Polycyclic groups in GAP

Bettina Eick

TU Braunschweig – Germany

Gap Days in Brussels, April 2025, Talk 1

Introduction

Introduction

Polycyclic groups

Definition

A group G is polycyclic if it has a subnormal series

$$G = G_1 \trianglerighteq G_2 \trianglerighteq \ldots \trianglerighteq G_n \trianglerighteq G_{n+1} = \{1\}$$

whose quotients G_i/G_{i+1} are cyclic for $1 \leq i \leq n$.

First comments

- The quotient G_i/G_{i+1} can be finite cyclic or infinite cyclic.

Polycyclic groups

Definition

A group G is polycyclic if it has a subnormal series

$$G = G_1 \trianglerighteq G_2 \trianglerighteq \ldots \trianglerighteq G_n \trianglerighteq G_{n+1} = \{1\}$$

whose quotients G_i/G_{i+1} are cyclic for $1 \leq i \leq n$.

First comments

- The quotient G_i/G_{i+1} can be finite cyclic or infinite cyclic.
- Example: Each finite solvable group is polycyclic.

Polycyclic groups

Definition

A group G is polycyclic if it has a subnormal series

$$G = G_1 \trianglerighteq G_2 \trianglerighteq \ldots \trianglerighteq G_n \trianglerighteq G_{n+1} = \{1\}$$

whose quotients G_i/G_{i+1} are cyclic for $1 \leq i \leq n$.

First comments

- The quotient G_i/G_{i+1} can be finite cyclic or infinite cyclic.
- Example: Each finite solvable group is polycyclic.
- Example: Space groups with solvable point groups are polycyclic.

Polycyclic groups

Definition

A group G is polycyclic if it has a subnormal series

$$G = G_1 \trianglerighteq G_2 \trianglerighteq \ldots \trianglerighteq G_n \trianglerighteq G_{n+1} = \{1\}$$

whose quotients G_i/G_{i+1} are cyclic for $1 \leq i \leq n$.

First comments

- The quotient G_i/G_{i+1} can be finite cyclic or infinite cyclic.
- Example: Each finite solvable group is polycyclic.
- Example: Space groups with solvable point groups are polycyclic.
- Example: Each finitely generated nilpotent group is polycyclic.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.
- Then $\mathcal{G} = (g_1, \ldots, g_n)$ is a *polycyclic generating set (Pcgs)* for G.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.
- Then $\mathcal{G} = (g_1, \ldots, g_n)$ is a *polycyclic generating set (Pcgs)* for G.
- Note that $G_i = \langle g_i, \ldots, g_n \rangle$ holds.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.
- Then $\mathcal{G} = (g_1, \ldots, g_n)$ is a *polycyclic generating set (Pcgs)* for G.
- Note that $G_i = \langle g_i, \ldots, g_n \rangle$ holds.
- Let $r_i = [G_i : G_{i+1}]$.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.
- Then $\mathcal{G} = (g_1, \ldots, g_n)$ is a *polycyclic generating set (Pcgs)* for G.
- Note that $G_i = \langle g_i, \ldots, g_n \rangle$ holds.
- Let $r_i = [G_i : G_{i+1}]$.
- Then $(r_1, \ldots, r_n)$ are the *relative orders* associated with $\mathcal{G}$.

Polycyclic generating sets

Polycyclic generating sets

Let G be polycyclic with series $G = G_1 \trianglerighteq \ldots \trianglerighteq G_{n+1}$.

- Choose $g_i \in G$ with $G_i = \langle g_i, G_{i+1} \rangle$.
- Then $\mathcal{G} = (g_1, \ldots, g_n)$ is a *polycyclic generating set (Pcgs)* for G.
- Note that $G_i = \langle g_i, \ldots, g_n \rangle$ holds.
- Let $r_i = [G_i : G_{i+1}]$.
- Then $(r_1, \ldots, r_n)$ are the *relative orders* associated with $\mathcal{G}$.
- Note that $|G| = r_1 \cdots r_n$ (finite or infinite).

Easy applications

Easy applications

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for G with relative orders $(r_1, \ldots, r_n)$.

Easy applications

Easy applications

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for G with relative orders $(r_1, \ldots, r_n)$.

- Each $g \in G$ can be written uniquely as

$$g = g_1^{e_1} \cdots g_n^{e_n}$$

  with $e_i \in \mathbb{Z}$ and $e_i \in \{0, \ldots, r_i - 1\}$ if $r_i < \infty$.

Easy applications

Easy applications

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for G with relative orders $(r_1, \ldots, r_n)$.

