The Computer Algebra System GAP

An overview with examples

Max Neunhöffer

max.neunhoeffer@math.rwth-aachen.de

What is GAP?

GAP is a computer algebra system . . .

- specialized but not limited to group theory and representation theory of groups and algebras
- intended for research and teaching purposes
- with a programming language that is easy to learn and yet powerful
- a huge collection of mathematical knowledge: in form of algorithms and data collections
- free software, distributed under the GNU public license
- well documented: >800 pages reference manual, tutorial, online help system
- mature system with an active group of developpers, who give support in case of problems or bugs.

History

- Initiated by Joachim Neubüser at Lehrstuhl D für Mathematik at RWTH Aachen in 1984.
- From beginning on freely available, with full source code.
- Implementation first by diploma students and PHD students.
- Today „the GAP Group“.
- 1994–1997: GAP versions 3.1 to 3.4.4
- 1997: Transfer of responsibility for GAP to St. Andrews.
- 1997–2002: GAP versions 4.1 to 4.3, major redesign.
- Packages!!! (user contributions, refereed)

Basic Data

Integers and Rationals

```
gap> 2^(4^3);
18446744073709551616
gap> 3+4
> /5;
19/5
gap> 3+4/*5;
Syntax error: expression expected
3+4/*5;
    ^
gap> 3+4/5;
19/5
gap> 2^(4^3);
18446744073709551616
gap> last*last;
340282366920938463463374607431768211456
```

Cyclotomic and finite fields

```
gap> F := GF(4);
GF(2^2)
gap> Elements(F);
[ 0*Z(2), Z(2)^0, Z(2^2), Z(2^2)^2 ]
gap> Z(2^2)+Z(2)^0;
Z(2^2)^2
gap> i := E(4);;
gap> i*i;
-1
gap> (1+i)^3;
-2+2*E(4)
gap> E(9)+E(4);
-E(36)^16-E(36)^21-E(36)^28-E(36)^33
gap> E(9)^3;
E(3)
```

Lists, sets, vectors, and matrices

```
gap> [1,2,3] * 2;
[ 2, 4, 6 ]
gap> v := Set([3,2,1]);
[ 1, 2, 3 ]
gap> Difference(v,[2,4]);
[ 1, 3 ]
gap> M := [[7,8,9],[4,5,6],[1,2,3]];;
gap> v * M;
[ 18, 24, 30 ]
gap> Display(M*M);
[ [   90,  114,  138 ],
  [   54,   69,   84 ],
  [   18,   24,   30 ] ]
gap> Print(M[2]," and ",M[2][3],".\n");
[ 4, 5, 6 ] and 6.
```

Permutations

```
gap> p := (1,2)*(2,3)(4,5);
(1,3,2)(4,5)
gap> ListPerm(last);
[ 3, 1, 2, 5, 4 ]
gap> PermList([4,3,5,1,2]);
(1,4)(2,3,5)
gap> last*last;
(2,5,3)
gap> 5^p;
4
gap> (2,5,3)^-1;
(2,3,5)
gap> SignPerm((1,2)*(2,3)*(1,2));
-1
gap> (1,2)^p = p^-1 * (1,2) * p;
true
```

Groups and group actions

Permutation groups

```
gap> m12 := MathieuGroup(12);
Group([ (1,2,3,4,5,6,7,8,9,10,11), (3,7,11,8)(4,10,5,6),
  (1,12)(2,11)(3,6)(4,8)(5,9)(7,10) ])
gap> Orbit(m12,1,OnPoints);
[ 1, 2, 12, 3, 11, 4, 7, 6, 8, 5, 10, 9 ]
gap> m10 := MathieuGroup(10);
Group([ (1,9,6,7,5)(2,10,3,8,4), (1,10,7,8)(2,9,4,6) ])
gap> IsSubgroup(m12,m10);
true
gap> Orbits(m10,[1..12],OnPoints);
[ [ 1, 9, 10, 6, 4, 3, 7, 2, 8, 5 ], [ 11 ], [ 12 ] ]
gap> Index(m12,m10);
132
gap> rc := RightCosets(m12,m10);;
gap> rc[17];
RightCoset(Group( [ ( 1, 9, 6, 7, 5)( 2,10, 3, 8, 4),
  ( 1,10, 7, 8)( 2, 9, 4, 6) ] ),( 1,11,10, 7, 9, 5, 6,12, 3, 2)
( 4, 8))
gap> hom := ActionHomomorphism(m12,rc,OnRight);
<action homomorphism>
gap> Source(hom);
Group([ (1,2,3,4,5,6,7,8,9,10,11), (3,7,11,8)(4,10,5,6),
  (1,12)(2,11)(3,6)(4,8)(5,9)(7,10) ])
```

