YangBaxter

Combinatorial Solutions for the Yang-Baxter equation

0.10.3

25 February 2023

Leandro Vendramin

Olexandr Konovalov

Leandro Vendramin
Email: Leandro.Vendramin@vub.be
Homepage: https://vendramin.github.io/
Address: Vrije Universiteit Brussel
Faculty of Sciences
Department of Mathematics and Data Science
Pleinlaan 2, B-1050
Brussel, Belgium

Olexandr Konovalov
Email: obk1@st-andrews.ac.uk
Homepage: https://olexandr-konovalov.github.io/
Address: School of Computer Science
University of St Andrews
Jack Cole Building, North Haugh,
St Andrews, Fife, KY16 9SX, Scotland
Chapter 1

Preliminaries

In this section we define skew braces and list some of their main properties [GV17].

1.1 Definition and examples

A skew brace is a triple \((A, +, \circ)\), where \((A, +)\) and \((A, \circ)\) are two (not necessarily abelian) groups such that the compatibility \(a \circ (b + c) = a \circ b - a + a \circ c\) holds for all \(a, b, c \in A\). Ones proves that the map \(\lambda: (A, \circ) \rightarrow \text{Aut}(A, +), a \mapsto \lambda_a(b), \lambda_a(b) = -a + a \circ b\), is a group homomorphism. Notation: For \(a, b \in A\), we write \(a \ast b = \lambda_a(b) - b\).

1.1.1 IsSkewbrace (for IsAttributeStoringRep)

\[ \text{IsSkewbrace}(\text{arg}) \]

Returns: true or false

1.1.2 Skewbrace (for IsList)

\[ \text{Skewbrace}(\text{list}) \]

Returns: a skew brace

The argument \text{list} is a list of pairs of elements in a group. By Proposition 5.11 of [GV17], skew braces over an abelian group \(A\) are equivalent to pairs \((G, \pi)\), where \(G\) is a group and \(\pi: G \rightarrow A\) is a bijective 1-cocycle, a finite skew brace can be constructed from the set \(\{(a_j, g_j) : 1 \leq j \leq n\}\), where \(G = \{g_1, \ldots, g_n\}\) and \(A = \{a_1, \ldots, a_n\}\) are permutation groups. This function is used to construct skew braces.

\[
\text{gap> Skewbrace}([[(),()]]);
<brace of size 1>
\]

\[
\text{gap> Skewbrace}([[(),()],[(1,2),(1,2)]]);
<brace of size 2>
\]

1.1.3 SmallSkewbrace (for IsInt, IsInt)

\[ \text{SmallSkewbrace}(n, k) \]

Returns: a skew brace

The function returns the \(k\)-th skew brace from the database of skew braces of order \(n\).
1.1.4 TrivialBrace (for IsGroup)

\texttt{TrivialBrace(abelian\_group)} 

\textbf{Returns:} a brace

This function returns the trivial brace over the abelian group \texttt{abelian\_group}. Here \texttt{abelian\_group} should be an abelian group!

\texttt{gap> TrivialBrace(CyclicGroup(IsPermGroup, 5));}
\texttt{<brace of size 5>}

1.1.5 TrivialSkewbrace (for IsGroup)

\texttt{TrivialSkewbrace(group)}

\textbf{Returns:} a skew brace

This function returns the trivial skew brace over \texttt{group}.

\texttt{gap> TrivialSkewbrace(DihedralGroup(10));}
\texttt{<skew brace of size 10>}

1.1.6 SmallBrace (for IsInt, IsInt)

\texttt{SmallBrace(n, k)}

\textbf{Returns:} a brace of abelian type

The function returns the \texttt{k}-th brace (of abelian type) from the database of braces of order \texttt{n}.

\texttt{gap> SmallBrace(8,3);}
\texttt{<brace of size 8>}

1.1.7 IdSkewbrace (for IsSkewbrace)

\texttt{IdSkewbrace(obj)}

\textbf{Returns:} a list

The function returns \texttt{[n, k]} if the skew brace \texttt{obj} is isomorphic to \texttt{SmallSkewbrace(n,k)}.

\texttt{gap> IdSkewbrace(SmallSkewbrace(8,5));}
\texttt{[ 8, 5 ]}

1.1.8 AutomorphismGroup (for IsSkewbrace)

\texttt{AutomorphismGroup(obj)}

\textbf{Returns:} a list

The function computes the automorphism group of a skew brace.
YangBaxter

\[ \text{Example} \]
\begin{verbatim}
gap> br := SmallSkewbrace(8,20);;
gap> AutomorphismGroup(br);
<group with 8 generators>
gap> StructureDescription(last);
"D8"
\end{verbatim}

\[ \text{Example} \]
\begin{verbatim}
gap> br := SmallSkewbrace(8,25);;
gap> aut := AutomorphismGroup(br);;
gap> f := Random(aut);;
gap> x := Random(br);;
gap> ImageElm(f, x) in br;
true
\end{verbatim}

1.1.9 \textbf{IdBrace (for IsSkewbrace)}

\[ \text{IDbrace(obj)} \]

\textbf{Returns:} a list

The function returns \([ n, k ]\) if the brace of abelian type \(obj\) is isomorphic to \(SmallBrace(n,k)\).

\begin{verbatim}
gap> IdBrace(SmallBrace(8,5));
[ 8, 5 ]
\end{verbatim}

1.1.10 \textbf{IsomorphismSkewbraces}

\[ \text{IsomorphismSkewbraces(obj1, obj2)} \]

\textbf{Returns:} an isomorphism of skew braces if \(obj1\) and \(obj2\) are isomorphic and \text{fail} otherwise.

