Type of a tableau, definition and properties

François Viard

ICJ

Mars 2014

Plan

Reduced decompositions in the symmetric group

- It is well known that the symmetric groups S_n is generated by the **simple transpositions** $s_i = (i, i+1)$, $i \in \mathbb{N}$.

Reduced decompositions in the symmetric group

- It is well known that the symmetric groups S_n is generated by the **simple transpositions** $s_i = (i, i+1)$, $i \in \mathbb{N}$.

- Set $\sigma \in S_n$, we define $\ell(\sigma)$ the minimal integer such that $\sigma = s_{i_1} \cdots s_{i_{\ell(\sigma)}}$. Such a product is called a **reduced decomposition**.

Reduced decompositions in the symmetric group

- It is well known that the symmetric groups S_n is generated by the **simple transpositions** $s_i = (i, i+1)$, $i \in \mathbb{N}$.
- Set $\sigma \in S_n$, we define $\ell(\sigma)$ the minimal integer such that $\sigma = s_{i_1} \cdots s_{i_{\ell(\sigma)}}$. Such a product is called a **reduced decomposition**.
- It is classical that $\ell(\sigma) = |\mathrm{Inv}(\sigma)|$, where

$$\mathrm{Inv}(\sigma) = \{(p, q) \mid p < q \text{ and } \sigma^{-1}(p) > \sigma^{-1}(q)\}$$

Partitions and standard tableaux

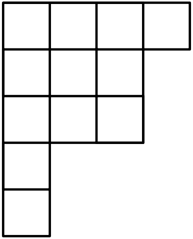
Definition

A partition of the integer n is a non-increasing sequence of non-negative integers $\lambda_1 \geq \lambda_2 \geq \ldots$ such that $\sum_i \lambda_i = n$.

Partitions and standard tableaux

Definition

A partition of the integer n is a non-increasing sequence of non-negative integers $\lambda_1 \geq \lambda_2 \geq \dots$ such that $\sum_i \lambda_i = n$.

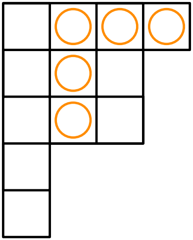


Ferrers diagram of the partition $\lambda = (4, 3, 3, 1, 1)$.

Partitions and standard tableaux

Definition

A partition of the integer n is a non-increasing sequence of non-negative integers $\lambda_1 \geq \lambda_2 \geq \ldots$ such that $\sum_i \lambda_i = n$.

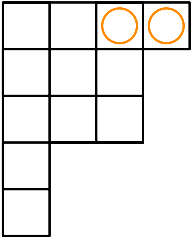


The hook based on $(1, 2)$, denoted $H_{(1,2)}(\lambda)$.

Partitions and standard tableaux

Definition

A partition of the integer n is a non-increasing sequence of non-negative integers $\lambda_1 \geq \lambda_2 \geq \ldots$ such that $\sum_i \lambda_i = n$.



Arm based on $(1, 2)$.

Partitions and standard tableaux

Definition Standard Tableaux

A *standard Young Tableau* of shape λ is a filling of λ with all the integers from 1 to n such that the integers are increasing from left to right and from top to bottom. The set of all such tableaux is denoted $SYT(\lambda)$ and $f^{\lambda} = |SYT(\lambda)|$.

1	2	5	10
3	7	9	
4	8	11	
6			
12			

A Standard Young Tableau of shape $(4, 3, 3, 1, 1)$.

Enumeration of reduced decompositions

Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Stanley, 1984)

$$\mathrm{red}(\omega_0) = f^{\lambda_n} = \frac{\binom{n}{2}!}{1^{n-1}3^{n-2}5^{n-3}\cdots(2n-3)^1}$$

Enumeration of reduced decompositions

Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Stanley, 1984)

$$\mathrm{red}(\omega_0) = f^{\lambda_n} = \frac{\binom{n}{2}!}{1^{n-1}3^{n-2}5^{n-3}\cdots(2n-3)^1}$$

- The proof is not bijective and is based on the study of a symmetric function.

Enumeration of reduced decompositions

Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Stanley, 1984)

$$\mathrm{red}(\omega_0) = f^{\lambda_n} = \frac{\binom{n}{2}!}{1^{n-1}3^{n-2}5^{n-3}\cdots(2n-3)^1}$$

- The proof is not bijective and is based on the study of a symmetric function.
- Stanley also conjectured that for all $\sigma \in S_n$,

$$\mathrm{red}(\sigma) = \sum_{\lambda} a_\lambda f^\lambda$$

where the sum is over the partitions of $\ell(\sigma)$ and $a_\lambda \geq 0$.

Enumeration of reduced decomposition

Theorem (Edelman-Greene / Lascoux-Schützenberger, 1987)

Set $\sigma \in S_n$. There exists a sequence of non-negative integers a_λ such that

$$\mathrm{red}(\sigma) = \sum_{\lambda \vdash \ell(\sigma)} a_\lambda f^\lambda$$

Enumeration of reduced decomposition

Theorem (Edelman-Greene / Lascoux-Schützenberger, 1987)

Set $\sigma \in S_n$. There exists a sequence of non-negative integers a_λ such that

$$\mathrm{red}(\sigma) = \sum_{\lambda \vdash \ell(\sigma)} a_\lambda f^\lambda$$

- LS : the proof is based on the study of Schubert polynomials (with this point of view $a_\lambda = \#\{\text{leafs of type } \lambda \text{ in the LS-Tree}\}$).