Permutation groups

```
gap> Range(hom);
Sym( [ 1 .. 132 ] )
gap> m12on132 := Image(hom);
<permutation group with 3 generators>
gap> Size(m12on132);
95040
gap> dom := Cartesian([1..132],[1..132]);;
gap> dom{[1..17]};
[ [ 1, 1 ], [ 1, 2 ], [ 1, 3 ], [ 1, 4 ], [ 1, 5 ], [ 1, 6 ],
  [ 1, 7 ], [ 1, 8 ], [ 1, 9 ], [ 1, 10 ], [ 1, 11 ], [ 1, 12 ],
  [ 1, 13 ], [ 1, 14 ], [ 1, 15 ], [ 1, 16 ], [ 1, 17 ] ]
gap> Length(dom);
17424
gap> o := Orbits(m12on132,dom,OnPairs);;
gap> time;
1120
gap> List(o,Length);
[ 132, 1320, 1320, 11880, 1320, 132, 1320 ]
gap> GeneratorsOfGroup(m12on132);
[ (1,66,100,78,130,89,45,111,67,34,12)(2,60,107,80,122,90,46,112,
    68,35,13)(3,65,110,86,128,96,52,113,69,36,14)(4,56,109,84,125,
    94,50,114,70,37,15)(5,59,101,81,124,99,55,115,71,38,16)(6,64,
  ...
```

Finitely presented groups

```
gap> f := FreeGroup("a","b","c","d");
<free group on the generators [ a, b, c, d ]>
gap> a := f.a;; b := f.b;; c := f.c;; d := f.d;;
gap> rels:=[a*a,b*b,c*c,d*d,a*c*(c*a)^-1,b*d*(d*b)^-1,a*d*(d*a)^-1
> ,a*b*a*(b*a*b)^-1,b*c*b*(c*b*c)^-1,c*d*c*(d*c*d)^-1];
[ a^2, b^2, c^2, d^2, a*c*a^-1*c^-1, b*d*b^-1*d^-1, a*d*a^-1*d^-1,
  a*b*a*b^-1*a^-1*b^-1, b*c*b*c^-1*b^-1*c^-1, c*d*c*d^-1*c^-1*d^-1
 ]
gap> g := f/rels;
<fp group on the generators [ a, b, c, d ]>
gap> KnownAttributesOfObject(g);
[ "GeneratorsOfMagmaWithInverses", "MultiplicativeNeutralElement" ]
gap> KnownPropertiesOfObject(g);
[ "IsWholeFamily", "IsDuplicateFree",
  "IsGeneratorsOfMagmaWithInverses", "IsAssociative" ]
gap> Size(g);
120
gap> KnownAttributesOfObject(g);
[ "Size", "GeneratorsOfMagmaWithInverses",
  "MultiplicativeNeutralElement", "FreeGeneratorsOfFpGroup",
  "RelatorsOfFpGroup", "FreeGroupOfFpGroup",
  "IndicesInvolutaryGenerators" ]
```

Attributes and Properties

```
gap> KnownPropertiesOfObject(g);
[ "IsEmpty", "IsTrivial", "IsNonTrivial", "IsFinite",
  "IsWholeFamily", "IsDuplicateFree",
  "IsGeneratorsOfMagmaWithInverses", "IsAssociative",
  "IsSubsetLocallyFiniteGroup" ]
gap> HasIsTrivial(g);
true
gap> IsTrivial(g);
false
gap> IsAbelian(g);
false
gap> IsSolvable(g);
false
gap> IdGroup(g);
[ 120, 34 ]
gap> SmallGroup(120,34);
Group([ (1,2,3,4,5), (1,2) ])
gap> i := IsomorphismPermGroup(g);
[ a, b, c, d ] -> [ (4,5), (3,4), (2,3), (1,2) ]
gap> Image(i);
Group([ (4,5), (3,4), (2,3), (1,2) ])
```

