Planes, Hyperplanes, and Beyond

Jeremy L. Martin
Department of Mathematics
University of Kansas

KU Mini College
June 6, 2012

Summary

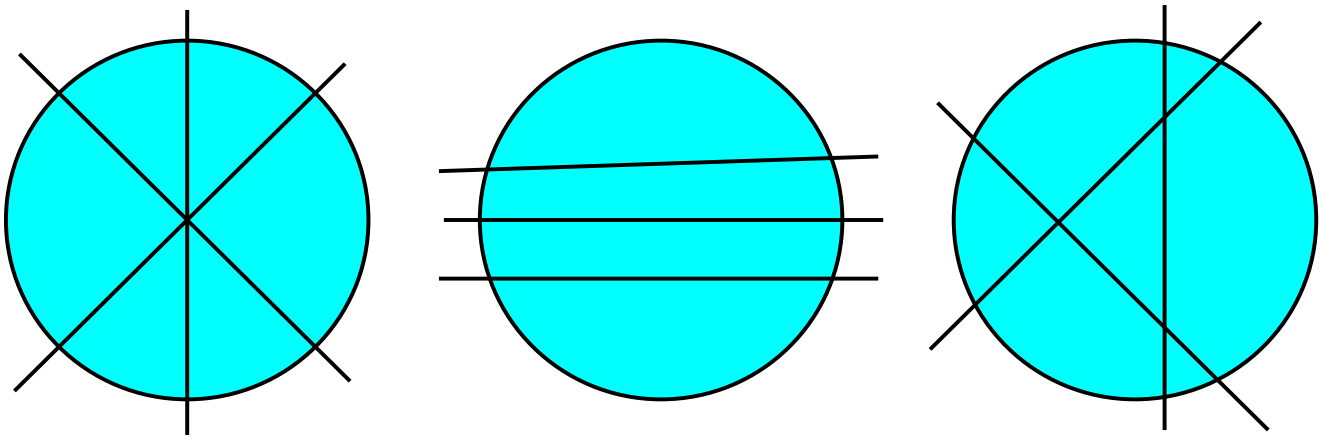
Suppose that you have a cake and are allowed to make ten straight-line slices. What is the greatest number of pieces you can produce? What if the slices have to be symmetric — or if the cake is four-dimensional? How can we possibly see what it looks like to slice space into pieces using lines, planes, or hyperplanes? Many of these questions have beautiful answers that can be revealed using unexpected, yet essentially simple mathematical techniques. Better yet, the seemingly abstract study of hyperplane arrangements has many surprising practical applications, ranging from optimization problems, to the theory of networks, to how a group of cars can find parking spots.

Themes

- Mathematics is about studying natural patterns using logic.
- Mathematics requires a high standard of proof — we can use evidence to make conjectures, but not to draw conclusion.
- In order to understand concepts we can't directly visualize (like four- and higher-dimensional space), we can use analogies.
- For example, understanding the two- and three-dimensional versions of a problem can help us understand the four-dimensional version.

The Cake-Cutting Problem

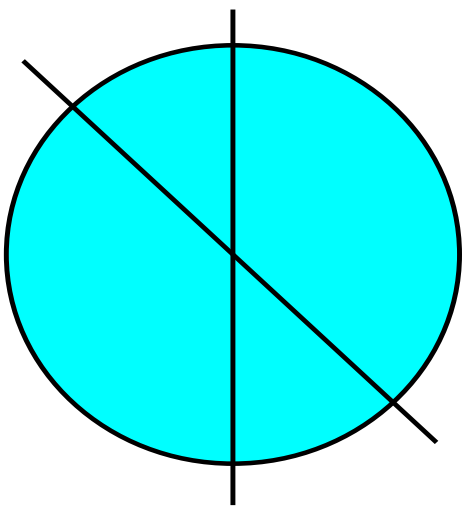
What is the greatest number of pieces that a cake can be cut into with a given number of cuts?



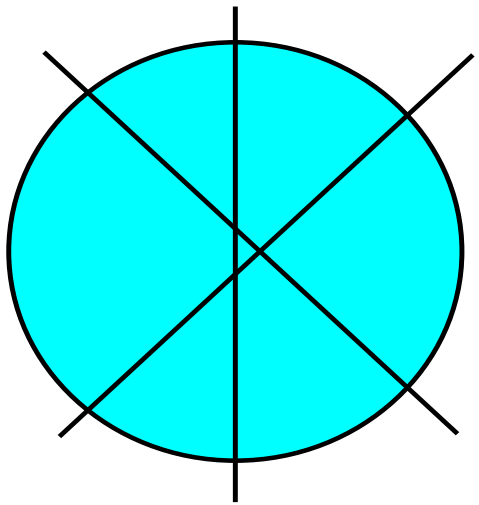
- The cuts must be straight lines and must go all the way through the cake.
- The sizes and shapes of the pieces don't matter.
- For the moment, we'll focus on 2-dimensional cakes (think of them as pancakes).

Solutions with 2, 3 or 4 Cuts

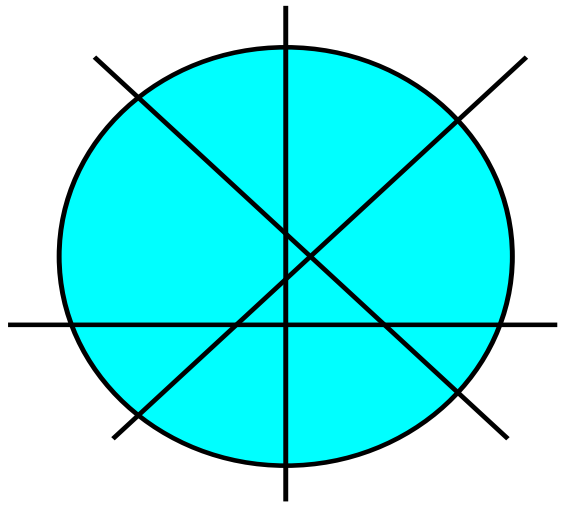
Let's write $P_2(N)$ for the maximum number of pieces obtainable using N cuts. (The 2 is a reminder of the dimension.)



2 cuts:
P(2) = 4



3 cuts:
P(3) = 7



4 cuts:
P(4) = 11

More Data

There is no way to obtain more than 2^N pieces with n cuts (since $P_2(N)$ can at most double with each cut)

In fact, $P_2(N)$ is much smaller than 2^N.

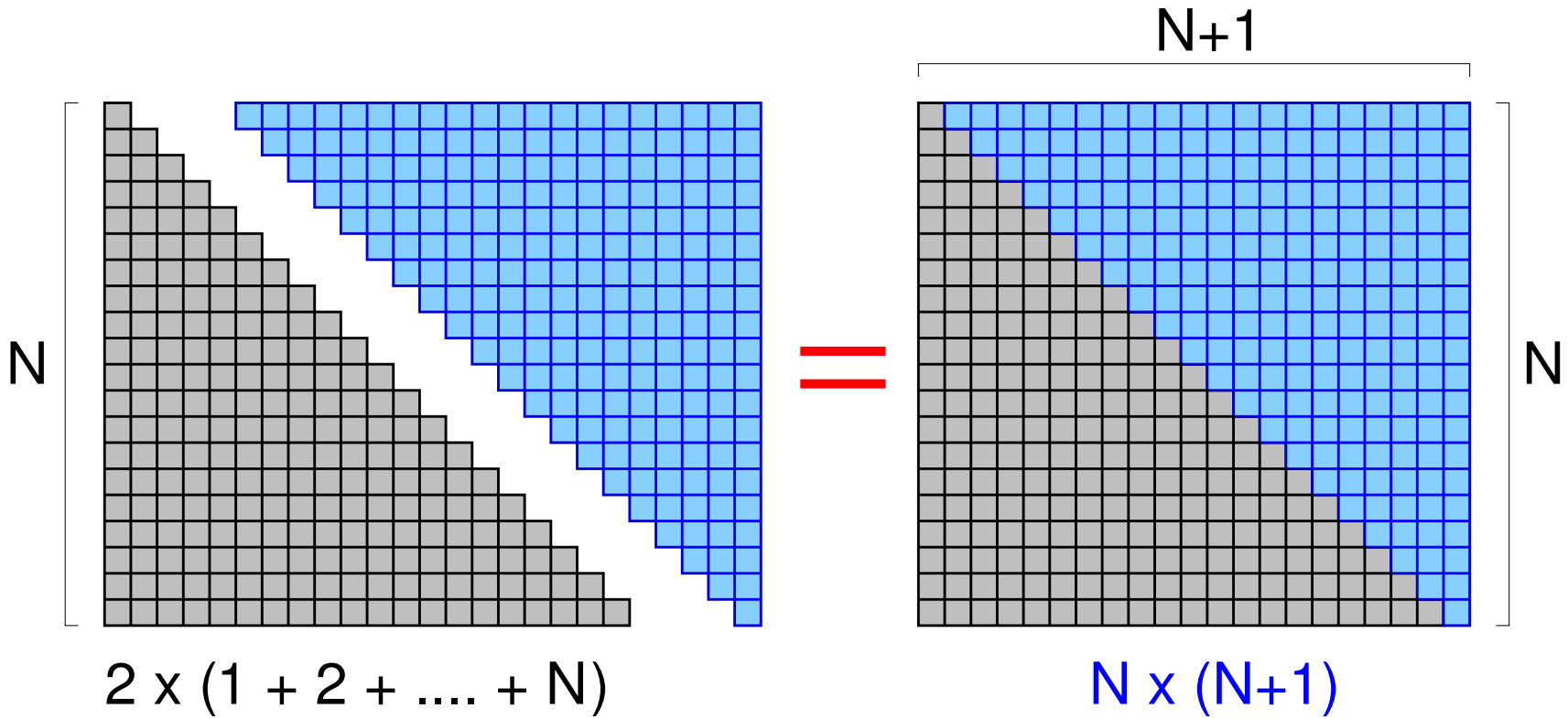
Cuts N	Pieces $P_2(N)$	2^N
1	2	2
2	4	4
3	7	8
4	11	16
5	16	32
...	...	...
20	211	1048576

