Graph Theory and Discrete Geometry

Jeremy L. Martin
Department of Mathematics
University of Kansas

C&PE Graduate Seminar
November 9, 2010

Graphs

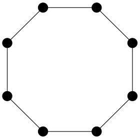
A **graph** is a pair $G = (V, E)$, where

- V is a finite set of *vertices*;
- E is a finite set of *edges*;
- Each edge connects two vertices called its *endpoints*.

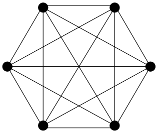
Graphs

A **graph** is a pair $G = (V, E)$, where

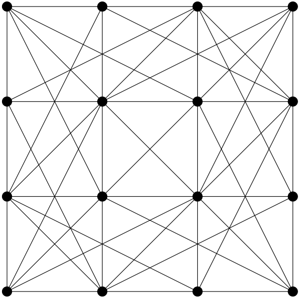
- V is a finite set of *vertices*;
- E is a finite set of *edges*;
- Each edge connects two vertices called its *endpoints*.



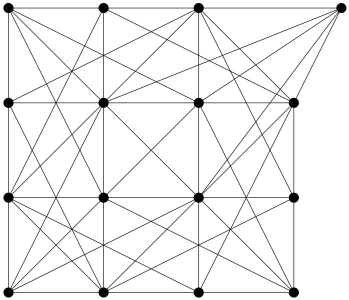
C_8



K_6



G



G

Why study graphs?

- Real-world applications
  - Combinatorial optimization (routing, scheduling. . .)
  - Computer science (data structures, sorting, searching. . .)
  - Biology (evolutionary descent. . .)
  - Chemistry (molecular structure. . .)
  - Engineering (roads, electrical circuits, rigidity. . .)
  - Network models (the Internet, Facebook!. . .)
- Pure mathematics
  - Combinatorics (ubiquitous!)
  - Discrete dynamical systems (chip-firing game. . .)
  - Abstract algebra. . .)
  - Discrete geometry (polytopes, sphere packing. . .)

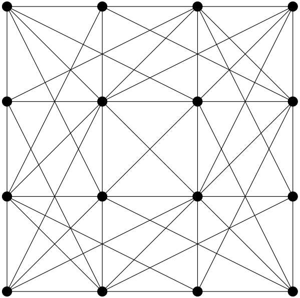
Spanning Trees

Definition A **spanning tree of G** is a set of edges T (or a subgraph (V, T)) such that:

1. (V, T) is **connected**: every pair of vertices is joined by a path
2. (V, T) is **acyclic**: there are no cycles
3. $|T| = |V| - 1$.

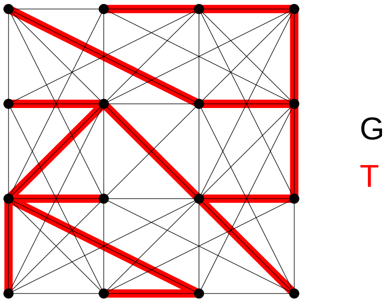
Any two of these conditions together imply the third.

Spanning Trees

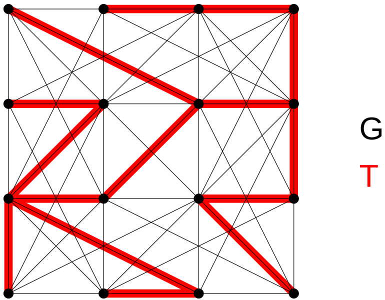


G

Spanning Trees



Spanning Trees



Counting Spanning Trees

Definition $\tau(G)$ = number of spanning trees of G

(Think of $\tau(G)$ as a rough measure of the complexity of G.)

- $\tau(\text{tree}) = 1$ (trivial)
- $\tau(C_n) = n$ (almost trivial)
- $\tau(K_n) = n^{n-2}$ (Cayley's formula; highly nontrivial!)
- Many other enumeration formulas for "nice" graphs

Deletion and Contraction

Let $e \in E(G)$.

Deletion and Contraction

Let $e \in E(G)$.

- *Deletion* $G - e$: Remove e

Deletion and Contraction

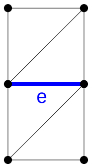
Let $e \in E(G)$.

- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point

Deletion and Contraction

Let $e \in E(G)$.

- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point

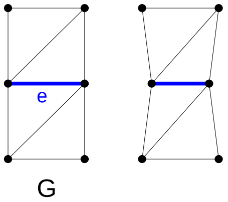


G

Deletion and Contraction

Let $e \in E(G)$.

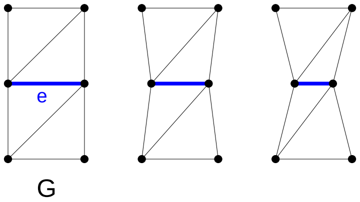
- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point



Deletion and Contraction

Let $e \in E(G)$.

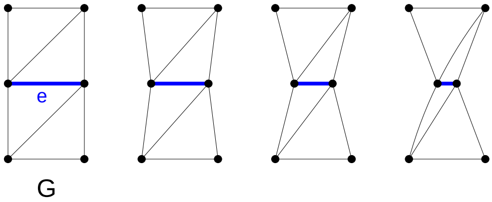
- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point



Deletion and Contraction

Let $e \in E(G)$.

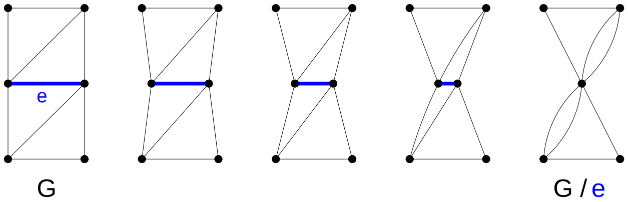
- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point



Deletion and Contraction

Let $e \in E(G)$.

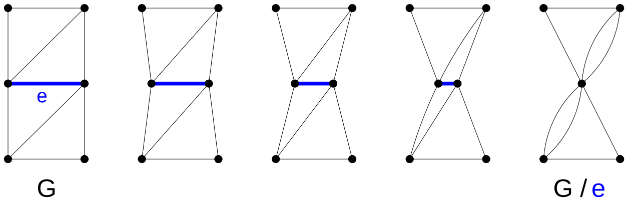
- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point



Deletion and Contraction

Let $e \in E(G)$.

- *Deletion* $G - e$: Remove e
- *Contraction* G/e: Shrink e to a point



Theorem $\tau(G) = \tau(G - e) + \tau(G/e)$.

Deletion and Contraction

Theorem $\tau(G) = \tau(G - e) + \tau(G/e).$

Deletion and Contraction

Theorem $\tau(G) = \tau(G - e) + \tau(G/e)$.

- Therefore, we can calculate $\tau(G)$ recursively...

Deletion and Contraction

Theorem $\tau(G) = \tau(G - e) + \tau(G/e)$.

- Therefore, we can calculate $\tau(G)$ recursively…
- …but this is computationally inefficient (since it requires $2^{|E|}$ steps)…

Deletion and Contraction

Theorem $\tau(G) = \tau(G - e) + \tau(G/e)$.

- Therefore, we can calculate $\tau(G)$ recursively...
- ...but this is computationally inefficient (since it requires $2^{|E|}$ steps)...
- ...and, in general, is not useful for proving enumerative results like Cayley's formula.

The Matrix-Tree Theorem

$G = (V, E)$: connected graph without loops (parallel edges OK)
$V = \{1, 2, \dots, n\}$

Definition The **Laplacian of G** is the $n \times n$ matrix $L = [\ell_{ij}]$:

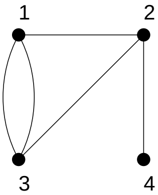
$$\ell_{ij} = \begin{cases} \deg_G(i) & \text{if } i = j \\ -(\#\text{ of edges between } i \text{ and } j) & \text{otherwise.} \end{cases}$$

- rank $L = n - 1$.

The Matrix-Tree Theorem

Example

$G =$



$$L = \begin{bmatrix} 3 & -1 & -2 & 0 \\ -1 & 3 & -1 & -1 \\ -2 & -1 & 3 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

The Matrix-Tree Theorem

The Matrix-Tree Theorem (Kirchhoff, 1847)

(1) Let $0, \lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ be the eigenvalues of L. Then the number of spanning trees of G is

$$\tau(G) = \frac{\lambda_1 \lambda_2 \cdots \lambda_{n-1}}{n} .$$

The Matrix-Tree Theorem

The Matrix-Tree Theorem (Kirchhoff, 1847)

(1) Let $0, \lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ be the eigenvalues of L. Then the number of spanning trees of G is

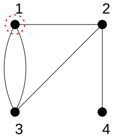
$$\tau(G) = \frac{\lambda_1 \lambda_2 \cdots \lambda_{n-1}}{n} .$$

(2) Pick any $i \in \{1, \ldots, n\}$. Form the *reduced Laplacian* $\tilde{L}$ by deleting the i^{th} row and i^{th} column of L. Then

$$\tau(G) = \det \tilde{L} .$$

The Matrix-Tree Theorem

Example

$G =$ 

$$L = \begin{bmatrix} 3 & -1 & -2 & 0 \\ -1 & 3 & -1 & -1 \\ -2 & -1 & 3 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \qquad \tilde{L} = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

Eigenvalues: 0, 1, 4, 5

$$(1 \cdot 4 \cdot 5)/4 = 5$$

$$\det \tilde{L} = 5$$

The Chip-Firing Game

- Discrete dynamical system on graphs discovered independently by many: Biggs, Dhar, Merino, . . .

- Essentially equivalent to the *abelian sandpile model*, *dollar game*, . . .

The Chip-Firing Game

- Let $G = (V, E)$ be a simple graph, $V = \{0, 1, \ldots, n\}$. Each vertex i has a finite number c_i of poker chips.

The Chip-Firing Game

- Let $G = (V, E)$ be a simple graph, $V = \{0, 1, \ldots, n\}$. Each vertex i has a finite number c_i of poker chips.
- A vertex *fires* by giving one chip to each of its neighbors.

The Chip-Firing Game

- Let $G = (V, E)$ be a simple graph, $V = \{0, 1, \dots, n\}$. Each vertex i has a finite number c_i of poker chips.
- A vertex *fires* by giving one chip to each of its neighbors.
- Vertex 0, the *bank*, only fires if no other vertex can fire.

The Chip-Firing Game

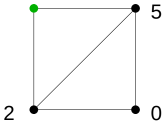
- Let $G = (V, E)$ be a simple graph, $V = \{0, 1, \ldots, n\}$. Each vertex i has a finite number c_i of poker chips.
- A vertex *fires* by giving one chip to each of its neighbors.
- Vertex 0, the *bank*, only fires if no other vertex can fire.
- Vertices other than the bank cannot go into debt

The Chip-Firing Game

- Let $G = (V, E)$ be a simple graph, $V = \{0, 1, \ldots, n\}$.
  Each vertex i has a finite number c_i of poker chips.
- A vertex *fires* by giving one chip to each of its neighbors.
- Vertex 0, the *bank*, only fires if no other vertex can fire.
- Vertices other than the bank cannot go into debt
- State of the system $= \mathbf{c} = (c_1, \ldots, c_n)$
  (We don't care how many chips the bank has.)

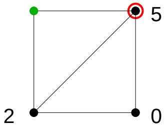
The Chip-Firing Game

Bank



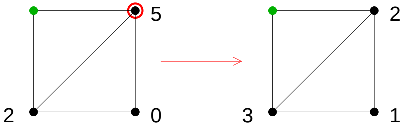
The Chip-Firing Game

Bank



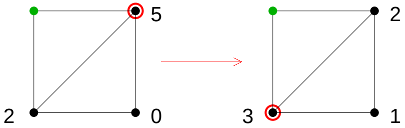
The Chip-Firing Game

Bank



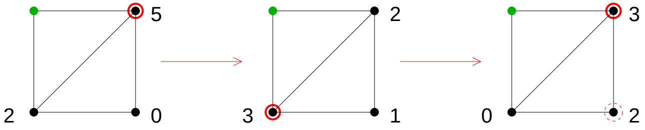
The Chip-Firing Game

Bank



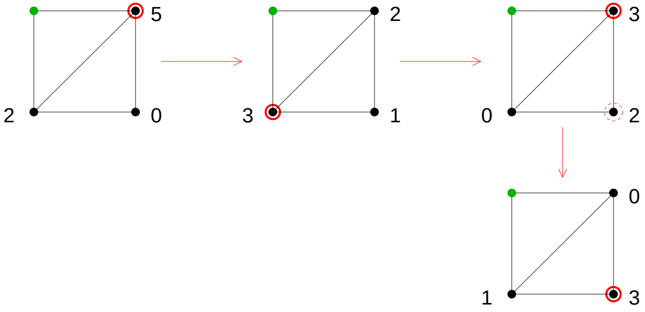
The Chip-Firing Game

Bank



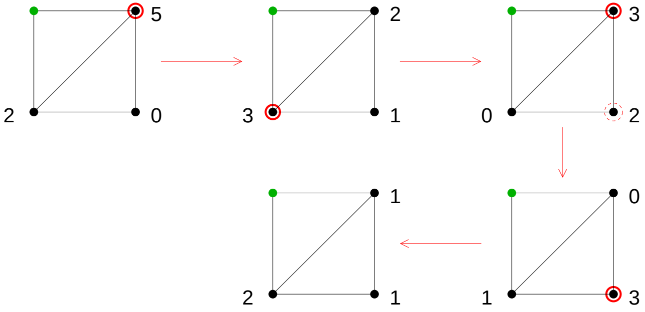
The Chip-Firing Game

Bank



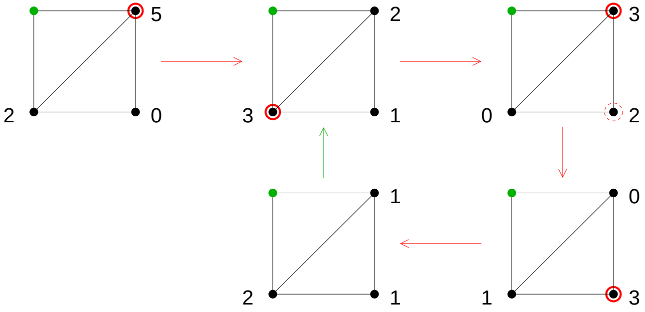
The Chip-Firing Game

Bank



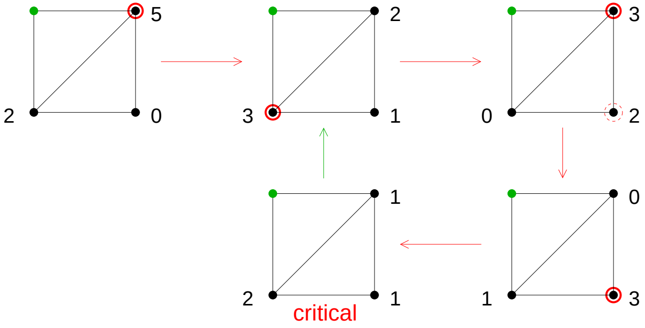
The Chip-Firing Game

Bank



The Chip-Firing Game

Bank



Chip-Firing and the Laplacian

- Recall: reduced Laplacian of G is $\tilde{L} = [\ell_{ij}]_{i,j=1\dots n}$, where

$$\ell_{ij} = \begin{cases} \deg_G(i) & \text{if } i = j \\ -1 & \text{if } i, j \text{ are adjacent} \\ 0 & \text{otherwise.} \end{cases}$$

Chip-Firing and the Laplacian

- Recall: reduced Laplacian of G is $\tilde{L} = [\ell_{ij}]_{i,j=1\dots n}$, where

$$\ell_{ij} = \begin{cases} \deg_G(i) & \text{if } i = j \\ -1 & \text{if } i,j \text{ are adjacent} \\ 0 & \text{otherwise.} \end{cases}$$

- Firing vertex $i \longleftrightarrow$ subtracting i^{th} column of $\tilde{L}$ from $\mathbf{c}$

Chip-Firing and the Laplacian

- Recall: reduced Laplacian of G is $\tilde{L} = [\ell_{ij}]_{i,j=1\ldots n}$, where

$$\ell_{ij} = \begin{cases} \deg_G(i) & \text{if } i = j \\ -1 & \text{if } i,j \text{ are adjacent} \\ 0 & \text{otherwise.} \end{cases}$$

- Firing vertex $i \longleftrightarrow$ subtracting i^{th} column of $\tilde{L}$ from $\mathbf{c}$

Fact Each starting state $\mathbf{c}$ eventually leads to a unique critical state $Crit(\mathbf{c})$.

Chip-Firing and Trees

Call two state vectors $\mathbf{c}, \mathbf{c}'$ *firing-equivalent* if their difference is in the column space of $\tilde{L}$.

Fact $\mathbf{c}, \mathbf{c}'$ are firing-equivalent if and only if $Crit(\mathbf{c}) = Crit(\mathbf{c}')$.

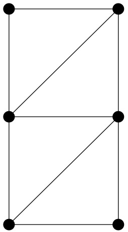
Fact Number of critical states $= \det \tilde{L} = \tau(G)$.

Acyclic Orientations

To *orient* a graph, place an arrow on each edge.

Acyclic Orientations

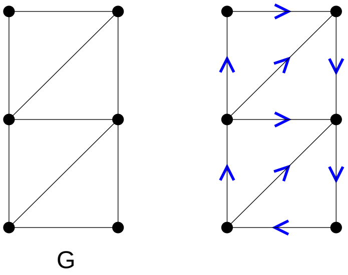
To *orient* a graph, place an arrow on each edge.



G

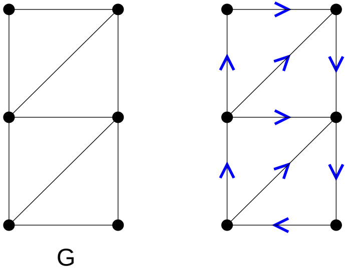
Acyclic Orientations

To *orient* a graph, place an arrow on each edge.



Acyclic Orientations

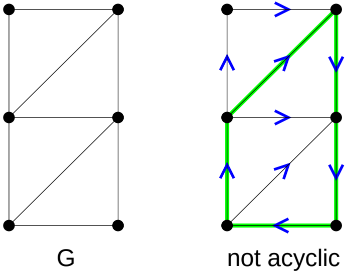
To *orient* a graph, place an arrow on each edge.



An orientation is *acyclic* if it contains no directed cycles.

Acyclic Orientations

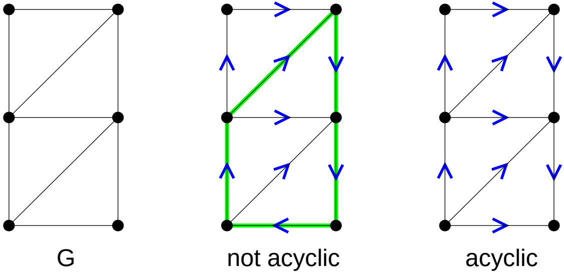
To *orient* a graph, place an arrow on each edge.



An orientation is *acyclic* if it contains no directed cycles.

Acyclic Orientations

To *orient* a graph, place an arrow on each edge.



An orientation is *acyclic* if it contains no directed cycles.

Counting Acyclic Orientations

$\alpha(G)$ = number of acyclic orientations of G

Counting Acyclic Orientations

$\alpha(G)$ = number of acyclic orientations of G

- $\alpha(\text{tree with } n \text{ vertices}) = 2^{n-1}$
- $\alpha(C_n) = 2^n - 2$
- $\alpha(K_n) = n!$

Counting Acyclic Orientations

$\alpha(G)$ = number of acyclic orientations of G

- $\alpha(\text{tree with } n \text{ vertices}) = 2^{n-1}$
- $\alpha(C_n) = 2^n - 2$
- $\alpha(K_n) = n!$

Theorem $\alpha(G) = \alpha(G - e) + \alpha(G/e)$.

Counting Acyclic Orientations

$\alpha(G)$ = number of acyclic orientations of G

- $\alpha(\text{tree with } n \text{ vertices}) = 2^{n-1}$
- $\alpha(C_n) = 2^n - 2$
- $\alpha(K_n) = n!$

Theorem $\alpha(G) = \alpha(G - e) + \alpha(G/e)$.

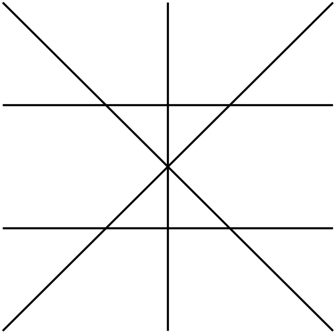
(Fact: Both $\alpha(G)$ and $\tau(G)$, as well as any other invariant satisfying a deletion-contraction recurrence, can be obtained from the *Tutte polynomial* $T_G(x, y)$.)

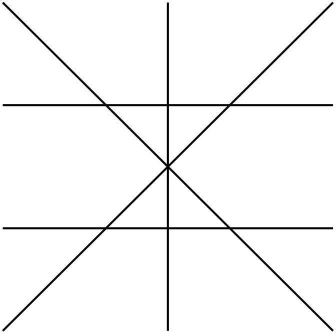
Hyperplane Arrangements

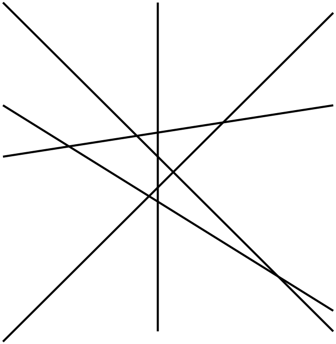
Definition A **hyperplane** H in $\mathbb{R}^n$ is an $(n-1)$-dimensional affine linear subspace.

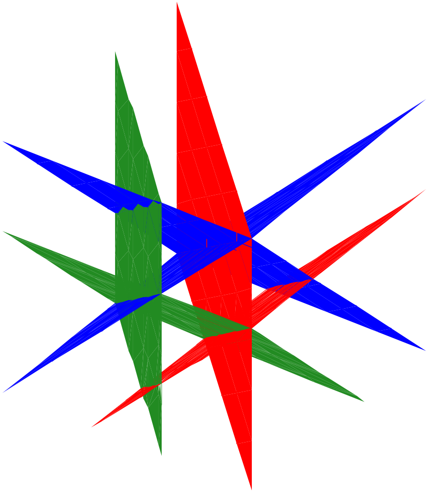
Definition A **hyperplane arrangement** $\mathcal{A} \subset \mathbb{R}^n$ is a finite collection of hyperplanes.

- $n = 1$: points on a line
- $n = 2$: lines on a plane
- $n = 3$: planes in 3-space







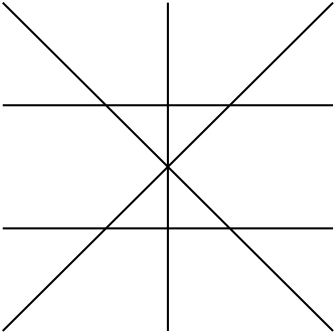


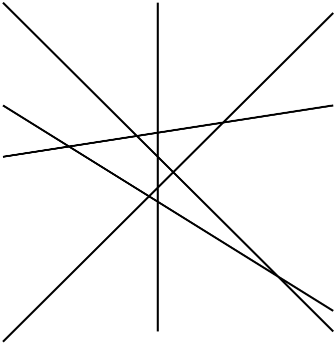
Counting Regions

$$\begin{aligned} r(\mathcal{A}) &:= \text{number of regions of } \mathcal{A} \\ &= \text{number of connected components of } \mathbb{R}^n \setminus \mathcal{A} \end{aligned}$$

Counting Regions

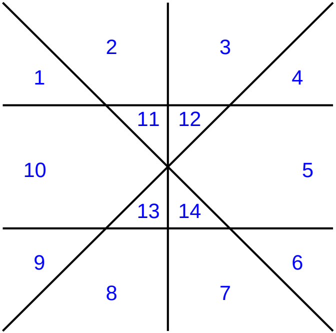
$$\begin{aligned} r(\mathcal{A}) &:= \text{number of regions of } \mathcal{A} \\ &= \text{number of connected components of } \mathbb{R}^n \setminus \mathcal{A} \end{aligned}$$



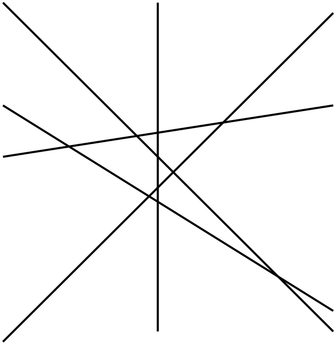


Counting Regions

$$\begin{aligned} r(\mathcal{A}) &:= \text{number of regions of } \mathcal{A} \\ &= \text{number of connected components of } \mathbb{R}^n \setminus \mathcal{A} \end{aligned}$$



14 regions



16 regions

Counting Regions

Example $\mathcal{A} = n$ lines in $\mathbb{R}^2$

- $2n \;\leq\; r(\mathcal{A}) \;\leq\; 1 + \binom{n+1}{2}$

Example $\mathcal{A} = n$ coordinate hyperplanes in $\mathbb{R}^n$

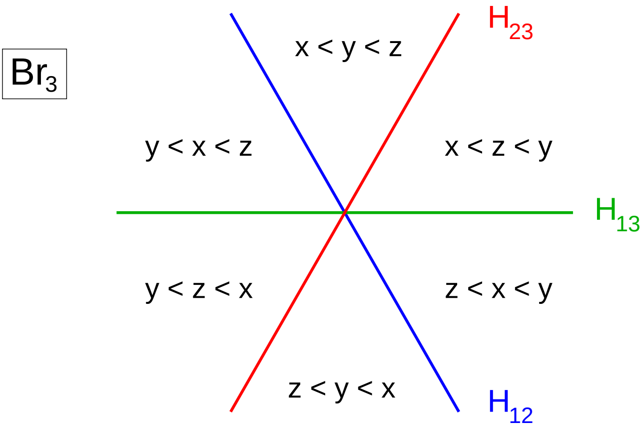
- Regions of $\mathcal{A}$ = orthants
- $r(\mathcal{A}) = 2^n$

The Braid Arrangement

The *braid arrangement* $Br_n \subset \mathbb{R}^n$ consists of the $\binom{n}{2}$ hyperplanes

$$\begin{aligned} H_{12} &= \{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_2\}, \\ H_{13} &= \{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_3\}, \\ &\dots \\ H_{n-1,n} &= \{\mathbf{x} \in \mathbb{R}^n \mid x_{n-1} = x_n\}. \end{aligned}$$

- $\mathbb{R}^n \setminus Br_n = \{\mathbf{x} \in \mathbb{R}^n \mid \text{all } x_i \text{ are distinct}\}$.
- **Problem:** Count the regions of Br_n.


Br3
H23
H13
H12
x < y < z
y < x < z
x < z < y
y < z < x
z < x < y
z < y < x

Graphic Arrangements

Let $G = (V, E)$ be a simple graph with $V = [n] = \{1, \ldots, n\}$. The *graphic arrangement* $\mathcal{A}_G \subset \mathbb{R}^n$ consists of the hyperplanes

$$\{H_{ij} : \; x_i = x_j \quad \Big| \quad ij \in E\}.$$

Graphic Arrangements

Let $G = (V, E)$ be a simple graph with $V = [n] = \{1, \ldots, n\}$. The *graphic arrangement* $\mathcal{A}_G \subset \mathbb{R}^n$ consists of the hyperplanes

$$\{H_{ij} : \; x_i = x_j \quad \Big| \quad ij \in E\}.$$

Theorem There is a bijection between regions of $\mathcal{A}_G$ and acyclic orientations of G. In particular,

$$r(\mathcal{A}_G) = \alpha(G).$$

(When $G = K_n$, the arrangement $\mathcal{A}_G$ is the braid arrangement.)

Graphic Arrangements

Theorem $r(\mathcal{A}_G) = \alpha(G)$.

Graphic Arrangements

Theorem $r(\mathcal{A}_G) = \alpha(G)$.

Sketch of proof: Suppose that $\mathbf{a} \in \mathbb{R}^n \setminus \mathcal{A}_G$.

Graphic Arrangements

Theorem $r(\mathcal{A}_G) = \alpha(G)$.

Sketch of proof: Suppose that $\mathbf{a} \in \mathbb{R}^n \setminus \mathcal{A}_G$.

In particular, $a_i \neq a_j$ for every edge ij. Orient that edge as

$$\begin{cases} i \to j & \text{if } a_i < a_j, \\ j \to i & \text{if } a_i > a_j. \end{cases}$$

Graphic Arrangements

Theorem $r(\mathcal{A}_G) = \alpha(G)$.

Sketch of proof: Suppose that $\mathbf{a} \in \mathbb{R}^n \setminus \mathcal{A}_G$.

In particular, $a_i \neq a_j$ for every edge ij. Orient that edge as

$$\begin{cases} i \to j & \text{if } a_i < a_j, \\ j \to i & \text{if } a_i > a_j. \end{cases}$$

The resulting orientation is acyclic.

Graphic Arrangements

Theorem $r(\mathcal{A}_G) = \alpha(G)$.

Sketch of proof: Suppose that $\mathbf{a} \in \mathbb{R}^n \setminus \mathcal{A}_G$.

In particular, $a_i \neq a_j$ for every edge ij. Orient that edge as

$$\begin{cases} i \to j & \text{if } a_i < a_j, \\ j \to i & \text{if } a_i > a_j. \end{cases}$$

The resulting orientation is acyclic.

Corollary $r(Br_n) = \alpha(K_n) = n!$.

Parking Functions

There are n parking spaces on a one-way street.

Cars $1, \ldots, n$ want to park in the spaces.

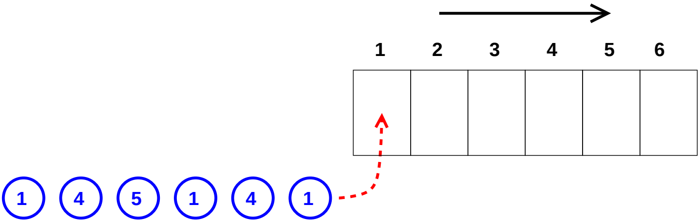
Each car has a preferred spot p_i.

Can all the cars park?

(Analogy: Hash table. . .)

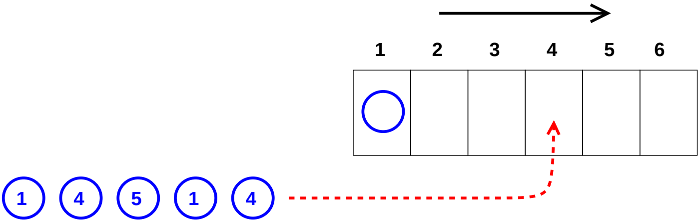
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



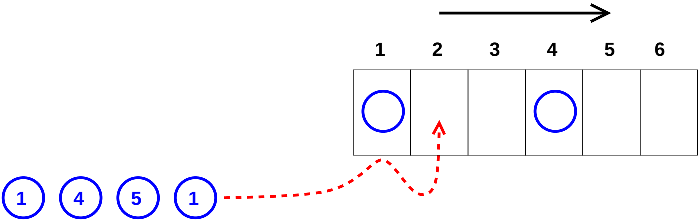
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



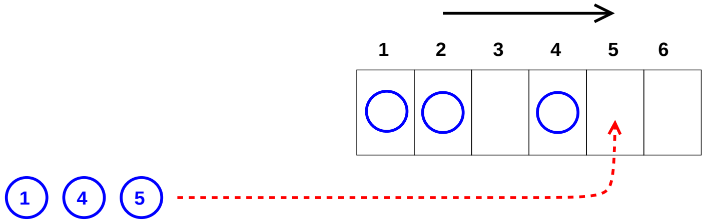
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



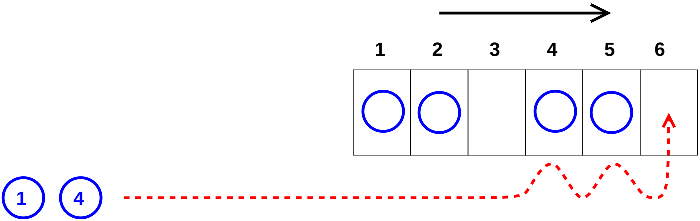
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



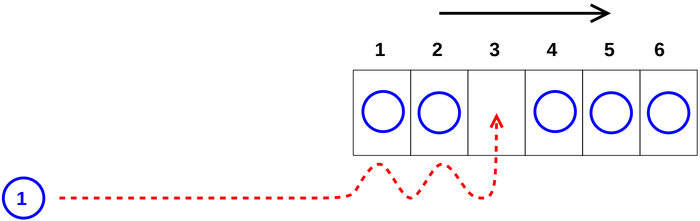
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



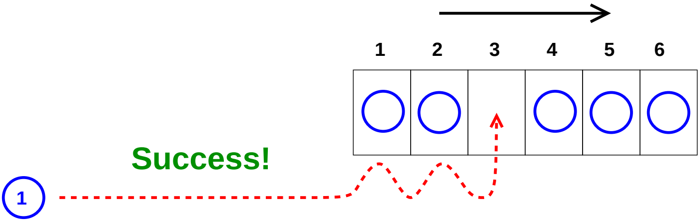
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



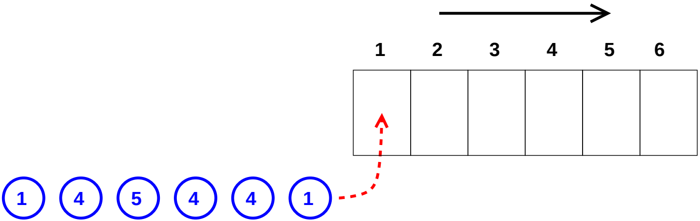
Parking Functions

Example #1: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 1, 5, 4, 1)$



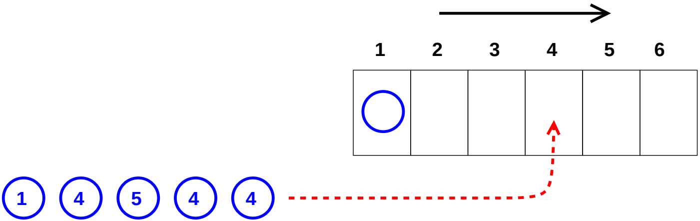
Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



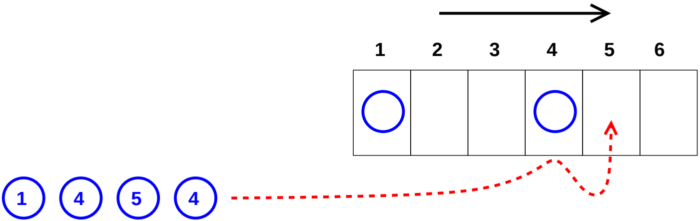
Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



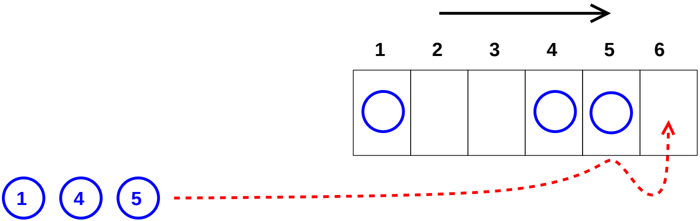
Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



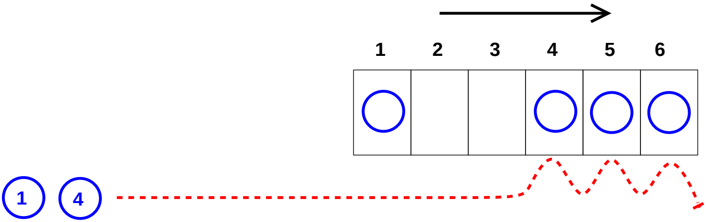
Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



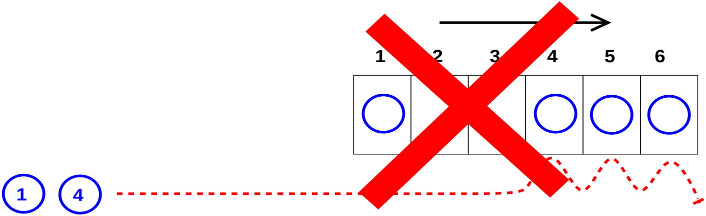
Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



Parking Functions

Example #2: $n = 6$; $(p_1, \ldots, p_6) = (1, 4, 4, 5, 4, 1)$



Parking Functions

- $(p_1, \ldots, p_n)$ is a parking function if and only if the i^{th} smallest entry is $\leq i$, for all i.

Parking Functions

- $(p_1, \ldots, p_n)$ is a parking function if and only if the i^{th} smallest entry is $\leq i$, for all i.

111	112	122	113	123 132
	121	212	131	213 231
	211	221	311	312 321

Parking Functions

- $(p_1, \ldots, p_n)$ is a parking function if and only if the i^{th} smallest entry is $\leq i$, for all i.

111	112	122	113	123 132
	121	212	131	213 231
	211	221	311	312 321

- In particular, parking functions are invariant up to permutation.

Parking Functions

- $(p_1, \ldots, p_n)$ is a parking function if and only if the i^{th} smallest entry is $\leq i$, for all i.

111	112	122	113	123 132
	121	212	131	213 231
	211	221	311	312 321

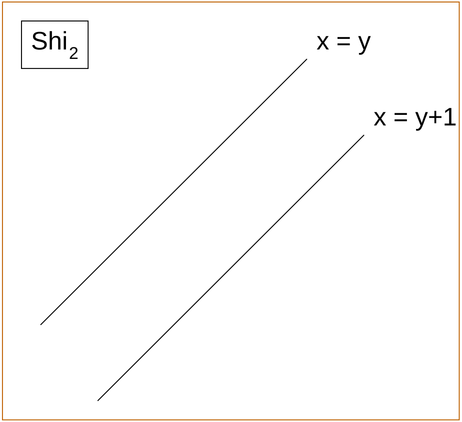
- In particular, parking functions are invariant up to permutation.
- The number of parking functions of length n is $(n+1)^{n-1}$.

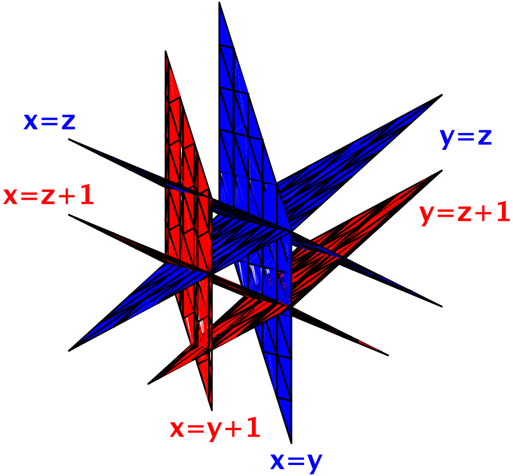
The Shi Arrangement

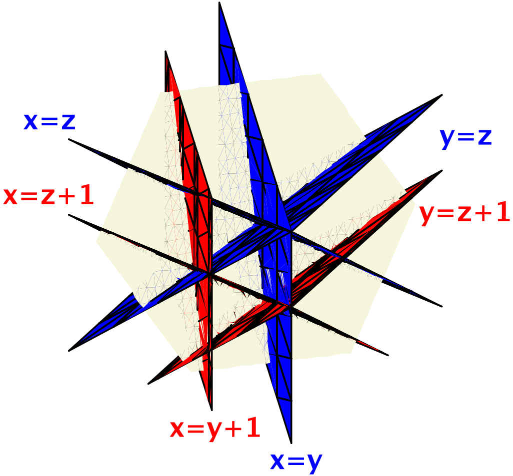
The *Shi arrangement* $Shi_n \subset \mathbb{R}^n$ consists of the $2\binom{n}{2}$ hyperplanes

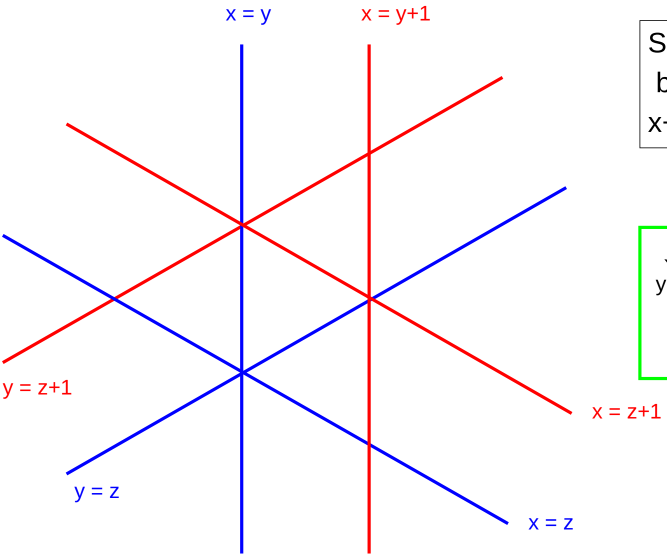
$$\begin{array}{ll}
\{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_2\}, & \{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_2 + 1\}, \\
\{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_3\}, & \{\mathbf{x} \in \mathbb{R}^n \mid x_1 = x_3 + 1\}, \\
\dots & \\
\{\mathbf{x} \in \mathbb{R}^n \mid x_{n-1} = x_n\}, & \{\mathbf{x} \in \mathbb{R}^n \mid x_{n-1} = x_n + 1\}.
\end{array}$$

The Shi Arrangement

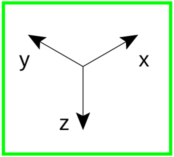



x=z
y=z
x=z+1
y=z+1
x=y+1
x=y


x=z
y=z
x=z+1
y=z+1
x=y+1
x=y



Slice of Shi_3
by plane
x+y+z = 0



The Shi Arrangement

Theorem The number of regions in Shi_n is $(n+1)^{n-1}$.

(Many proofs known: Shi, Athanasiadis-Linusson, Stanley . . .)

Score Vectors

Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

Score Vectors

Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

- If $x_i < x_j$, then j scores a point.

Score Vectors

Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

- If $x_i < x_j$, then j scores a point.
- If $x_j < x_i < x_j + 1$, then no one scores a point.

Score Vectors

Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

- If $x_i < x_j$, then j scores a point.
- If $x_j < x_i < x_j + 1$, then no one scores a point.
- If $x_j + 1 < x_i$, then i scores a point.

Score Vectors

Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

- If $x_i < x_j$, then j scores a point.
- If $x_j < x_i < x_j + 1$, then no one scores a point.
- If $x_j + 1 < x_i$, then i scores a point.

$\mathbf{s} = (s_1, \ldots, s_n) =$ **score vector**
(where $s_i =$ number of points scored by i).

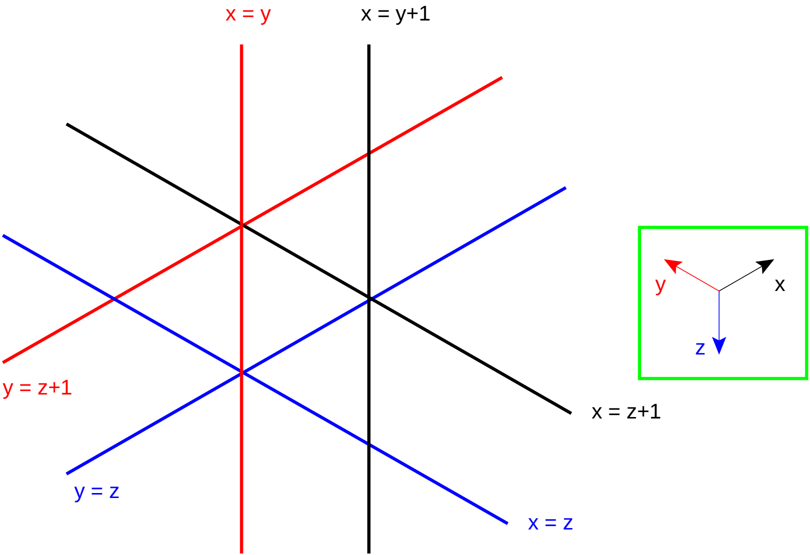
Score Vectors

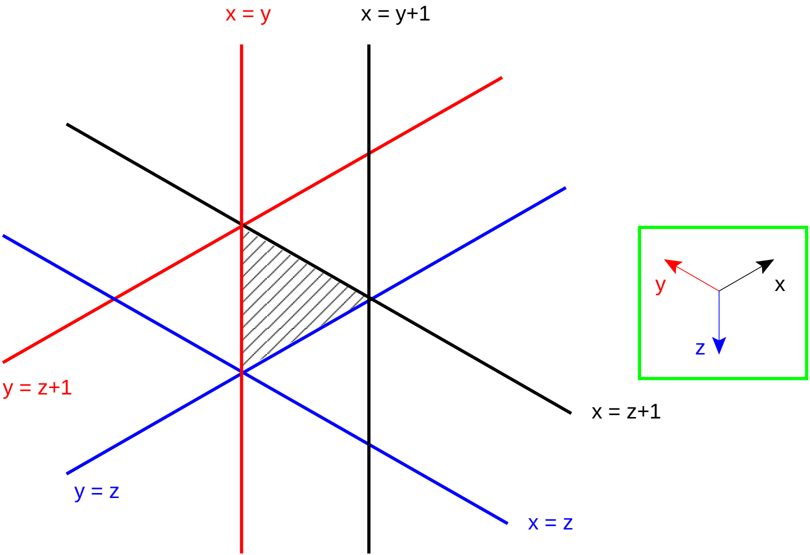
Let $\mathbf{x} \in \mathbb{R}^n \setminus Shi_n$. For every $1 \leq i < j \leq n$:

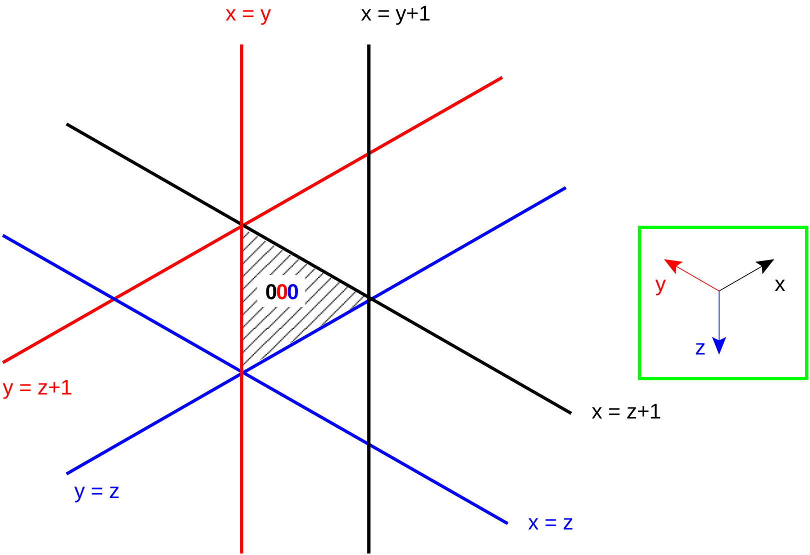
- If $x_i < x_j$, then j scores a point.
- If $x_j < x_i < x_j + 1$, then no one scores a point.
- If $x_j + 1 < x_i$, then i scores a point.

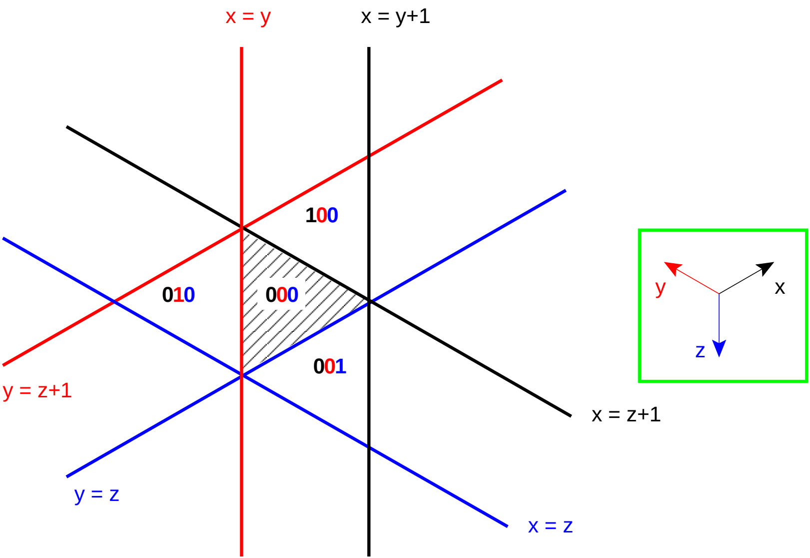
$\mathbf{s} = (s_1, \ldots, s_n) =$ **score vector**
(where $s_i =$ number of points scored by i).