- Each $g \in G$ can be written uniquely as

$$g = g_1^{e_1} \cdots g_n^{e_n}$$

  with $e_i \in \mathbb{Z}$ and $e_i \in \{0, \ldots, r_i - 1\}$ if $r_i < \infty$.
- This is the *normal form* of g.

Easy applications

Easy applications

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for G with relative orders $(r_1, \ldots, r_n)$.

- Each $g \in G$ can be written uniquely as
$$g = g_1^{e_1} \cdots g_n^{e_n}$$
with $e_i \in \mathbb{Z}$ and $e_i \in \{0, \ldots, r_i - 1\}$ if $r_i < \infty$.
- This is the *normal form* of g.
- We call $exp(g) = (e_1, \ldots, e_n)$ the *exponent vector* of g.

Easy applications

Easy applications

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for G with relative orders $(r_1, \ldots, r_n)$.

- Each $g \in G$ can be written uniquely as

  $$g = g_1^{e_1} \cdots g_n^{e_n}$$

  with $e_i \in \mathbb{Z}$ and $e_i \in \{0, \ldots, r_i - 1\}$ if $r_i < \infty$.
- This is the *normal form* of g.
- We call $exp(g) = (e_1, \ldots, e_n)$ the *exponent vector* of g.
- If d is minimal with $e_d \neq 0$, then $d = dep(g)$ is the *depth* of g.

Examples

Examples

Examples

Examples

- $G = S_4$ is finite solvable, hence polycyclic.

Examples

Examples

- $G = S_4$ is finite solvable, hence polycyclic.
- $G = D_\infty \leq GL(2, \mathbb{Z})$ is infinite polycyclic.

Examples

Examples

- $G = S_4$ is finite solvable, hence polycyclic.
- $G = D_\infty \leq GL(2, \mathbb{Z})$ is infinite polycyclic.
- The upper unitriangular matrices in $GL(n, \mathbb{Z})$ are finitely generated nilpotent, hence polycyclic.

Pc-presentations

Pcgs to Pc-Presentation

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for the finite group G with relative orders $(r_1, \ldots, r_n)$.

Pc-presentations

Pcgs to Pc-Presentation

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for the finite group G with relative orders $(r_1, \ldots, r_n)$.

- Write $exp(g_i^{r_i}) = (e_{i,1}, \ldots, e_{i,n})$, and

Pc-presentations

Pcgs to Pc-Presentation

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for the finite group G with relative orders $(r_1, \ldots, r_n)$.

- Write $exp(g_i^{r_i}) = (e_{i,1}, \ldots, e_{i,n})$, and
- Write $exp(g_i^{g_j}) = (e_{j,i,1}, \ldots, e_{j,i,n})$ for $j < i$.

Pc-presentations

Pcgs to Pc-Presentation

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for the finite group G with relative orders $(r_1, \ldots, r_n)$.

- Write $exp(g_i^{r_i}) = (e_{i,1}, \ldots, e_{i,n})$, and
- Write $exp(g_i^{g_j}) = (e_{j,i,1}, \ldots, e_{j,i,n})$ for $j < i$.
- Then G has a presentation on the generators $g_1, \ldots, g_n$ with the relations

$$\begin{array}{rcll} g_i^{r_i} & = & g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} & 1 \leq i \leq n \\ g_i g_j & = & g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} & 1 \leq j < i \leq n \end{array}$$

Pc-presentations

Pcgs to Pc-Presentation

Let $\mathcal{G} = (g_1, \ldots, g_n)$ be a pcgs for the finite group G with relative orders $(r_1, \ldots, r_n)$.

- Write $exp(g_i^{r_i}) = (e_{i,1}, \ldots, e_{i,n})$, and
- Write $exp(g_i^{g_j}) = (e_{j,i,1}, \ldots, e_{j,i,n})$ for $j < i$.
- Then G has a presentation on the generators $g_1, \ldots, g_n$ with the relations

$$\begin{array}{rcl} g_i^{r_i} & = & g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} \quad 1 \leq i \leq n \\ g_i g_j & = & g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} \quad 1 \leq j < i \leq n \end{array}$$

- This is a Pc-presentation for G.

