.173.504,$

where the $n_i$ are given as words in $a$ and $b$.

Known: $G = Ng_1 N \cup Ng_2 N \cup \cdots \cup Ng_{36} N,$

where $NgN = \{ n \cdot g \cdot n' \mid n, n' \in N \}$.

Problem: Find $\{ g_1, \ldots, g_{36} \}$ as words in $a$ and $b$.

Application: Let $K \triangleleft N$ and $F := GF(2)$ and $2 \nmid |K|$. Then we have for $e^2 = e := \frac{1}{|K|} \sum_{k \in K} k \in FG$:

$$eFGe = \langle eg_1 e, \ldots, eg_{36} e, en_1 e, en_2 e, en_3 e \rangle_{F_{-\text{Alg}}}.$$ (F. Noeske, 2005)
Double cosets and suborbits

\[ G = N g_1 N \cup N g_2 N \cup \cdots \cup N g_{36} N \]

\( G \) acts on the set \( N \backslash G := \{N g \mid g \in G\} \)

\( N \backslash G = N g_1 \cdot N \cup \cdots \cup N g_{36} \cdot N \)

Thus: \( \text{double cosets} \leftrightarrow \text{suborbits} \)

Problems:

- \( |N \backslash G| = 1.252.451.200 \approx 1.25 \cdot 10^9 \)
- Permutations for \( a, b, n_1, n_2, n_3 \) would need about 5 GB
- Not easy to determine
- \( G \) is given as a permutation group on 31.671 points
- To determine elements \( g_i \) would take too long
Realisation of the action on $N \backslash G$

Linear representation of $G$ on $V := F^{1 \times 1494} (F = GF(2))$:

- group homomorphism $\rho : G \rightarrow GL_{1494}(F)$

Find vector $v_1 \in V$ such that $v_1 \cdot \rho(N) = \{v_1\}$.

$\implies$ The orbit $v_1 \cdot \rho(G)$ is isomorphic to $N \backslash G$ as $G$-set.

Thus we can:
- Store and compare points of $N \backslash G$ as vectors in $V$
- Act with group elements (words in $a, b$) on them

Still too large:
- Each vector needs about 200 bytes ($\approx 1494/8$)
- Altogether about 250 GB (main memory!)
- It takes uncomfortably long to enumerate all vectors!
A Trick

Let $U < N$ with $|U| = 6561$.

**Idea:** do things “by $U$-orbits”:
- $N$-orbits are unions of $U$-orbits
- enumerate $U$-orbits

To this end:
- choose in each $U$-orbit $B$ a subset $\min(B) \subseteq B$ ("$U$-minimal" vectors), such that
- we have an algorithm, that computes, given an $U$-orbit $B$ and a $v \in B$, a $u(v) \in U$ such that $v \cdot \rho(u(v)) \in \min(B)$.
- store $B$ by storing $\min(B)$

$\implies$ Save about a factor of 250 of memory and time!
Introduction and Condensation

The 2-modular characters of $\text{Fi}_{23}$

Problem, Perfidy, Tricks, and Tackling them

Verification, Overview and Outlook

Still tedious

Progression of an $N$-orbit enumeration by $U$-orbits:

- 25%
- 50%
- 75%
- 100%

N–orbit: 90,699,264 vectors

- 90%: 5.3 min
- 95%: 5.3 min
- 98%: 6.7 min
- 100%: 18.3 min

Max Neunhöffer
Computing the 2-modular characters of $\text{Fi}_{23}$
Halves of orbits

We enumerate only one half of each $N$-orbit!

Suppose we know $H \subseteq v \cdot \rho(N)$ with $|H| > |v \cdot \rho(N)|/2$.

**Question:** Does $w \in V$ lie in the orbit $v \cdot \rho(N)$?

**Answer:**

Apply random elements $\{m_1, \ldots, m_{40}\}$ of $N$ to $w$ and test whether $\{w \cdot \rho(m_i) \mid 1 \leq i \leq 40\} \cap H \neq \emptyset$.