If \(A\) and \(B\) are skew braces, a skew brace homomorphism is a map \(f:A \rightarrow B\) such that

\[ f(a + b) = f(a) + f(b) \quad f(a \circ b) = f(a) \circ f(b) \]

hold for all \(a, b \in A\). A skew brace isomorphism is a bijective skew brace homomorphism. \(IsomorphismSkewbraces\) first computes all injective homomorphisms from \((A, +)\) to \((B, +)\) and then tries to find one \(f\) such that \(f(a \circ b) = f(a) \circ f(b)\) for all \(a, b \in A\).

1.1.11 \textbf{DirectProductSkewbraces (for IsSkewbrace, IsSkewbrace)}

\[ \text{DirectProductSkewbraces(obj1, obj2)} \]

\textbf{Returns:} the direct product of \(obj1\) and \(obj2\)

\begin{verbatim}
gap> br1 := SmallBrace(8,18);;
gap> br2 := SmallBrace(12,2);;
gap> br := DirectProductSkewbraces(br1,br2);;
gap> IsLeftNilpotent(br);
false
gap> IsRightNilpotent(br);
false
gap> IsSolvable(br);
true
\end{verbatim}
1.1.12 DirectProductOp (for IsList, IsSkewbrace)

DirectProductOp(arg1, arg2) (operation)

1.1.13 IsTwoSided (for IsSkewbrace)

IsTwoSided(obj) (property)

Returns: true if the skew brace is two sided, false otherwise

A skew brace $A$ is said to be two-sided if $(a + b) \circ c = a \circ c - c + b \circ c$ holds for all $a, b, c \in A$.

Example

```
gap> IsTwoSided(SmallSkewbrace(8,2));
false

gap> IsTwoSided(SmallSkewbrace(8,4));
true
```

1.1.14 IsAutomorphismGroupOfSkewbrace (for IsAutomorphismGroup)

IsAutomorphismGroupOfSkewbrace(obj) (property)

Returns: true if the group is the automorphism group of a skew braces, false otherwise

Example

```
gap> br := SmallSkewbrace(8,25);;
gap> aut := AutomorphismGroup(br);;
gap> Order(aut);
4

gap> IsAutomorphismGroupOfSkewbrace(aut);
true
```

1.1.15 IsClassical (for IsSkewbrace)

IsClassical(obj) (property)

Returns: true if the skew brace is of abelian type, false otherwise

Let $\mathcal{X}$ be a property of groups. A skew brace $A$ is said to be of $\mathcal{X}$-type if its additive group belongs to $\mathcal{X}$. In particular, skew braces of abelian type are those skew braces with abelian additive group. Such skew braces were introduced by Rump in [Rum07].

1.1.16 IsOfAbelianType (for IsSkewbrace)

IsOfAbelianType(arg) (property)

Returns: true or false

1.1.17 IsBiSkewbrace (for IsSkewbrace)

IsBiSkewbrace(obj) (property)

Returns: true if the skew brace is a bi-skew brace, false otherwise

A skew brace $(\cdot, +, \circ)$ is said to be a bi-skew brace if $(\cdot, \circ, +)$ is a skew brace
1.1.18  IsOfNilpotentType (for IsSkewbrace)

\[ \text{IsOfNilpotentType(obj)} \]

Returns: true if the skew brace is of nilpotent type, false otherwise

Let $\mathcal{X}$ be a property of groups. A skew brace $A$ is said to be of $\mathcal{X}$-type if its additive group belongs to $\mathcal{X}$. In particular, skew braces of nilpotent type are those skew braces with nilpotent additive group.

1.1.19  IsTrivialSkewbracer (for IsSkewbrace)

\[ \text{IsTrivialSkewbracer(obj)} \]

Returns: true if the skew brace is trivial, false otherwise

The function returns true if the skew brace $A$ is trivial, i.e., $a \circ b = a + b$ for all $a, b \in A$. WARNING: The property IsTrivial applied to a skew brace will return true if and only if the skew brace has only one element.

Example

```gap
gap> br := SmallSkewbracer(9,1);;
gap> IsTrivialSkewbracer(br);
true
gap> IsTrivial(br);
false
```

1.1.20  Skewbrace2YB (for IsSkewbrace)

\[ \text{Skewbrace2YB(obj)} \]

Returns: the set-theoretic solution associated with the skew brace $obj$.

If $A$ is a skew brace, the map $r_A:A \times A \to A \times A$

\[ r_A(a, b) = (\lambda_a(b), \lambda_b(a \circ b)) \]

is a non-degenerate set-theoretic solution of the Yang–Baxter equation. Furthermore, $r_A$ is involutive if and only if $A$ is of abelian type (i.e., the additive group of $A$ is abelian).

Example

```gap
gap> Skewbrace2YB(TrivialBracer(CyclicGroup(6)));
<A set-theoretical solution of size 6>
```

1.1.21  Brace2YB (for IsSkewbrace)

\[ \text{Brace2YB(arg)} \]

Returns: the set-theoretic solution associated with a given subset of a skew brace.

Example

```gap
gap> br := TrivialSkewbracer(SymmetricGroup(3));;
gap> AsList(br);
[ <()>, <(2,3)>, <(1,2)>, <(1,2,3)>, <(1,3,2)>, <(1,3)> ]
gap> SkewbraceSubset2YB(br, last{[4,5]});
<A set-theoretical solution of size 2>
```

1.1.22  SkewbraceSubset2YB (for IsSkewbrace, IsCollection)

\[ \text{SkewbraceSubset2YB(obj)} \]

Returns: the set-theoretic solution associated with a given subset of a skew brace.

