smallantimagmas

A fork of smallgrp package for small antiassociative magmas

0.2.8

28 August 2024

Kamil Zabielski

Kamil Zabielski

Address: Department of Theoretical Computer Science
Białystok University of Technology
Wiejska 45A
15-325 Białystok
Poland

Copyright

Acknowledgements

Contents

Chapter 1

smallantimagmas automatic generated documentation

1.1 smallantimagmas automatic generated documentation of properties

1.1.1 IsAntiassociative (for IsMagma)

▷ IsAntiassociative(*M*)   (property)

Returns: `true` or `false`

identifies whether magma *M* is antiassociative.

```
——————————— Example ———————————
gap> IsAntiassociative(OneSmallGroup(16));
false
gap> IsAntiassociative(OneSmallAntimagma(2));
true
gap> IsAntiassociative(OneSmallAntimagma(3));
true
```

1.1.2 IsLeftCyclic (for IsMagma)

▷ IsLeftCyclic(*M*)   (property)

Returns: `true` or `false`

if magma is left cyclic *m*.

1.1.3 IsRightCyclic (for IsMagma)

▷ IsRightCyclic(*M*)   (property)

Returns: `true` or `false`

if magma is left cyclic *m*.

1.1.4 IsLeftCancellative (for IsMagma)

▷ IsLeftCancellative(*M*)   (property)

Returns: `true` or `false`

if magma is left cancellative *m*.

```
                          ──── Example ────
  gap> M := SmallAntimagma(2, 1);
  <magma with 2 generators>
  gap> Display( MultiplicationTable(M) );
  [ [  2,  1 ],
    [  2,  1 ] ]
  gap> IsRightCancellative(M);
  false
  gap> IsLeftCancellative(M);
  true
  gap> List(AllSmallAntimagmas(2), M -> IsLeftCancellative(M));
  [ true, false ]
```

1.1.5 IsRightCancellative (for IsMagma)

▷ IsRightCancellative(*M*)   (property)

Returns: true or false

if magma is right cancellative *m*.

```
                          ──── Example ────
  gap> List(AllSmallAntimagmas(2), M -> IsRightCancellative(M));
  [ false, true ]
```

1.1.6 IsCancellative (for IsMagma)

▷ IsCancellative(*M*)   (property)

Returns: true or false

if magma is cancellative *m*.

```
                          ──── Example ────
  gap> List(AllSmallAntimagmas(2), M -> IsCancellative(M));
  [ false, false ]
```

1.2 smallantimagmas automatic generated documentation of attributes

1.2.1 AssociativityIndex (for IsMagma)

▷ AssociativityIndex(*M*)   (attribute)

identifies associativity index of *M*.

```
                          ──── Example ────
  gap> OneSmallAntimagma(2);
  <magma with 2 generators>
  gap> AssociativityIndex(OneSmallAntimagma(2));
  0
  gap> OneSmallGroup(4);
  <pc group of size 4 with 2 generators>
  gap> AssociativityIndex(OneSmallGroup(4));
  64
  gap> AssociativityIndex(OneSmallGroup(4)) = 4 ^ 3;
  true
```

1.2.2 DiagonalOfMultiplicationTable (for IsMagma)

▷ DiagonalOfMultiplicationTable(*M*)   (attribute)

  computes diaognal of multiplication table of *M*.

```
——————————————————————— Example ———————————————————————
gap> List(AllSmallAntimagmas(3), M -> DiagonalOfMultiplicationTable((M)));
[ [ 2, 1, 1 ], [ 2, 1, 1 ],
  [ 2, 3, 2 ], [ 2, 1, 1 ],
  [ 2, 1, 1 ], [ 2, 1, 2 ],
  [ 2, 3, 2 ], [ 2, 1, 2 ],
  [ 2, 3, 1 ], [ 2, 3, 1 ]
]
```

1.2.3 CommutativityIndex (for IsMagma)

▷ CommutativityIndex(*M*)   (attribute)

  identifies commutativity index of *M*.

```
——————————————————————— Example ———————————————————————
```

1.2.4 AnticommutativityIndex (for IsMagma)

▷ AnticommutativityIndex(*M*)   (attribute)

  calculates anticommutativity index of *M*.

```
——————————————————————— Example ———————————————————————
```

