The computer calculation of modular character tables

Gerhard Hiss

Nikolaus Conference 2024
Session in Memory of Richard Parker
RWTH Aachen University, December 7, 2024

Richard 15 years ago

What is MOC?

MOC (for MOdular Characters)
is a collection of stand-alone programs (in C and FORTRAN),
linked via shell scripts,
for computing with Brauer characters and projective **characters**.

Purpose: Assist the computation of decomposition numbers
(equivalently:
Brauer character tables = modular character tables).

Authors: Richard Parker, Christoph Jansen, Klaus Lux, H.
(1984 – 1987)

Gap-Package MOC: Lukas Maas and Felix Noeske
(2012–2014)

Decomposition numbers

Let G be a finite group and p be prime, fixed throughout this talk.

Write $\mathrm{IBr}(G) = \{\varphi_1, \ldots, \varphi_l\}$, the set of p-modular irreducible character (= Brauer characters).

For $\chi \in \mathrm{Irr}(G) = \{\chi_1, \ldots, \chi_k\}$, write χ° for the restriction of χ to $G_{p'} := \{g \in G \mid p \nmid |g|\}$. Then

$$\chi_i^\circ = \sum_{j=1}^{l} d_{ij}\, \varphi_j$$

for non-negative integers, d_{ij}, the decomposition numbers of G modulo p.

The matrix $D = [d_{ij}]$ is the decomposition matrix of G.

Knowing D is equivalent to knowing $\mathrm{IBr}(G)$ (if $\mathrm{Irr}(G)$ is known).

Projective characters

Let $D = [d_{ij}]$ be the decomposition matrix.

The (ordinary) character

$$\Phi_i := \sum_{j=1}^{k} d_{ji}\chi_j$$

is called the projective indecomposable character (PIM) associated to φ_i ($1 \leq i \leq l$). Put $\mathrm{IPr}(G) := \{\Phi_1, \ldots, \Phi_l\}$.

A projective character is a sum of PIMs.

Expanding a projective character in $\mathrm{Irr}(G)$ yields a sum of columns of the decomposition matrix.

The orthogonality relations

$\langle \mathrm{IBr}(G)\rangle_{\mathbb{Z}}$: set of generalised Brauer characters and
$\langle \mathrm{IPr}(G)\rangle_{\mathbb{Z}}$: set of generalised projective characters

These are free abelian groups with bases $\mathrm{IBr}(G)$ resp. $\mathrm{IPr}(G)$.

Define

$$\langle \chi, \psi\rangle' := \frac{1}{|G|} \sum_{g \in G_{p'}} \chi(g)\,\psi(g^{-1})$$

for $\chi \in \langle \mathrm{IBr}(G)\rangle_{\mathbb{Z}}, \psi \in \langle \mathrm{IPr}(G)\rangle_{\mathbb{Z}}$.

Theorem (Orthogonality relations)

$\langle \varphi_i, \Phi_j\rangle' = \delta_{ij}$.

Basic sets according to Richard Parker

1. A basic set of Brauer characters is a $\mathbb{Z}$-basis of $\langle \mathrm{IBr}(G) \rangle_{\mathbb{Z}}$ consisting of **Brauer characters**.
2. A basic set of projective characters is a $\mathbb{Z}$-basis of $\langle \mathrm{IPr}(G) \rangle_{\mathbb{Z}}$ consisting of **projective characters**.

Let $\mathcal{B}$ and $\mathcal{P}$ be sets of Brauer characters, respectively projective characters.

1. $\mathcal{B}$ is a basic set, if and only if $\mathcal{B}$ is linearly independent and $\{\chi^\circ \mid \chi \in \mathrm{Irr}(G)\} \subseteq \langle \mathcal{B} \rangle_{\mathbb{Z}}$.
2. $\mathcal{B}$ and $\mathcal{P}$ are basic sets if and only if $U := \langle \mathcal{B}, \mathcal{P} \rangle'$ is an $l \times l$ matrix, invertible over $\mathbb{Z}$.

FBA and special basic sets

Let $\mathcal{X}$ be a set of Brauer characters.

FBA is an algorithm of Richard Parker to compute a $\mathbb{Z}$-basis $\mathcal{B}$ of $\langle \mathcal{X} \rangle_{\mathbb{Z}}$ consisting of **Brauer characters**.

Applied with $\mathcal{X}' = \{\chi^\circ \mid \chi \in \mathrm{Irr}(G)\}$, $\mathcal{B}$ is a basic set of Brauer characters, most of the time with $\mathcal{B} \subseteq \mathcal{X}'$.

Definition (Parker, from 1984)

A basic set $\mathcal{B}$ is special, if $\mathcal{B} \subseteq \{\chi^\circ \mid \chi \in \mathrm{Irr}(G)\}$.

Question

Do special basic sets always exist?

This question is still open.

Special basic sets are of considerable relevance in the representation theory of groups of Lie type.