Enumeration of reduced decomposition

Theorem (Edelman-Greene / Lascoux-Schützenberger, 1987)

Set $\sigma \in S_n$. There exists a sequence of non-negative integers a_λ such that

$$\mathrm{red}(\sigma) = \sum_{\lambda \vdash \ell(\sigma)} a_\lambda f^\lambda$$

- LS : the proof is based on the study of Schubert polynomials (with this point of view $a_\lambda = \#\{\text{leafs of type } \lambda \text{ in the LS-Tree}\}$).
- The proof of Edelman and Greene is purely bijective and is based on a RSK-like insertion (here $a_\lambda = \#\{\text{EG-tableaux of shape } \lambda\}$).

Balanced tableaux

Where they come from

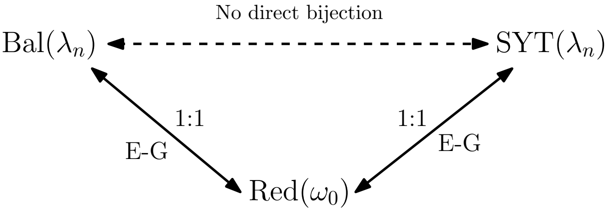
In their first attempt to find a combinatorial proof of the Stanley's theorem, Edelman and Greene introduced a new set of tableaux $\mathrm{Bal}(\lambda)$ of shape λ called **balanced tableaux**.

Balanced tableaux

Where they come from

In their first attempt to find a combinatorial proof of the Stanley's theorem, Edelman and Greene introduced a new set of tableaux $\mathrm{Bal}(\lambda)$ of shape λ called **balanced tableaux**.

Recall : $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$ the staircase partition.



Balanced tableaux

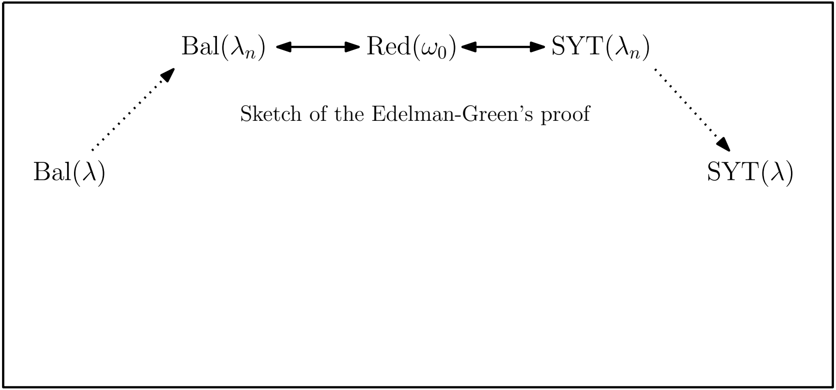
Theorem (Edelman-Greene, 1987)

For any partition λ, we have that $|\mathrm{SYT}(\lambda)| = |\mathrm{Bal}(\lambda)|$.

Balanced tableaux

Theorem (Edelman-Greene, 1987)

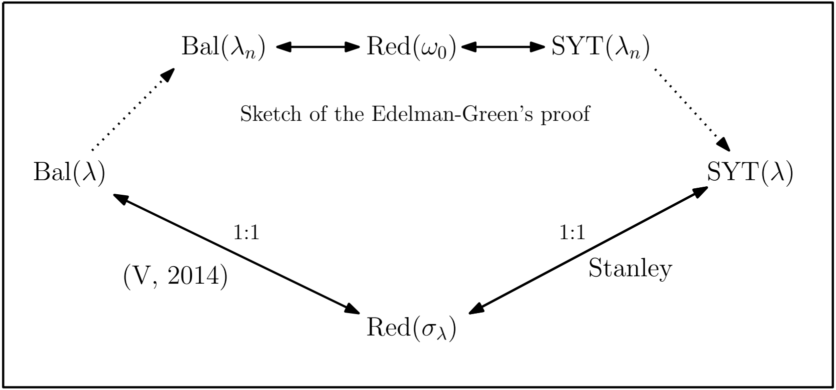
For any partition λ, we have that $|\mathrm{SYT}(\lambda)| = |\mathrm{Bal}(\lambda)|$.



Balanced tableaux

Theorem (Edelman-Greene, 1987)

For any partition λ, we have that $|\mathrm{SYT}(\lambda)| = |\mathrm{Bal}(\lambda)|$.



Balanced tableaux

Definition of balanced tableaux

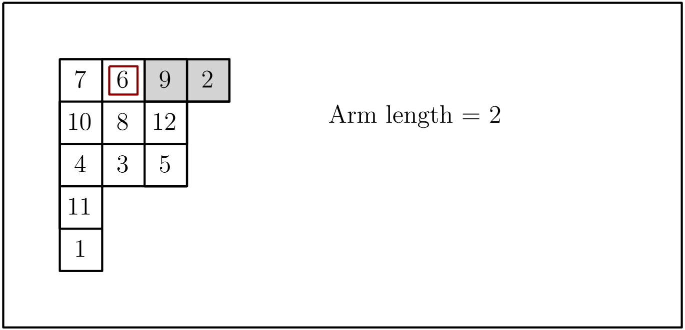
Set $T = (t_c)_{c\in\lambda}$ a tableau of shape λ. T is a balanced tableau if and only if for all boxes $c \in \lambda$ we have $|\{z \in H_c(\lambda) \mid t_z > t_c\}| = a_c$.

7	6	9	2
10	8	12	
4	3	5	
11			
1			

Balanced tableaux

Definition of balanced tableaux

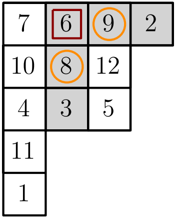
Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. T is a balanced tableau if and only if for all boxes $c \in \lambda$ we have $|\{z \in H_c(\lambda) \mid t_z > t_c\}| = a_c$.



Balanced tableaux

Definition of balanced tableaux

Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. T is a balanced tableau if and only if for all boxes $c \in \lambda$ we have $|\{z \in H_c(\lambda) \mid t_z > t_c\}| = a_c$.