The Pattern

N	$P_2(N)$				
0	1				
1	2	$=$	$1+1$		
2	4	$=$	$2+2$	$=$	$1+1+2$
3	7	$=$	$4+3$	$=$	$1+1+2+3$
4	11	$=$	$7+4$	$=$	$1+1+2+3+4$
5	16	$=$	$11+5$	$=$	$1+1+2+3+4+5$
$\dots$	$\dots$		$\dots$		$\dots$
20	211			$=$	$1+1+2+3+\cdots+19+20$

- How do we prove that the pattern works for every N?

Interlude: Counting Stairs

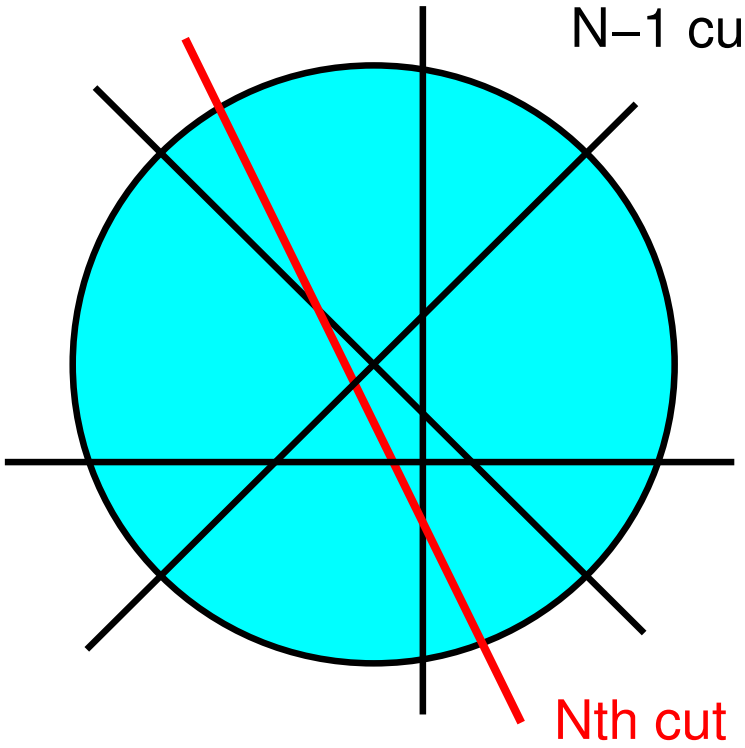


$$1 + 2 + \ldots + N = N(N+1) / 2$$

Maximizing the Number of Pieces

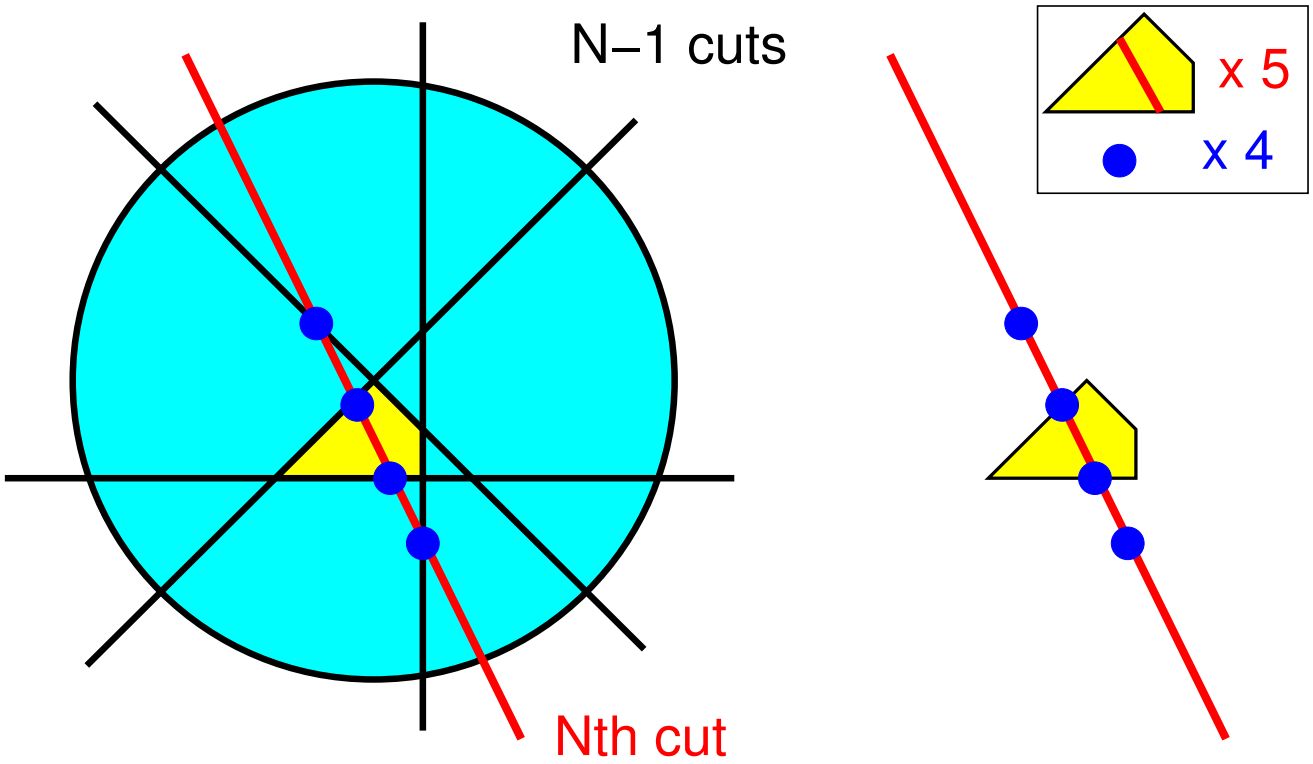
How can we ensure obtaining as many pieces as possible?

- First cut the pancake into $P_2(N-1)$ pieces using $N-1$ cuts.
- Now make the Nth cut, hitting as many pieces as possible.



Maximizing the Number of Pieces

Key observation: The number of pieces subdivided by the Nth cut equals one more than the number of previous cuts it meets.



