Action ’c’ and Kashiwara-Nakashima tableaux

Release 1.0

Harsh Patel

Aug 30, 2019

CONTENTS

CHAPTER

ONE

ALPHABET MODULE

Module containing definition of Alphabet class

class `alphabet.`**`Alphabet`**(*num*)

Bases: `object`

This class defines the generalized alphabet and its behaviour.

`__eq__`(*other*)

Function to check equality of the two Alphabet objects, using ==

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if both alphabet object are equal, and False otherwise.

`__ge__`(*other*)

Function to check if invoker is greater than of equal to the other Alphabet object, using >=

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if invoker is greater than or equal to the other, False otherwise.

`__gt__`(*other*)

Function to check invoker object is greater than the other Alphabet object, using >

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if invoker is greater than other, False otherwise.

`__init__`(*num*)

Initialization function for Alphabet

Parameters **`num`** (`int`) – Number that is to be decorated with a specific type of alphabet

`__le__`(*other*)

Function to check if invoker is less than or equal to the other Alphabet object, using <=

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if invoker is less than or equal to the other, False otherwise.

`__ne__`(*other*)

Function to check inequality of two Alphabet objects, using !=

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if both alphabet object are not equal, and False otherwise.

class `alphabet.`**`Barred`**(*num*)

Bases: *`alphabet.Alphabet`*

This class defines the Barred alphabet, which is a specialization of Alphabet.

`__lt__`(*other*)

Function to check if invoker is less than the other Alphabet object, using <

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if invoker is less than the other, False otherwise.

`__str__`()

Function to get the string representation of the Barred object.

Returns string representation of invoking object.

Return type str

`clone`()

Function to make a copy of the invoking Barred object.

Returns Copy of invoking object

Return type *Barred*

class `alphabet.`**`Ordinary`**(*num*)

Bases: *`alphabet.Alphabet`*

This class defines the Ordinary alphabet, which is a specialization of Alphabet.

`__lt__`(*other*)

Function to check if invoker is less than the other Alphabet object, using <

Parameters **`other`** (`Alphabet`) – alphabet object that we are comparing with.

Returns True if invoker is less than the other, False otherwise.

`__str__`()

Function to get the string representation of the Ordinary object.

Returns string representation of invoking object.

Return type str

`clone`()

Function to make a copy of the invoking Ordinary object.

Returns Copy of invoking object

Return type *Ordinary*

CHAPTER TWO

OPERATION MODULE

This module contains all the elementary function to apply action c on a kn tableau.

operation.**action_c**(*t*, *m=None*, *mutable=True*)
Function to apply action c on tableau.

Parameters

- **t** – tableau.
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar
- **mutable** – optional. Default set to True, where it directly make changes to the tableau t. If passed in as False, it creates a copy of t and operates on that copy, leaving t unaffected.

Returns tableau obtained after applying c on t.

operation.**e_i**(*t*, *i*, *m*, *mutable=True*)
Function to apply e_i on tableau.

Parameters

- **t** – tableau.
- **i** – i as in e_i
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar
- **mutable** – optional. Default set to True, where it directly make changes to the tableau t. If passed in as False, it creates a copy of t and operates on that copy, leaving t unaffected.

Returns tableau after applying e_i

operation.**evaluate**(*poly*, *q*, *inp*)
Function to evaluate polynomial at a particular value.

Parameters

- **poly** – polynomial.
- **q** – variable in polynomial.
- **inp** – input value where we need to evaluate the polynomial.

Returns value obtained after evaluation.

operation.**f_i**(*t*, *i*, *m*, *mutable=True*)
Function to apply f_i on tableau.

Parameters

- **t** – tableau.

- **i** – i as in f_i
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar
- **mutable** – optional. Default set to True, where it directly make changes to the tableau t. If passed in as False, it creates a copy of t and operates on that copy, leaving t unaffected.

Returns tableau after applying f_i

`operation.`**`get_crystal_graph`**(*shape*, *m=None*)

Function to create crystal graph data.

