GRPS1024

Library of the groups of order 1024 of p-class at least 3, those of p-class two and rank 4 and those of p-class 1 and rank 10.

0.0.4

20 June 2022

David Burrell

David Burrell
Email: davidburrell@ufl.edu
Homepage: https://davidburrell.github.io/

Contents

Chapter 1

Groups of Order 1024

1.1 Overview

This library gives explicit access to the following groups of order 1024:

- The Rank 1 group
- All Rank 2 groups
- All Rank 3 groups
- All Rank 4 groups
- Rank 5 groups with p-class at least 3
- Rank 6 groups with p-class at least 3
- Rank 7 groups with p-class at least 3
- Rank 8 groups with p-class at least 3
- Rank 9 groups with p-class at least 3
- The Rank 10 group

This library gives partial information on the remaining groups of order 1024:

- Rank 5 groups with p-class 2
- Rank 6 groups with p-class 2
- Rank 7 groups with p-class 2
- Rank 8 groups with p-class 2
- Rank 9 groups with p-class 2

For the groups that are not explicitly available the following information is available:

- Parent Group ID

- Parent Group Order
- p-class
- Rank
- Age

The groups are sorted first by their parent group ids and then by the pc codes of the standard presentations for the groups. The data contained in this library was used in the 2021 enumeration of the groups of order 1024 [Bur21]. The available groups were generated using the p-group generation algorithm [O'B90] as implemented in the ANUPQ package [GNOH19]. The information on the remaining groups was calculated using the cohomological methods for enumerating p-groups of p-class 2 as introduced in [EO99].

Chapter 2

Functionality

2.1 Methods

Once the package is loaded the user may call `SmallGroup(1024,i)` and receive either a group if available or a *partially constructed group* which has the following attributes set

- p-class
- Rank
- Heritage
- Order

Example

```
gap> G:=SmallGroup(1024,1); #this group is available
<pc group of size 1024 with 10 generators>
gap> RankPGroup(G);
4
gap> PClassPGroup(G);
2
gap> Heritage(G);
[ 16, 14, 1 ]
gap> H:=SmallGroup(1024,3568); #this is a partially constructed group
<pc group with 0 generators>
gap> PClassPGroup(H);
2
gap> RankPGroup(H);
5
gap> Heritage(H);
[ 32, 51, 1 ]
gap> K:=SmallGroup(1024,3569); #this is a partially constructed group
<pc group with 0 generators>
gap> PClassPGroup(K);
2
gap> RankPGroup(K);
5
gap> Heritage(K);
[ 32, 51, 2 ]
#notice that H,K have the same parent group but their age differs
```

2.1.1 AvailableMap

▷ `AvailableMap(N)` (function)

Returns: `int`

For $1 \leq i \leq 683,875,133$ this function will return the SmallGroup ID of the ith available group among all the groups of order 1024.

Example
```
#groups 1-3567 are available SmallGroup(1024,3568) is not available
gap> g:=SmallGroup(1024,3567);
<pc group of size 1024 with 10 generators> #this is an available group
gap> g:=SmallGroup(1024,3568);
<pc group with 0 generators> #this is a partially constructed group
#the next available group has index 378632399
gap> AvailableMap(3568)
378632399
#access the ith available group of order 1024
gap> SmallGroup(1024,AvailableMap(i)) #for i <= 683,875,1333
```

2.1.2 InverseAvailableMap

▷ `InverseAvailableMap(N)` (function)

Returns: `int`

For $1 \leq i \leq 49,487,367,289$ if `SmallGroup(1024,i)` is available this will return its position in the available groups list or else it will print a message telling you that it is not available and return 0.

Example
```
gap> InverseAvailableMap(AvailableMap(i)) = i;
gap> InverseAvailableMap(3568);
This is an immediate descendant of the elementary abelian group of order 32 and is not available
0
```

2.1.3 Heritage (for IsGroup)

▷ `Heritage(G)` (attribute)

Returns: `list`

Returns as a list the following information for a group of order 1024 loaded from the library `[ParentGroupID, ParentGroupOrder, Age]`. The *Age* of a group is the position of the group among its siblings in the ordered list of their standard PC codes.

References

[Bur21] D. Burrell. On The Number of Groups of Order 1024. *Communications in Algebra*, 0(0):1–3, 2021. 4

[EO99] B. Eick and E. A. O'Brien. Enumerating p -Groups. *Journal of the Australian Mathematical Society. Series A. Pure Mathematics and Statistics*, 67(2):191–205, dec 1999. 4

[GNOH19] G. Gamble, W. Nickel, E. O'Brien, and M. Horn. ANU p-Quotient, 2019. 4

[O'B90] E. A. O'Brien. The p-group generation algorithm. *Journal of Symbolic Computation*, 9(5):677–698, oct 1990. 4

Index