Principal MOC strategy

1. Compute large sets $\mathcal{X}$ and $\mathcal{Y}$ of Brauer characters, resp. projective characters.
2. Construct "minimal" basic sets $\mathcal{B}$ and $\mathcal{P}$ from $\mathcal{X}$ resp. $\mathcal{Y}$.
3. Compute $U := \langle \mathcal{B}, \mathcal{P} \rangle'$.
4. Then $\mathcal{B} = X_1 \cdot \mathrm{IBr}(G)$, and $\mathcal{P} = X_2 \cdot \mathrm{IPr}(G)$ with (unknown) $X_1, X_2 \in \mathbb{N}^{l \times l}$, invertible over $\mathbb{Z}$, such that
   - $U = X_1 X_2^{tr}$,
   - $\langle \mathcal{X}, \mathcal{P} \rangle' \cdot (X_2^{tr})^{-1} \geq 0$,
   - $X_1^{-1} \cdot \langle \mathcal{B}, \mathcal{Y} \rangle' \geq 0$.

Try to determine X_1, X_2 from these conditions (using methods of integer linear programming).

Works well, if U is "sparse".

The MeatAxe

Let F be a finite field of characteristic p and let $\mathfrak{X} : G \to \mathrm{GL}_d(F)$ be an F-**representation** of G.

Aim: Find the composition factors of $\mathfrak{X}$ (chopping $\mathfrak{X}$).

The MeatAxe is a collection of programs that perform this task.

It was invented and developed by Richard Parker and Jon Thackray from 1978.

Since then, it has been improved and enhanced by many people, including Derek Holt, Gábor Ivanyos, Klaus Lux, Jürgen Müller, Sarah Rees, Michael Ringe, and by Richard Parker himself.

In fact he worked continuously on his vision to drive the MeatAxe to perfection.

This culminated in the MeatAxe64 (2017).

The MeatAxe: basic strategy

Suppose that $G = \langle g_1, \ldots, g_n \rangle$. Put $A_i := \mathfrak{X}(g_i)$, $1 \leq i \leq n$.

Let $\mathbf{A} := \langle A_1, \ldots, A_n \rangle_{F\text{-alg.}} \leq F^{d \times d}$
and $V := F^{1 \times d}$, the natural right $\mathbf{A}$-module.

Find singular $B \in \mathbf{A}$ (by a random search) with nullspace
$N \leq V$ of small dimension (preferably 1).

For **all** $0 \neq w \in N$ test if $w.\mathbf{A} = V$.
(Note that $w.\mathbf{A}$ is G-invariant.)

If YES

For **one** $0 \neq w$ in the nullspace of B^{tr} test if $w.\mathbf{A}^{tr} = V$.

If YES, $\mathfrak{X}$ is irreducible.

Condensation: motivation

The MeatAxe64 can chop representations of degrees of up to two million over $\mathbb{F}_2$. Over larger fields, only smaller dimensions are feasible.

A two million square matrix over $\mathbb{F}_2$ needs about 0.5TB.

Frank has recently chopped a 131072-dimensional module over $\mathbb{F}_7$ with the MeatAxe64.

Michael Ringe's C-MeatAxe can chop representations of degree up to 200 000 over $\mathbb{F}_2$.

To expand the range of applications of the MeatAxe, Condensation is used

(Thackray, Parker, from 1980; Lux, Müller, Noeske, Ryba).

Condensation: theory [J. A. Green, J. Thackray]

Let $e \in \mathbf{A}$ an idempotent, i.e., $0 \neq e = e^2$ (a projection).

Get exact functor (Condensation):

$$\text{mod-}\mathbf{A} \to \text{mod-}e\,\mathbf{A}e, \qquad V \mapsto Ve.$$

If $S \in$ mod-$\mathbf{A}$-mod is simple, then $Se = 0$ or simple.

Let $S_1, \ldots, S_n$ be the simple $\mathbf{A}$-modules.

Suppose that $S_1 e \neq 0, \ldots, S_m e \neq 0$, $S_{m+1} e = \cdots = S_n e = 0$.

Then $S_1 e, \ldots, S_m e$ are exactly the simple $e\,\mathbf{A}e$-modules.
If $m = n$, the algebras $\mathbf{A}$ and $e\,\mathbf{A}e$ are Morita equivalent.

As the Condensation functor is exact, it sends a composition series of $V \in$ mod-$\mathbf{A}$ to a composition series of $Ve \in$ mod-$e\,\mathbf{A}e$.

How to condense a permutation module?

Let $H \leq G$ with $\mathrm{char}(F) \nmid |H|$. Then a suitable idempotent is

$$e := \frac{1}{|H|} \sum_{x \in H} x \in FG.$$

Let V be the permutation FG-module w.r.t. an action of G on the finite set Ω. Then Ve is the set of H-fixed points in V.

