Contents

Chapter 1

Classes

1.1 group – algorithms for finite groups

- **Classes**
  - **Group**
  - **GroupElement**
  - **GenerateGroup**
  - **AbelianGenerate**

1.1.1 †Group – group structure

Initialize (Constructor)

Group(value: *class*, operation: *int*=-1) → Group

Create an object which wraps value (typically a ring or a field) only to expose its group structure.

The instance has methods defined for (abstract) group. For example, **identity** returns the identity element of the group from wrapped value.

value must be an instance of a class expresses group structure. operation must be 0 or 1; If operation is 0, value is regarded as the additive group. On the other hand, if operation is 1, value is considered as the multiplicative group. The default value of operation is 0.
†You can input an instance of **Group** itself as value. In this case, the default value of operation is the attribute **operation** of the instance.

Attributes

entity :
The wrapped object.

operation :
It expresses the mode of operation; 0 means additive, while 1 means multiplicative.

Operations

operator	explanation
`A==B`	Return whether A and B are equal or not.
`A!=B`	Check whether A and B are not equal.
`repr(A)`	representation
`str(A)`	simple representation

Examples

```
>>> G1=group.Group(finitefield.FinitePrimeField(37), 1)
>>> print G1
F_37
>>> G2=group.Group(intresidue.IntegerResidueClassRing(6), 0)
```

```
>>> print G2
Z/6Z
```

Methods

1.1.1.1 setOperation – change operation

setOperation(self, operation: *int*) → (None)

群のタイプを加法 (0) または乗法 (1) に変える。

operation は 0 または 1。

1.1.1.2 †createElement – generate a GroupElement instance

createElement(self, *value) → *GroupElement*

Return **GroupElement** object whose group is self, initialized with value.

† この方法は self と呼ぶ。linkingtwogroupGroupentity.createElement.

value must fit the form of argument for self.**entity**.createElement.

1.1.1.3 †identity – identity element

identity(self) → GroupElement

群の単位元の値を返す。

operation によって 0 (加法) または 1 (乗法) を返す。† この方法は paramself.**entity** と呼ばれている。identity または **entity** が属性をもたないときは 0 か 1 を返す。

1.1.1.4 grouporder – order of the group

grouporder(self) → long

paramself の要素の個数の値を返す。.

† この方法は self と呼ばれている。**entity**.grouporder, card or __len__.
ここではこの群は有限と考え、 返す値は long 型の整数である。もしこの群が無限の場合、この方法では出力は定義できない。

Examples

```
>>> G1=group.Group(finitefield.FinitePrimeField(37), 1)
>>> G1.grouporder()
36
>>> G1.setOperation(0)
>>> print G1.identity()
FinitePrimeField,0 in F_37
>>> G1.grouporder()
37
```

1.1.2 GroupElement – elements of group structure

Initialize (Constructor)

GroupElement(value: *class*, operation: *int*=-1) → GroupElement

Create an object which wraps value (typically a ring element or a field element) to make it behave as an element of group.

The instance has methods defined for an (abstract) element of group. For example, **inverse** returns the inverse element of value as the element of group object.

value must be an instance of a class expresses an element of group structure. operation must be 0 or 1; If operation is 0, value is regarded as the additive group. On the other hand, if operation is 1, value is considered as the multiplicative group. The default value of operation is 0.
†You can input an instance of **GroupElement** itself as value. In this case, the default value of operation is the attribute **operation** of the instance.

Attributes

entity :
The wrapped object.

set :
It is an instance of **Group**, which expresses the group to which self belongs.

operation :
It expresses the mode of operation; 0 means additive, while 1 means multiplicative.