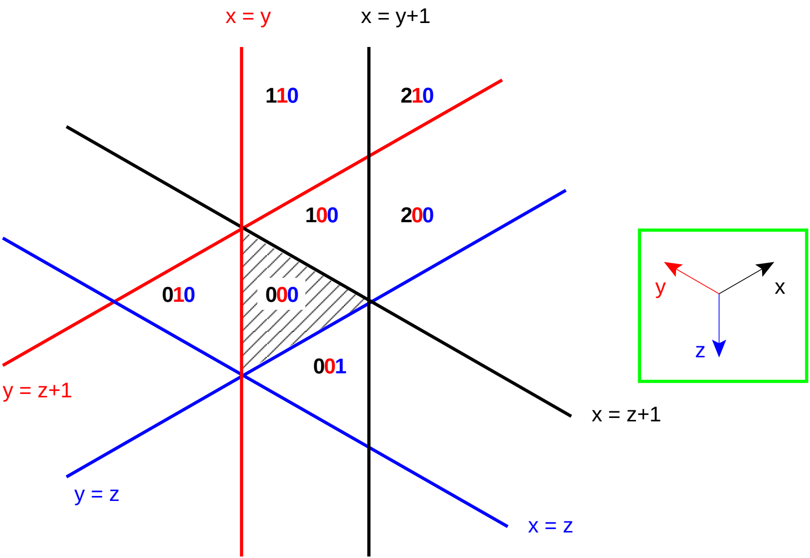
Example The score vector of $\mathbf{x} = (3.142,\ 2.010,\ 2.718)$ is $\mathbf{s} = (1, 0, 1)$.

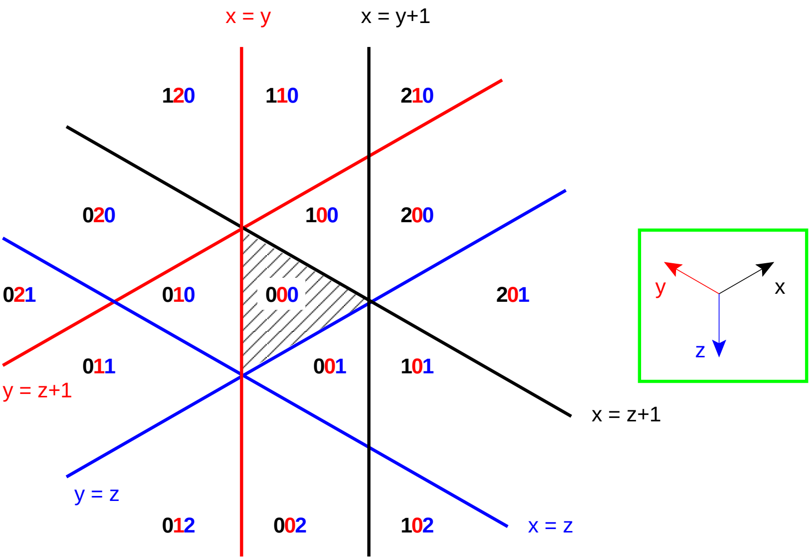

x = y
x = y+1
y = z+1
x = z+1
y = z
x = z
y
x
z


x = y
x = y+1
y = z+1
x = z+1
y = z
x = z
y
x
z


x = y
x = y+1
000
y = z+1
x = z+1
y = z
x = z
y
x
z


x = y
x = y+1
100
010
000
001
y = z+1
x = z+1
y = z
x = z
y
x
z


x = y
x = y+1
110
210
100
200
010
000
001
y = z+1
x = z+1
y = z
x = z
y
x
z


x = y
x = y+1
120
110
210
020
100
200
021
010
000
201
011
001
101
y = z+1
x = z+1
y = z
012
002
102
x = z
y
x
z

Score Vectors and Parking Functions

Theorem $(s_1, \ldots, s_n)$ is the score vector of some region of Shi_n $\iff$ $(s_1 + 1, \ldots, s_n + 1)$ is a parking function of length n.

Score Vectors and Parking Functions

Theorem $(s_1, \ldots, s_n)$ is the score vector of some region of Shi_n $\iff$ $(s_1 + 1, \ldots, s_n + 1)$ is a parking function of length n.

Theorem

$$\sum_{\text{regions } R \text{ of } Shi_n} y^{d(R_0,R)} = \sum_{\substack{\text{parking fns} \\ (p_1,\ldots,p_n)}} y^{p_1+\cdots+p_n}$$

where d = distance, R_0 = base region.

Score Vectors and Parking Functions

Theorem $(s_1, \ldots, s_n)$ is the score vector of some region of Shi_n $\iff$ $(s_1+1, \ldots, s_n+1)$ is a parking function of length n.

Theorem

$$\sum_{\text{regions } R \text{ of } Shi_n} y^{d(R_0,R)} = \sum_{\substack{\text{parking fns} \\ (p_1,\ldots,p_n)}} y^{p_1+\cdots+p_n}$$

where d = distance, R_0 = base region.

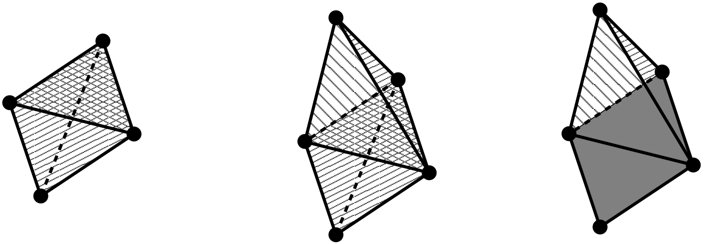
Example For $n=3$: $T_{K_4}(1,y) = 1+3y+6y^2+6y^3$.

Simplicial Complexes

Definition A *simplicial complex* is a space built out of

- vertices (dimension 0)
- edges (dimension 1)
- triangles (dimension 2)
- tetrahedra (dimension 3)
- higher-dimensional simplices

Simplicial complexes are the natural higher-dimensional analogues of graphs.



Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009. . . , extending Bolker 1978, Kalai 1983, Adin 1992]

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009..., extending Bolker 1978, Kalai 1983, Adin 1992]
- Critical group: yes [Duval–Klivans–JLM 2010]

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009. . . , extending Bolker 1978, Kalai 1983, Adin 1992]
- Critical group: yes [Duval–Klivans–JLM 2010]
- Acyclic orientations: maybe

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009..., extending Bolker 1978, Kalai 1983, Adin 1992]
- Critical group: yes [Duval–Klivans–JLM 2010]
- Acyclic orientations: maybe
- Chip-firing game: sort of ("stable state" is problematic)

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009. . . , extending Bolker 1978, Kalai 1983, Adin 1992]
- Critical group: yes [Duval–Klivans–JLM 2010]
- Acyclic orientations: maybe
- Chip-firing game: sort of ("stable state" is problematic)
- Parking functions: also doubtful

Open Questions

Do the links between graph theory and geometry generalize to higher dimension?

- Definition of spanning trees: yes (in several ways)
- Matrix-Tree Theorem: yes [Duval–Klivans–JLM 2007, 2009. . . , extending Bolker 1978, Kalai 1983, Adin 1992]
- Critical group: yes [Duval–Klivans–JLM 2010]
- Acyclic orientations: maybe
- Chip-firing game: sort of ("stable state" is problematic)
- Parking functions: also doubtful
- Hyperplane arrangements: ???