The Small Groups Library

```
gap> NrSmallGroups(120);
47
gap> l := AllSmallGroups(120);;
gap> List(l,IsAbelian);
[ false, false, false, true, false, false, false, false, false,
  false, false, false, false, false, false, false, false, false,
  false, false, false, false, false, false, false, false, false,
  false, false, false, true, false, false, false, false, false,
  false, false, false, false, false, false, false, false, false,
  false, true ]
gap> List(l,IsSolvable);
[ true, true, true, true, false, true, true, true, true, true,
  true, true, true, true, true, true, true, true, true, true,
  true, true, true, true, true, true, true, true, true, true,
  true, true, true, false, false, true, true, true, true, true,
  true, true, true, true, true, true, true ]
gap> gg := SmallGroup(120,5);
Group(
[ (1,2,4,8)(3,6,9,5)(7,12,13,17)(10,14,11,15)(16,20,21,24)(18,22,
    19,23), (1,3,7)(2,5,10)(4,9,13)(6,11,8)(12,16,20)(14,18,22)(15,
    19,23)(17,21,24) ])
gap> Length(ConjugacyClasses(gg));
9
```

The Small Groups Library

```
gap> ll := NormalSubgroups(gg);
[ Group(()),
  Group([ (1,4)(2,8)(3,9)(5,6)(7,13)(10,11)(12,17)(14,15)(16,
        21)(18,19)(20,24)(22,23) ]),
  Group([ (1,2,4,8)(3,6,9,5)(7,12,13,17)(10,14,11,15)(16,20,21,
        24)(18,22,19,23), (1,3,7)(2,5,10)(4,9,13)(6,11,8)(12,16,
        20)(14,18,22)(15,19,23)(17,21,24) ]) ]
gap> List(last,Size);
[ 1, 2, 120 ]
gap> ggg := gg/ll[2];
Group([ (1,2)(3,4)(5,7)(6,8)(9,11)(10,12),
  (1,3,5)(2,4,6)(7,9,11)(8,10,12) ])
gap> IdGroup(ggg);
[ 60, 5 ]
gap> IdGroup(AlternatingGroup(5));
[ 60, 5 ]
```

Data Collections in GAP

Data Collections in GAP

- SmallGroups: All groups of order ≤ 2000 (except 1024), altogether $423\,164\,062$ groups (Besche, Eick, O'Brien).
- Transitive Groups: Up to degree 30 (Hulpke).
- Primitive Groups: Up to degree 256, completeness guaranteed up to degree 50 (Sims, Short, Theißen).
- Character Tables: All ATLAS tables and more, Brauer tables (Breuer).
- Tables of Marks: Compact description of subgroup lattice (Breuer, Merkwitz).
- WWW ATLAS of Finite Group Representations: Wilson, Walsh, Tripp, Suleiman, Rogers, Parker, Norton, Linton, Bray, Breuer.

Writing your own programs...

Writing your own programs...

```
mygcd := function(a,b)
  local r;
  repeat
    r := RemInt(a,b);
    a := b;
    b := r;
    Print("a=",a," b=",b,"\n");
  until r = 0;
  return a;
end;
gap> mygcd(Fibonacci(10),Fibonacci(11));
a=89 b=55
a=55 b=34
a=34 b=21
a=21 b=13
a=13 b=8
a=8 b=5
a=5 b=3
a=3 b=2
a=2 b=1
a=1 b=0
1
```

Integer Factorization

The GAP Package FactInt by Stefan Kohl

```
gap> RequirePackage("factint");

Loading FactInt 1.2 (Routines for Integer Factorization)
by Stefan Kohl, kohl@mathematik.uni-stuttgart.de
true
gap> a := Factorial(34)-1;
295232799039604140847618609643519999999
gap> FactInfo(1);
gap> IntegerFactorization(a);
#I  Check for n = b^k +/- 1
#I  Factors already found : [  ]
#I
#I  Trial division by all primes p < 1000
#I  Trial division by some already known primes
#I  Check for perfect powers
#I  Pollard's Rho
Steps = 16384, Cluster = 161
Number to be factored :
295232799039604140847618609643519999999
#I  Pollard's p - 1
Limit1 = 10000, Limit2 = 400000
Number to be factored :
295232799039604140847618609643519999999
```