Maximizing the Number of Pieces

If we make sure that

- every pair of cuts meets in some point, and
- no more than two cuts meet at any point,

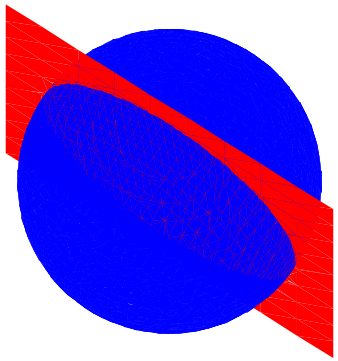
then the N^{th} cut will meet each of the previous $N-1$ cuts, and therefore will make N new pieces.

Since the original pancake had one piece, we have proved that

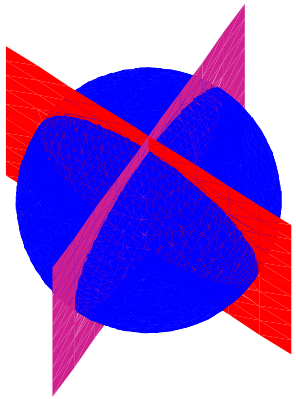
$$\boxed{P_2(N) = 1 + (1 + 2 + \cdots + N) = 1 + \frac{N(N+1)}{2}.}$$

From 2D to 3D

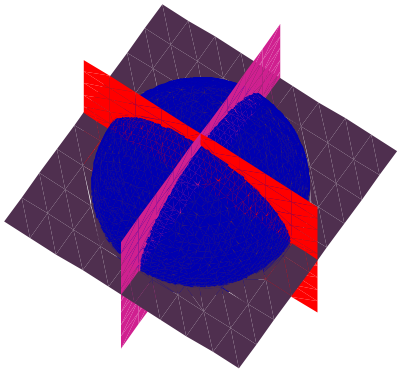
Let's write $P_3(N)$ for the maximum number of pieces obtainable from a 3-dimensional cake with N cuts.



$P_3(1) = 2$



$P_3(2) = 4$

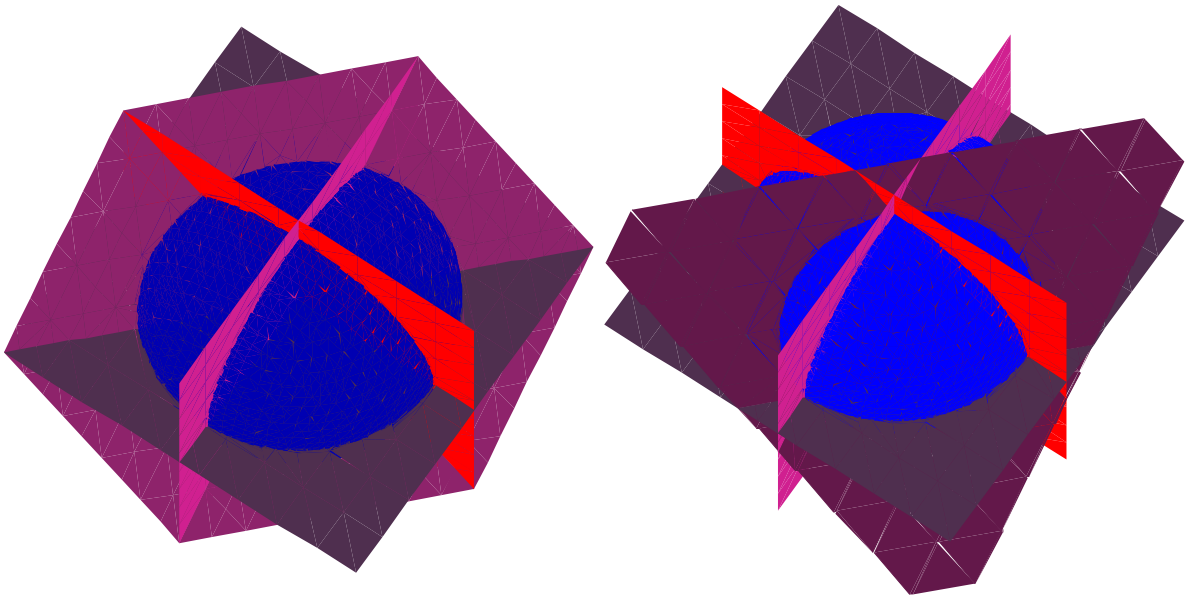


$P_3(3) = 8$

Compare 2D: $P(1) = 2$, $P(2) = 4$, $P(3) = 7$.

From 2D to 3D

At first glance it seems that $P_3(4) = 14$, but in the left-hand diagram, the four planes all meet at a point, so jiggling one plane a little increases the number of pieces to 15.



From 2D to 3D

N	0	1	2	3	4	5	6	7	8
$P_2(N)$	1	2	4	7	11	16	22	29	37
$P_3(N)$	1	2	4	8	15	26	42	64	93

The pattern is

$$\boxed{P_3(N) = P_3(N-1) + P_2(N-1).}$$

(This observation, together with the formula $P_2(N) = \frac{N(N+1)}{2} + 1$ and a little algebra, can be used to prove the formula $P_3(N) = \frac{N^3+5N+6}{6}$. But what is really important is the pattern!)

Pancakes, Cakes and Beyond

How about four-dimensional pancakes?

(Never mind whether they actually exist!)

More generally, if you have a d-dimensional cake and you can make N cuts, how many pieces can you make? (Call this $P_d(N)$.)

We already know the answer for $d = 2$ and $d = 3$. If $d = 1$, it is pretty clear that N cuts give $N + 1$ pieces.

	N								
	0	1	2	3	4	5	6	7	8
$P_1(N)$	1	2	3	4	5	6	7	8	9
$P_2(N)$	1	2	4	7	11	16	22	29	37
$P_3(N)$	1	2	4	8	15	26	42	64	93

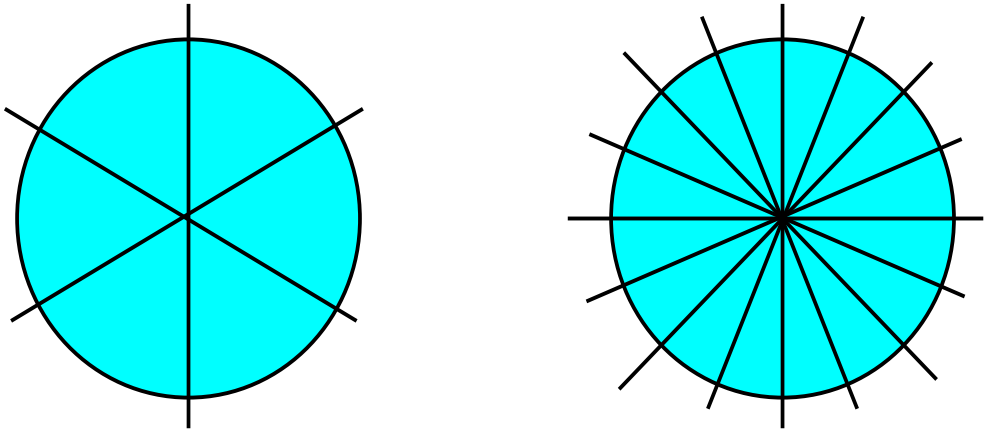
Pancakes, Cakes and Beyond

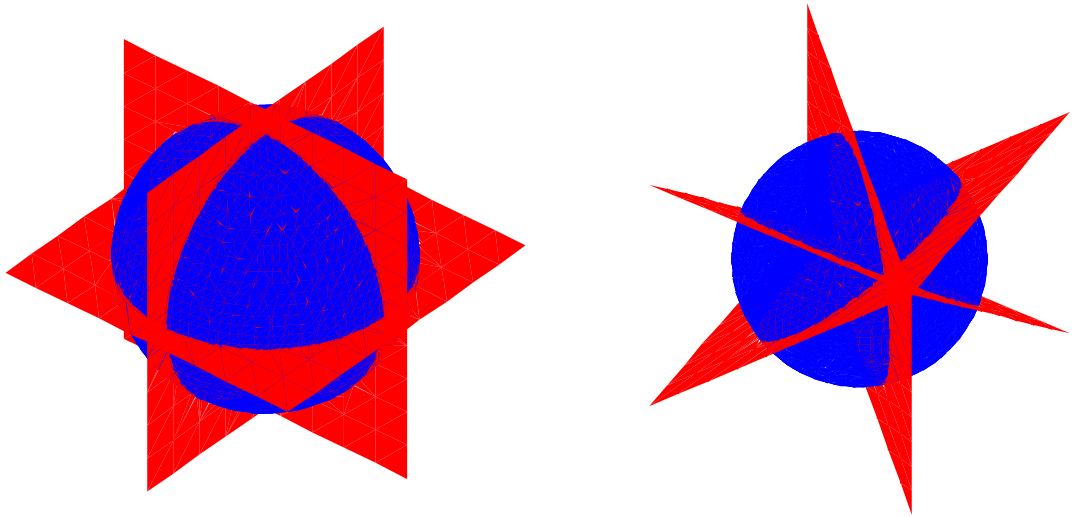
- Each number is the sum of the numbers immediately "west" ($\leftarrow$) and "northwest" ($\nwarrow$).
- Formula: $P_d N) = P_d(N-1) + P_{d-1}(N-1)$.