Operations

operator	explanation
`A==B`	Return whether A and B are equal or not.
`A!=B`	Check whether A and B are not equal.
`A.ope(B)`	Basic operation (additive +, multiplicative ∗)
`A.ope2(n)`	Extended operation (additive ∗, multiplicative ∗∗)
`A.inverse()`	Return the inverse element of self
`repr(A)`	representation
`str(A)`	simple representation

Examples

```
>>> G1=group.GroupElement(finitefield.FinitePrimeFieldElement(18, 37), 1)
>>> print G1
FinitePrimeField,18 in F_37
>>> G2=group.Group(intresidue.IntegerResidueClass(3, 6), 0)
IntegerResidueClass(3, 6)
```

Methods

1.1.2.1 setOperation – change operation

setOperation(self, operation: *int*) → (None)

群のタイプを加法 (0) または乗法 (1) に変える。

operation は 0 か 1。

1.1.2.2 †getGroup – generate a Group instance

getGroup(self) → *Group*

Return **Group** object to which self belongs.

†This method calls self.**entity**.getRing or getGroup.
†In an initialization of **GroupElement**, the attribute **set** is set as the value returned from the method.

1.1.2.3 order – order by factorization method

order(self) → long

self の位数の値を返す。

† この方法は群の位数の因数分解を使う。
† ここではこの群は有限と考え、 返す値は long 型の整数である。† もしここの群が無限ならば、この方法はエラーを返すか有効でない値を返す。

1.1.2.4 t_order – order by baby-step giant-step

t_order(self, v: *int*=2) → long

self の位数の値を返す。

† この方法は Terry's baby-step giant-step algorithm を使う。
この方法は群の位数を使わない。v に baby-step の数を入れる。† ここではこの群は有限と考え、 返す値は long 型の整数である。† もしここの群が無限ならば、この方法はエラーを返すか有効でない値を返す。

v は int 型の整数。

Examples

```
>>> G1=group.GroupElement(finitefield.FinitePrimeFieldElement(18, 37), 1)
>>> G1.order()
36
>>> G1.t_order()
36
```

1.1.3 †GenerateGroup – group structure with generator

Initialize (Constructor)

GenerateGroup(value: *class*, operation: *int*=-1) → GroupElement

Create an object which is generated by value as the element of group structure.

This initializes a group 'including' the group elements, not a group with generators, now. We do not recommend using this module now. The instance has methods defined for an (abstract) element of group. For example, **inverse** returns the inverse element of value as the element of group object.
The class inherits the class **Group**.

value must be a list of generators. Each generator should be an instance of a class expresses an element of group structure. operation must be 0 or 1; If operation is 0, value is regarded as the additive group. On the other hand, if operation is 1, value is considered as the multiplicative group. The default value of operation is 0.

Examples

```
>>> G1=group.GenerateGroup([intresidue.IntegerResidueClass(2, 20),
... intresidue.IntegerResidueClass(6, 20)])
>>> G1.identity()
intresidue.IntegerResidueClass(0, 20)
```

1.1.4 AbelianGenerate – abelian group structure with generator

Initialize (Constructor)

GenerateGroup のクラスを継承する。

1.1.4.1 relationLattice – relation between generators

relationLattice(self) → Matrix

格子原理関係にある数のリストを返す。as a square matrix each of whose column vector is a relation basis.

関係の原理 V は $\prod_i \text{generator}_i V_i = 1$ を充たす。

1.1.4.2 computeStructure – abelian group structure

computeStructure(self) → tuple

有限アーベル群構造を計算する。

もし self $G \simeq \oplus_i < h_i >$ で、 $[(h_1, \text{ord}(h_1)),..(h_n, \text{ord}(h_n))]$ と $^{\#}G$ を返す。$< h_i >$ は変数 h_i の巡回群。

出力は二つずつ要素を持つ三つの組である。; 最初の要素は h_i とのそ位数のリストである。, また、二番目の要素は群の位数である。

Examples

```
>>> G=AbelianGenerate([intresidue.IntegerResidueClass(2, 20),
... intresidue.IntegerResidueClass(6, 20)])
>>> G.relationLattice()
10 7
 0 1
>>> G.computeStructure()
([IntegerResidueClassRing,IntegerResidueClass(2, 20), 10)], 10L)
```

Bibliography