Arm length $= 2$

$$|\{z \in H_c(\lambda) \mid t_z > t_c\}| = 2$$

Definition of the type of a tableau

Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition of the type of a tableau

Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition

Set λ a partition of n.

- Set $T = (t_c)_{c\in\lambda}$ a tableau of shape λ. The **type** of T is the filling of λ with the integers $\theta_c = |\{z \in H_c(\lambda) \mid t_z > t_c\}|$.

Definition of the type of a tableau

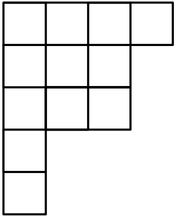
Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition

Set λ a partition of n.

- Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. The **type** of T is the filling of λ with the integers $\theta_c = |\{z \in H_c(\lambda) \mid t_z > t_c\}|$.

6	10	5	3
7	12	9	
11	2	4	
8			
1			



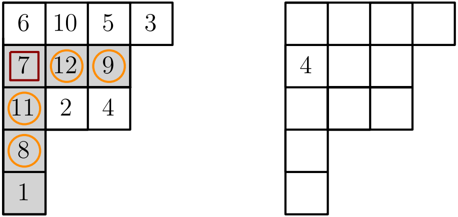
Definition of the type of a tableau

Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition

Set λ a partition of n.

- Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. The **type** of T is the filling of λ with the integers $\theta_c = |\{z \in H_c(\lambda) \mid t_z > t_c\}|$.



Definition of the type of a tableau

Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition

Set λ a partition of n.

- Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. The **type** of T is the filling of λ with the integers $\theta_c = |\{z \in H_c(\lambda) \mid t_z > t_c\}|$.

6	10	5	3
7	12	9	
11	2	4	
8			
1			

4			
4			

Definition of the type of a tableau

Now we will introduce a new combinatorial object associated to each tableau, in order to classify ALL of them (even if they are not standard and not balanced).

Definition

Set λ a partition of n.

- Set $T = (t_c)_{c \in \lambda}$ a tableau of shape λ. The **type** of T is the filling of λ with the integers $\theta_c = |\{z \in H_c(\lambda) \mid t_z > t_c\}|$.

6	10	5	3
7	12	9	
11	2	4	
8			
1			

4	1	1	0
4	0	0	
0	1	0	
0			
0			

Classification of all tableaux using their types

Definition

The set of all the tableaux which are of type $\mathcal{T}$ is denoted $\mathrm{Tab}(\mathcal{T})$.

Classification of all tableaux using their types

Definition

The set of all the tableaux which are of type $\mathcal{T}$ is denoted $\mathrm{Tab}(\mathcal{T})$.

Type $\begin{array}{\|c\|c\|c\|}\hline 3 & 2 & 0 \\ \hline 1 & 0 & \\ \cline{1-2}\end{array}$	Type $\begin{array}{\|c\|c\|c\|}\hline 2 & 1 & 0 \\ \hline 1 & 0 & \\ \cline{1-2}\end{array}$

Classification of all tableaux using their types

Definition

The set of all the tableaux which are of type $\mathcal{T}$ is denoted $\mathrm{Tab}(\mathcal{T})$.

Type

3	2	0
1	0	

Type

2	1	0
1	0	

Standard Young Tableaux

1	2	3
4	5	

1	2	4
3	5	

1	2	5
3	4	

1	3	4
2	5	

1	3	5
2	4	

Classification of all tableaux using their types

Definition

The set of all the tableaux which are of type $\mathcal{T}$ is denoted $\mathrm{Tab}(\mathcal{T})$.

Type	Type
$\begin{array}{\|c\|c\|c\|}\hline 3 & 2 & 0 \\ \hline 1 & 0 & \\ \cline{1-2}\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 2 & 1 & 0 \\ \hline 1 & 0 & \\ \cline{1-2}\end{array}$
Standard Young Tableaux	Balanced tableaux
$\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 3 \\ \hline 4 & 5 & \\ \cline{1-2}\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 3 & 4 & 5 \\ \hline 1 & 2 & \\ \cline{1-2}\end{array}$
$\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 4 \\ \hline 3 & 5 & \\ \cline{1-2}\end{array}$ $\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 5 \\ \hline 3 & 4 & \\ \cline{1-2}\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 2 & 4 & 5 \\ \hline 1 & 3 & \\ \cline{1-2}\end{array}$ $\begin{array}{\|c\|c\|c\|}\hline 2 & 4 & 3 \\ \hline 1 & 5 & \\ \cline{1-2}\end{array}$
$\begin{array}{\|c\|c\|c\|}\hline 1 & 3 & 4 \\ \hline 2 & 5 & \\ \cline{1-2}\end{array}$ $\begin{array}{\|c\|c\|c\|}\hline 1 & 3 & 5 \\ \hline 2 & 4 & \\ \cline{1-2}\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 2 & 4 & 1 \\ \hline 3 & 5 & \\ \cline{1-2}\end{array}$ $\begin{array}{\|c\|c\|c\|}\hline 2 & 3 & 1 \\ \hline 4 & 5 & \\ \cline{1-2}\end{array}$

A natural question

Lemma

Set λ a partition.

- Set $\mathcal{S}t_\lambda = (h_c - 1)_{c \in \lambda}$, then $\mathrm{Tab}(\mathcal{S}t_\lambda) = \mathrm{SYT}(\lambda)$.
- Set $\mathcal{B}_\lambda = (a_c)_{c \in \lambda}$, then $\mathrm{Tab}(\mathcal{B}_\lambda) = \mathrm{Bal}(\lambda)$.

A natural question

Lemma

Set λ a partition.