	N								
	0	1	2	3	4	5	6	7	8
$P_1(N)$	1	2	3	4	5	6	7	8	9
$P_2(N)$	1	2	4	7	11	16	22	29	37
$P_3(N)$	1	2	4	8	15	26	42	64	93
$P_4(N)$	1	2	4	8	16	31	57	99	163
$P_5(N)$	1	2	4	8	16	32	63	120	219
...	...								

Symmetric Cake-Cutting

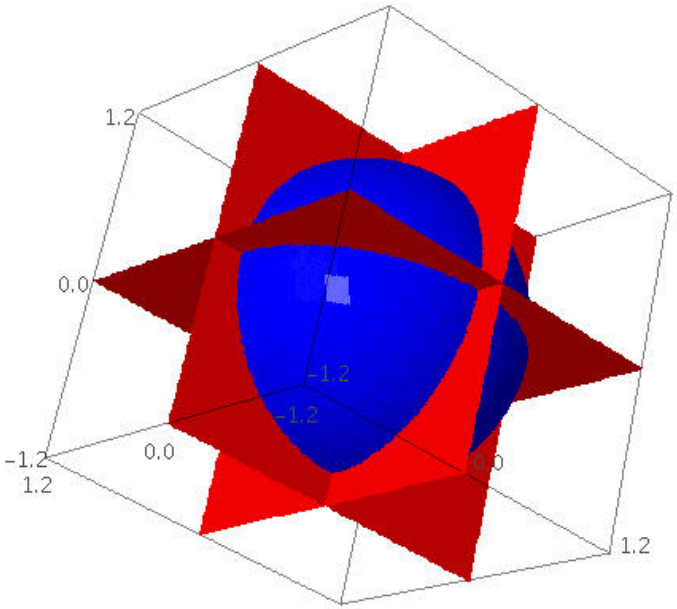
What are the possible ways to cut a perfectly round cake so that all pieces are congruent (i.e., geometrically the same)?



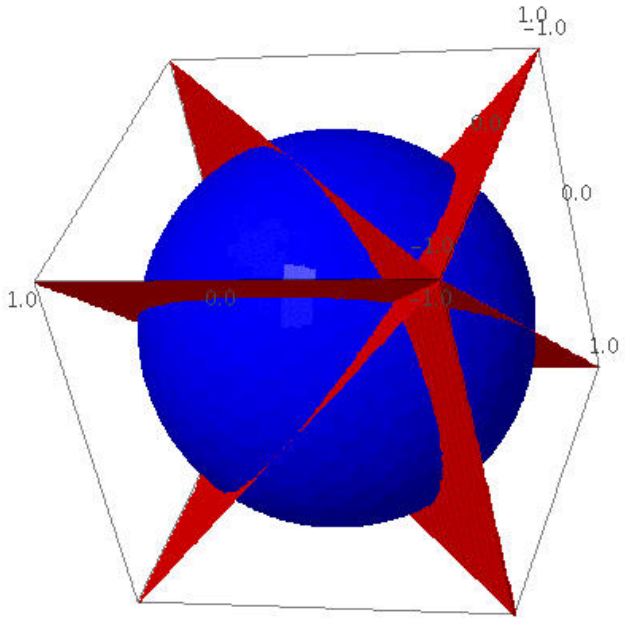


Symmetric Cake-Cutting

The symmetry shows up in the equations.



$x = 0,\ y = 0,\ z = 0$



$x = y,\ x = z,\ y = z$

Symmetric Cake-Cutting in Higher Dimensions

Question: If we can cut up a 3-dimensional sphere into congruent pieces, using planes defined by the equations

$$\boxed{x = 0,\ y = 0,\ z = 0} \qquad \text{or} \qquad \boxed{x = y,\ x = z,\ y = z}$$

then what happens if we cut up a 4-dimensional sphere into pieces using the "hyperplanes"

$$\boxed{\begin{array}{ll} w = 0, & x = 0, \\ y = 0, & z = 0 \end{array}} \quad \text{or} \quad \boxed{\begin{array}{lll} w = x, & w = y, & w = z, \\ x = y, & x = z, & y = z \end{array}} \ ?$$

(Note: We can replace "sphere" with "space".)

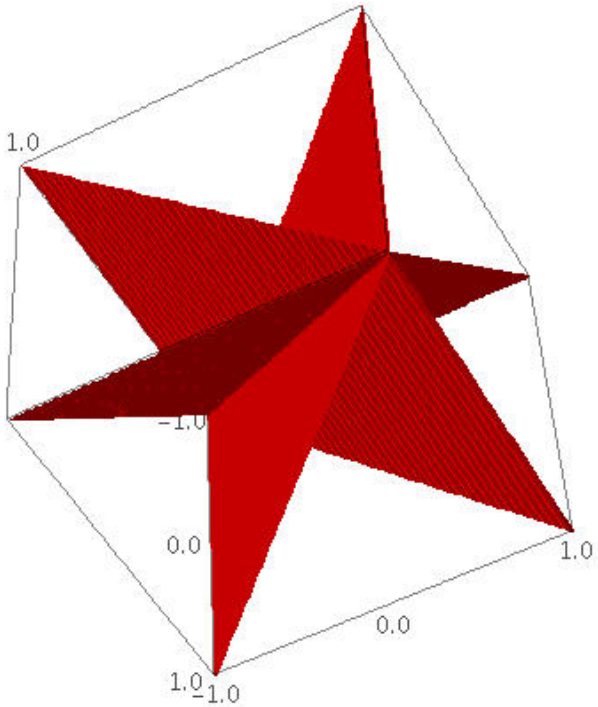
Symmetric Cake-Cutting in Higher Dimensions

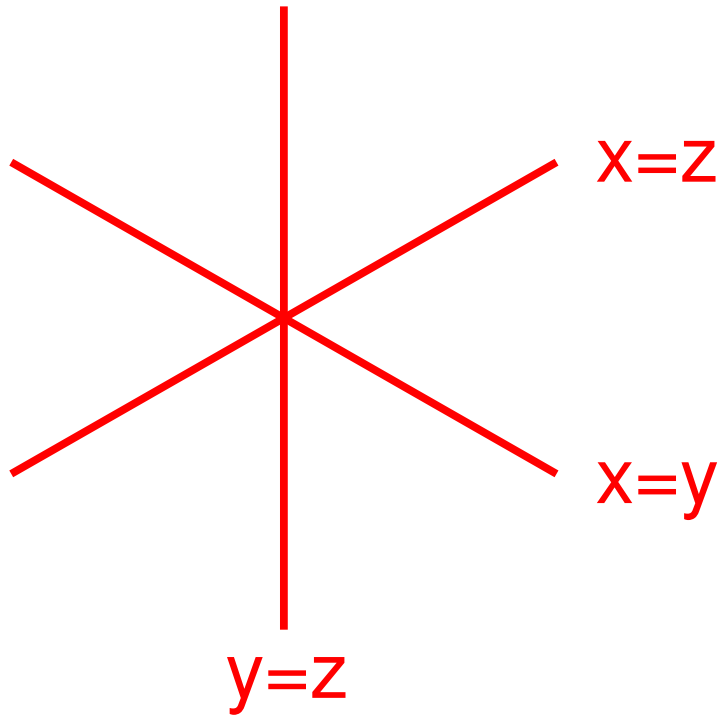
Some tools for visualizing 4-dimensional space:

- Work by analogy: understanding low-dimensional space can help us understand higher dimensions
- Project into lower dimension to make visualization easier
- Reexpress high-dimensional problems mathematically

The Braid Arrangement

The arrangement of planes $x = y$, $x = z$, $y = z$ is called the *3-dimensional braid arrangement* (**Braid3** for short).