Example

```gap
gap> br := TrivialSkewbracer(SymmetricGroup(3));;
gap> AsList(br);
[ <()>, <(2,3)>, <(1,2)>, <(1,2,3)>, <(1,3,2)>, <(1,3)> ]
gap> SkewbraceSubset2YB(br, last{[4,5]});
<A set-theoretical solution of size 2>
```
1.1.23 SemidirectProduct (for IsSkewbrace, IsSkewbrace, IsGeneralMapping)

- **SemidirectProduct** \((A, B, s)\)  
  **Returns:** the semidirect product of skew braces  
  Let \(A\) and \(B\) be two skew braces and \(\sigma\) be a skew brace action of \(B\) on \(A\), this is a group homomorphism \(\sigma: (B, \circ) \to \text{Aut}_{Br}(A)\) from the multiplicative group of \(B\) to the skew brace automorphism of \(A\). The semidirect product of \(A\) and \(B\) with respect to \(\sigma\) is the skew brace \(A \rtimes_{\sigma} B\) with operations

\[
(a_1, b_1) + (a_2, b_2) = (a_1 + a_2, b_1 + b_2), \quad (a_1, b_1) \circ (b_2, b_2) = (a_1 \circ \sigma(b_1)(a_2), b_1 \circ b_2)
\]

Example

```gap
gap> A := SmallSkewbrace(4,2);;
gap> B := SmallSkewbrace(3,1);;
gap> s := SkewbraceActions(B,A);;
gap> Size(s); 1
gap> IdSkewbrace(SemidirectProduct(A,B,s[1]));
[ 12, 11 ]
gap> IdSkewbrace(DirectProduct(A,B));
[ 12, 11 ]
```

1.1.24 UnderlyingAdditiveGroup (for IsSkewbrace)

**UnderlyingAdditiveGroup** \((A)\)  
**Returns:** the underlying multiplicative group of the skew brace

Example

```gap
gap> br := SmallBrace(4,2);;
gap> G:=UnderlyingMultiplicativeGroup(br);;
gap> StructureDescription(G);
"C2 x C2"
```

1.1.25 UnderlyingMultiplicativeGroup (for IsSkewbrace)

**UnderlyingMultiplicativeGroup** \((A)\)  
**Returns:** the underlying additive group of the skew brace

Example

```gap
gap> br := SmallSkewbrace(6,2);;
gap> G:=UnderlyingAdditiveGroup(br);;
gap> IsAbelian(G);
false
```
## 2.1 Braces and Radical Rings

### 2.1.1 AdditiveGroupOfRing (for IsRing)

- **AdditiveGroupOfRing**<br>
  - **Returns:** a group<br>
  - This function returns a permutation representation of the additive group of the given ring.<br>

  ```gap<br>gap> rg := SmallRing(8,10);;<br>gap> StructureDescription(AdditiveGroupOfRing(rg));<br>"C4 x C2"
  ```

### 2.1.2 IsJacobsonRadical (for IsRing)

- **IsJacobsonRadical**<br>
  - **Returns:** true if the ring is radical and false otherwise.<br>

  ```gap<br>gap> rg := SmallRing(8,11);;<br>gap> IsJacobsonRadical(rg);<br>true<br>gap> rg := SmallRing(8,20);;<br>gap> IsJacobsonRadical(rg);<br>false
  ```

## 2.2 Braces and Yang-Baxter Equation

### 2.2.1 Table2YB (for IsList)

- **Table2YB**<br>
  - **Returns:** the solution given by the table<br>

  ```gap<br>gap> l := Table(SmallIYB(4,13));;<br>gap> t := Table2YB(l);;
  ```
2.2.2 Evaluate (for IsYB, IsList)

Evaluate(obj, pair)  
Returns: a pair of two integers

Example

```gap
gap> cs := SmallCycleSet(4,13);;
gap> yb := CycleSet2YB(cs);;
gap> Permutations(yb);
[ [ (3,4), (1,3,2,4), (1,4,2,3), (1,2) ],
  [ (2,4), (1,4,3,2), (1,2,3,4), (1,3) ] ]
gap> Evaluate(yb, [1,2]);
[ 2, 4 ]
gap> Evaluate(yb, [1,3]);
[ 4, 2 ]
```

2.2.3 LyubashenkoYB (for IsInt, IsPerm, IsPerm)

LyubashenkoYB(size, f, g)  
Returns: a permutation solution to the YBE

Example

```gap
gap> yb := LyubashenkoYB(4, (1,2),(3,4));
<A set-theoretical solution of size 4>
gap> Permutations(last);
[ [ (1,2), (1,2), (1,2), (1,2) ],
  [ (3,4), (3,4), (3,4), (3,4) ] ]
```

2.2.4 IsIndecomposable (for IsYB)

IsIndecomposable(X)  
Returns: true if the involutive solutions is indecomposable

2.2.5 Table (for IsYB)

Table(obj)  
Returns: a table with the image of the solution

Example

```gap
gap> yb := SmallIYB(3,2);;
gap> Table(yb);
[ [ [ 1, 1 ], [ 2, 1 ], [ 3, 2 ] ], [ [ 1, 2 ], [ 2, 2 ], [ 3, 1 ] ], [ [ 2, 3 ], [ 1, 3 ], [ 3, ] ]
```
2.2.6 DehornoyClass (for IsYB)

- **DehornoyClass(obj)** (attribute)
  - **Returns**: The class of an involutive solution

  ```gap
  gap> cs := SmallCycleSet(4,13);;
gap> yb := CycleSet2YB(cs);
gap> DehornoyClass(yb);
  2
  gap> cs := SmallCycleSet(4,19);;
gap> yb := CycleSet2YB(cs);
gap> DehornoyClass(yb);
  4
  ```