- Set $\mathcal{S}t_\lambda = (h_c - 1)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{S}t_\lambda) = \mathrm{SYT}(\lambda)$.
- Set $\mathcal{B}_\lambda = (a_c)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{B}_\lambda) = \mathrm{Bal}(\lambda)$.

With the Edelman-Greene's Theorem, we have that

$$|\mathrm{Tab}(\mathcal{B}_\lambda)| = |\mathrm{Tab}(\mathcal{S}t_\lambda)| = f^\lambda.$$

A natural question

Lemma

Set λ a partition.

- Set $\mathcal{S}t_\lambda = (h_c - 1)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{S}t_\lambda) = \mathrm{SYT}(\lambda)$.
- Set $\mathcal{B}_\lambda = (a_c)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{B}_\lambda) = \mathrm{Bal}(\lambda)$.

With the Edelman-Greene's Theorem, we have that

$$|\mathrm{Tab}(\mathcal{B}_\lambda)| = |\mathrm{Tab}(\mathcal{S}t_\lambda)| = f^\lambda.$$

Problem

Fix $\mathcal{T}$ a type. Can we find a formula for $|\mathrm{Tab}(\mathcal{T})|$?

A natural question

Lemma

Set λ a partition.

- Set $\mathcal{S}t_\lambda = (h_c - 1)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{S}t_\lambda) = \mathrm{SYT}(\lambda)$.
- Set $\mathcal{B}_\lambda = (a_c)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{B}_\lambda) = \mathrm{Bal}(\lambda)$.

With the Edelman-Greene's Theorem, we have that

$$|\mathrm{Tab}(\mathcal{B}_\lambda)| = |\mathrm{Tab}(\mathcal{S}t_\lambda)| = f^\lambda.$$

Problem

Fix $\mathcal{T}$ a type. Can we find a formula for $|\mathrm{Tab}(\mathcal{T})|$?

- In some cases yes.
- In general, the problem is open.

A natural question

Lemma

Set λ a partition.

- Set $\mathcal{S}t_\lambda = (h_c - 1)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{S}t_\lambda) = \mathrm{SYT}(\lambda)$.
- Set $\mathcal{B}_\lambda = (a_c)_{c\in\lambda}$, then $\mathrm{Tab}(\mathcal{B}_\lambda) = \mathrm{Bal}(\lambda)$.

With the Edelman-Greene's Theorem, we have that

$$|\mathrm{Tab}(\mathcal{B}_\lambda)| = |\mathrm{Tab}(\mathcal{S}t_\lambda)| = f^\lambda.$$

Problem

Fix $\mathcal{T}$ a type. Can we find a formula for $|\mathrm{Tab}(\mathcal{T})|$?

- In some cases yes.
- In general, the problem is open.
- We have a probabilistic result : the expected value for $|\mathrm{Tab}(\mathcal{T})|$, when we uniformly pick a type $\mathcal{T}$, is f^λ.

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

Remaining boxes : 12

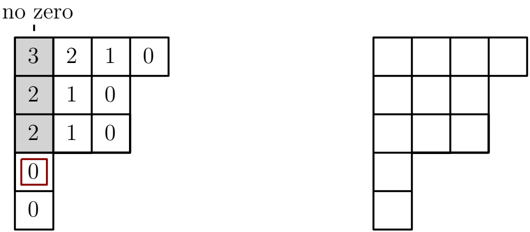
$L =$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



Remaining boxes : 12

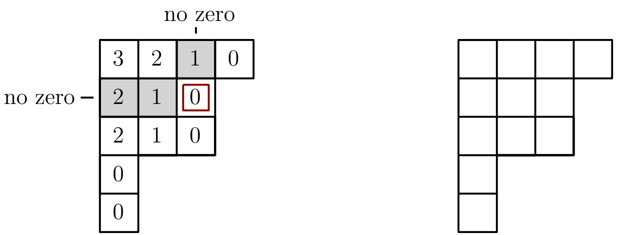
$L =$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



Remaining boxes : 12

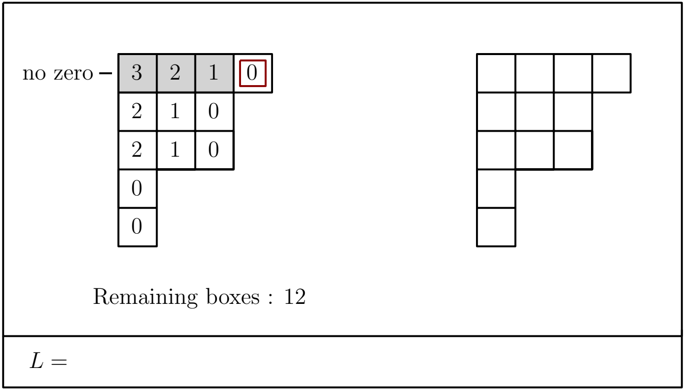
$L =$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

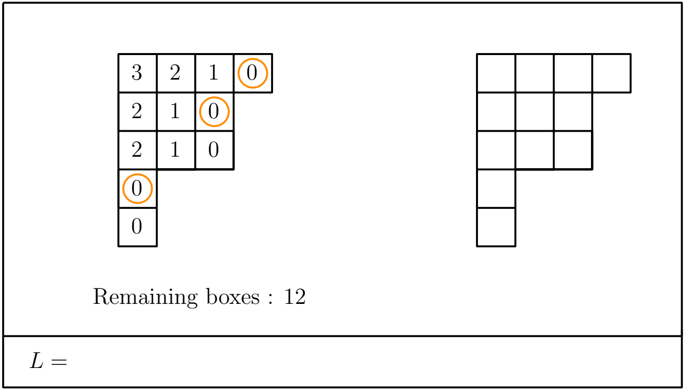


The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

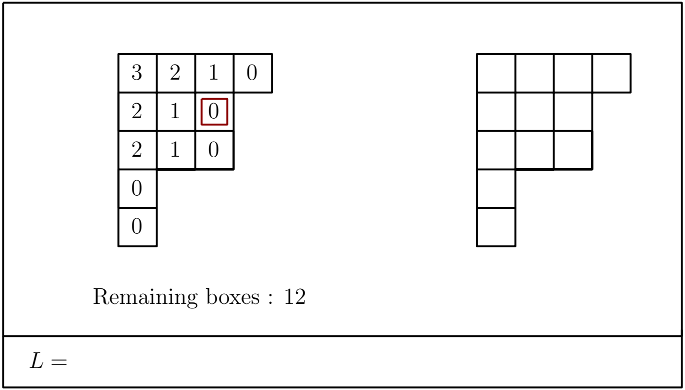


The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

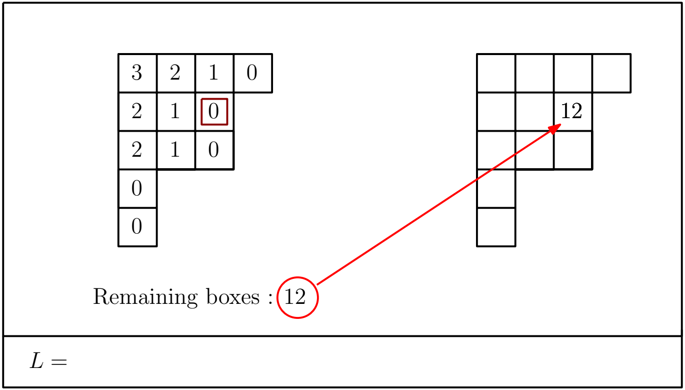


The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

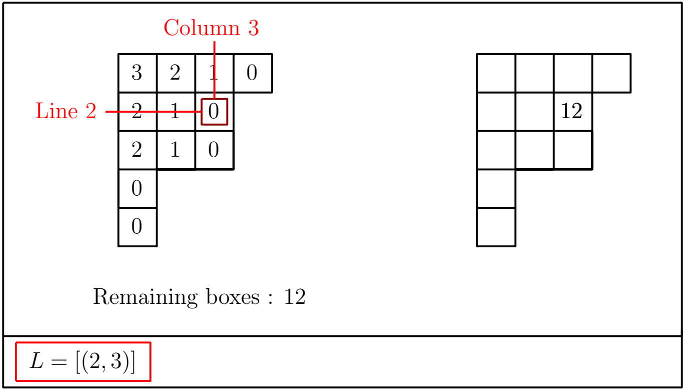


The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

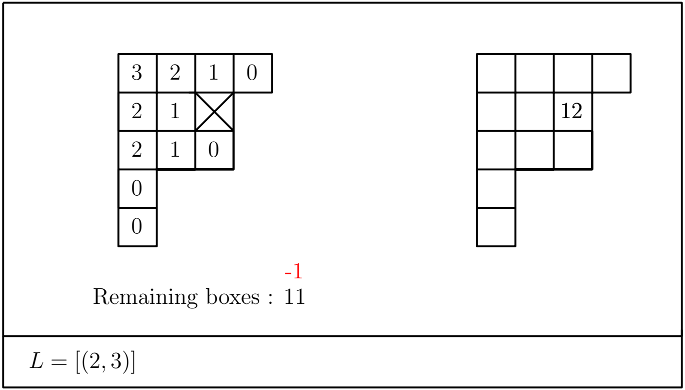


The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

Remaining boxes : 11

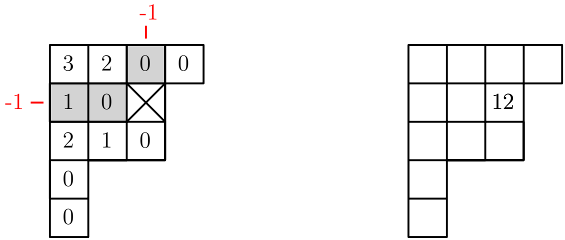
$L = [(2, 3)]$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



Remaining boxes : 11

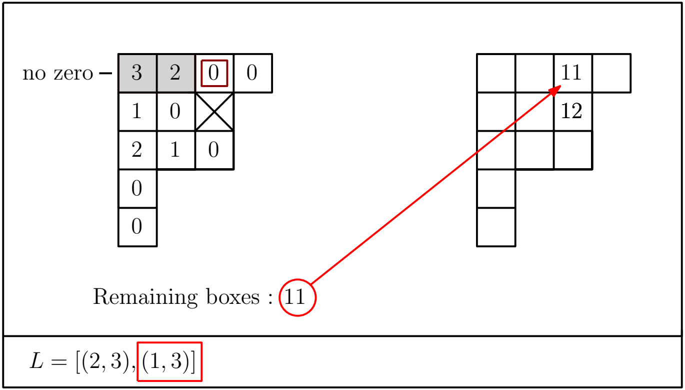
$L = [(2, 3)]$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

-1

Remaining boxes : 10

$L = [(2,3),(1,3)]$

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

Remaining boxes : 9

$L = [(2,3),(1,3),(4,1)]$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.

Remaining boxes : 8

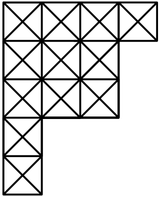
$L = [(2,3),(1,3),(4,1),(3,3)]$

The Filling algorithm (V, 2014)

Question

Can we find an algorithm in order to construct all the tableaux of a given type ?

- Yes ! The Filling algorithm.



7	6	11	8
5	3	12	
4	1	9	
10			
2			

Remaining boxes : 0

Filling sequence

$L = [(2,3),(1,3),(4,1),(3,3),(1,4),(1,1),\dots,(3,2)]$

The Filling algorithm

Definition

Set $\mathcal{T}$ a type.

- A sequence L which come from the algorithm is called a **filling sequence** of $\mathcal{T}$.
- We define $\mathrm{Fil}(\mathcal{T}) = \{\text{Filling sequences of } \mathcal{T}\}$.

The Filling algorithm

Definition

Set $\mathcal{T}$ a type.

- A sequence L which come from the algorithm is called a **filling sequence** of $\mathcal{T}$.
- We define $\mathrm{Fil}(\mathcal{T}) = \{\text{Filling sequences of } \mathcal{T}\}$.
- To each filling sequence $L = [c_1, c_2, \ldots, c_n]$ we associate the tableau $T_L = (t_c)_{c \in \lambda}$ such that $t_{c_k} = n + 1 - k$.

The Filling algorithm

Definition

Set $\mathcal{T}$ a type.

- A sequence L which come from the algorithm is called a **filling sequence** of $\mathcal{T}$.
- We define $\mathrm{Fil}(\mathcal{T}) = \{\text{Filling sequences of } \mathcal{T}\}$.
- To each filling sequence $L = [c_1, c_2, \ldots, c_n]$ we associate the tableau $T_L = (t_c)_{c\in\lambda}$ such that $t_{c_k} = n + 1 - k$.

Theorem (V., 2014)

- For any filling sequence L of $\mathcal{T}$, $T_L \in \mathrm{Tab}(\mathcal{T})$.

The Filling algorithm

Definition

Set $\mathcal{T}$ a type.

- A sequence L which come from the algorithm is called a **filling sequence** of $\mathcal{T}$.
- We define $\mathrm{Fil}(\mathcal{T}) = \{\text{Filling sequences of } \mathcal{T}\}$.
- To each filling sequence $L = [c_1, c_2, \ldots, c_n]$ we associate the tableau $T_L = (t_c)_{c \in \lambda}$ such that $t_{c_k} = n + 1 - k$.

Theorem (V., 2014)

- For any filling sequence L of $\mathcal{T}$, $T_L \in \mathrm{Tab}(\mathcal{T})$.
- The application $L \to T_L$ is a bijection.

$$\mathrm{Fil}(\mathcal{T}) \xleftrightarrow{\;1:1\;} \mathrm{Tab}(\mathcal{T})$$

$$L \longmapsto T_L$$

Reformulation of Edelman-Greene's theorem using types

Motivation

In the sequence, we will show how some types are connected to the theory of reduced decompositions in the symmetric group.

Reformulation of Edelman-Greene's theorem using types

Motivation

In the sequence, we will show how some types are connected to the theory of reduced decompositions in the symmetric group.

Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of Edelman-Greene's theorem using types

Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of Edelman-Greene's theorem using types

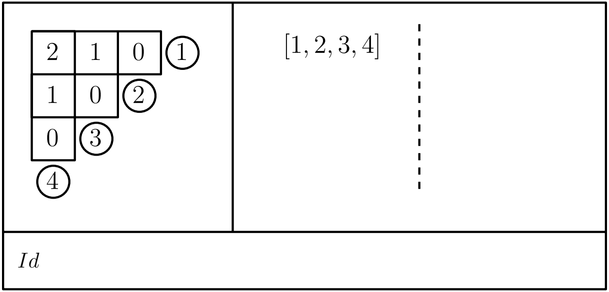
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

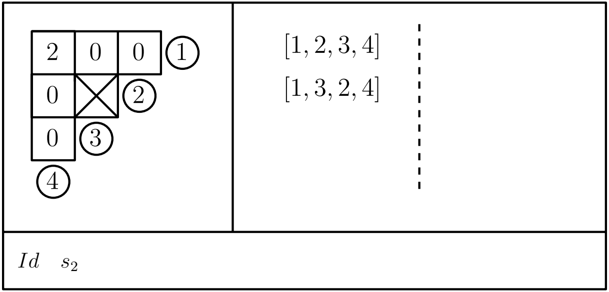
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

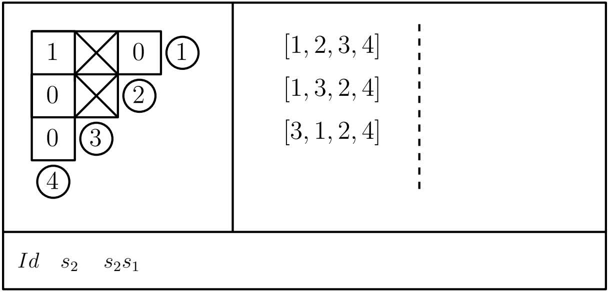
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

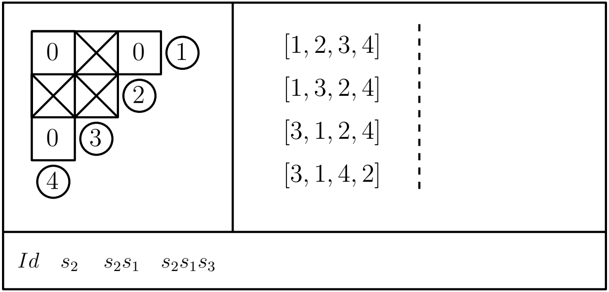
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

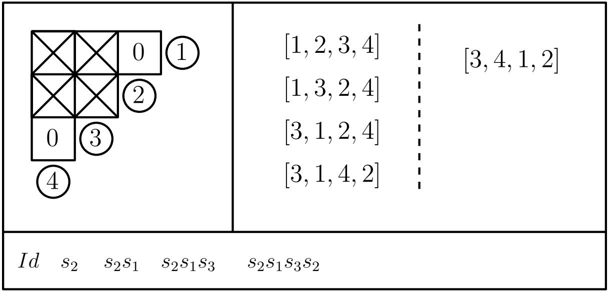
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

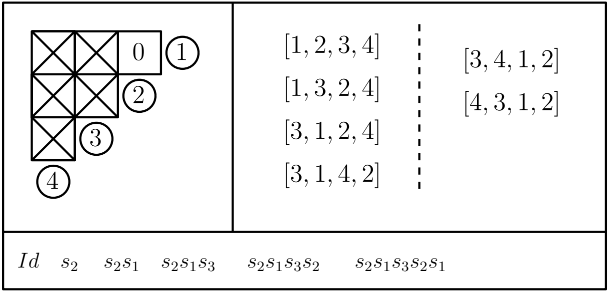
Set $\omega_0 = [n, n-1, \ldots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \ldots, 1)$.

Theorem (Edelman-Greene, 1987)

There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



Reformulation of Edelman-Greene's theorem using types

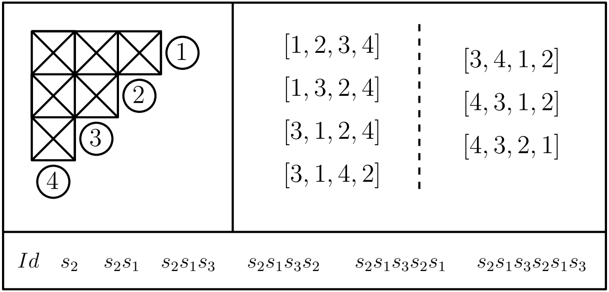
Set $\omega_0 = [n, n-1, \dots, 1] \in S_n$ and $\lambda_n = (n-1, n-2, \dots, 1)$.

Theorem (Edelman-Greene, 1987)

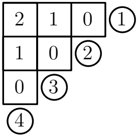
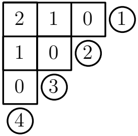
There exists a bijection between $\mathrm{Tab}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.

Reformulation of the theorem

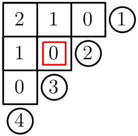
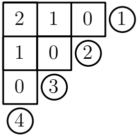
There exists a bijection between $\mathrm{Fil}(\mathcal{B}_{\lambda_n})$ and $\mathrm{Red}(\omega_0)$.



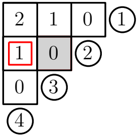
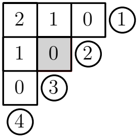
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3, 1, 4, 2]$	$\text{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4)\ ,\ (1,3)\ \}$	
	$[1, 2, 3, 4]$	$\text{Red}(\sigma)$ - - - - - - - - - -
Id		
	$[1, 2, 3, 4]$	
Id		

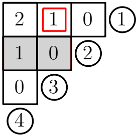
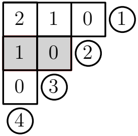
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\text{Inv}(\sigma) = \{ (2,3), \ (2,4) \ , \ (1,3) \ \}$	
	$[1,2,3,4]$	$\text{Red}(\sigma)$ ----------
Id		
	$[1,2,3,4]$	
Id		

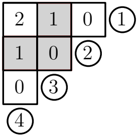
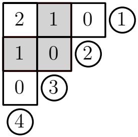
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\mathrm{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4),\ (1,3)\ \}$	
	$[1,2,3,4]$	$\mathrm{Red}(\sigma)$ - - - - - - - - - -
Id		
	$[1,2,3,4]$	
Id		

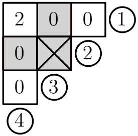
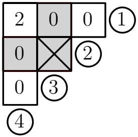
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\mathrm{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4)\ ,\ (1,3)\ \}$	
	$[1,2,3,4]$	$\mathrm{Red}(\sigma)$ ----------
Id		
	$[1,2,3,4]$	
Id		

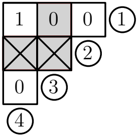
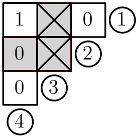
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\mathrm{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4)\ ,\ (1,3)\ \}$	
	$[1,2,3,4]$	$\mathrm{Red}(\sigma)$ - - - - - - - - - -
Id		
	$[1,2,3,4]$	
Id		

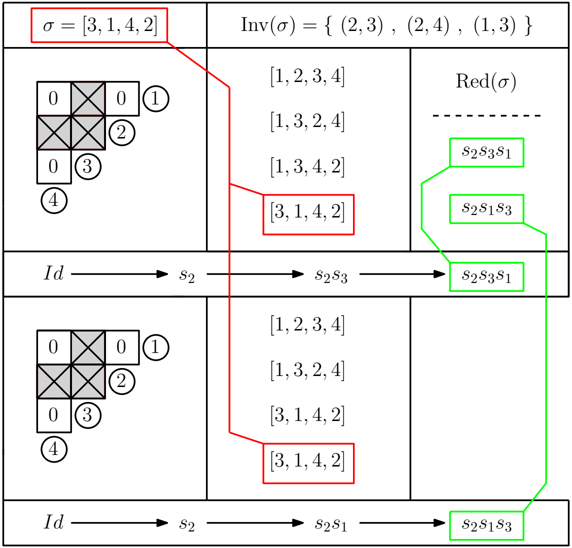
How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\mathrm{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4)\ ,\ (1,3)\ \}$	
	$[1,2,3,4]$ $[1,3,2,4]$	$\mathrm{Red}(\sigma)$ ----------
$Id \longrightarrow s_2$		
	$[1,2,3,4]$ $[1,3,2,4]$	
$Id \longrightarrow s_2$		

How to obtain all the reduced decompositions of any permutation with the Filling algorithm