Projecting from 3D to 2D makes the diagrams simpler, and retains both the number and symmetry of the regions.

A Brief Note on Algebra

- The equations of the planes are $x = y$, $x = z$, $y = z$.
- All three planes meet in the line L defined by $x = y = z$.
- Passing from 3D to 2D can be viewed either as "squashing" the arrangement along L, or as "slicing" perpendicularly to L.
- The two sides of each plane are defined by inequalities. E.g., the two sides of $y = x$ are $x < y$ and $y < x$.

Regions Between The Planes of Braid3

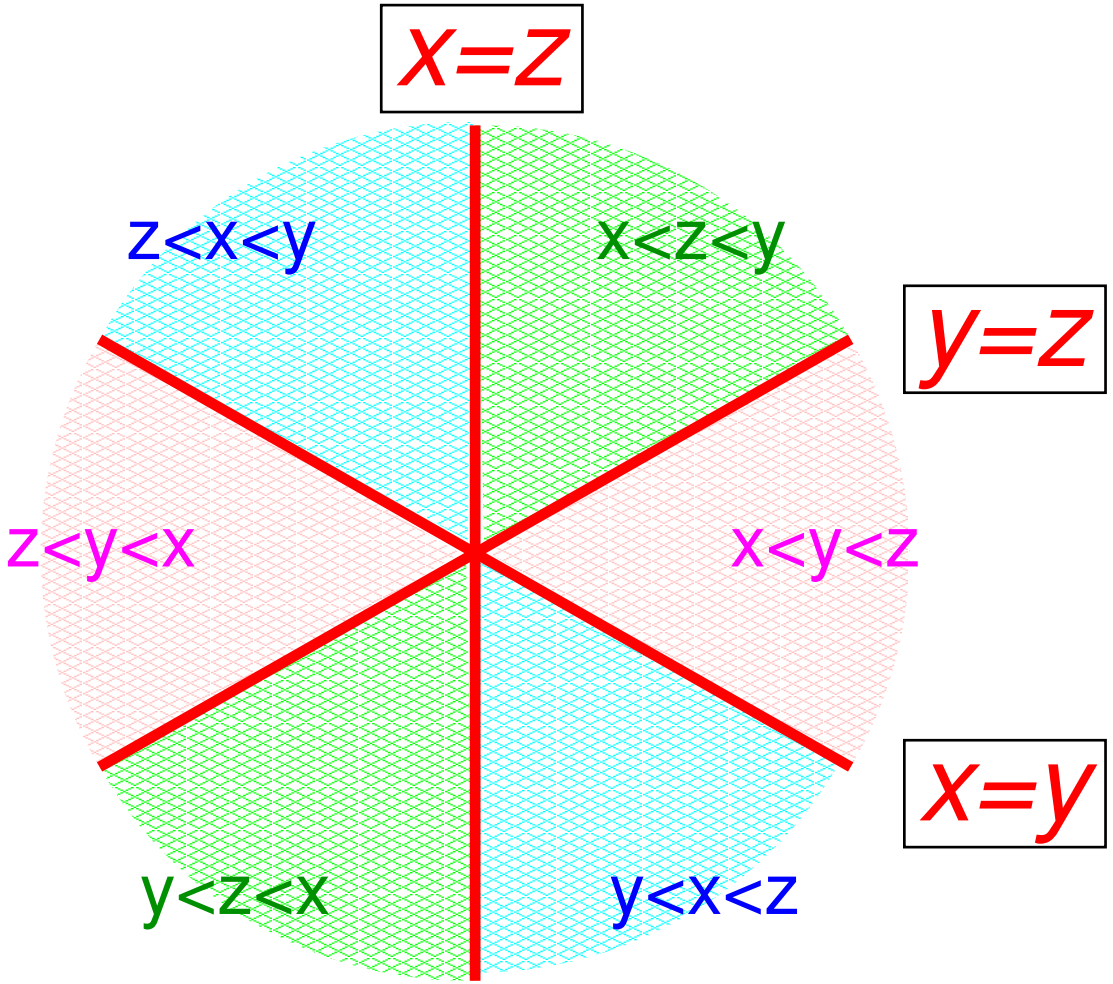
Each region of **Braid3** is described by three inequalities:

- either $x < y$ or $y < x$, and
- either $x < z$ or $z < x$, and
- either $y < z$ or $z < y$.

Therefore, which region you are in corresponds to the order of the three coordinates x, y, z. There are six possibilities:

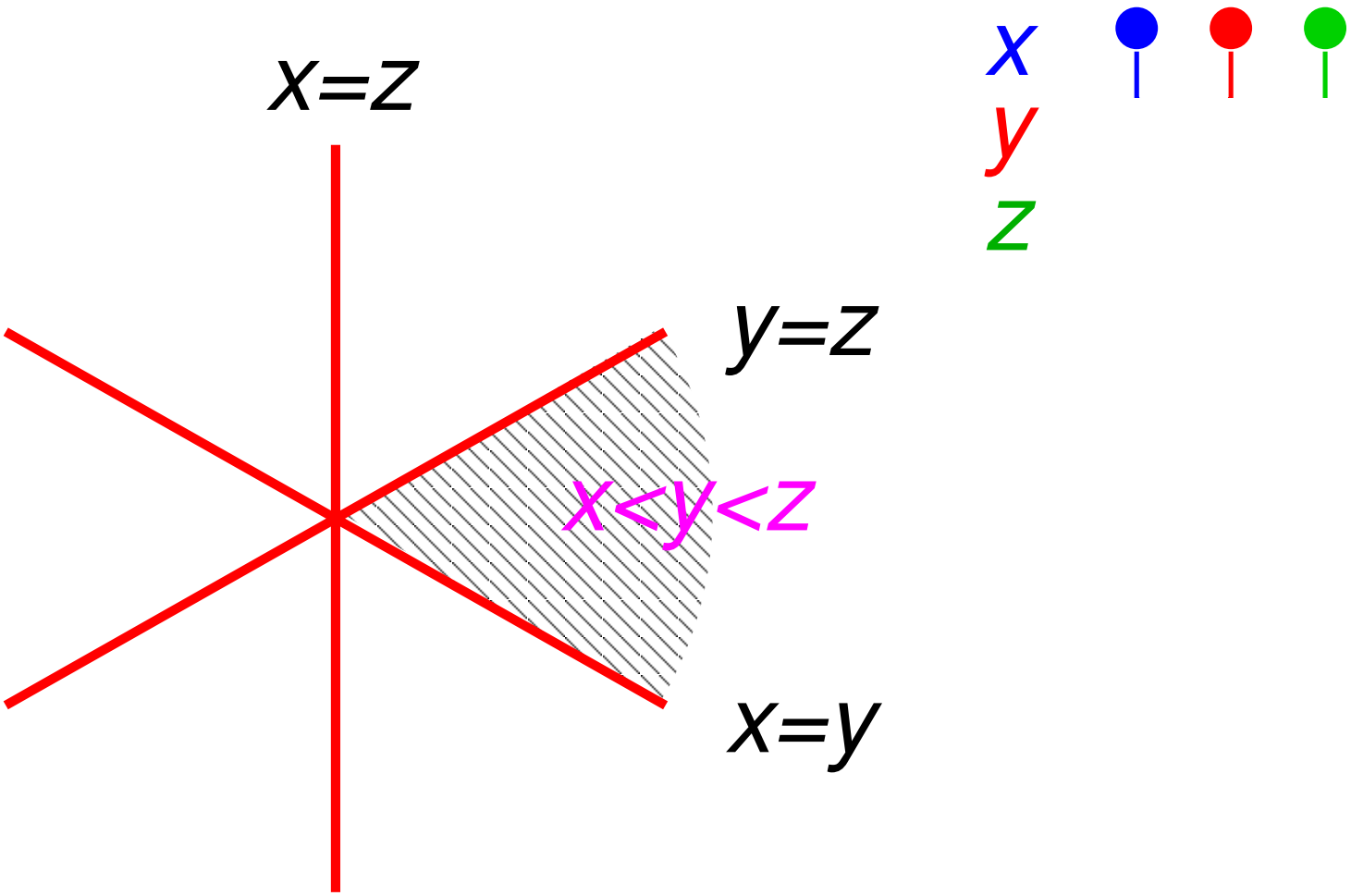
$$x < y < z \qquad y < x < z \qquad z < x < y$$
$$x < z < y \qquad y < z < x \qquad z < y < x$$