Pc-presentations II

Comments

Pc-presentations II

Comments

- Example: S_3
  Pcgs $(g_1, g_2) = ((1,2),(1,2,3))$ with relative orders $(2,3)$ and relations

$$\begin{aligned} g_1^2 &= g_2^3 = 1 \\ g_2 g_1 &= g_1 g_2^2 \end{aligned}$$

Pc-presentations II

Comments

- Example: S_3
  Pcgs $(g_1, g_2) = ((1,2),(1,2,3))$ with relative orders $(2,3)$ and relations

$$\begin{aligned} g_1^2 &= g_2^3 = 1 \\ g_2 g_1 &= g_1 g_2^2 \end{aligned}$$

- Note that the relations allow to determine normal forms (Algorithm *collection*)

Pc-presentations II

Comments

- Example: S_3
  Pcgs $(g_1, g_2) = ((1,2),(1,2,3))$ with relative orders $(2,3)$ and relations

$$\begin{aligned} g_1^2 &= g_2^3 = 1 \\ g_2 g_1 &= g_1 g_2^2 \end{aligned}$$

- Note that the relations allow to determine normal forms (Algorithm *collection*)
- Also infinite polycyclic groups have Pc-presentations (Example D_∞)

Pc-presentations III

Pc-Presentation to Pcgs

Let F be free on $\mathcal{G} = (g_1, \ldots, g_n)$ and let $(r_1, \ldots, r_n) \in \mathbb{N}^n$. Let G be finitely presented on $\mathcal{G}$ with relations

$$\begin{array}{rcl} g_i^{r_i} & = & g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} \;\; 1 \leq i \leq n \\ g_i g_j & = & g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} \;\; 1 \leq j < i \leq n \end{array}$$

for certain $e_{i,k}, e_{j,i,k} \in \{0, \ldots, r_k - 1\}$.

Pc-presentations III

Pc-Presentation to Pcgs

Let F be free on $\mathcal{G} = (g_1, \ldots, g_n)$ and let $(r_1, \ldots, r_n) \in \mathbb{N}^n$. Let G be finitely presented on $\mathcal{G}$ with relations

$$\begin{array}{rcl} g_i^{r_i} & = & g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} \;\; 1 \leq i \leq n \\ g_i g_j & = & g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} \;\; 1 \leq j < i \leq n \end{array}$$

for certain $e_{i,k}, e_{j,i,k} \in \{0, \ldots, r_k - 1\}$.

- Then G is finite polycyclic with pcgs $\mathcal{G}$.

Pc-presentations III

Pc-Presentation to Pcgs

Let F be free on $\mathcal{G} = (g_1, \ldots, g_n)$ and let $(r_1, \ldots, r_n) \in \mathbb{N}^n$. Let G be finitely presented on $\mathcal{G}$ with relations

$$\begin{array}{rcl} g_i^{r_i} & = & g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} \;\; 1 \leq i \leq n \\ g_i g_j & = & g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} \;\; 1 \leq j < i \leq n \end{array}$$

for certain $e_{i,k}, e_{j,i,k} \in \{0, \ldots, r_k - 1\}$.

- Then G is finite polycyclic with pcgs $\mathcal{G}$.
- The relative orders $(s_1, \ldots, s_n)$ of $\mathcal{G}$ satisfy $s_i \leq r_i$.

Pc-presentations III

Pc-Presentation to Pcgs

Let F be free on $\mathcal{G} = (g_1, \ldots, g_n)$ and let $(r_1, \ldots, r_n) \in \mathbb{N}^n$. Let G be finitely presented on $\mathcal{G}$ with relations

$$\begin{aligned} g_i^{r_i} &= g_{i+1}^{e_{i,i+1}} \cdots g_n^{e_{i,n}} \quad 1 \leq i \leq n \\ g_i g_j &= g_j g_{j+1}^{e_{j,i,j+1}} \cdots g_n^{e_{j,i,n}} \quad 1 \leq j < i \leq n \end{aligned}$$

for certain $e_{i,k}, e_{j,i,k} \in \{0, \ldots, r_k - 1\}$.

- Then G is finite polycyclic with pcgs $\mathcal{G}$.
- The relative orders $(s_1, \ldots, s_n)$ of $\mathcal{G}$ satisfy $s_i \leq r_i$.
- It is a *consistent* Pc-presentation if $s_i = r_i$ for $1 \leq i \leq n$ holds.

GAP Session 1

GAP Session 1

Permutation groups I

Permutation Groups I

```
gap> G := SymmetricGroup(3);
Sym( [ 1 .. 3 ] )
gap> IsomorphismPcGroup(G);
Pcgs([ (2,3), (1,2,3) ]) -> [ f1, f2 ]
gap> H := Image(last);
Group([ f1, f2 ])
gap> PrintPcpPresentation(PcGroupToPcpGroup(H));
g1^2 = id
g2^3 = id
g2 ^ g1 = g2^2
```