2.2.7 DehornoyRepresentationOfStructureGroup (for IsYB, IsObject)

- **DehornoyRepresentationOfStructureGroup(obj, variable)** (operation)
  - **Returns**: A faithful linear representation of the structure group of obj

  ```gap
  gap> cs := SmallCycleSet(4,13);;
gap> yb := CycleSet2YB(cs);
gap> Permutations(yb);
[ [ (3,4), (1,3,2,4), (1,4,2,3), (1,2) ],
  [ (2,4), (1,4,3,2), (1,2,3,4), (1,3) ] ]
gap> field := FunctionField(Rationals, 1);;
gap> q := IndeterminatesOfFunctionField(field)[1];;
gap> G := DehornoyRepresentationOfStructureGroup(yb, q);;
gap> x1 := G.1;;
gap> x2 := G.2;;
gap> x3 := G.3;;
gap> x4 := G.4;;
gap> x1*x2=x2*x4;
true
gap> x1*x3=x4*x2;
true
gap> x1*x4=x3*x3;
true
gap> x2*x1=x3*x4;
true
gap> x2*x2=x4*x1;
true
gap> x3*x1=x4*x3;
true
  ```

2.2.8 IdYB (for IsYB)

- **IdYB(obj)** (attribute)
  - **Returns**: the identification number of obj

  ```gap
  gap> cs := SmallCycleSet(5,10);;
gap> IdCycleSet(cs);
  ```
2.2.9 LinearRepresentationOfStructureGroup (for IsYB)

▷ LinearRepresentationOfStructureGroup(obj) (attribute)

**Returns:** the permutation brace of the involutive solution of obj a linear representation of the structure group of a finite involutive solution

```
gap> yb := SmallIYB(5,86);;
gap> IdBrace(IYBBrace(yb));
[ 6, 2 ]
```

```
gap> yb := SmallIYB(5,86);;
gap> gr := LinearRepresentationOfStructureGroup(yb);;
gap> gens := GeneratorsOfGroup(gr);
gap> Display(gens[1]);
[ [ 0, 1, 0, 0, 0, 1 ],
  [ 1, 0, 0, 0, 0, 0 ],
  [ 0, 0, 0, 0, 1, 0 ],
  [ 0, 0, 1, 0, 0, 0 ],
  [ 0, 0, 0, 1, 0, 0 ],
  [ 0, 0, 0, 0, 0, 1 ] ]
```
3.1 YangBaxter automatic generated documentation of properties

3.1.1 IsIndecomposable (for IsCycleSet)

def IsIndecomposable(arg):
    # (property)
    Returns: true if the cycle set is indecomposable

    Let X be a cycle set. We say that X is indecomposable if the group \( \mathcal{G}(X) = \langle \varphi_x : x \in X \rangle \) acts transitively on X.
Chapter 4

Ideals and left ideals

In this section we describe several functions related to ideals and left ideals of skew braces. References: [GV17] and [SV18].

4.1 Left ideals

An left ideal \( I \) of a skew brace \( A \) is a subgroup \( I \) of the additive group of \( A \) such that \( \lambda_a(I) \subseteq I \) for all \( a \in A \).

4.1.1 LeftIdeals (for IsSkewbrace)

\[ \text{LeftIdeals(obj)} \]

Returns: a list with the left ideals of the skew brace \( obj \)

4.1.2 StrongLeftIdeals (for IsSkewbrace)

\[ \text{StrongLeftIdeals(obj)} \]

Returns: a list with the left ideals of the skew brace \( obj \) that are normal in the additive group of \( A \)

4.1.3 IsLeftIdeal (for IsSkewbrace, IsCollection)

\[ \text{IsLeftIdeal(obj)} \]

Returns: true if the subset is a left ideal of \( obj \)

Example

\[
\text{gap> br := SmallBrace(8,4);}
<brace of size 8>
\text{gap> leftideals := LeftIdeals(br);}
[ <brace of size 1>, <brace of size 2>, <brace of size 4>, <brace of size 8> ]
\text{gap> List(leftideals, x->IsLeftIdeal(br, x));}
[ true, true, true, true ]
\text{gap> List(leftideals, IdBrace);}
[ [ 1, 1 ], [ 2, 1 ], [ 4, 1 ], [ 8, 4 ] ]
\]
4.2 Ideals

An ideal $I$ of a skew brace $A$ is a normal subgroup $I$ of the additive group of $A$ such that $\lambda_a(I) \subseteq I$ and $a \circ I = I \circ a$ for all $a \in A$.

4.2.1 IsIdeal (for IsSkewbrace, IsCollection)

▷ IsIdeal(obj, subset)  
    Returns: true if the subset is a left ideal of obj

Example

```gap
gap> br := SmallBrace(8,4);
<brace of size 8>
gap> leftideals := LeftIdeals(br);
[ <brace of size 1>, <brace of size 2>, <brace of size 4>, <brace of size 8> ]
gap> List(leftideals, x->IsLeftIdeal(br, x));
[ true, true, true, true ]
gap> List(leftideals, IdBrace);
[ [ 1, 1 ], [ 2, 1 ], [ 4, 1 ], [ 8, 4 ] ]
```

4.2.2 Ideals (for IsSkewbrace)

▷ Ideals(obj)  
    Returns: a list with the ideals of the skew brace obj

4.2.3 AsIdeal (for IsSkewbrace, IsCollection)

▷ AsIdeal(arg1, arg2)

4.2.4 IdealGeneratedBy (for IsSkewbrace, IsCollection)

▷ IdealGeneratedBy(obj, subset)  
    Returns: the ideal of obj generated by the given subset

The ideal of a skew brace $A$ generated by a subset $X$ is the intersection of all the ideals of $A$ containing $X$.