Regions of Braid3



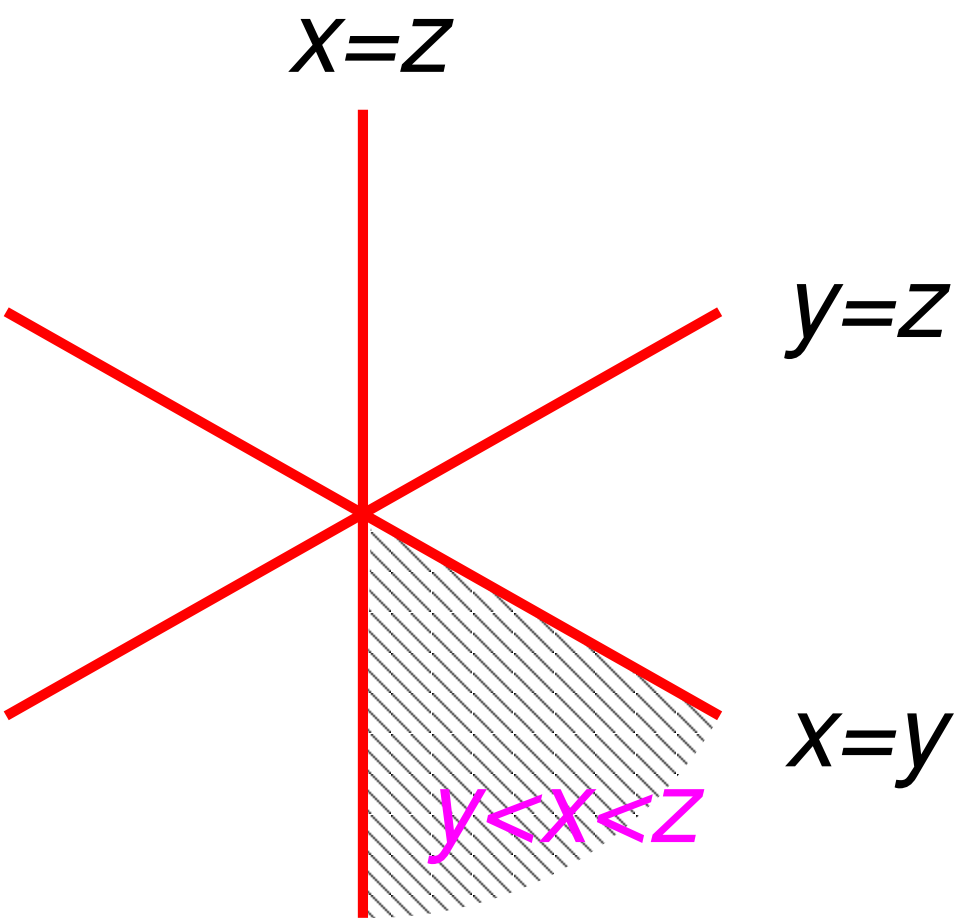
Why "Braid"?

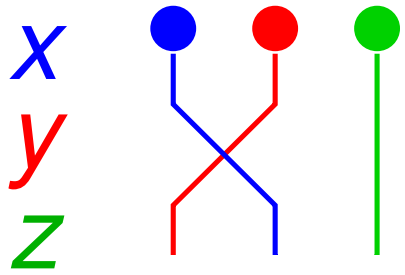
Crossing a border corresponds to reversing one inequality.



Why "Braid"?

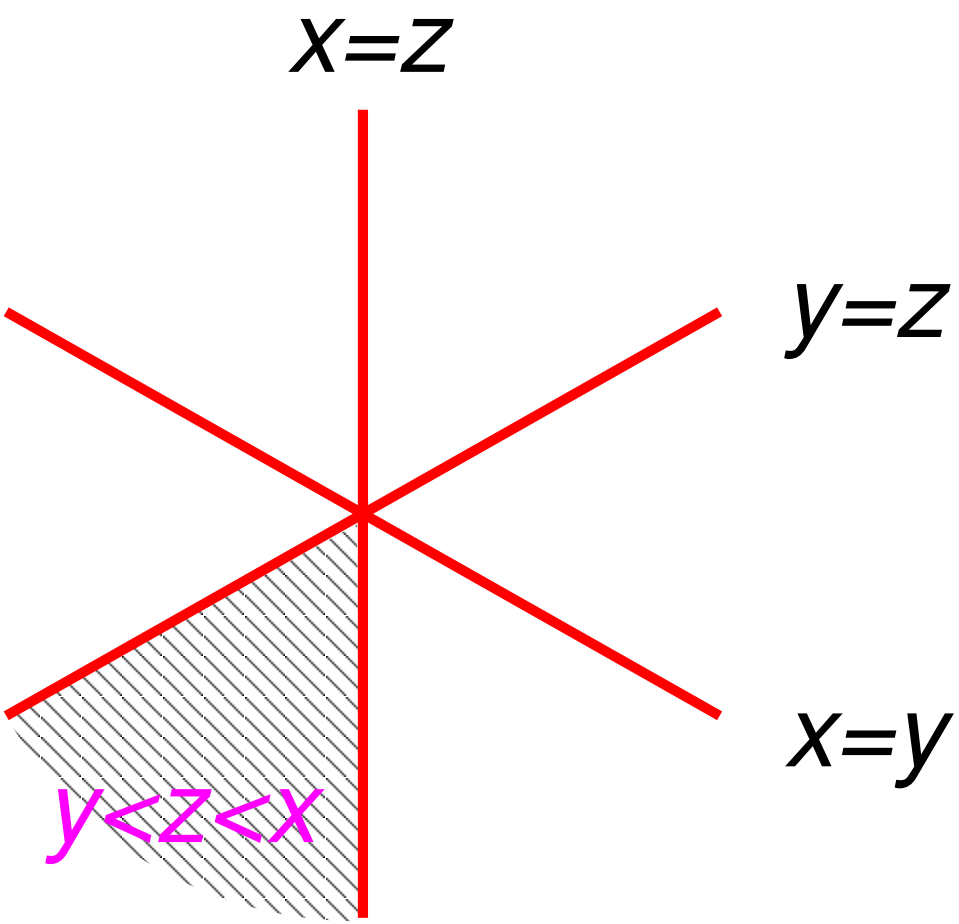
Crossing a border corresponds to reversing one inequality.

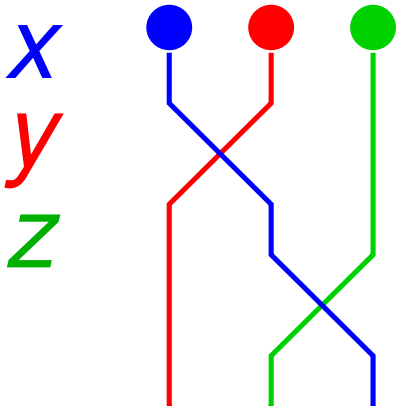




Why "Braid"?

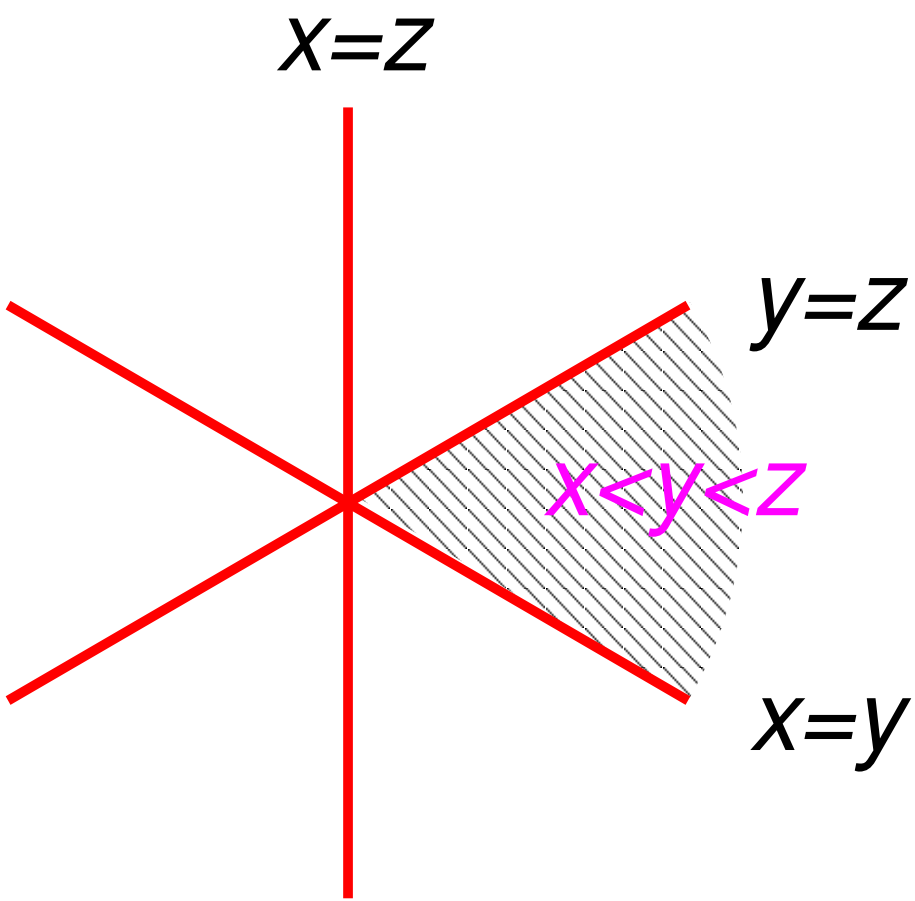
Crossing a border corresponds to reversing one inequality.

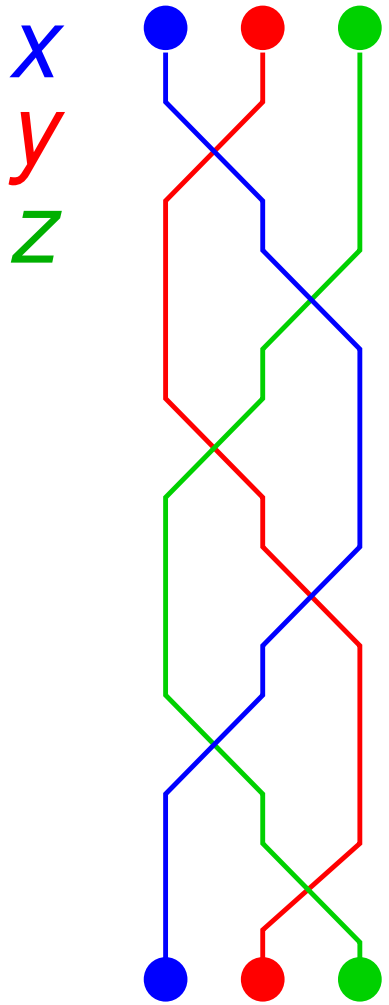




Why "Braid"?

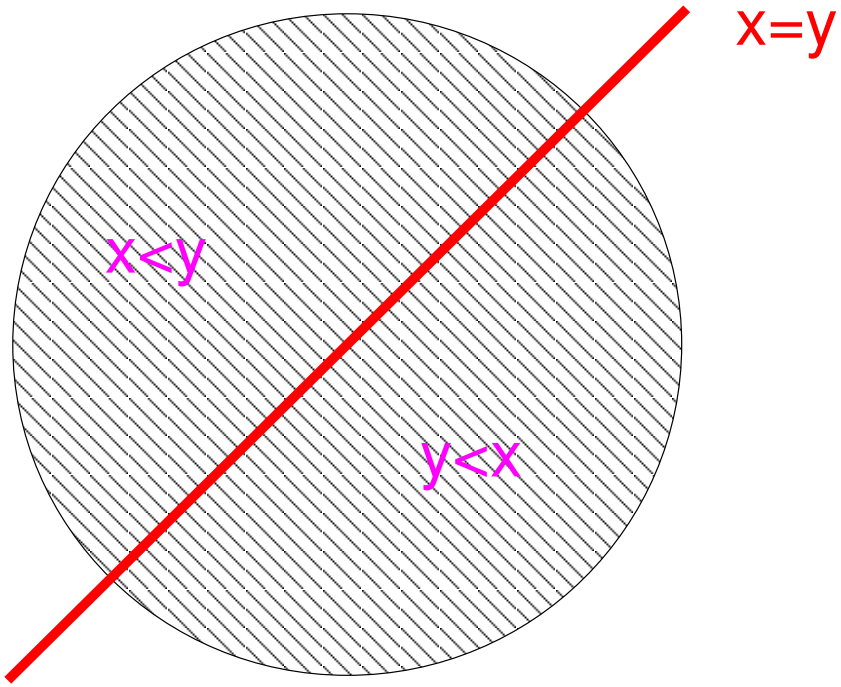
Crossing a border corresponds to reversing one inequality.



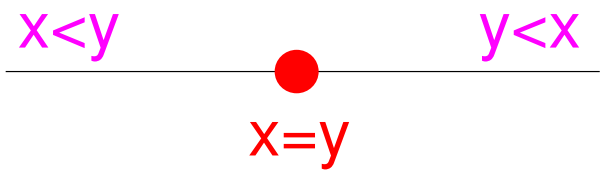


The 2-Dimensional Braid Arrangement

Before we tackle **Braid4**, let's make sure we understand **Braid2**.



Braid2



Projection into 1D

Note that there are 2 regions.

The 4-Dimensional Braid Arrangement

The arrangement **Braid4** consists of the hyperplanes defined by the equations

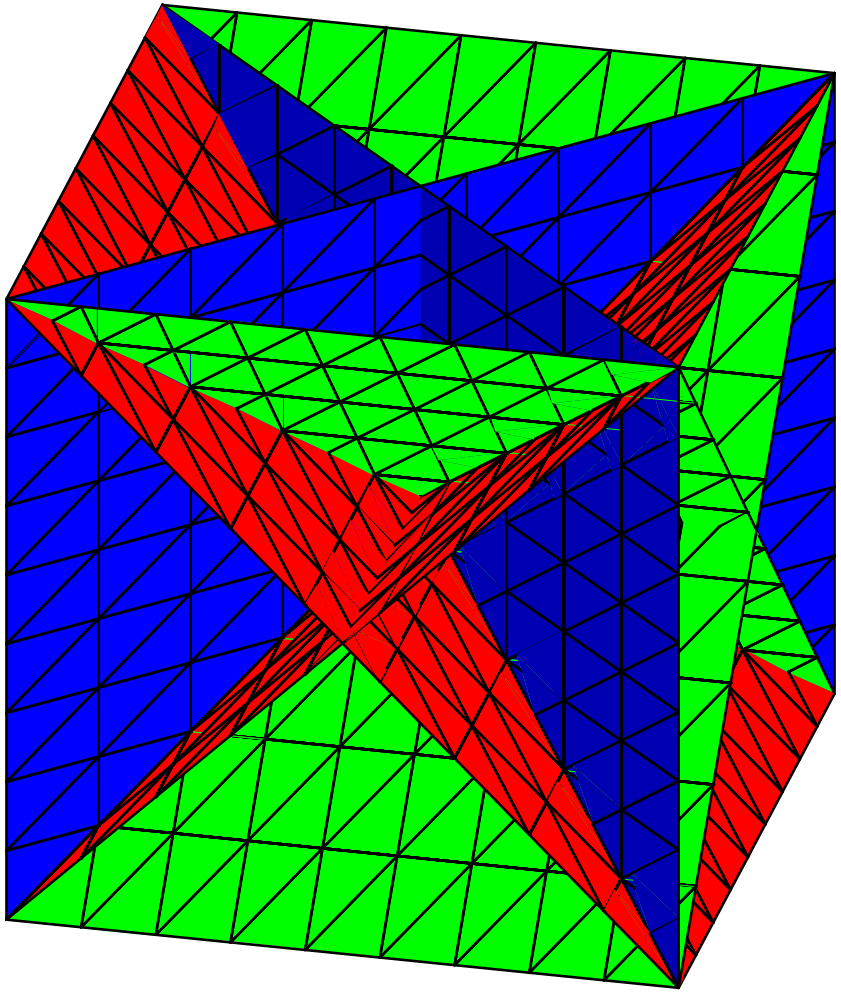
$$w = x, \quad w = y, \quad w = z, \quad x = y, \quad x = z, \quad y = z$$

in four-dimensional space.