Task: Given $g \in G$, determine the action of ege on Ve, without the explicit computation of the action of g on V.

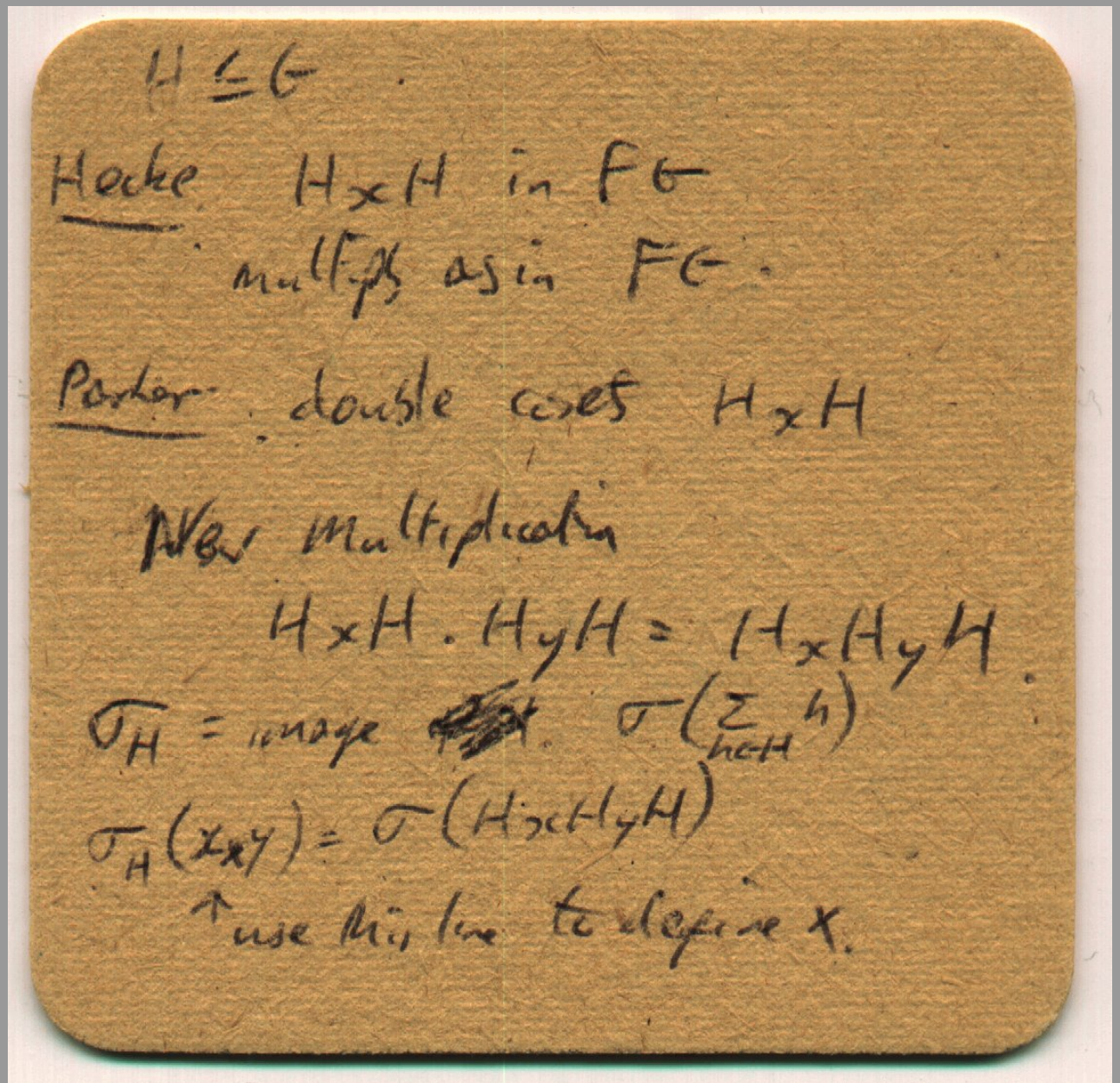
Let $\Omega_1, \ldots, \Omega_m$ be the H-orbits on Ω.

The orbits sums $\sum_{\omega \in \Omega_j} \omega \in V$, $j = 1, \ldots, m$, form a basis of Ve.

Then the (i, j)-entry of the matrix of ege on Ve equals

$$\frac{1}{|\Omega_j|} |\Omega_i.g \cap \Omega_j|.$$

CONDENSATION: HISTORY, I



$H \leq G$.

<u>Hecke</u> HxH in FG

mult'ps as in FG.

<u>Parker</u>: double cosets HxH

New multiplication

$$HxH \cdot HyH = HxHyH.$$

σ_H = image $\sigma\left(\sum_{h \in H} h\right)$

$\sigma_H(x \times y) = \sigma(HxHyH)$

↑ use this line to define $\times$.

CONDENSATION: HISTORY, II

Thank you for your attention!