Parameters

- **shape** – shape of tableaux in graph
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns a list of all k-n tableaux for given shape and value of m, an f_i dictionary with all the kn tableaux as keys with a list as value, where (i-1)st index in list refer to tableau obtained by applying f_i to the key tableau. A similar e_i tableaux dictionary.

`operation.`**`get_kn_schur_polynomial`**(*shape*, *m=None*)

Function to compute the kn-Schur polynomial in q for a given shape and value of m.

Parameters

- **shape** – shape of tableaux in set.
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns Schur polynomial in variable q. and variable q.

`operation.`**`get_power_of_q`**(*t*, *m*)

Power of q in the monomial that corresponds to tableau.

Parameters

- **t** – tableau.
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns power of q in the monomial that corresponds to tableau t.

`operation.`**`get_tableaux_set`**(*shape*, *m=None*)

Function to obtain all the kn tableaux of given shape and value of m.

Parameters

- **shape** – shape of tableaux.
- **m** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns list of all the tableaux of given shape and value of m.

`operation.`**`k_of_shape`**(*shape*)

statistic of shape as defined in OH's and PARK's paper.

Parameters **shape** – shape.

Returns statistic of shape.

`operation.`**`nth_root_of_unity`**(*n*)

Function to compute n-th root of unity.

Parameters **n** – n as in n-th root of unity.

Returns n-th root of unity in form of sympy complex number expression.

`operation.`**`order_of_set`**`(`*shape*, *m=None*`)`

Function to compute order of a set of kn tableaux of a given shape and value of m.

Parameters

- **`shape`** – shape of tableau.
- **`m`** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns dictionary with key as order and frequency of that order in set as value of that key.

`operation.`**`order_of_tableau`**`(`*t*, *m=None*`)`

Function to compute the order of action c.

Parameters

- **`t`** – tableau.
- **`m`** – m such that allowed entries in t are 1 to m and 1-bar to m-bar

Returns order of tableau t under action c.

`operation.`**`s_i`**`(`*t*, *i*, *m=None*, *mutable=True*`)`

Function to apply s_i to the tableau.

Parameters

- **`t`** – tableau.
- **`i`** – i as in s_i
- **`m`** – m such that allowed entries in t are 1 to m and 1-bar to m-bar
- **`mutable`** – optional. Default set to True, where it directly make changes to the tableau t. If passed in as False, it creates a copy of t and operates on that copy, leaving t unaffected.

Returns tableau obtained after applying s_i on t.

CHAPTER

THREE

TABLEAU MODULE

This module contains definition of Tableau class.

class tableau.**Tableau**(*body*)

Bases: object

This class defines a young tableau and provide functions to operate on it.

__eq__(*other*)

Function to check if two Tableau object are same in the sense that they have same entries and same location for all location of boxes, using ==

Parameters **other** – Tableau object with which invoking Tableau object is being compared.

Returns True if both Tableau object are same. False otherwise.

__init__(*body*)

Initialization function for Tableau object.

Parameters **body** – body of tableau.

__ne__(*other*)

Function to check if two Tableau object are not same.

Parameters **other** –

param other Tableau object with which invoking Tableau object is being compared.

Returns False if both Tableau object are same. True otherwise.

__str__()

Function to get string representation of the Tableau object.

Returns string representation of the Tableau object.

Return type str

add_new_box(*i*, *element*)

Function to add a new box at the end of a row.

Parameters

- **i** – row index.
- **element** – element to be added in new box.

Returns None

clone()

Function to creat a copy of the invoking Tableau object

Returns copy of invoking tableau object.

Return type *Tableau*

`del_row_lastbox`(*i*)

Function to delete the last box in the row.

Parameters **`i`** – row index

Returns None

`static get_shape_transpose`(*shape*)

Function to compute the transpose of a given shape.

Parameters **`shape`** – shape for which we need to calculate transpose.

Returns list representing transpose shape.

Return type list

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX