Sandpiles and pmaj for $\nabla^k e_n$

joint with: Michele D'Adderio

Alessio Sgubin

Department of Mathematics
University of Pisa

q,t-Combinatorics in Cortona - June 13, 2025

Index

Section 1

The sorted sandpile model

Basic definitions: graphs

Graphs: finite, undirected, connected, without loops.

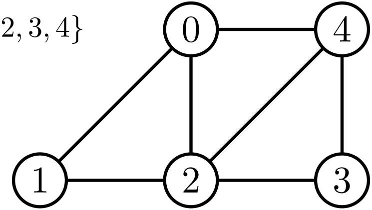
Let $G = (V, E)$ be a graph, with

→ $V = \{0, 1, \ldots, n\}$ **vertex set**,

→ E **edge set**.

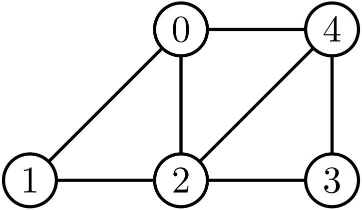
For example:

$V = \{0, 1, 2, 3, 4\}$



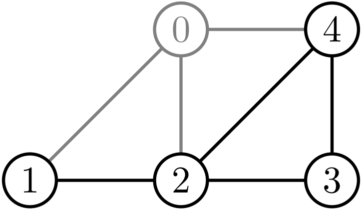
Basic definitions: configurations

Fix a vertex called **sink**, in our case let it be $0 \in V$.



Basic definitions: configurations

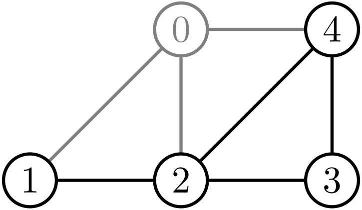
Fix a vertex called **sink**, in our case let it be $0 \in V$.



Basic definitions: configurations

Fix a vertex called **sink**, in our case let it be $0 \in V$.

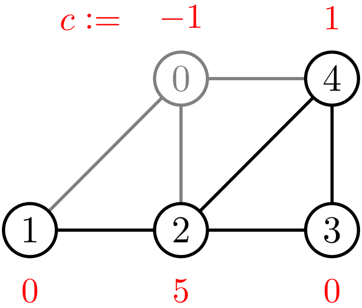
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.



Basic definitions: configurations

Fix a vertex called **sink**, in our case let it be $0 \in V$.

A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

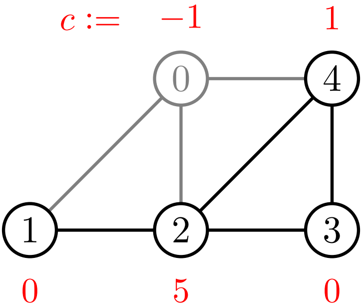


Basic definitions: configurations

Fix a vertex called **sink**, in our case let it be $0 \in V$.
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$



Basic definitions: configurations

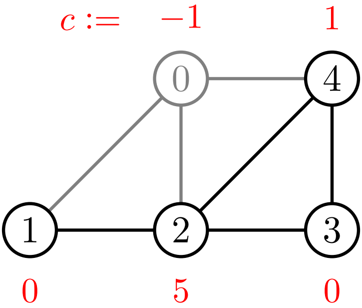
Fix a vertex called **sink**, in our case let it be $0 \in V$.
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$

For example:

$$c' := \phi_4(c)$$



Basic definitions: configurations

Fix a vertex called **sink**, in our case let it be $0 \in V$.

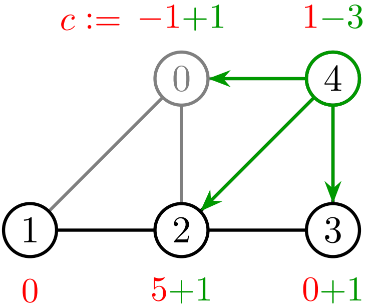
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$

For example:

$$c' := \phi_4(c)$$



Basic definitions: configurations

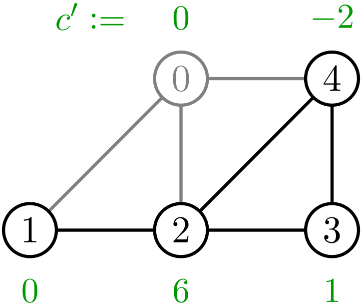
Fix a vertex called **sink**, in our case let it be $0 \in V$.
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$

For example:

$$c' := \phi_4(c)$$



Basic definitions: configurations

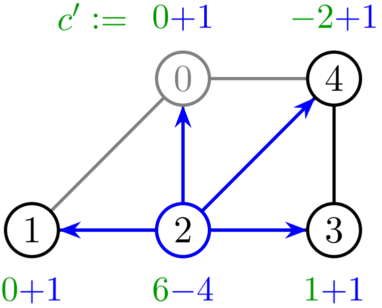
Fix a vertex called **sink**, in our case let it be $0 \in V$.
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$

For example:

$$c'' := \phi_2(c')$$



Basic definitions: configurations

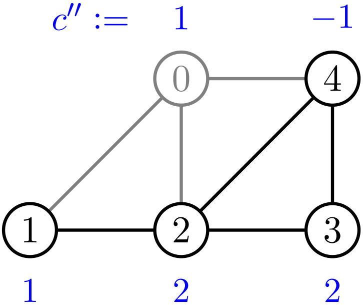
Fix a vertex called **sink**, in our case let it be $0 \in V$.
A **configuration** is an element $c \in \mathbb{Z}^{|V|}$.

The **toppling** of vertex $v \in V$ is defined by

$$\phi_v(c) := c - \sum_{wv \in E} (w - v).$$

For example:

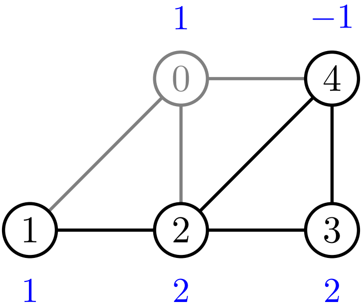
$$c'' := \phi_2(c')$$



Basic definitions: sandpiles

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.

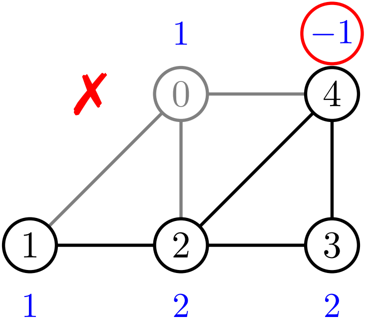
A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.



Basic definitions: sandpiles

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.

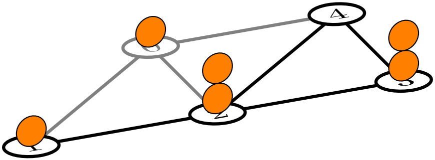
A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.

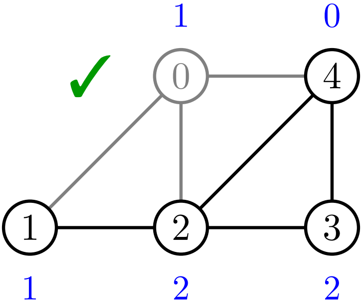


Basic definitions: sandpiles

Let $V = \{0, 1, \dots, n\}$ and let 0 be the sink.

A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.



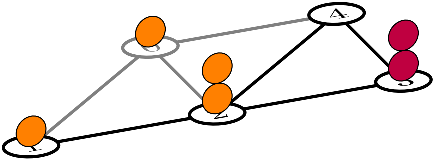


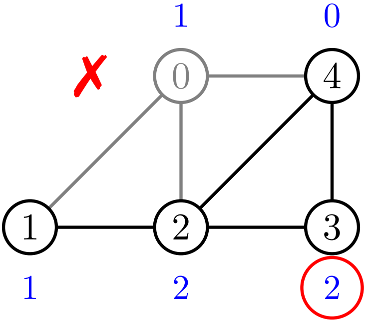
Basic definitions: sandpiles

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.

A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.

A configuration is **stable** if $c(v) < \deg_G(v)$ for all $v \geq 1$.



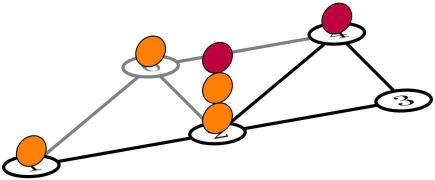


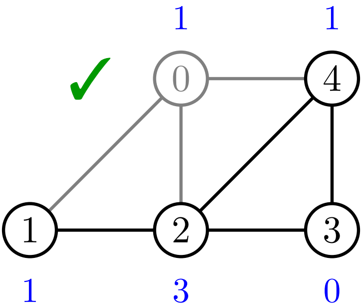
Basic definitions: sandpiles

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.

A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.

A configuration is **stable** if $c(v) < \deg_G(v)$ for all $v \geq 1$.





Basic definitions: sandpiles

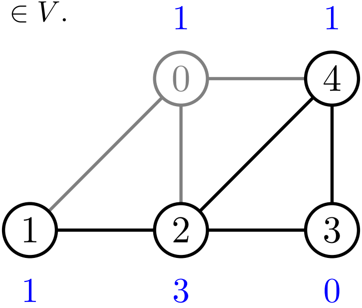
Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.

A configuration is **non-negative** if $c(v) \geq 0$ for all $v \geq 1$.

A configuration is **stable** if $c(v) < \deg_G(v)$ for all $v \geq 1$.

We ignore the values on the sink $0 \in V$.

Thus, consider $c \in \mathbb{Z}^{V \setminus \{0\}}$.



Basic definitions: recurrent configurations

Let $V = \{0, 1, \dots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that

$$c \rightsquigarrow \phi_0(c) \rightsquigarrow \phi_{\sigma(1)}\phi_0(c) \rightsquigarrow \dots \rightsquigarrow \phi_{\sigma(n)} \dots \phi_{\sigma(1)}\phi_0(c)$$

are all non-negative configurations.

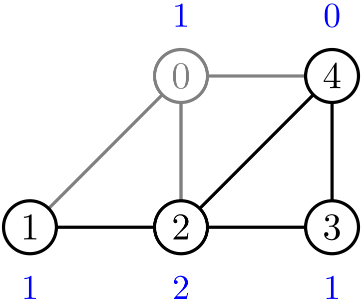
Basic definitions: recurrent configurations

Let $V = \{0, 1, \dots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma =$

Configuration: c



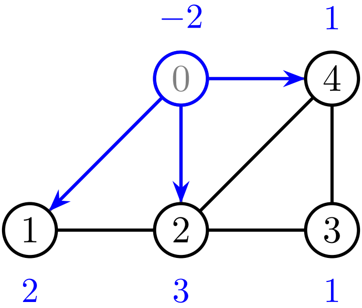
Basic definitions: recurrent configurations

Let $V = \{0, 1, \dots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma =$

Configuration: $\phi_0(c)$



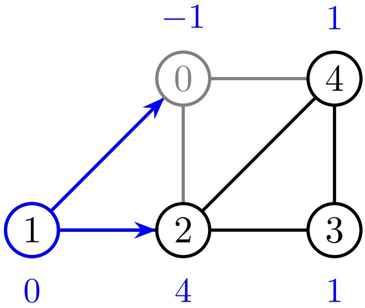
Basic definitions: recurrent configurations

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma = 1$

Configuration: $\phi_1\phi_0(c)$



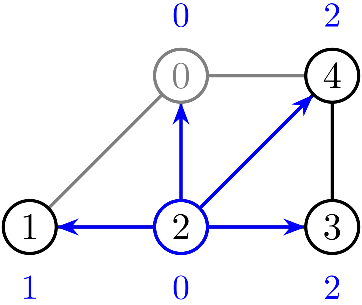
Basic definitions: recurrent configurations

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V \setminus \{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma = 1\ 2$

Configuration: $\phi_2\phi_1\phi_0(c)$



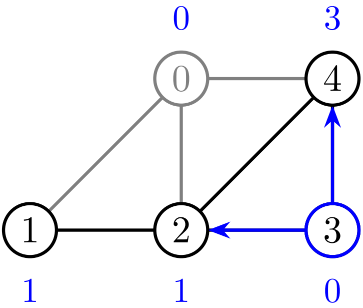
Basic definitions: recurrent configurations

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma = 1\ 2\ 3$

Configuration: $\phi_3\phi_2\phi_1\phi_0(c)$



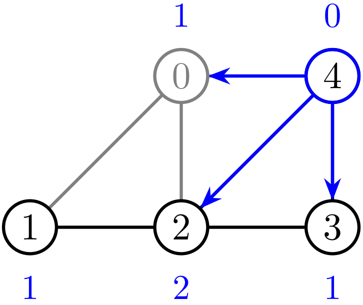
Basic definitions: recurrent configurations

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V \setminus \{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma = 1\ 2\ 3\ 4$

Configuration: $\phi_4\phi_3\phi_2\phi_1\phi_0(c)$



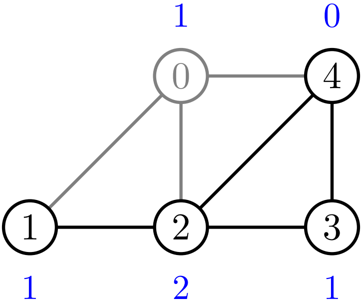
Basic definitions: recurrent configurations

Let $V = \{0, 1, \dots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V\setminus\{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

For example:

$\sigma = 1\ 2\ 3\ 4$ $\quad$ c is recurrent ✓

Configuration: $\phi_4\phi_3\phi_2\phi_1\phi_0(c)$



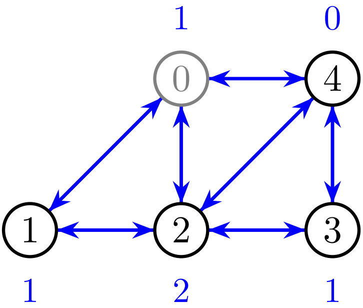
Basic definitions: recurrent configurations

Let $V = \{0, 1, \ldots, n\}$ and let 0 be the sink.
A configuration $c \in \mathbb{Z}^{V \setminus \{0\}}$ is **recurrent** if it is stable and there exist $\sigma \in \mathfrak{S}_n$ such that c, $\phi_0(c)$, $\phi_{\sigma(1)}\phi_0(c)$, $\phi_{\sigma(2)}\phi_{\sigma(1)}\phi_0(c)$, ... are all non-negative configurations.

Observe that

$$\phi_{\sigma(n)} \circ \cdots \circ \phi_{\sigma(1)} \circ \phi_0(c) = c.$$

We say $c \in \mathrm{Rec}(G)$.

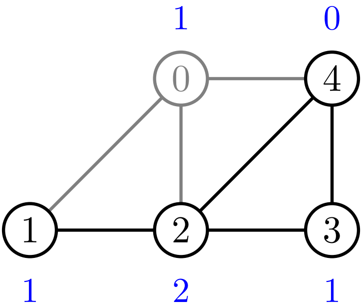


Basic definitions: level statistic

Let G be a graph on $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{Rec}(G)$.

We define the **level** of c as:

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to } 0\}.$$



Basic definitions: level statistic

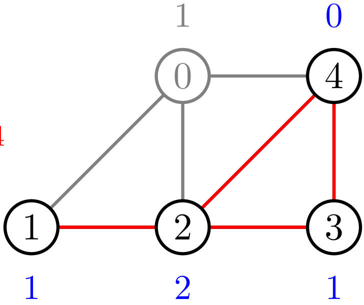
Let G be a graph on $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{Rec}(G)$.

We define the **level** of c as:

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to } 0\}.$$

In the example:

$$\mathrm{level}(c) = (1 + 2 + 1 + 0) - 4$$
$$= 0$$



Sandpile model: some motivation

- Mathematical Physics: self-organized criticality (Bak-Tang-Wiesenfeld, 1987)
- Geometry: divisors on tropical curves
- Probability: limit configurations for Markov chains on sandpiles
- Combinatorics:

Theorem

The number of recurrent configurations on a graph G equals the number of its spanning trees.

Sandpile model: some motivation

- Mathematical Physics: self-organized criticality (Bak-Tang-Wiesenfeld, 1987)
- Geometry: divisors on tropical curves
- Probability: limit configurations for Markov chains on sandpiles
- Combinatorics:

Theorem

The number of recurrent configurations on a graph G equals the number of its spanning trees.

Sandpile model: some motivation

- Mathematical Physics: self-organized criticality (Bak-Tang-Wiesenfeld, 1987)
- Geometry: divisors on tropical curves
- Probability: limit configurations for Markov chains on sandpiles
- Combinatorics:

Theorem

The number of recurrent configurations on a graph G equals the number of its spanning trees.

Sandpile model: some motivation

- Mathematical Physics: self-organized criticality (Bak-Tang-Wiesenfeld, 1987)
- Geometry: divisors on tropical curves
- Probability: limit configurations for Markov chains on sandpiles
- Combinatorics:

Theorem

The number of recurrent configurations on a graph G equals the number of its spanning trees.

A variation: sorted sandpiles

Consider a graph G on $V = \{0, 1, \ldots, n\}$ and fix sink 0.

Let $\Gamma < \mathrm{Aut}(G)$ be a subgroup of the stabilizer of 0 (the sink).

We define **sorted recurrent configurations** the elements of

$$\mathrm{SortRec}_\Gamma(G) := {}^{\mathrm{Rec}(G)}\!\big/_{\Gamma}.$$

Theorem

Let K_{n+1} be the complete graph on $V = \{0, 1, \ldots, n\}$ and consider $\Gamma = \mathfrak{S}_n$ the stabilizer of 0. Then:

$$|\mathrm{Rec}(K_{n+1})| = (n+1)^{n-1}$$

$$|\mathrm{SortRec}_\Gamma(K_{n+1})| = C_n := n^{th}\text{-Catalan number.}$$

A variation: sorted sandpiles

Consider a graph G on $V = \{0, 1, \dots, n\}$ and fix sink 0.

Let $\Gamma < \mathrm{Aut}(G)$ be a subgroup of the stabilizer of 0 (the sink).

We define **sorted recurrent configurations** the elements of

$$\mathrm{SortRec}_\Gamma(G) := {}^{\mathrm{Rec}(G)}\!\big/_{\Gamma}.$$

Theorem

Let K_{n+1} be the complete graph on $V = \{0, 1, \dots, n\}$ and consider $\Gamma = \mathfrak{S}_n$ the stabilizer of 0. Then:

$$|\mathrm{Rec}(K_{n+1})| = (n+1)^{n-1}$$

$$|\mathrm{SortRec}_\Gamma(K_{n+1})| = C_n := n^{th}\text{-Catalan number.}$$

A variation: sorted sandpiles

Consider a graph G on $V = \{0, 1, \dots, n\}$ and fix sink 0.
Let $\Gamma < \mathrm{Aut}(G)$ be a subgroup of the stabilizer of 0 (the sink).

We define **sorted recurrent configurations** the elements of

$$\mathrm{SortRec}_\Gamma(G) := {}^{\mathrm{Rec}(G)}\!\diagup_{\Gamma}.$$

Theorem

Let K_{n+1} be the complete graph on $V = \{0, 1, \dots, n\}$ and consider $\Gamma = \mathfrak{S}_n$ the stabilizer of 0. Then:

$$|\,\mathrm{Rec}(K_{n+1})| = (n+1)^{n-1}$$

$$|\,\mathrm{SortRec}_\Gamma(K_{n+1})| = C_n := n^{th}\text{-Catalan number.}$$

Section 2

...and the Shuffle Theorem

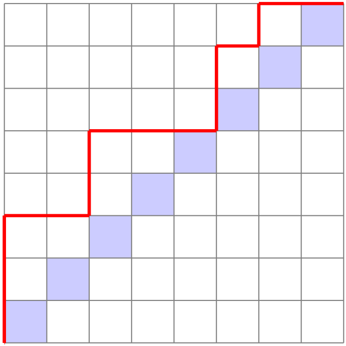
Catalan numbers: Dyck paths

Catalan numbers are counted by **Dyck paths**:

$$\mathrm{Dyck}(n) := \{\text{Dyck paths of size } n\}.$$

It follows that:

$$C_n = \sum_{P \in \mathrm{Dyck}(n)} 1.$$



q-Catalan numbers: Dyck paths

Catalan numbers are counted by **Dyck paths**:

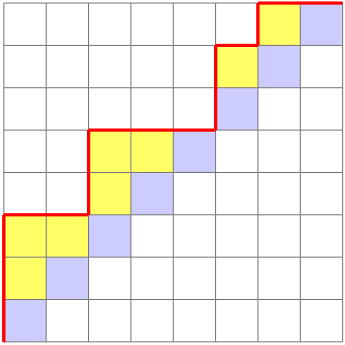
$$\mathrm{Dyck}(n) := \{\text{Dyck paths of size } n\}.$$

It follows that:

$$C_n(q) = \sum_{P \in \mathrm{Dyck}(n)} q^{\mathrm{area}(P)}$$

where in the example:

$$\mathrm{area}(P) = 8.$$



q,t-Catalan numbers: Dyck paths

Catalan numbers are counted by **Dyck paths**:

$$\mathrm{Dyck}(n) := \{\text{Dyck paths of size } n\}.$$

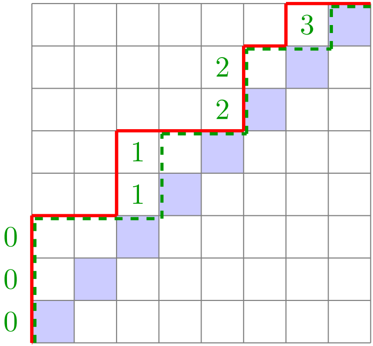
It follows that:

$$C_n(q,t) = \sum_{P \in \mathrm{Dyck}(n)} q^{\mathrm{area}(P)} t^{\mathrm{bounce}(P)}$$

where in the example:

$$\mathrm{area}(P) = 8$$

$$\mathrm{bounce}(P) = 9.$$



q,t-Catalan numbers: Dyck paths

Catalan numbers are counted by **Dyck paths**:

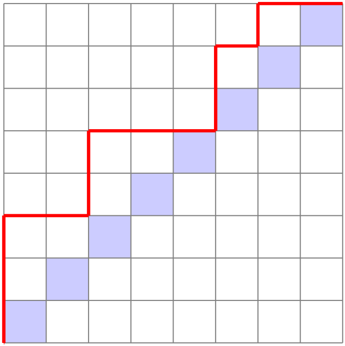
$$\mathrm{Dyck}(n) := \{\text{Dyck paths of size } n\}.$$

It follows that:

$$C_n(q,t) = \sum_{P \in \mathrm{Dyck}(n)} q^{\mathrm{area}(P)} t^{\mathrm{bounce}(P)}$$

where in the example:

$$\mathrm{area}(P) = 8$$
$$\mathrm{bounce}(P) = 9.$$



q,t-Catalan numbers: parking functions

Dyck paths are described by *some* **parking functions**:

$$\overline{\mathrm{PF}}_n\big((n);\varnothing\big) := \{n\text{-labelled Dyck paths with increasing labels}\}.$$

In particular for $\pi \in \overline{\mathrm{PF}}_n((n);\varnothing)$:

$$\operatorname{area}(P(\pi)) = \operatorname{area}(\pi)$$

$$\operatorname{bounce}(P(\pi)) = \boxed{\operatorname{pmaj}(\pi)}.$$

q,t-Catalan numbers: parking functions

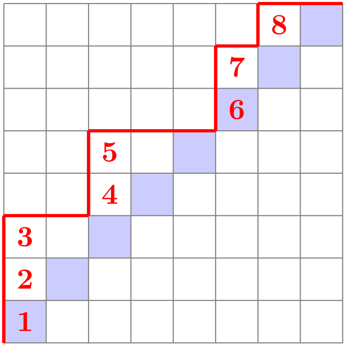
Dyck paths are described by *some* **parking functions**:

$$\overline{\mathrm{PF}}_n\big((n);\varnothing\big) := \{n\text{-labelled Dyck paths with increasing labels}\}.$$

In particular for $\pi \in \overline{\mathrm{PF}}_n((n);\varnothing)$:

$$\mathrm{area}(P(\pi)) = \mathrm{area}(\pi)$$

$$\mathrm{bounce}(P(\pi)) = \boxed{\mathrm{pmaj}(\pi)}.$$



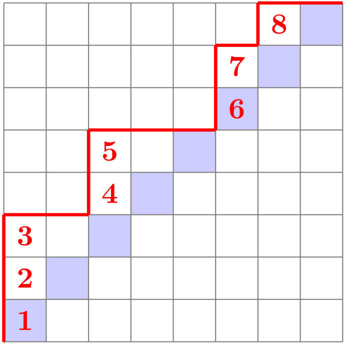
The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$



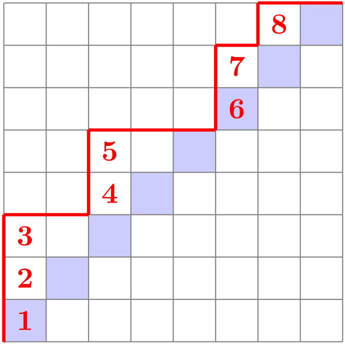
The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

For $m = 1, 2, \dots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$



$m = 0$	$B = \varnothing$
$\sigma_0 = 9$	$X = \varnothing$

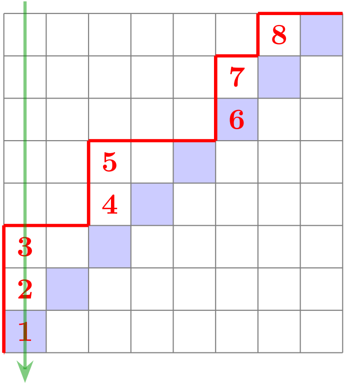
The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

For $m = 1, 2, \dots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$



$m = 1$ $\quad B = \{1, 2, 3\}$

$\sigma_0 = 9$ $\quad X = \varnothing$

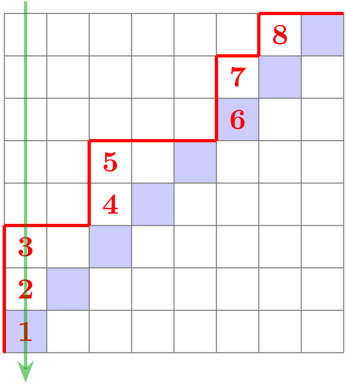
The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$



$m = 1$ $\qquad B = \{1, 2, 3\}$

$\sigma_0 = 9$ $\qquad X = \{1, 2, 3\}$

The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

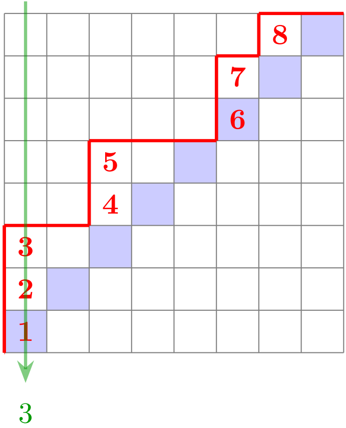
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 1$ $\quad$ $B = \{1, 2, \cancel{3}\}$

$\sigma_0 = 9$ $\quad$ $X = \{1, 2, 3\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

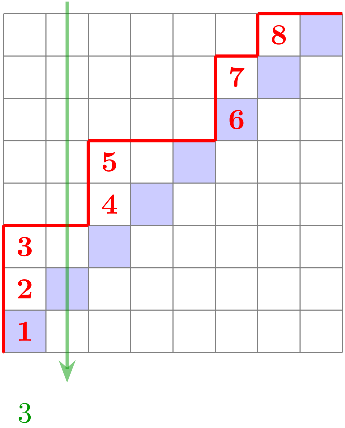
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 2$ $\quad B = \{1, 2\}$

$\sigma_1 = 3$ $\quad X = \{1, 2, 3\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

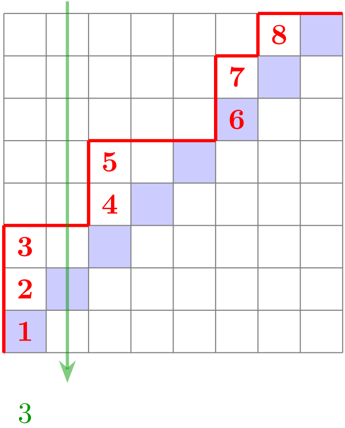
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 2$ $\quad B = \{1, 2\}$

$\sigma_1 = 3$ $\quad X = \{1, 2\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n+1$.

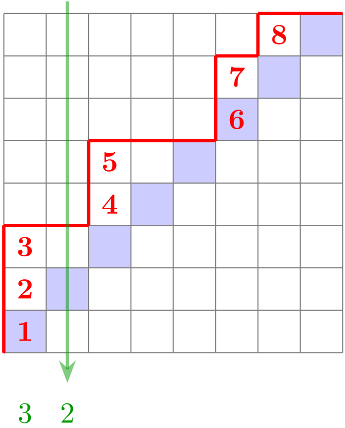
For $m = 1, 2, \dots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 2$ $\qquad B = \{1, \cancel{2}\}$

$\sigma_1 = 3$ $\qquad X = \{1, 2\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

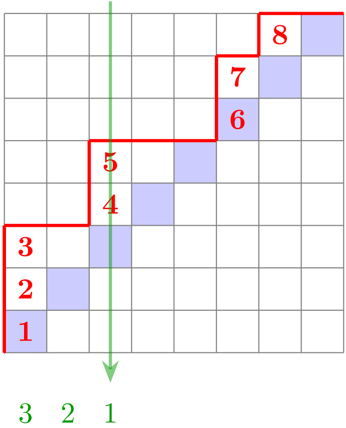
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 3 \qquad B = \{1, 4, 5\}$

$\sigma_2 = 2 \qquad X = \{1\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

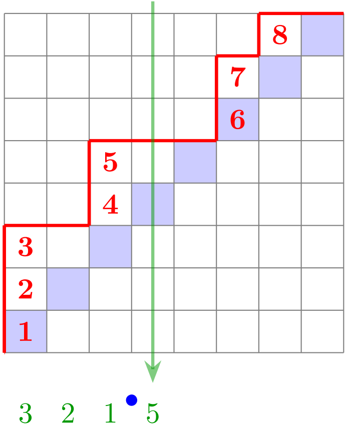
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 4$ $\quad B = \{4, 5\}$

$\sigma_3 = 1$ $\quad X = \varnothing$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

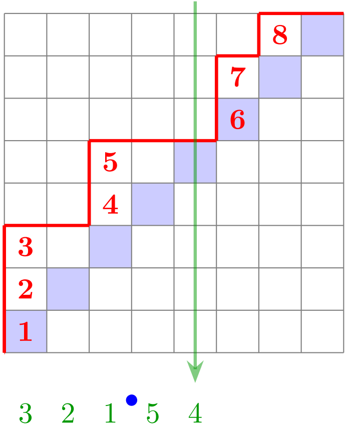
For $m = 1, 2, \dots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 5 \qquad B = \{4\}$

$\sigma_4 = 5 \qquad X = \{4\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

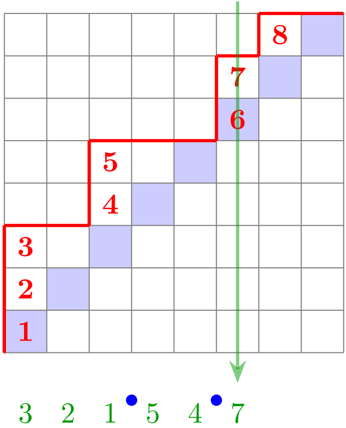
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 6$ $\quad B = \{6, 7\}$

$\sigma_5 = 4$ $\quad X = \varnothing$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n+1$.

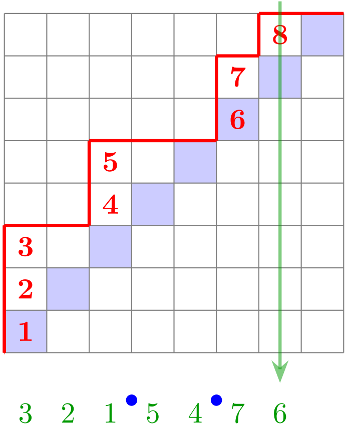
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 7$ $\quad B = \{6, 8\}$

$\sigma_6 = 7$ $\quad X = \{6\}$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

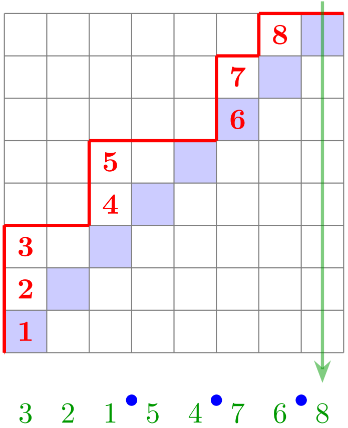
For $m = 1, 2, \dots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

$m = 8$ $\quad B = \{8\}$

$\sigma_7 = 6$ $\quad X = \varnothing$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

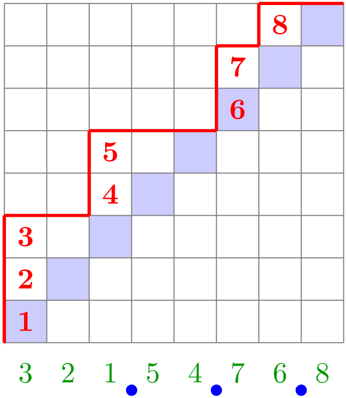
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

Then:

$$\operatorname{pmaj}(\pi) := \operatorname{maj}(\sigma_n \sigma_{n-1} \ldots \sigma_1)$$



The pmaj statistic

We compute pmaj using an algorithm. Let $B = \varnothing$, $\sigma_0 = n + 1$.

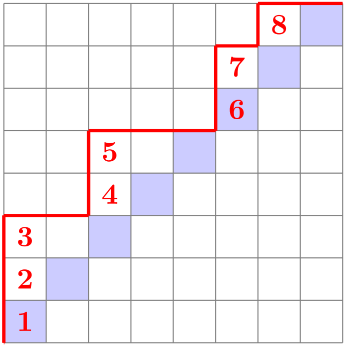
For $m = 1, 2, \ldots, n$:

- add labels of column m in B.
- let $X = \{i \in B \mid i < \sigma_{m-1}\}$.
- remove from B element

$$\sigma_m := \begin{cases} \max(X) & X \neq \varnothing \\ \max(B) & X = \varnothing. \end{cases}$$

Then:

$$\operatorname{pmaj}(\pi) := \operatorname{maj}(\sigma_n \sigma_{n-1} \ldots \sigma_1)$$



3 2 1 • 5 4 • 7 6 • 8

$\operatorname{pmaj}(\pi) = 9 =$ sum of 0 0 0 1 1 2 2 3

Some milestones: why parking functions?

- 1988: introduction of Macdonald polynomials
- ∼2000: Haglund and Haiman define bounce and dinv for Dyck paths
- 2003: HHLRU state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{dinv}, \mathrm{area})$
- 2004: Loehr-Remmel state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{area}, \mathrm{pmaj})$
- 2018: Carlsson-Mellit prove the Shuffle conjecture

For μ, ν compositions, $|\mu| + |\nu| = n$ we have:

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} = \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

Some milestones: why parking functions?

- 1988: introduction of Macdonald polynomials
- ∼2000: Haglund and Haiman define bounce and dinv for Dyck paths
- 2003: HHLRU state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{dinv}, \mathrm{area})$
- 2004: Loehr-Remmel state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{area}, \mathrm{pmaj})$
- 2018: Carlsson-Mellit prove the Shuffle conjecture

For μ, ν compositions, $|\mu| + |\nu| = n$ we have:

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} = \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

Some milestones: why parking functions?

- 1988: introduction of Macdonald polynomials
- ∼2000: Haglund and Haiman define bounce and dinv for Dyck paths
- 2003: HHLRU state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{dinv}, \mathrm{area})$
- 2004: Loehr-Remmel state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{area}, \mathrm{pmaj})$
- 2018: Carlsson-Mellit prove the Shuffle conjecture

For μ, ν compositions, $|\mu| + |\nu| = n$ we have:

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} = \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

Some milestones: why parking functions?

- 1988: introduction of Macdonald polynomials
- ∼2000: Haglund and Haiman define bounce and dinv for Dyck paths
- 2003: HHLRU state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{dinv}, \mathrm{area})$
- 2004: Loehr-Remmel state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{area}, \mathrm{pmaj})$
- 2018: Carlsson-Mellit prove the Shuffle conjecture

For μ, ν compositions, $|\mu| + |\nu| = n$ we have:

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} = \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

Some milestones: why parking functions?

- 1988: introduction of Macdonald polynomials
- ∼2000: Haglund and Haiman define bounce and dinv for Dyck paths
- 2003: HHLRU state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{dinv}, \mathrm{area})$
- 2004: Loehr-Remmel state Shuffle conjecture for ∇e_n with parking functions and bistatistic $(\mathrm{area}, \mathrm{pmaj})$
- 2018: Carlsson-Mellit prove the Shuffle conjecture

For μ, ν compositions, $|\mu| + |\nu| = n$ we have:

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} = \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

The connection

- 2014: ADDHL the case $\mu = \varnothing$ and $\nu = (k, n-k)$.
- 2023: DDL the case $\mu = (k)$ and $\nu = (n-k)$.
- 2024: DDILLV the general case:

Theorem - D'Adderio, Dukes, Iraci, Lazar, Le Borgne, Vander Wyngaerd (2025)

Consider $|\mu| + |\nu| = n$. Then

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}$$

The connection

- 2014: ADDHL the case $\mu = \varnothing$ and $\nu = (k, n-k)$.
- 2023: DDL the case $\mu = (k)$ and $\nu = (n-k)$.
- 2024: DDILLV the general case:

Theorem - D'Adderio, Dukes, Iraci, Lazar, Le Borgne, Vander Wyngaerd (2025)

Consider $|\mu| + |\nu| = n$. Then

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}$$

The connection

- 2014: ADDHL the case $\mu = \varnothing$ and $\nu = (k, n-k)$.
- 2023: DDL the case $\mu = (k)$ and $\nu = (n-k)$.
- 2024: DDILLV the general case:

Theorem - D'Adderio, Dukes, Iraci, Lazar, Le Borgne, Vander Wyngaerd (2025)

Consider $|\mu| + |\nu| = n$. *Then*

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}$$

The proof idea: the identities

Show the last identity:

$$\langle \nabla e_n, e_\mu h_\nu \rangle \overset{\text{[CM18]}}{=} \sum_{\pi \in \mathrm{PF}_n(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)}$$

$$\overset{\text{[LR04]}}{=} \sum_{\pi \in \overline{\mathrm{PF}}_n(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)}$$

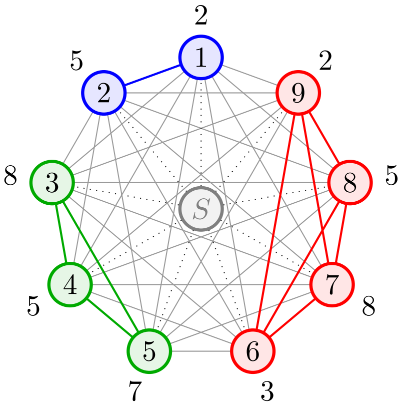
$$\overset{\text{[DDI}^+\text{25]}}{=} \sum_{c \in \mathrm{SortRec}(G(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}$$

via a bijection between:

$$\overline{\mathrm{PF}}_n(\mu;\nu) \text{ with } (\mathrm{area}, \mathrm{pmaj}) \quad \longleftrightarrow \quad \mathrm{SortRec}(G(\mu;\nu)) \text{ with } (\mathrm{level}, \mathrm{delay})$$

The proof idea: the bijection

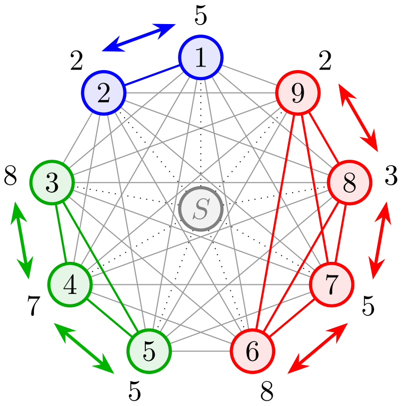
Consider $\mu = (4, 3, 2)$ and $\nu = \varnothing$.



The proof idea: the bijection

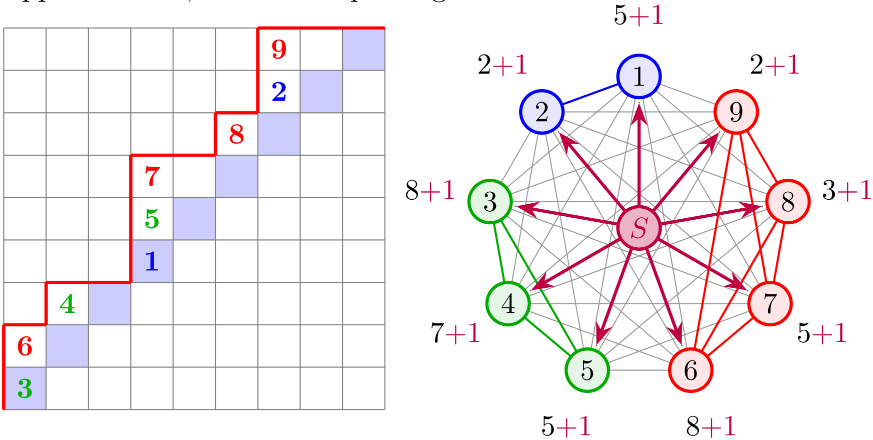
Consider $\mu = (4, 3, 2)$ and $\nu = \varnothing$.

Re-order entries in each subset, decreasingly.



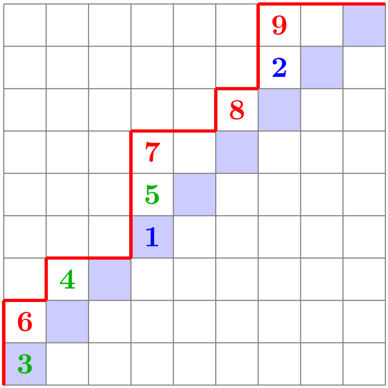
The proof idea: the bijection

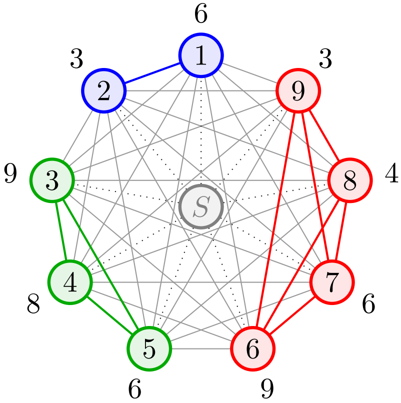
Topple the sink, associate a parking function.



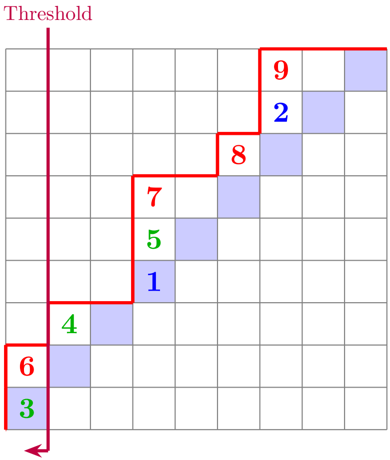
The proof idea: the bijection

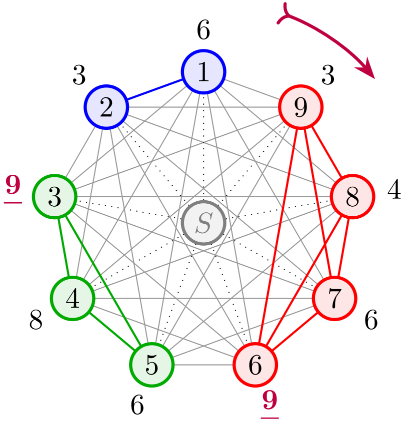
Dyck path condition $\equiv$ Recurrent configuration condition



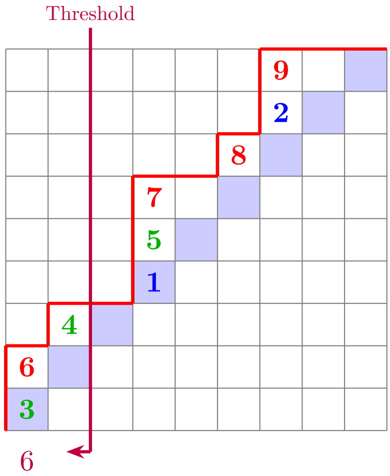


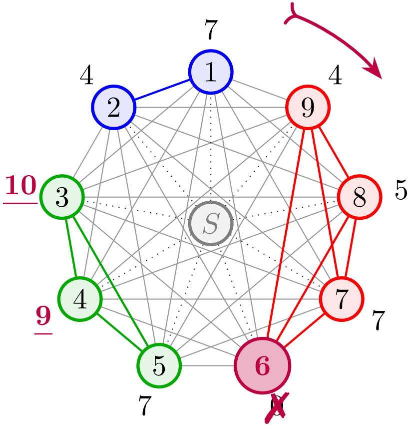
The proof idea: the delay



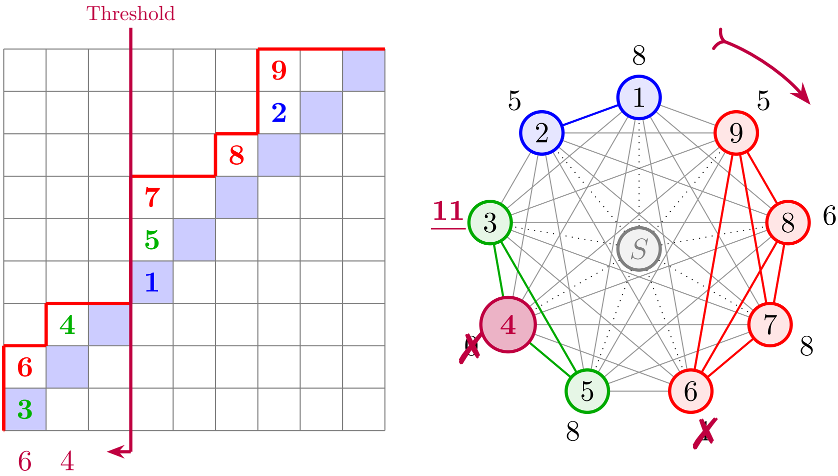


The proof idea: the delay

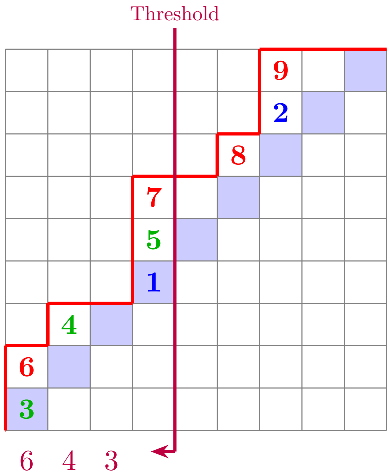


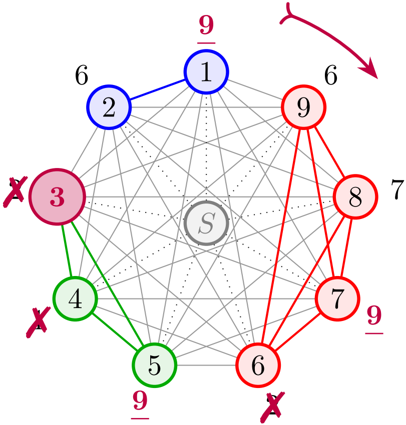


The proof idea: the delay

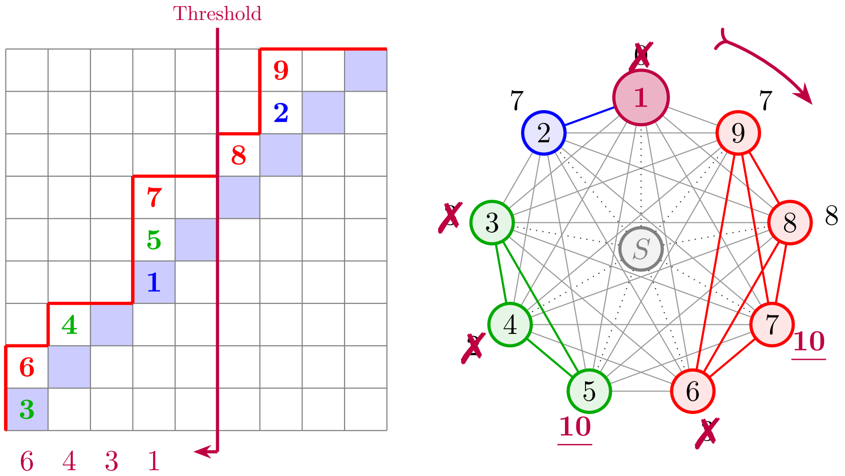


The proof idea: the delay

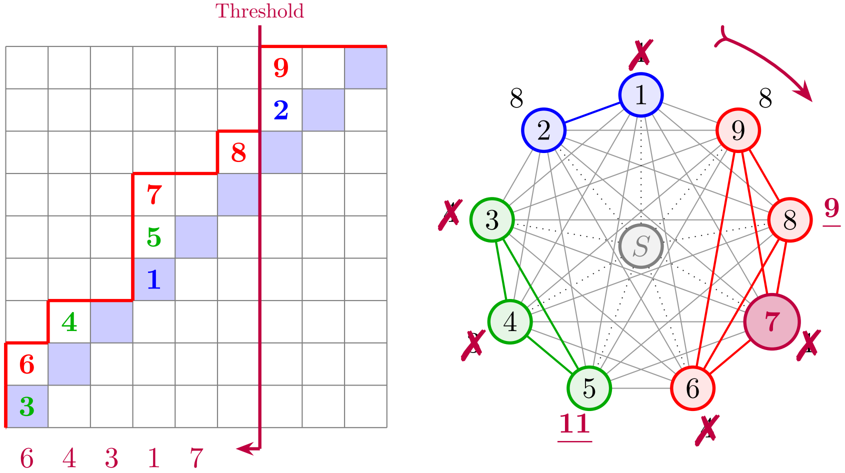




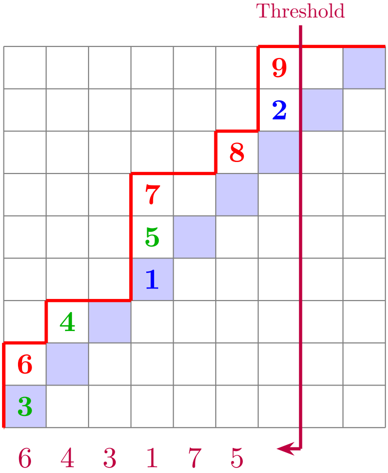
The proof idea: the delay

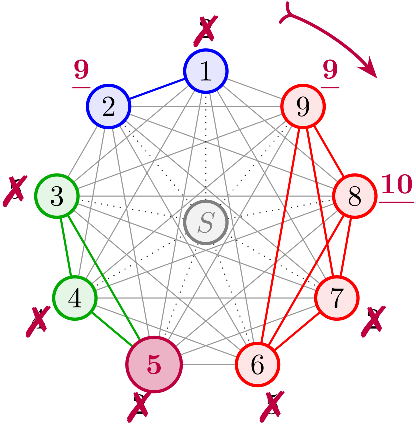


The proof idea: the delay

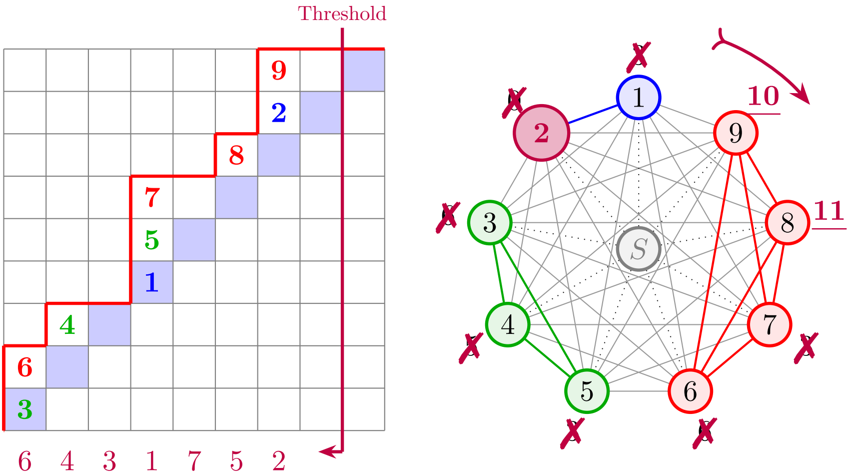


The proof idea: the delay

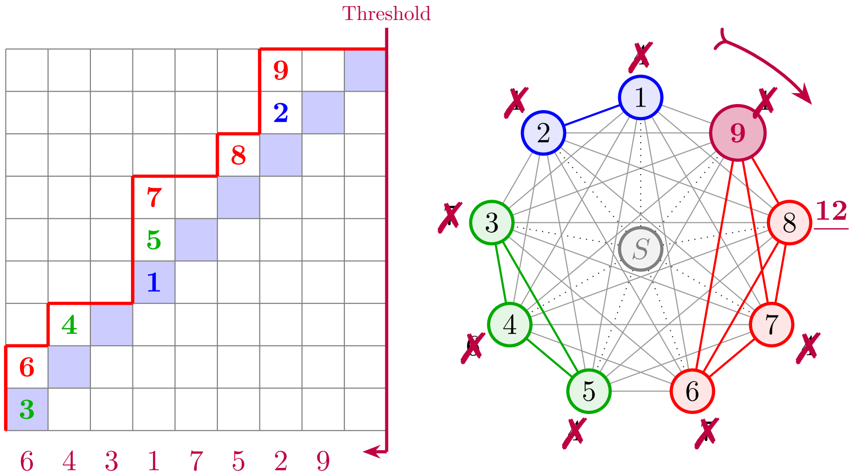




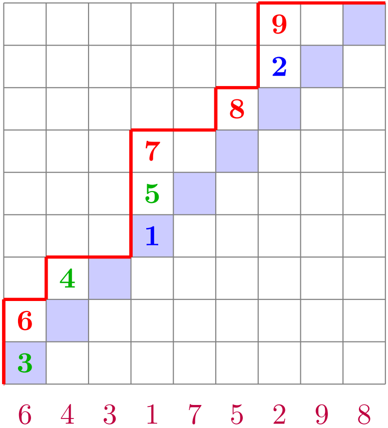
The proof idea: the delay

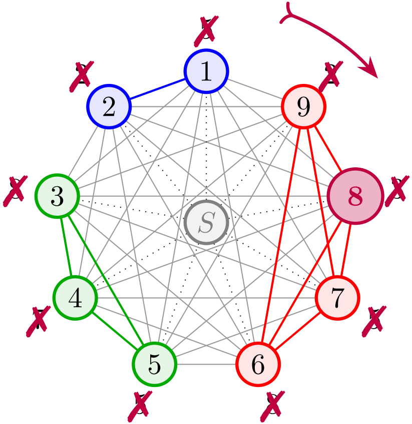


The proof idea: the delay



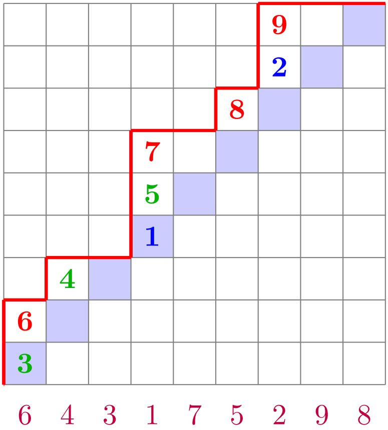
The proof idea: the delay

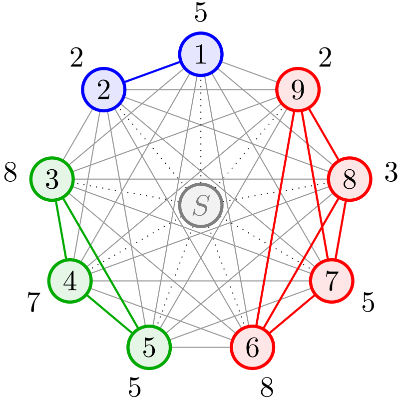




The proof idea: the delay

pmaj contribute of label λ = # of loops before toppling label λ





Section 3

A generalization of delay and pmaj

Our study case

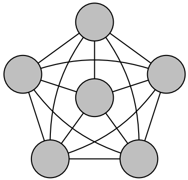
Goal: study the sandpile model on other families of graphs.

Starting from the family of graphs from [DDI+25], we allow edges with **multeplicities**:

Our study case

Goal: study the sandpile model on other families of graphs.

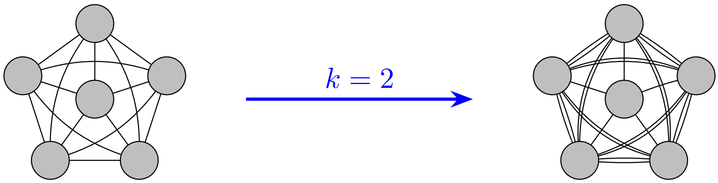
Starting from the family of graphs from [DDI+25], we allow edges with **multeplicities**:



Our study case

Goal: study the sandpile model on other families of graphs.

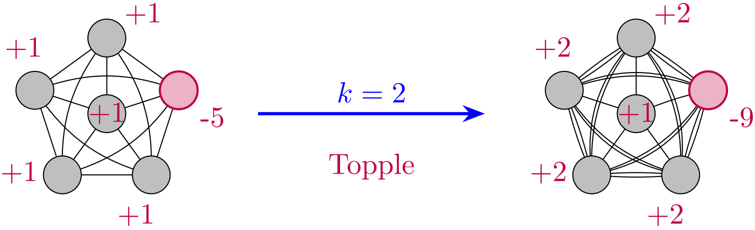
Starting from the family of graphs from [DDI+25], we allow edges with **multeplicities**:



Our study case

Goal: study the sandpile model on other families of graphs.

Starting from the family of graphs from [DDI+25], we allow edges with **multeplicities**:



New sorted sandpile statistics

The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "slow" release occurs

New sorted sandpile statistics

The definition of (sorted) recurrent configurations is the same.
For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.

New sorted sandpile statistics

The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \dots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.

New sorted sandpile statistics

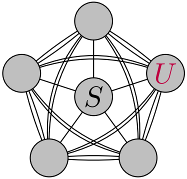
The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.



New sorted sandpile statistics

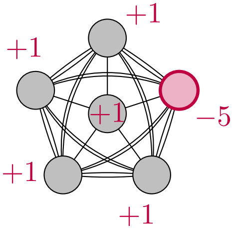
The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.



New sorted sandpile statistics

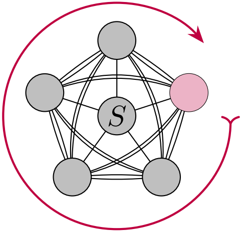
The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \dots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.



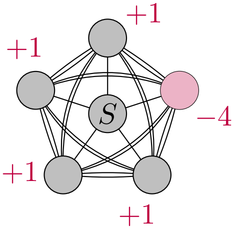
New sorted sandpile statistics

The definition of (sorted) recurrent configurations is the same.
For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.



New sorted sandpile statistics

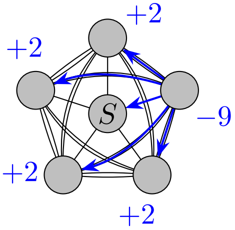
The definition of (sorted) recurrent configurations is the same.

For $V = \{0, 1, \dots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.



New sorted sandpile statistics

The definition of (sorted) recurrent configurations is the same.

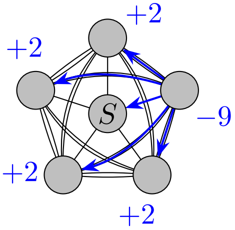
For $V = \{0, 1, \ldots, n\}$ and $c \in \mathrm{SortRec}(G)$ the **statistics** are:

- level: same definition as before

$$\mathrm{level}(c) := \sum_{i=1}^{n} c(i) - \#\{\text{edges non-incident to the sink}\}.$$

- delay: the toppling algorithm must be changed, when a vertex is unstable a "**slow release**" starts.

  → Implementation on [Sgu24]!



Interpretation of $\nabla^k e_n$

Theorem - D'Adderio, Dukes, Iraci, Lazar, Le Borgne, Vander Wyngaerd (2025)

Let $|\mu| + |\nu| = n$. *Then*

$$\langle \nabla e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}.$$

Interpretation of $\nabla^k e_n$

Conjecture - D'Adderio, S. (In preparation)

Let $|\mu|+|\nu|=n$ *and* $k\geq 1$. *Then*

$$\langle \nabla^{k} e_n, e_\mu h_\nu \rangle = \sum_{c\in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}.$$

Interpretation of $\nabla^k e_n$

Conjecture - D'Adderio, S. (In preparation)

Let $|\mu| + |\nu| = n$ *and* $k \geq 1$. *Then*

$$\langle \nabla^k e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}.$$

The idea is to follow the same proof of [DDI+25]:

- Mellit proves an interpretation of $\langle \nabla^k e_n, e_\mu h_\nu \rangle$ by $n \times nk$ parking functions with $(\mathrm{dinv}, \mathrm{area})$.
- No known statistic pmaj for $nk \times n$ parking functions.

Interpretation of $\nabla^k e_n$

Conjecture - D'Adderio, S. (In preparation)

Let $|\mu| + |\nu| = n$ *and* $k \geq 1$. *Then*

$$\langle \nabla^{k} e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}.$$

The idea is to follow the same proof of [DDI+25]:

- Mellit proves an interpretation of $\langle \nabla^k e_n, e_\mu h_\nu \rangle$ by $n \times nk$ parking functions with $(\mathrm{dinv}, \mathrm{area})$.
- No known statistic pmaj for $nk \times n$ parking functions.

Interpretation of $\nabla^k e_n$

Conjecture - D'Adderio, S. (In preparation)

Let $|\mu| + |\nu| = n$ *and* $k \geq 1$. *Then*

$$\langle \nabla^k e_n, e_\mu h_\nu \rangle = \sum_{c \in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}.$$

The idea is to follow the same proof of [DDI+25]:

- Mellit proves an interpretation of $\langle \nabla^k e_n, e_\mu h_\nu \rangle$ by $n \times nk$ parking functions with $(\mathrm{dinv}, \mathrm{area})$.
- No known statistic pmaj for $nk \times n$ parking functions.

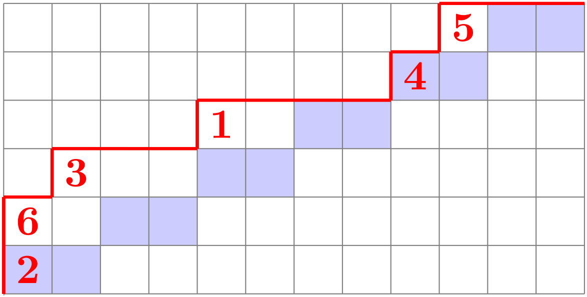
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \varnothing$



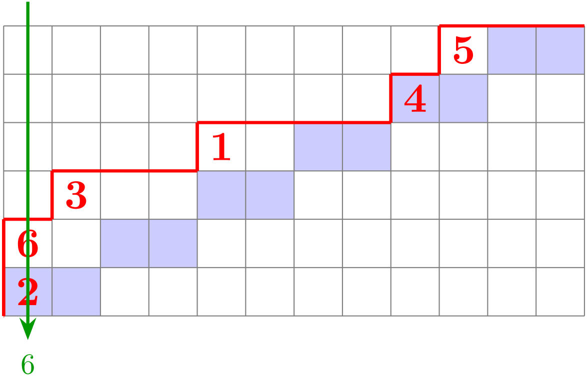
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{2, 2, 6, 6\}$



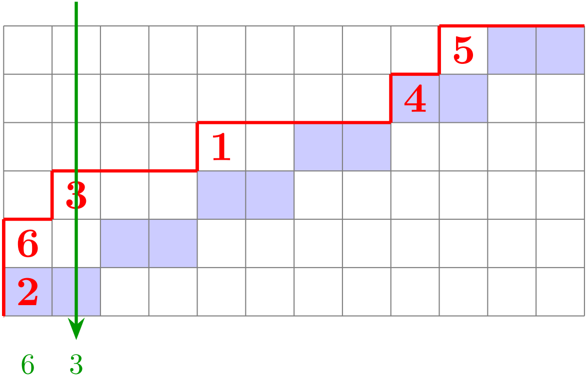
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{2, 2, 3, 3, 6\}$



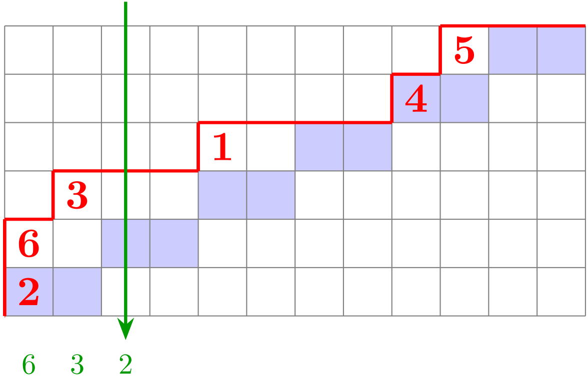
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{2, 2, 3, 6\}$



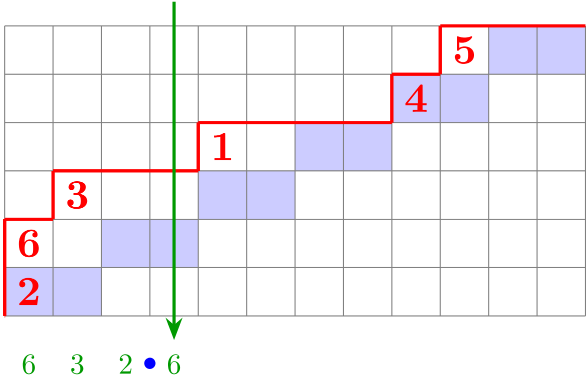
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{2, 3, 6\}$



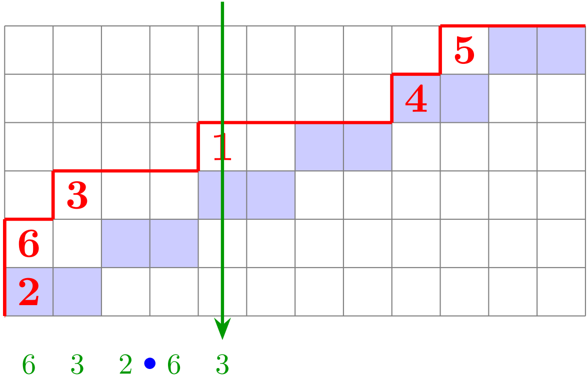
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{1, 1, 2, 3\}$



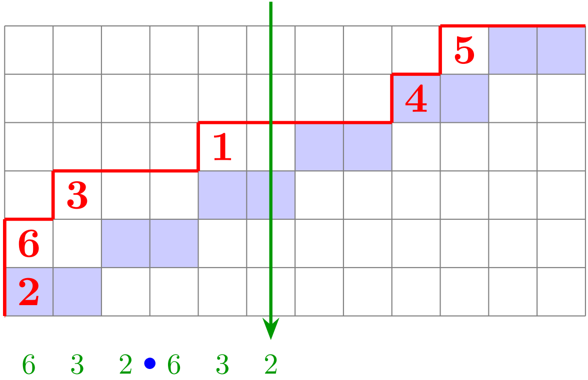
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{1, 1, 2\}$



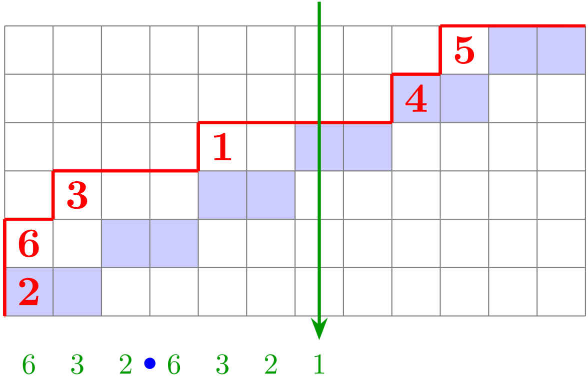
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{1, 1\}$



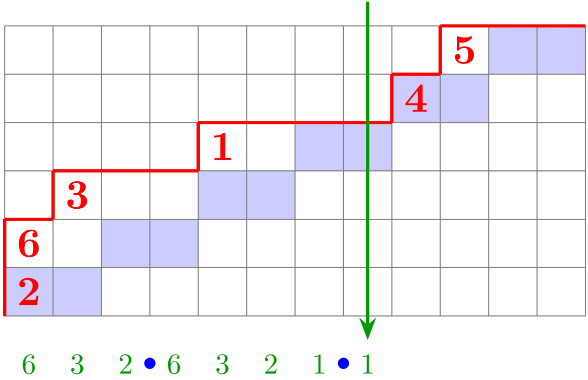
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{1\}$



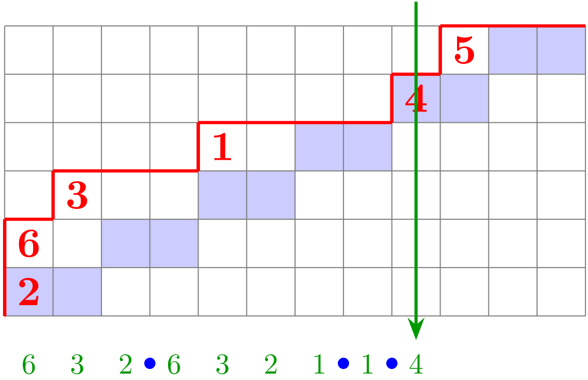
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{4, 4\}$



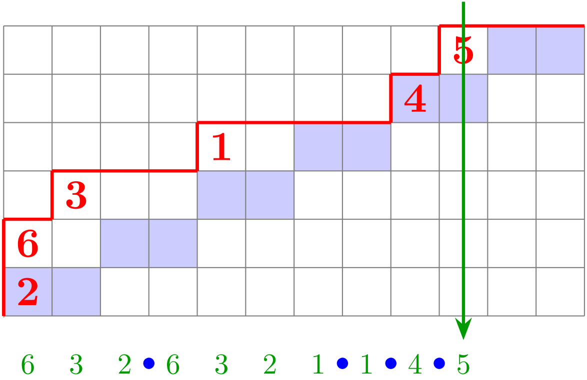
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{4, 5, 5\}$



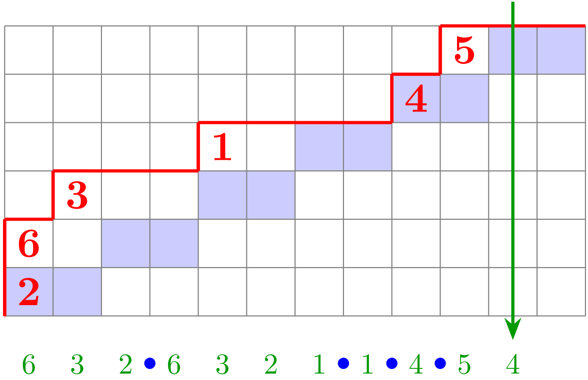
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{4, 5\}$



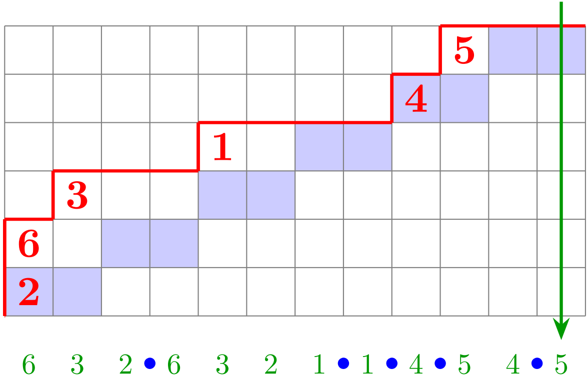
New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

$B = \{5\}$

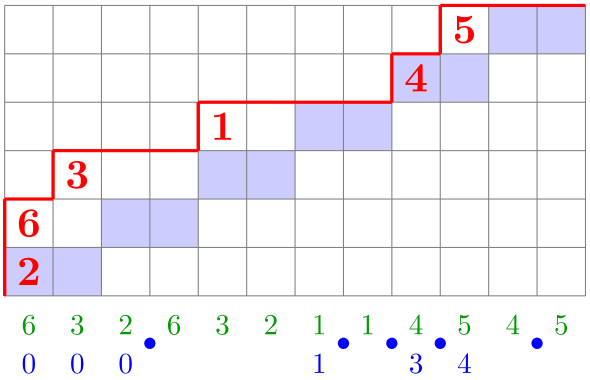


New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$

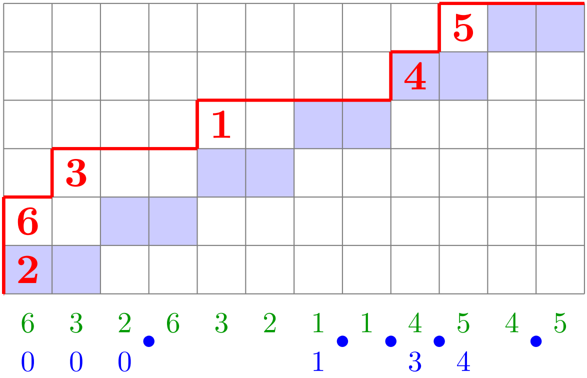


New pmaj statistic

The new delay statistic on sandpiles gives an idea for a pmaj on $n \times nk$ parking functions.

$n = 6$

$k = 2$



$\mathrm{pmaj}(\pi) = 8$

The proof

Given the new statistics, we show that:

$$\begin{aligned}\langle \nabla^k e_n, e_\mu h_\nu \rangle &\overset{\text{Mellit}}{=} \sum_{\pi \in \mathrm{PF}_{n,nk}(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} \\ &\overset{\text{New}}{=} \sum_{\pi \in \overline{\mathrm{PF}}_{n,nk}(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)} \\ &\overset{\text{New}}{=} \sum_{c \in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}\end{aligned}$$

via the following bijections:

$$\begin{array}{ccccc} \mathrm{PF}_{n,nk}(\mu;\nu) & \longleftrightarrow & \overline{\mathrm{PF}}_{n,nk}(\mu;\nu) & \longleftrightarrow & \mathrm{SortRec}(G_k(\mu;\nu)) \\ (\mathrm{dinv}, \mathrm{area}) & & (\mathrm{area}, \mathrm{pmaj}) & & (\mathrm{level}, \mathrm{delay}). \end{array}$$

The proof

Given the new statistics, we show that:

$$\begin{aligned}
\langle \nabla^k e_n, e_\mu h_\nu \rangle &\overset{\text{Mellit}}{=} \sum_{\pi \in \mathrm{PF}_{n,nk}(\mu;\nu)} q^{\mathrm{dinv}(\pi)} t^{\mathrm{area}(\pi)} \\
&\overset{\text{New}}{=} \sum_{\pi \in \overline{\mathrm{PF}}_{n,nk}(\mu;\nu)} q^{\mathrm{area}(\pi)} t^{\mathrm{pmaj}(\pi)} \\
&\overset{\text{New}}{=} \sum_{c \in \mathrm{SortRec}(G_k(\mu;\nu))} q^{\mathrm{level}(c)} t^{\mathrm{delay}(c)}
\end{aligned}$$

via the following bijections:

$$\begin{array}{ccccc}
\mathrm{PF}_{n,nk}(\mu;\nu) & \longleftrightarrow & \overline{\mathrm{PF}}_{n,nk}(\mu;\nu) & \longleftrightarrow & \mathrm{SortRec}(G_k(\mu;\nu)) \\
(\mathrm{dinv}, \mathrm{area}) & & (\mathrm{area}, \mathrm{pmaj}) & & (\mathrm{level}, \mathrm{delay}).
\end{array}$$

Bibliography

[DDI+25] Michele D'Adderio, Mark Dukes, Alessandro Iraci, Alexander Lazar, Yvan Le Borgne, and Anna Vanden Wyngaerd, *Shuffle Theorems and Sandpiles*, Comm. Math. Phys. **406** (2025), no. 4, Paper No. 83. MR 4881031

[DS25] Michele D'Adderio and Alessio Sgubin, *Sorted sandpiles and a new pmaj statistic for* $\nabla^k e_n$, In preparation (2025).

[LR04] Nicholas A. Loehr and Jeffrey B. Remmel, *Conjectured combinatorial models for the Hilbert series of generalized diagonal harmonics modules*, Electron. J. Combin. **11** (2004), no. 1, Research Paper 68, 64. MR 2097334

[Mel21] Anton Mellit, *Toric braids and (m,n)-parking functions*, Duke Mathematical Journal **170** (2021), no. 18.

[Sgu24] Alessio Sgubin, *Sorted sandpile model for SageMath*, `https://github.com/AlessioSgubin/sandpiles2`, 2024.