The GAP Package FactInt by Stefan Kohl

```
#I  Williams' p + 1
Residues = 2, Limit1 = 2000, Limit2 = 80000
Number to be factored :
29523279903960414084761860964351999999
#I  Elliptic Curves Method (ECM)
Curves = <func.>
Init. Limit1 = <func.>, Init. Limit2 = <func.>, Delta = <func.>
Number to be factored :
29523279903960414084761860964351999999
#I  Multiple Polynomial Quadratic Sieve (MPQS)
Number to be factored :
29523279903960414084761860964351999999
#I  The factors are
[ 2839169786733973241, 10398560889846739639 ]
#I  Intermediate result :
[ [ 10398560889846739639, 2839169786733973241 ], [  ] ]
#I
#I  The result is
[ [ 10398560889846739639, 2839169786733973241 ], [  ] ]

[ 10398560889846739639, 2839169786733973241 ]
gap> time;
23750
```

Character tables

The Character Table of S_5

```
gap> c := CharacterTable("Symmetric",5);
CharacterTable( "Sym(5)" )
gap> Display(c);
Sym(5)

     2  3  2  3  1  1  2  .
     3  1  1  .  1  1  .  .
     5  1  .  .  .  .  .  1

       1a 2a 2b 3a 6a 4a 5a
    2P 1a 1a 1a 3a 3a 2b 5a
    3P 1a 2a 2b 1a 2a 4a 5a
    5P 1a 2a 2b 3a 6a 4a 1a

X.1     1 -1  1  1 -1 -1  1
X.2     4 -2  .  1  1  . -1
X.3     5 -1  1 -1 -1  1  .
X.4     6  . -2  .  .  .  1
X.5     5  1  1 -1  1 -1  .
X.6     4  2  .  1 -1  . -1
X.7     1  1  1  1  1  1  1
```

The Character Table Library

```
gap> CharacterParameters(c);
[ [ 1, [ 1, 1, 1, 1, 1 ] ], [ 1, [ 2, 1, 1, 1 ] ],
  [ 1, [ 2, 2, 1 ] ], [ 1, [ 3, 1, 1 ] ], [ 1, [ 3, 2 ] ],
  [ 1, [ 4, 1 ] ], [ 1, [ 5 ] ] ]
gap> ClassParameters(c);
[ [ 1, [ 1, 1, 1, 1, 1 ] ], [ 1, [ 2, 1, 1, 1 ] ],
  [ 1, [ 2, 2, 1 ] ], [ 1, [ 3, 1, 1 ] ], [ 1, [ 3, 2 ] ],
  [ 1, [ 4, 1 ] ], [ 1, [ 5 ] ] ]
gap> B := CharacterTable("B");
CharacterTable( "B" )
gap> Size(B);
4154781481226426191177580544000000
gap> fi23 := CharacterTable("Fi23");
CharacterTable( "Fi23" )
gap> Size(fi23);
4089470473293004800
gap> Size(B)/Size(fi23);
1015970529280000
gap> 10^15;
1000000000000000
```

The Character Table of Fi_{23}

```
gap> List(Irr(fi23),c->c[1]);
[ 1, 782, 3588, 5083, 25806, 30888, 60996, 106743, 111826, 274482,
  279565, 752675, 789360, 812889, 837200, 837200, 850850, 850850,
  1677390, 1951872, 2236520, 2322540, 3913910, 5533110, 6709560,
  7468032, 8783424, 9108736, 9108736, 10567557, 10674300,
  12077208, 15096510, 17892160, 18812574, 20322225, 21135114,
  21348600, 22644765, 26838240, 28464800, 29354325, 35225190,
  37573536, 40840800, 42270228, 48034350, 48308832, 55740960,
  56360304, 56360304, 57254912, 58708650, 58708650, 65875680,
  73531392, 73531392, 78278200, 93933840, 97976320, 133398252,
  153014400, 153014400, 166559744, 166559744, 176125950,
  176125950, 203802885, 207793431, 211351140, 216154575,
  216770400, 244563462, 263376036, 264188925, 264536064,
  264536064, 286274560, 287721720, 289027200, 289027200,
  289103904, 313112800, 313112800, 313112800, 322058880,
  336061440, 341577600, 343529472, 352251900, 362316240,
  476702577, 496897335, 504627200, 504627200, 526752072,
  528377850, 559458900 ]
gap> ch := Irr(fi23)[1];
Character( CharacterTable( "Fi23" ),
[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
...
```