Example

```gap
br := SmallSkewbrace(6,6);
gap> AsList(br);
[ <()>, <(1,2,3)(4,5,6)>, <(1,3,2)(4,6,5)>, <(1,4)(2,5)(3,6)>,
  <(1,5,3,4,2,6)>, <(1,6,2,4,3,5)> ]
gap> IdealGeneratedBy(br, [last[2]]);
<brace of size 3>
```

4.2.5 IntersectionOfTwoIdeals (for IsSkewbrace and IsIdealInParent, IsSkewbrace and IsIdealInParent)

▷ IntersectionOfTwoIdeals(ideal1, ideal2)  
    Returns: the intersection of ideal1 and ideal2
Example

\begin{verbatim}
gap> br := SmallSkewbrace(6,6);;
gap> Ideals(br);;
gap> IntersectionOfTwoIdeals(last[2],last[3]);
<brace of size 1>
\end{verbatim}

4.2.6 SumOfTwoIdeals (for IsSkewbrace and IsIdealInParent, IsSkewbrace and IsIdealInParent)

▷ SumOfTwoIdeals(ideal1, ideal2) (operation)

Returns: the sum of ideal1 and ideal2

Example

\begin{verbatim}
gap> br := SmallSkewbrace(6,6);;
gap> Ideals(br);;
gap> SumOfTwoIdeals(last[2],last[3]);
<brace of size 6>
\end{verbatim}

4.3 Sequences (left) ideals

4.3.1 LeftSeries (for IsSkewbrace)

▷ LeftSeries(obj) (attribute)

Returns: the left ideals of the left series of obj

The left series of a skew brace $A$ is defined recursively as $A^1 = A$ and $A^{n+1} = A * A^n$ for $n \geq 1$, where $a * b = \lambda_a(b) - b$. Each $A^n$ is a left ideal.

Example

\begin{verbatim}
gap> br := SmallSkewbrace(8,20);
gap> LeftSeries(br);
[ <skew brace of size 8>, <brace of size 2>, <brace of size 1> ]
\end{verbatim}

4.3.2 RightSeries (for IsSkewbrace)

▷ RightSeries(obj) (attribute)

Returns: the ideals of the right series of obj

The right series of a skew brace $0A$ is defined recursively as $A^{(1)} = A$ and $A^{(n+1)} = A * A^{(n)}$ for $n \geq 1$, where $a * b = \lambda_a(b) - b$

Example

\begin{verbatim}
gap> br := SmallSkewbrace(8,20);
gap> RightSeries(br);
[ <skew brace of size 8>, <brace of size 2>, <brace of size 1> ]
\end{verbatim}

4.3.3 IsLeftNilpotent (for IsSkewbrace)

▷ IsLeftNilpotent(obj) (property)

Returns: true if the skew brace obj is left nilpotent.

A skew brace $A$ is said to be left nilpotent if there exists $n \geq 1$ such that $A^n = 0$. 
Example

```gap
gap> IsLeftNilpotent(SmallBrace(8,18));
true
gap> IsLeftNilpotent(SmallBrace(12,2));
false
```

4.3.4 **IsSimpleSkewbrace (for IsSkewbrace)**

▷ **IsSimpleSkewbrace(obj)** (property)

**Returns**: true if the skew brace `obj` is simple.
A skew brace $A$ is said to be simple if $\{0\}$ and $A$ are its only ideals.

Example

```gap
gap> IsSimple(SmallSkewbrace(12,22));
true
gap> IsSimple(SmallSkewbrace(12,21));
false
```

4.3.5 **IsRightNilpotent (for IsSkewbrace)**

▷ **IsRightNilpotent(obj)** (property)

**Returns**: true if the skew brace `obj` is right nilpotent.
A skew brace $A$ is said to be right nilpotent if there exists $n \geq 1$ such that $A^n = 0$.

Example

```gap
gap> IsRightNilpotent(SmallBrace(8,18));
false
gap> IsRightNilpotent(SmallBrace(12,2));
true
```

4.3.6 **LeftNilpotentIdeals (for IsSkewbrace)**

▷ **LeftNilpotentIdeals(obj)** (attribute)

**Returns**: the list of right or left nilpotent ideals of `obj`.
An ideal $I$ of a skew brace $A$ is said to be left if it is left nilpotent as a skew brace.

4.3.7 **RightNilpotentIdeals (for IsSkewbrace)**

▷ **RightNilpotentIdeals(obj)** (attribute)

**Returns**: the list of right or left nilpotent ideals of `obj`.
An ideal $I$ of a skew brace $A$ is said to be right nilpotent if $A$ is said to be left if it is right nilpotent as a skew brace.

```gap
br := SmallBrace(8,18);
gap> IsLeftNilpotent(br);
true
gap> IsRightNilpotent(br);
false
gap> Length(LeftNilpotentIdeals(br));
3
gap> Length(RightNilpotentIdeals(br));
2
```
4.3.8 SmoktunowiczSeries (for IsSkewbrace, IsInt)

\textbf{SmoktunowiczSeries}(obj, bound) \hspace{1cm} \text{(operation)}

\textbf{Returns:} a list of bound left ideals of the Smoktunowicz’s series of obj

The Smoktunowicz’s series of a skew brace $A$ is defined recursively as $A^{[1]} = A$ and $A^{[n+1]}$ is the additive subgroup of $A$ generated by $A^{[i]} * A^{[n+1-i]}$ for $1 \leq i + j \leq n + 1$, where $a * b = \lambda_a(b) - b$.