Permutation groups II

Permutation Groups II

```
gap> Elements(G);
[ (), (2,3), (1,2), (1,2,3), (1,3,2), (1,3) ]
gap> Elements(H);
[ <identity> of ..., f1, f2, f1*f2, f2^2, f1*f2^2 ]

gap> h := Pcgs(H);
Pcgs([ f1, f2 ])
gap> RelativeOrders(h);
[ 2, 3 ]
gap> w := h[2]*h[1];
f1*f2^2
gap> ExponentsOfPcElement(h,w);
[ 1, 2 ]
```

Permutation groups

Permutation Groups

```
gap> G := SymmetricGroup(100);
Sym( [ 1 .. 100 ] )
gap> Collected(Factors(Size(G)));
[ [2, 97], [3, 48], [5, 24], [7, 16], [11, 9], [13, 7],
  [17, 5], [19, 5], [23, 4], [29, 3], [31, 3], [37, 2],
  [41, 2], [43, 2], [47, 2], [53, 1], [59, 1], [61, 1],
  [67, 1], [71, 1], [73, 1], [79, 1], [83, 1], [89, 1],
  [97, 1] ]
gap> H := SylowSubgroup(G, 7);
<permutation group of size 33232930569601 ...>
gap> iso := IsomorphismPcGroup(H);;
gap> U := Image(iso);
<pc group of size 33232930569601 with 16 generators>
gap> RelativeOrders(Pcgs(U));
[ 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7 ]
```

Matrix groups

Matrix Groups

```
gap> G := GL(4,9);
GL(4,9)
gap> U := SylowSubgroup(G, 3);
<matrix group of size 531441 with 6 generators>
gap> H := Image(IsomorphismPcGroup(U));
Group([f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12])
gap> LowerCentralSeries(H);
[ Group([ f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12 ]),
  Group([ f4*f6^2,f5,f8*f11^2,f9*f11*f12,f10,f11*f12^2 ]),
  Group([ f8*f11^2, f10 ]),
  Group([ <identity> of ... ]) ]
```

SmallGroups Library

SmallGroups Library in GAP

```
gap> ll := AllSmallGroups(8);
[ <pc group of size 8 with 3 generators>,
  <pc group of size 8 with 3 generators>,
  <pc group of size 8 with 3 generators>,
  <pc group of size 8 with 3 generators>,
  <pc group of size 8 with 3 generators> ]

gap> List(ll, StructureDescription);
[ "C8", "C4 x C2", "D8", "Q8", "C2 x C2 x C2" ]

gap> PrintPcpPresentation(PcGroupToPcpGroup(ll[4]));
g1^2 = g3
g2^2 = g3
g3^2 = id
g2 ^ g1 = g2 * g3
```

Algorithms

Algorithms

What can one do with Pc-groups?

Algorithms

There are many algorithms available for Pc-groups

- Compute centralizers, normalizers and intersections

What can one do with Pc-groups?

Algorithms

There are many algorithms available for Pc-groups

- Compute centralizers, normalizers and intersections
- Compute lower central or derived series

What can one do with Pc-groups?

Algorithms

There are many algorithms available for Pc-groups

- Compute centralizers, normalizers and intersections
- Compute lower central or derived series
- Compute Sylow subgroups, Hall subgroups, Frattini subgroup

What can one do with Pc-groups?

Algorithms

There are many algorithms available for Pc-groups

- Compute centralizers, normalizers and intersections
- Compute lower central or derived series
- Compute Sylow subgroups, Hall subgroups, Frattini subgroup
- Compute maximal subgroups

What can one do with Pc-groups?

Algorithms

There are many algorithms available for Pc-groups

- Compute centralizers, normalizers and intersections
- Compute lower central or derived series
- Compute Sylow subgroups, Hall subgroups, Frattini subgroup
- Compute maximal subgroups
- ... and many more

Effectivity

Note

Algorithms for Pc-groups often proceed by induction

- upwards along the defining subnormal series
  (example: orbits + stabilizers)

Effectivity

Note

Algorithms for Pc-groups often proceed by induction

- upwards along the defining subnormal series
  (example: orbits + stabilizers)
- downwards along a normal series with el.-ab. quotients
  (example: conjugacy classes + centralizers)

Some comments

Comments

- Finite solvable groups: Pc-groups in GAP

Some comments

Comments

- Finite solvable groups: Pc-groups in GAP
- It is assumed that the relative orders are all primes

Some comments

Comments

- Finite solvable groups: Pc-groups in GAP
- It is assumed that the relative orders are all primes
- Infinite polycyclic groups: Pcp-groups in GAP

Some comments

Comments

- Finite solvable groups: Pc-groups in GAP
- It is assumed that the relative orders are all primes
- Infinite polycyclic groups: Pcp-groups in GAP
- Arbitrary relative orders allowed.