$\sigma = [3,1,4,2]$	$\mathrm{Inv}(\sigma) = \{\ (2,3)\ ,\ (2,4)\ ,\ (1,3)\ \}$	
	$[1,2,3,4]$ $[1,3,2,4]$ $[1,3,4,2]$	$\mathrm{Red}(\sigma)$ ----------
$Id \longrightarrow s_2 \longrightarrow s_2 s_3$		
	$[1,2,3,4]$ $[1,3,2,4]$ $[3,1,4,2]$	
$Id \longrightarrow s_2 \longrightarrow s_2 s_1$		

How to obtain all the reduced decompositions of any permutation with the Filling algorithm



Subtype of $\mathcal{B}_{\lambda_n}$

Definition

Let $\mathcal{A}$ be a diagram contained in $\mathcal{B}_{\lambda_n}$. We call $\mathcal{A}$ a *sub-type* of $\mathcal{B}_{\lambda_n}$ if and only if by using the filling algorithm we can fill it with crosses without putting any cross outside $\mathcal{A}$. The set of all the subtypes is denoted $\mathrm{Sub}(\mathcal{B}_{\lambda_n})$.

Subtype of $\mathcal{B}_{\lambda_n}$

Definition

Let $\mathcal{A}$ be a diagram contained in $\mathcal{B}_{\lambda_n}$. We call $\mathcal{A}$ a *sub-type* of $\mathcal{B}_{\lambda_n}$ if and only if by using the filling algorithm we can fill it with crosses without putting any cross outside $\mathcal{A}$. The set of all the subtypes is denoted $\mathrm{Sub}(\mathcal{B}_{\lambda_n})$.

Theorem (V, 2014)

Set $\phi : \sigma \to \mathrm{Inv}(\sigma)$ (seen as a subset of boxes of $\mathcal{B}_{\lambda_n}$). Then we have the following situation.

$$\begin{array}{ccc}
S_n & \overset{\phi}{\underset{1:1}{\longleftrightarrow}} & \mathrm{Sub}(\mathcal{B}_{\lambda_n}) \\
\sigma & & \phi(\sigma) \\
\vdots\downarrow & & \vdots\downarrow \\
\mathrm{Red}(\sigma) & \underset{1:1}{\longleftrightarrow} & \mathrm{Fil}(\phi(\sigma))
\end{array}$$

Link with balanced tableaux ?

Definition

Set $\sigma \in S_n$. The permutation σ is called vexillary if and only if σ is 2143-avoiding.

Link with balanced tableaux ?

Definition

Set $\sigma \in S_n$. The permutation σ is called vexillary if and only if σ is 2143-avoiding.

Theorem (Stanley, 1984)

If σ is vexillary, then there exists a partition $\lambda(\sigma)$ of the integer $\ell(\sigma)$ such that $\mathrm{red}(\sigma) = f^{\lambda(\sigma)}$.

Link with balanced tableaux ?

Definition

Set $\sigma \in S_n$. The permutation σ is called vexillary if and only if σ is 2143-avoiding.

Theorem (Stanley, 1984)

If σ is vexillary, then there exists a partition $\lambda(\sigma)$ of the integer $\ell(\sigma)$ such that $\mathrm{red}(\sigma) = f^{\lambda(\sigma)}$.

Definition

We denote

$$\mathrm{Vex}(\lambda) = \{\sigma \mid \sigma \text{ vexillary and } \lambda(\sigma) = \lambda\}$$

(It is an infinite set, the permutations are taken in ALL symmetric groups)

Link with balanced tableaux ?

Theorem (V, 2014)

Using ϕ and one more combinatorial tool, we can explicitly construct an application Ψ from $\mathrm{Vex}(\lambda)$ to $\mathrm{Typ}(\lambda)$ (the set of the types of shape λ) such that:

Link with balanced tableaux ?

Theorem (V, 2014)

Using ϕ and one more combinatorial tool, we can explicitly construct an application Ψ from $\mathrm{Vex}(\lambda)$ to $\mathrm{Typ}(\lambda)$ (the set of the types of shape λ) such that:

- for all $\sigma \in \mathrm{Vex}(\lambda)$, $|\mathrm{Tab}(\Psi(\sigma))| = \mathrm{red}(\sigma) = f^{\lambda}$.

Link with balanced tableaux ?

Theorem (V, 2014)

Using ϕ and one more combinatorial tool, we can explicitly construct an application Ψ from $\mathrm{Vex}(\lambda)$ to $\mathrm{Typ}(\lambda)$ (the set of the types of shape λ) such that:

- for all $\sigma \in \mathrm{Vex}(\lambda)$, $|\mathrm{Tab}(\Psi(\sigma))| = \mathrm{red}(\sigma) = f^{\lambda}$.
- We can construct σ_λ such that $\Psi(\sigma_\lambda) = \mathcal{B}_\lambda$ (recall: $\mathcal{B}_\lambda$ is the type associated with balanced tableaux of shape λ).

Link with balanced tableaux ?

Theorem (V, 2014)

Using ϕ and one more combinatorial tool, we can explicitly construct an application Ψ from $\mathrm{Vex}(\lambda)$ to $\mathrm{Typ}(\lambda)$ (the set of the types of shape λ) such that:

- for all $\sigma \in \mathrm{Vex}(\lambda)$, $|\mathrm{Tab}(\Psi(\sigma))| = \mathrm{red}(\sigma) = f^{\lambda}$.
- We can construct σ_λ such that $\Psi(\sigma_\lambda) = \mathcal{B}_\lambda$ (recall: $\mathcal{B}_\lambda$ is the type associated with balanced tableaux of shape λ).

Corollary

We have that $|\mathrm{Bal}(\lambda)| = f^{\lambda} = |\mathrm{SYT}(\lambda)|$.

Thank you for your attention.