\begin{verbatim}
gap> br := SmallBrace(16,145);;
gap> SmoktunowiczSeries(br,4);
[ <brace of size 16>, <brace of size 8>, <brace of size 4>, <brace of size 2>,
  <brace of size 2> ]
gap> SmoktunowiczSeries(br,5);
[ <brace of size 16>, <brace of size 8>, <brace of size 4>, <brace of size 2>,
  <brace of size 2>, <brace of size 1> ]
\end{verbatim}

4.3.9 Socle (for IsSkewbrace)

\textbf{Socle}(obj) \hspace{1cm} \text{(attribute)}

\textbf{Returns:} the socle of obj

The socle of a skew brace $A$ is the ideal ker $\lambda \cap Z(A, +)$.

\begin{verbatim}
gap> Socle(SmallSkewbrace(6,2));
<brace of size 1>
gap> Socle(SmallBrace(8,20));
<brace of size 8>
gap> Socle(SmallBrace(8,2));
<brace of size 4>
\end{verbatim}

4.3.10 Annihilator (for IsSkewbrace)

\textbf{Annihilator}(obj) \hspace{1cm} \text{(attribute)}

\textbf{Returns:} the annihilator of obj

The socle of a skew brace $A$ is the ideal ker $\lambda \cap Z(A, +) \cap Z(A, \circ)$.

\begin{verbatim}
gap> Annihilator(SmallSkewbrace(8,12));
<brace of size 2>
gap> Annihilator(SmallSkewbrace(4,2));
<brace of size 2>
gap> Annihilator(SmallSkewbrace(8,14));
<brace of size 4>
\end{verbatim}

4.4 Mutipermutation skew braces

4.4.1 SocleSeries (for IsSkewbrace)

\textbf{SocleSeries}(obj) \hspace{1cm} \text{(operation)}

\textbf{Returns:} the socle series of obj

The socle series of a skew brace $A$ is defined recursively as $A_1 = A$ and $A_{n+1} = A_n / \text{Soc}(A_n)$, see [SV18].
4.4.2 MultipermutationLevel (for IsSkewbrace)

▷ MultipermutationLevel(obj) (attribute)

Returns: the multipermutation level of the skew brace obj

The multipermutation level of a skew brace $A$ is defined as the smallest positive integer $n$ such that the $n$-th term $A_n$ of the socle series has only one element, see Definition 5.17 of [SV18].

\begin{verbatim}
gap> br := SmallBrace(8,20);;
gap> SocleSeries(br); 
[ <brace of size 8>, <brace of size 1> ]
gap> MultipermutationLevel(br); 2  
\end{verbatim}

4.4.3 IsMultipermutation (for IsSkewbrace)

▷ IsMultipermutation(obj) (property)

Returns: true if the skew brace obj has finite multipermutation level and false otherwise

4.4.4 Fix (for IsSkewbrace)

▷ Fix(obj) (attribute)

Returns: the left ideal $\{x \in A : \lambda(a)(x) = x \forall a \in A\}$ of the skew brace $A$.

\begin{verbatim}
gap> br := SmallSkewbrace(6,1);;
gap> IsTrivialSkewbrace(br); true
gap> Fix(br);
[ (), <(1,2,3)(4,5,6)>, <(1,3,2)(4,6,5)>, <(1,4)(2,6)(3,5)>, <(1,5)(2,4)(3,6)>, <(1,6)(2,5)(3,4)> ]  
\end{verbatim}

4.4.5 KernelOfLambda (for IsSkewbrace)

▷ KernelOfLambda(obj) (attribute)

Returns: the kernel of the map $\lambda$ as a subset of elements of the skew brace obj.

\begin{verbatim}
gap> br := SmallBrace(6,1);;
gap> KernelOfLambda(br); [ (), <(1,2,3)(4,5,6)>, <(1,3,2)(4,5,6)> ]  
\end{verbatim}

4.4.6 Quotient (for IsSkewbrace, IsSkewbrace)

▷ Quotient(obj, ideal) (operation)

Returns: the quotient obj by ideal

\begin{verbatim}
gap> br := SmallBrace(8,10);;
gap> ideals := Ideals(br);;
gap> Quotient(br, ideals[3]); <brace of size 4>
gap> br/ideals[3]; <brace of size 4>  
\end{verbatim}
4.5 Prime and semiprime ideals

4.5.1 IsPrimeBrace (for IsSkewbrace)

\[ \text{IsPrimeBrace}(\text{obj}) \]

\textbf{Returns:} \texttt{true} if the skew brace \texttt{obj} is prime
A skew brace \( A \) is said to be prime if for all non-zero ideals \( I \) and \( J \) one has \( I \ast J \neq 0 \)

\begin{verbatim}
gap> IsPrimeBrace(SmallBrace(24,12));
false

\end{verbatim}

\begin{verbatim}
gap> IsPrimeBrace(SmallBrace(24,94));
true

\end{verbatim}

4.5.2 IsPrimeIdeal (for IsSkewbrace and IsIdealInParent)

\[ \text{IsPrimeIdeal}(\text{obj}) \]

\textbf{Returns:} \texttt{true} if the ideal \texttt{obj} is prime
An ideal \( I \) of a skew brace \( A \) is said to be prime if \( A/I \) is a prime skew brace.

\begin{verbatim}
gap> br := SmallBrace(24,94);
<brace of size 24>

\end{verbatim}

\begin{verbatim}
gap> IsPrimeBrace(br);
true

\end{verbatim}

\begin{verbatim}
gap> Ideals(br);;
\gap> IsPrimeIdeal(last[2]);
true

\end{verbatim}

4.5.3 PrimeIdeals (for IsSkewbrace)

\[ \text{PrimeIdeals}(\text{obj}) \]