1.2.5 SquaresIndex (for IsMagma)

▷ SquaresIndex(*M*)   (attribute)

  computes squares index of M so the order of $\{m^2 | m \in\}$.

```
——————————————————————— Example ———————————————————————
gap> List(AllSmallAntimagmas(2), M -> List(M, m -> m * m) );
[ [ m2, m1 ], [ m2, m1 ] ]
gap> List(AllSmallAntimagmas(2), M -> SquaresIndex(M ));
[ 2, 2 ]
gap> List(AllSmallAntimagmas(3), M -> SquaresIndex(M ));
[ 2, 2, 2, 2, 2, 2, 2, 2, 3, 3 ]
```

1.2.6 IdSmallAntimagma (for IsMagma)

▷ IdSmallAntimagma(*M*)   (attribute)

  identifies class of antiassociative magma *M*.

```
─────────────── Example ───────────────
gap> IsAntiassociative(OneSmallGroup(16));
false
gap> IsAntiassociative(OneSmallAntimagma(2));
true
gap> IsAntiassociative(OneSmallAntimagma(3));
true
```

1.2.7 LeftOrder (for IsExtLElement)

▷ LeftOrder(*[m]*)   (attribute)

returns a left order of element *m*.

1.2.8 RightOrder (for IsExtRElement)

▷ RightOrder(*[m]*)   (attribute)

returns a right order of element *m*.

1.2.9 LeftOrdersOfElements (for IsMagma)

▷ LeftOrdersOfElements(*[m]*)   (attribute)

returns a left order of element *m*.

1.2.10 RightOrdersOfElements (for IsMagma)

▷ RightOrdersOfElements(*[m]*)   (attribute)

returns a left order of element *m*.

1.3 smallantimagmas automatic generated documentation of global functions

1.3.1 AllSubmagmas

▷ AllSubmagmas(*M*)   (function)

builds a collection of non-isomorphic submagmas of *M*.

```
─────────────── Example ───────────────
gap> AllSmallAntimagmas(2);
[ <magma with 2 generators>, <magma with 2 generators> ]
gap> List(AllSmallAntimagmas(2), M -> AllSubmagmas(M));
[ [ <magma with 1 generator> ], [ <magma with 1 generator> ] ]
```

1.3.2   MagmaIsomorphismInvariantsMatch

▷ MagmaIsomorphismInvariantsMatch(*M*)   (function)

computes isomorphism invariants of *M*.

1.3.3   IsMagmaIsomorphic

▷ IsMagmaIsomorphic(*M, N*)   (function)

identifies whether magmas *M*, *N* are isomorphic.

```
───────────────────────────────── Example ─────────────────────────────────
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> N := SmallAntimagma(2, 2);
<magma with 2 generators>
gap> T := MagmaByMultiplicationTable([ [2, 1], [2, 1] ]);
<magma with 2 generators>
gap> IsMagmaIsomorphic(M, M);
true
gap> IsMagmaIsomorphic(M, T);
true
gap> IsMagmaIsomorphic(M, N);
false
```

1.3.4   IsMagmaAntiisomorphic

▷ IsMagmaAntiisomorphic(*[M, N]*)   (function)

identifies whether magmas *M*, *N* are antiisomorphic.

```
───────────────────────────────── Example ─────────────────────────────────
gap> N := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> N := SmallAntimagma(2, 2);
<magma with 2 generators>
gap> IsMagmaAntiisomorphic(M, M);
false
gap> IsMagmaAntiisomorphic(M, N);
true
gap> IsMagmaAntiisomorphic(M, TransposedMagma(M));
true
```

1.3.5   TransposedMagma

▷ TransposedMagma(*[M]*)   (function)

generates transposed magma *M*.

```
――――――――――――― Example ―――――――――――――
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> IsMagmaAntiisomorphic(M, TransposedMagma(M));
true
gap> IsMagmaIsomorphic(M, TransposedMagma(TransposedMagma(M)));
true
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> Display(MultiplicationTable(M));
[ [  2,   1 ],
  [  2,   1 ] ]
gap> Display(MultiplicationTable(TransposedMagma(M)));
[ [  2,   2 ],
  [  1,   1 ] ]
```

1.3.6 LeftPower

▷ LeftPower(*[m, k]*)   (function)

   returns a left *k*-power of element *m*.

1.3.7 RightPower

▷ RightPower(*[m, k]*)   (function)

   returns a right *k*-power of element *m*.