Some comments

Comments

- Finite solvable groups: Pc-groups in GAP
- It is assumed that the relative orders are all primes
- Infinite polycyclic groups: Pcp-groups in GAP
- Arbitrary relative orders allowed.
- In both cases: consistency is assumed!

Determining Pc-Presentations

Computing Pc-Presentations

- Permutation groups: IsomorphismPcGroup

Determining Pc-Presentations

Computing Pc-Presentations

- Permutation groups: IsomorphismPcGroup
- Matrix groups: IsomorphismPcGroup

Determining Pc-Presentations

Computing Pc-Presentations

- Permutation groups: IsomorphismPcGroup
- Matrix groups: IsomorphismPcGroup
- Fp groups: Quotient algorithms

Determining Pc-Presentations

Computing Pc-Presentations

- Permutation groups: IsomorphismPcGroup
- Matrix groups: IsomorphismPcGroup
- Fp groups: Quotient algorithms
- ANUPQ: finite p-quotients

Determining Pc-Presentations

Computing Pc-Presentations

- Permutation groups: IsomorphismPcGroup
- Matrix groups: IsomorphismPcGroup
- Fp groups: Quotient algorithms
- ANUPQ: finite p-quotients
- NQ: arbitrary nilpotent quotients

GAP Session 2

GAP Session 2

Example 1

Example 1

```
gap> gg := [ (2,3)(7,9)(13,14)(19,20)(25,27),
      (1,21,26,8)(2,19,25,7)(3,20,27,9)(5,6)(10,11)(16,17)
      (23,24)(29,30),
      (1,22,26,28,15,12,21,4,3,23,25,30,14,11,20,6,2, 24,
       27,29,13,10,19,5)(7,16,9,17)(8,18) ];
gap> G := Group(gg);
gap> IsSolvable(G);
true
gap> H := Image(IsomorphismPcGroup(G));
<pc group of size 48372940800 with 27 generators>
gap> Collected(Factors(Size(H)));
[ [ 2, 15 ], [ 3, 10 ], [ 5, 2 ] ]
```

Example 1

Example 1

```
gap> AbelianInvariants(H);
[ 2, 2, 4 ]
gap> DerivedSeries(H);
[ <pc group of size 48372940800 with 27 generators>,
  <pc group of size 3023308800 with 23 generators>,
  <pc group of size 377913600 with 20 generators>,
  <pc group of size 15116544 with 18 generators>,
  <pc group of size 59049 with 10 generators>,
  Group([  ]) ]
gap> Center(H);
Group([  ])
gap> F := FittingSubgroup(H);
<pc group with 10 generators>
gap> Collected(Factors(Size(F)));
[ [ 3, 10 ] ]
```

Example 1

Example 1

```
gap> cc := ConjugacyClasses(H);;
gap> Length(cc);
2079
gap> Set(List(cc, x -> Order(Representative(x))));
[ 1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 40, 48,
  60, 120 ]
gap> MaximalSubgroupClassReps(H);;
gap> List(last, x -> Index(H,x));
[ 2, 2, 2, 2, 2, 2, 2, 25, 256, 59049 ]
gap> List([2,3,5], x -> SylowSubgroup(H,x));;
gap> List(last, IsAbelian);
[ false, true, true ]
```

Example 2

Example 2

```
IComm := function( a, b, x)
    local c, i;
    c := a;
    for i in [1..x] do
        c := Comm(c,b);
    od;
    return c;
end;
```

Example 2

Example 2

```
gap> F := FreeGroup(3);
<free group on the generators [ f1, f2, f3 ]>
gap> f := GeneratorsOfGroup(F);
[ f1, f2, f3 ]
gap> R := [IComm(f[1], f[2], 4),
           IComm(f[1], f[3], 4),
           IComm(f[2],f[3],4)];;
gap> NilpotentQuotient(F/R, 4);
Pcp-group with orders [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]
```

Example 2

Example 2

```
gap> Add(R, f[1]^4);
gap> NilpotentQuotient(F/R, 5);
Pcp-group with orders [4,0,0, 4, 4,0, 4, 4, 4, 4, 4, 4,0,
 0, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,0, 4,0,0, 4,
 2, 4, 4, 4, 4, 4,2,2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,0,0, 4, 4, 4, 4,
 0,0, 4,0]
gap> Size(TorsionSubgroup(G));
42535295865117307932921825928971026432        # = 2^125
gap> HirschLength(G);
13
gap> G/TorsionSubgroup(G);
Pcp-group with orders [0,0,0,0,0,0,0,0,0,0,0,0,0]
```