\textbf{Returns:} the list of prime ideals of the skew brace \texttt{obj}

\begin{verbatim}
gap> Length(PrimeIdeals(SmallBrace(24,94)));
2

\end{verbatim}

4.5.4 IsSemiprime (for IsSkewbrace)

\[ \text{IsSemiprime}(\text{obj}) \]

\textbf{Returns:} \texttt{true} if the skew brace \texttt{obj} is semiprime
An ideal \( I \) of a skew brace \( A \) is said to be semiprime if \( A/I \) is a semiprime skew brace.

\begin{verbatim}
gap> br := DirectProductSkewbraces(SmallSkewbrace(12,22),SmallSkewbrace(12,22));;

\end{verbatim}

\begin{verbatim}
gap> IsSemiprime(br);
true

\end{verbatim}
4.5.5  IsSemiprimeIdeal (for IsSkewbrace and IsIdealInParent)

\[ \text{IsSemiprimeIdeal}(\text{obj}) \]

\text{Returns:} \quad \text{true if the ideal obj is semiprime}

Example

\begin{verbatim}
gap> SemiprimeIdeals(SmallSkewbrace(12,24));
[ <skew brace of size 12> ]
gap> IsSemiprimeIdeal(last[1]);
true
\end{verbatim}

4.5.6  SemiprimeIdeals (for IsSkewbrace)

\[ \text{SemiprimeIdeals}(\text{obj}) \]

\text{Returns:} \quad \text{the list of semiprime ideals of the skew brace obj}

Example

\begin{verbatim}
gap> SemiprimeIdeals(SmallSkewbrace(12,24));
[ <skew brace of size 12> ]
gap> Length(SemiprimeIdeals(SmallSkewbrace(12,22)));
2
\end{verbatim}

4.5.7  BaerRadical (for IsSkewbrace)

\[ \text{BaerRadical}(\text{obj}) \]

\text{Returns:} \quad \text{the Baer radical of the skew brace obj}

Example

\begin{verbatim}
gap> br := SmallSkewbrace(6,2);
gap> BaerRadical(br);
<skew brace of size 6>
\end{verbatim}

4.5.8  IsBaer (for IsSkewbrace)

\[ \text{IsBaer}(\text{obj}) \]

\text{Returns:} \quad \text{true if the skew brace obj is a Baer radical skew brace.}

A skew brace A is said to be Baer radical if \( A = B(A) \), where \( B(A) \) is the Baer radical of A (i.e., the intersection of all prime ideals of A).

Example

\begin{verbatim}
gap> br := SmallSkewbrace(6,2);
gap> IsBaer(br);
true
\end{verbatim}

4.5.9  WedderburnRadical (for IsSkewbrace)

\[ \text{WedderburnRadical}(\text{obj}) \]

\text{Returns:} \quad \text{the Wedderburn radical of the skew brace obj}

The Wedderburn radical of a skew brace is the intersection of all its prime ideals

Example

\begin{verbatim}
gap> br := SmallSkewbrace(6,2);
gap> WedderburnRadical(br);
<brace of size 3>
\end{verbatim}
4.5.10 SolvableSeries (for IsSkewbrace)

\[ \text{SolvableSeries}(\text{obj}) \]

\textbf{Returns:} a list with the solvable series of the skew brace \text{obj}

The solvable series of a skew brace \( A \) is defined recursively as \( A_1 = A \) and \( A_{n+1} = A_n \ast A_n \) for \( n \geq 1 \), where \( a \ast b = \lambda_a(b) - b \).

\textbf{Example}

\begin{verbatim}
gap> br := SmallSkewbrace(8,20);;
gap> IsSolvable(br);
true
gap> SolvableSeries(br);
[ <skew brace of size 8>, <brace of size 2>, <brace of size 1> ]
gap> br := SmallSkewbrace(12,23);
gap> IsSolvable(br);
false
\end{verbatim}

4.5.11 IsMinimalIdeal (for IsSkewbrace and IsIdealInParent)

\[ \text{IsMinimalIdeal}(\text{obj}, \text{ideal}) \]

\textbf{Returns:} true if \text{ideal} is a minimal ideal of \text{obj} An ideal \( I \) of \( A \) is said to be \textit{minimal} if does not contain any other ideal of \( A \). To check if an ideal \( I \) of \( A \) is minimal, one computes the ideals of \( I \) and keep only those that are simple as a skew brace.

4.5.12 MinimalIdeals (for IsSkewbrace)

\[ \text{MinimalIdeals}(\text{obj}) \]

\textbf{Returns:} a list of minimal ideals of the skew brace \text{obj}
References