Restriction of Characters

```
gap> chars := List(Irr(B),c->RestrictedClassFunction(c,fi23));;
gap> l := List(chars,c->ScalarProduct(c,ch));
[ 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
gap> chi := l * Irr(B);
VirtualCharacter( CharacterTable( "B" ),
[ 101597052928000, 2370830336, 4823449600, 0, 36700160, 10278400,
  69160, 0, 0, 0, 32768, 286720, 0, 65536, 8192, 0, 0, 17600, 0,
  2816, 22400, 20992, 1160, 3968, 0, 40, 1280, 2600, 392, 0, 672,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 512, 256, 0, 0, 0, 13, 76, 336, 320,
  0, 0, 80, 0, 30, 0, 0, 0, 0, 32, 40, 0, 128, 0, 64, 8, 160, 0,
  52, 0, 32, 8, 0, 20, 0, 8, 0, 56, 0, 32, 0, 50, 0, 0, 0, 0, 0,
  0, 0, 16, 0, 4, 2, 14, 4, 2, 5, 0, 0, 0, 0, 0, 0, 8, 16, 0, 0,
  0, 0, 6, 6, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 4, 0, 0,
  0, 0, 4, 1, 0, 0, 0, 8, 0, 0, 6, 0, 2, 8, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 2, 0, 2, 1, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0 ] )
```

A huge Application

A huge Application

We want to study the permutation module P for the Baby-Monster B acting on the 1 015 970 529 280 000 right cosets of $Fi_{23} < B$ (Jürgen Müller and Rob Wilson).

Aim: Fi_{23} suborbit lengths, character table of End P.

- Representation of B of dimension 4371 over GF(2).
- One vector needs 500 bytes $\Longrightarrow$ 462009 terabyte!
- Use helper subgroup $U_3 < Fi_{23}$ of order $\sim 10^{10}$, some nice mathematical tricks and „archive“ U_3-suborbits.
- Use theory and luck to find Fi_{23} suborbits and enumerate them.
- Use GAP, standard PCs (2 GB), some weeks of CPU-time.
- Get suborbit lengths and character table.

Fi_{23} suborbits in B orbit on $Fi_{23} \backslash B$

No.	Length	Stabilizer Order	Iso-Type
1	1	4.089.470.473.293.004.800	Fi_{23}
2	12.678.169.870.080	322.560	$2^2.L_3(4).2^2$
3	283.991.005.089.792	14.400	$(A_5 \times A_5) : (2 \times 2)$
4	190.172.548.051.200	21.504	$2^6 : (L_3(2) : 2)$
5	262.954.634.342.400	15.552	$(3^4 : 2^{1+4}).S_3$
6	43.028.940.165.120	95.040	M_{12}
7	21.514.470.082.560	190.080	$2 \times M_{12}$
8	7.888.639.030.272	518.400	$(A_6 \times A_6) : (2 \times 2)$
9	50.712.679.480.320	80.640	$2 \times (2.L_3(4))$
10	33.816.182.400	120.932.352	$3.3^8.2.2^5.2^4.3^2.2$
11	1.044.084.577.536	3.916.800	$S_4(4) : 4$
12	133.120.783.635.840	30.720	$2^4.2^4.A_5.2$
13	5.282.570.779.200	774.144	$2^7.(U_3(3) : 2)$
14	1.584.771.233.760	2.580.480	$2^6.(2 \times A_8)$
15	86.316.516	47.377.612.800	$S_8(2)$
16	412.896	9.904.359.628.800	$O_8^+(3).2_2$
17	195.747.435	20.891.566.080	$2^{11}.M_{23}$
18	8.537.488.128	479.001.600	S_{12}
19	113.778.447.552	35.942.400	$2 \times {}^2F_4(2)'$
20	160.533.964.800	25.474.176	$S_3 \times G_2(3)$
21	1.152.560.897.280	3.548.160	$2 \times 2.M_{22}.2$
22	504.245.392.560	8.110.080	$2^{10}.M_{11}$
23	23.478.092.352	174.182.400	$O_8^+(2)$

http://www.gap-system.org/