1.3.8 AllSmallAntimagmas

▷ AllSmallAntimagmas(*n*)   (function)

   returns all antiassociative magmas of specified size *n* (a number)

```
――――――――――――― Example ―――――――――――――
gap> AllSmallAntimagmas(2);
[ <magma with 2 generators>, <magma with 2 generators> ]
gap> AllSmallAntimagmas(3);
[
  <magma with 3 generators>, <magma with 3 generators>, <magma with 3 generators>,
  <magma with 3 generators>, <magma with 3 generators>, <magma with 3 generators>,
  <magma with 3 generators>, <magma with 3 generators>,
  <magma with 3 generators>, <magma with 3 generators>
]
```

1.3.9 NrSmallAntimagmas

▷ NrSmallAntimagmas(*n*)   (function)

   counts number of antiassociative magmas of specified size *n* (a number).

```
                              ── Example ──────────────
  gap> NrSmallAntimagmas(2);
  2
  gap> NrSmallAntimagmas(3);
  10
  gap> NrSmallAntimagmas(4);
  17780
```

1.3.10   SmallAntimagma

▷ SmallAntimagma(*n, i*)   (function)

returns antiassociative magma of id *[n, i]*.

```
                              ── Example ──────────────
  gap> SmallAntimagma(2, 1);
  <magma with 2 generators>
  gap> SmallAntimagma(4, 5);
  <magma with 4 generators>
```

1.3.11   OneSmallAntimagma

▷ OneSmallAntimagma(*n*)   (function)

returns a random antiassociative magma of size *n*.

```
                              ── Example ──────────────
  gap> OneSmallAntimagma(2);
  <magma with 2 generators>

  gap> OneSmallAntimagma(3);
  <magma with 3 generators>
```

1.3.12   ReallyAllSmallAntimagmas

▷ ReallyAllSmallAntimagmas(*n*)   (function)

returns really-all antiassociative magmas, isomorphic, of specified size *n* (a number)

```
                              ── Example ──────────────
  gap> ReallyAllSmallAntimagmas(2);
  [ <magma with 2 generators>, <magma with 2 generators> ]
```

1.3.13   ReallyNrSmallAntimagmas

▷ ReallyNrSmallAntimagmas(*n*)   (function)

counts number of antiassociative magmas of specified size *n* (a number)

```
                              ── Example ──────────────
  gap> ReallyNrSmallAntimagmas(3);
  52
```

1.3.14   AntimagmaGeneratorPossibleDiagonals

▷ AntimagmaGeneratorPossibleDiagonals(*n*) (function)

returns all possible diagonals of multiplication table for `[n]`-antimagma.

```
─────────── Example ───────────
gap> AntimagmaGeneratorPossibleDiagonals(2);
[ [ 2, 1 ] ]
gap> AntimagmaGeneratorPossibleDiagonals(3);
[
  [ 2, 1, 1 ], [ 2, 1, 2 ], [ 2, 3, 1 ], [ 2, 3, 2 ],
  [ 3, 1, 1 ], [ 3, 1, 2 ], [ 3, 3, 1 ], [ 3, 3, 2 ]
]
```

1.3.15   AntimagmaGeneratorFilterNonIsomorphicMagmas

▷ AntimagmaGeneratorFilterNonIsomorphicMagmas(*Ms*) (function)

filters non-isomorphic magmas `m`.

1.4   smallantimagmas automatic generated documentation of methods

1.4.1   MagmaIsomorphism (for IsMagma, IsMagma)

▷ MagmaIsomorphism(*M, N*) (operation)

computes an isomoprhism between magmas *M*, *N*.

```
─────────── Example ───────────
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> N := MagmaByMultiplicationTable([ [2, 1], [2, 1] ]);
<magma with 2 generators>
gap> MagmaIsomorphism(M, N);
<mapping: Domain([ m1, m2 ]) -> Domain([ m1, m2 ]) >
```

1.4.2   MagmaAntiisomorphism (for IsMagma, IsMagma)

▷ MagmaAntiisomorphism(*M, N*) (operation)

creates an antiisomoprhism between magmas *M*, *N*.

```
─────────── Example ───────────
gap> M := SmallAntimagma(2, 1);
<magma with 2 generators>
gap> N := SmallAntimagma(2, 2);
<magma with 2 generators>
gap> MagmaAntiisomorphism(M, N);
<mapping: Domain([ m1, m2 ]) -> Domain([ m1, m2 ]) >
```

Index