# Index

<table>
<thead>
<tr>
<th>AdditiveGroupOfRing</th>
<th>for IsRing, 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annihilator</td>
<td>for IsSkewbrace, 18</td>
</tr>
<tr>
<td>AsIdeal</td>
<td>for IsSkewbrace, IsCollection, 15</td>
</tr>
<tr>
<td>AutomorphismGroup</td>
<td>for IsSkewbrace, 4</td>
</tr>
<tr>
<td>BaerRadical</td>
<td>for IsSkewbrace, 21</td>
</tr>
<tr>
<td>Brace2YB</td>
<td>for IsSkewbrace, 7</td>
</tr>
<tr>
<td>DehornoyClass</td>
<td>for IsYB, 11</td>
</tr>
<tr>
<td>DehornoyRepresentationOfStructureGroup</td>
<td>for IsYB, IsObject, 11</td>
</tr>
<tr>
<td>DirectProductOp</td>
<td>for IList, IsSkewbrace, 6</td>
</tr>
<tr>
<td>DirectProductSkewbraces</td>
<td>for IsSkewbrace, IsSkewbrace, 5</td>
</tr>
<tr>
<td>Evaluate</td>
<td>for IsYB, IsList, 10</td>
</tr>
<tr>
<td>Fix</td>
<td>for IsSkewbrace, 19</td>
</tr>
<tr>
<td>IdBrace</td>
<td>for IsSkewbrace, 5</td>
</tr>
<tr>
<td>IdealGeneratedBy</td>
<td>for IsSkewbrace, IsCollection, 15</td>
</tr>
<tr>
<td>Ideals</td>
<td>for IsSkewbrace, 15</td>
</tr>
<tr>
<td>IdSkewbrace</td>
<td>for IsSkewbrace, 4</td>
</tr>
<tr>
<td>IdYB</td>
<td>for IsYB, 11</td>
</tr>
<tr>
<td>IntersectionOfTwoIdeals</td>
<td>for IsSkewbrace and IsIdealInParent, IsSkewbrace and IsIdealInParent, 15</td>
</tr>
<tr>
<td>IsAutomorphismGroupOfSkewbrace</td>
<td>for IsAutomorphismGroup, 6</td>
</tr>
<tr>
<td>IsBaer</td>
<td>for IsSkewbrace, 21</td>
</tr>
<tr>
<td>IsBiSkewbrace</td>
<td>for IsSkewbrace, 6</td>
</tr>
<tr>
<td>IsClassical</td>
<td>for IsSkewbrace, 6</td>
</tr>
<tr>
<td>IsIdeal</td>
<td>for IsSkewbrace, IsCollection, 15</td>
</tr>
<tr>
<td>IsIndecomposable</td>
<td>for IsCycleSet, 13</td>
</tr>
<tr>
<td></td>
<td>for IsYB, 10</td>
</tr>
<tr>
<td>IsJacobsonRadical</td>
<td>for IsRing, 9</td>
</tr>
<tr>
<td>IsLeftIdeal</td>
<td>for IsSkewbrace, IsCollection, 14</td>
</tr>
<tr>
<td>IsLeftNilpotent</td>
<td>for IsSkewbrace, 16</td>
</tr>
<tr>
<td>IsMinimalIdeal</td>
<td>for IsSkewbrace and IsIdealInParent, 22</td>
</tr>
<tr>
<td>IsMultipermutation</td>
<td>for IsSkewbrace, 19</td>
</tr>
<tr>
<td>IsOfAbelianType</td>
<td>for IsSkewbrace, 6</td>
</tr>
<tr>
<td>IsOfNilpotentType</td>
<td>for IsSkewbrace, 7</td>
</tr>
<tr>
<td>IsomorphismSkewbraces</td>
<td>5</td>
</tr>
<tr>
<td>IsPrimeBrace</td>
<td>for IsSkewbrace, 20</td>
</tr>
<tr>
<td>IsPrimeIdeal</td>
<td>for IsSkewbrace and IsIdealInParent, 20</td>
</tr>
<tr>
<td>IsRightNilpotent</td>
<td>for IsSkewbrace, 20</td>
</tr>
<tr>
<td>IsSemiprime</td>
<td>for IsSkewbrace, 20</td>
</tr>
</tbody>
</table>

24
IsSemiprimeIdeal
  for IsSkewbrace and IsIdealInParent, 21
IsSimpleSkewbrace
  for IsSkewbrace, 17
IsSkewbrace
  for IsAttributeStoringRep, 3
IsTrivialSkewbrace
  for IsSkewbrace, 7
IsTwoSided
  for IsSkewbrace, 6
KernelOfLambda
  for IsSkewbrace, 19
LeftIdeals
  for IsSkewbrace, 14
LeftNilpotentIdeals
  for IsSkewbrace, 17
LeftSeries
  for IsSkewbrace, 16
LinearRepresentationOfStructureGroup
  for IsYB, 12
LyubashenkoYB
  for IsInt, IsPerm, IsPerm, 10
MinimalIdeals
  for IsSkewbrace, 22
MultipermutationLevel
  for IsSkewbrace, 19
PrimeIdeals
  for IsSkewbrace, 20
Quotient
  for IsSkewbrace, IsSkewbrace, 19
RightNilpotentIdeals
  for IsSkewbrace, 17
RightSeries
  for IsSkewbrace, 16
SemidirectProduct
  for IsSkewbrace, IsSkewbrace, IsGeneralMapping, 8
SemiprimeIdeals
  for IsSkewbrace, 21
Skewbrace
  for IsList, 3
Skewbrace2YB
  for IsSkewbrace, 7
SkewbraceSubset2YB
  for IsSkewbrace, IsCollection, 7
SmallBrace
  for IsInt, IsInt, 4
SmallSkewbrace
  for IsInt, IsInt, 3
SmoktunowiczSeries
  for IsSkewbrace, IsInt, 18
Socle
  for IsSkewbrace, 18
SocleSeries
  for IsSkewbrace, 18
SolvableSeries
  for IsSkewbrace, 22
StrongLeftIdeals
  for IsSkewbrace, 14
SumOfTwoIdeals
  for IsSkewbrace and IsIdealInParent,
    IsSkewbrace and IsIdealInParent, 16
Table
  for IsYB, 10
Table2YB
  for IsList, 9
TrivialBrace
  for IsGroup, 4
TrivialSkewbrace
  for IsGroup, 4
UnderlyingAdditiveGroup
  for IsSkewbrace, 8
UnderlyingMultiplicativeGroup
  for IsSkewbrace, 8
WedderburnRadical
  for IsSkewbrace, 21