If yes: $w$ lies in $v \cdot \rho(N)$ with certainty

If no: $w$ probably does not lie in $v \cdot \rho(N)$

$\implies$ Enough to find different $N$-orbits in $v_1 \cdot \rho(G)$. 
The first 35 are now doable

\[ N\text{-suborbit lengths in } v_1 \cdot \rho(G) : \]

<table>
<thead>
<tr>
<th></th>
<th>10.077.696</th>
<th>20.155.392</th>
<th>3.888</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.077.696</td>
<td>20.155.392</td>
<td>3.888</td>
</tr>
<tr>
<td>22.674.816</td>
<td>90.699.264</td>
<td>5.038.848</td>
<td>78.732</td>
</tr>
<tr>
<td>34.012.224</td>
<td>10.077.696</td>
<td>3.359.232</td>
<td>19.683</td>
</tr>
<tr>
<td>272.097.792</td>
<td>68.024.448</td>
<td>5.038.848</td>
<td>186.624</td>
</tr>
<tr>
<td>90.699.264</td>
<td>136.048.896</td>
<td>7.558.272</td>
<td>62.208</td>
</tr>
<tr>
<td>90.699.264</td>
<td>30.233.088</td>
<td>3.779.136</td>
<td>124.416</td>
</tr>
<tr>
<td>272.097.792</td>
<td>30.233.088</td>
<td>3.779.136</td>
<td>15.552</td>
</tr>
<tr>
<td>10.077.696</td>
<td>944.784</td>
<td>1.679.616</td>
<td>15.552</td>
</tr>
<tr>
<td>944.784</td>
<td>30.233.088</td>
<td>1.679.616</td>
<td>768</td>
</tr>
</tbody>
</table>

Apply orbit enumeration to \( v_1 \cdot \rho(G) \), do it by \( N \)-orbits.

We find all \( N \)-orbits, except for the one with 768 vectors.

Not yet proved, that these 35 orbits are pairwise disjoint!
Criminal search

We are looking for one of 768 vectors $v$ with the following properties:

- $v \in v_1 \cdot \rho(G)$
- $S := Stab_N(v) < N$ has index 768 in $N$

**Approach:**

- guess $S < N$ with $[N : S] = 768$ (not unique!)
- compute candidates $C := \{ v \in V \mid v \cdot \rho(S) = \{ v \} \}$
- check for all $v \in C$ whether $v \cdot \rho(a)$ lies in one of the 35 (half) known $N$-orbits (remember: $G = \langle a, b \rangle$)
- if yes, it follows that $v \in v_1 \cdot \rho(G)$ (proven!).
- If $v$ itself is not in one of the 35 orbits, then we are ready.
- $\implies$ produces in fact vector $v_{36}$
The Last Representative

We still need a word \( g_{36} \) in \( a \) and \( b \), that maps \( v_1 \) to \( v_{36} \)!

Do the following:

- enumerate vectors in \( v_1 \cdot \rho(G) \) with a “breadth first” search by applying \( a \) and \( b \) until the memory is full
- search backwards starting with \( v_{36} \) for a “known” vector using a “depth first” search (apply \( a^{-1} \) and \( b^{-1} \))
- put forward and backward search together
- \( \Rightarrow \) finds word within a few minutes
- possible improvement: “by \( U \)-orbits”
  (here not necessary)
For candidates for $g_1, \ldots, g_{36}$:

Enumerate all $N$-suborbits $v_1 \cdot \rho(g_i) \cdot \rho(N)$ completely.

Needs about 3 h on a machine with 4 GB main memory.

$\Rightarrow$ Proven:

All $N$-orbits lie in $v_1 \cdot \rho(G)$ and are pairwise disjoint.

Remark: There is still a lot of potential for improvements here:

- One only has to compare orbits of equal length.
- To test whether two orbits are disjoint, only one has to be enumerated.
- So only one orbit has to fit into main memory at a time.
We have
- constructed a permutation representation of $\text{Fi}_{23}$ on $1.252.451.200$ points,
- enumerated all $36$ $N$-suborbits and determined their lengths,
- found $N$-$N$-double coset representatives $g_1, \ldots, g_{36}$ as words in $a, b$,
- fulfilled the requirements for the condensation computations and
- computed the 2-modular character table of $\text{Fi}_{23}$ (joint work with Gerhard Hiß and Felix Noeske).
Outlook

These enumeration methods can be applied if
- there is an appropriate (small) linear representation,
- there is an appropriate vector $v_1$ (or something similar),
- one (or a few) appropriate helper subgroups can be found,
- one needs large orbits or double coset representatives.

The condensation methods can be applied if
- there is an appropriate condensation subgroup $K$,
- things are “small enough” with resp. to memory and time,
- the generation problem can be solved by using Noeske’s Theorem and determining double coset representatives.

All this is implemented in GAP and will be published as a package soon.