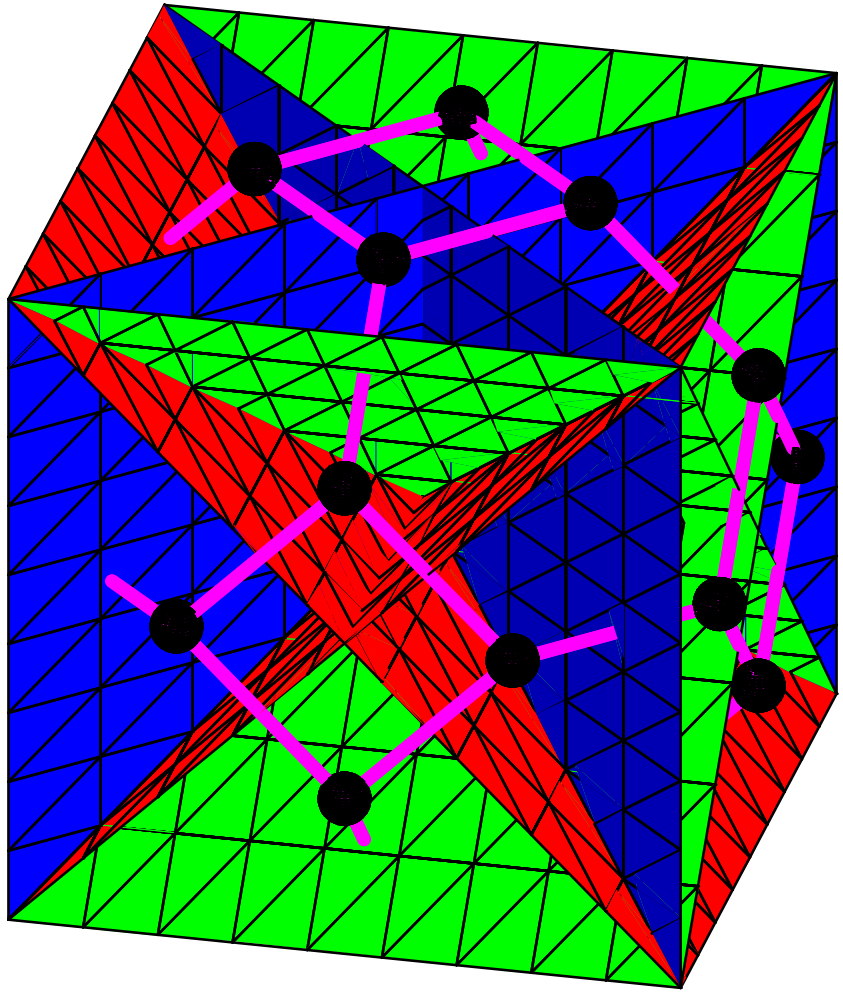
Key observation: We can project **Braid2** from 2D to 1D, and **Braid3** from 3D to 2D,

so, by analogy, we should be able to project **Braid4** from 4D to 3D!

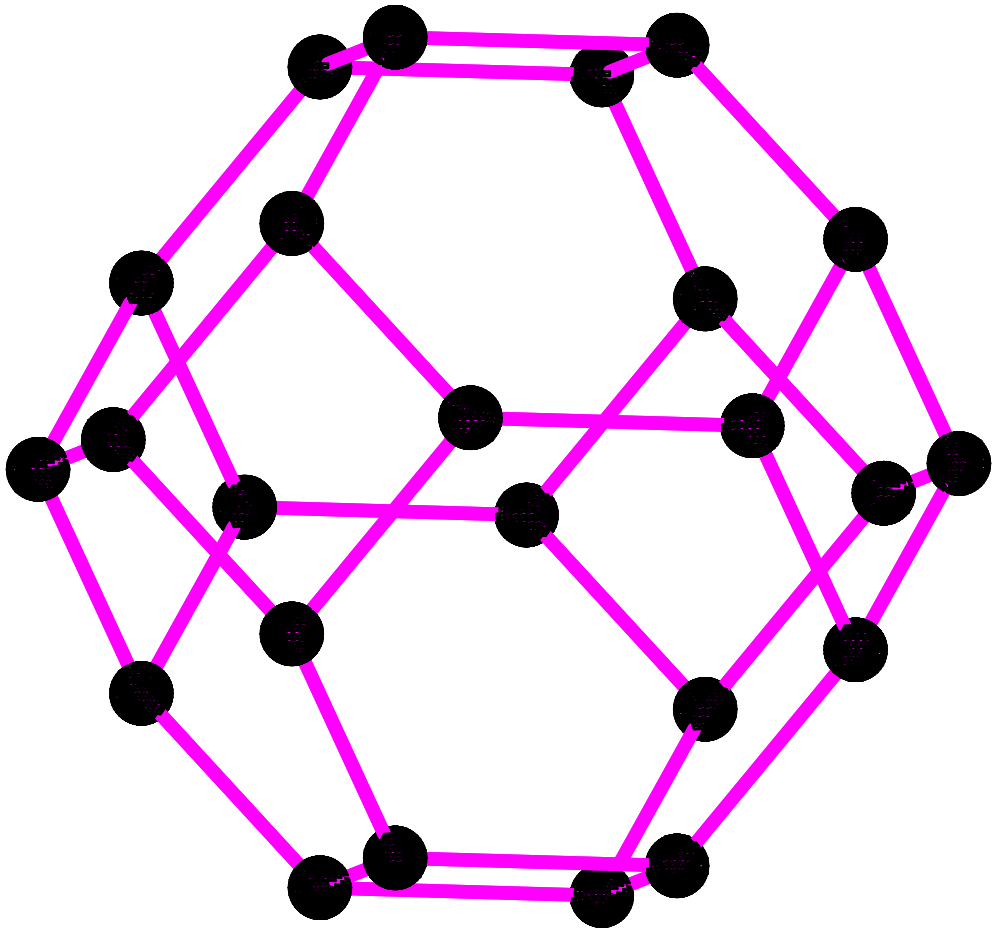
Here's what **Braid4** looks like!



Suppose we put a dot in each region and connect adjacent dots. . .



. . . and then remove the hyperplanes, leaving only the dots.



Regions of Braid4

The regions of **Braid4** correspond to the orderings of the four coordinates w, x, y, z:

wxyz	*wxzy*	*wyxz*	*wyzx*	*wzxy*	*wzyx*
xwyz	*xwzy*	*xywz*	*xyzw*	*xzwy*	*xzyw*
ywxz	*ywzx*	*yxwz*	*yxzw*	*yzwx*	*yzxw*
zwxy	*zwyx*	*zxwy*	*zxyw*	*zywx*	*zyxw*

- There are 4 possibilities for the first letter;
- 3 possibilities for the second, once the first is determined;
- 2 possibilities for the third, once the first two are determined;
- only 1 possibility for the last letter.

Total: $4 \times 3 \times 2 \times 1 =$ 24 orderings $=$ 24 regions.

Regions of Braid4

- We have just seen that **Braid4** has 24 regions.
- The regions correspond to permutations of w, x, y, z.
- If two regions are adjacent, the corresponding permutations differ by a single flip:

$$\text{x } \mathbf{z}\ \mathbf{w}\text{y} \quad \longleftrightarrow \quad \text{x } \mathbf{w}\ \mathbf{z}\text{y}$$

(In particular, each region has exactly 3 neighboring regions.)

Beyond the Fourth Dimension

The *n*-dimensional braid arrangement consists of the hyperplanes defined by the equations

$$\begin{aligned} &x_1 = x_2, \\ &x_1 = x_3, \quad x_2 = x_3, \\ &\dots \\ &x_1 = x_n, \quad x_2 = x_n, \quad \dots, \quad x_{n-1} = x_n \end{aligned}$$

- There are $n(n-1)/2$ hyperplanes (by the staircase formula!)
- The regions correspond to the possible orderings of the coordinates $x_1, \dots, x_n$.
- The number of regions is $n \times (n-1) \times \cdots \times 3 \times 2 \times 1$ (also known as *n* factorial; notation: $n!$).
- Each region has $n-1$ neighboring regions.

Parking Cars

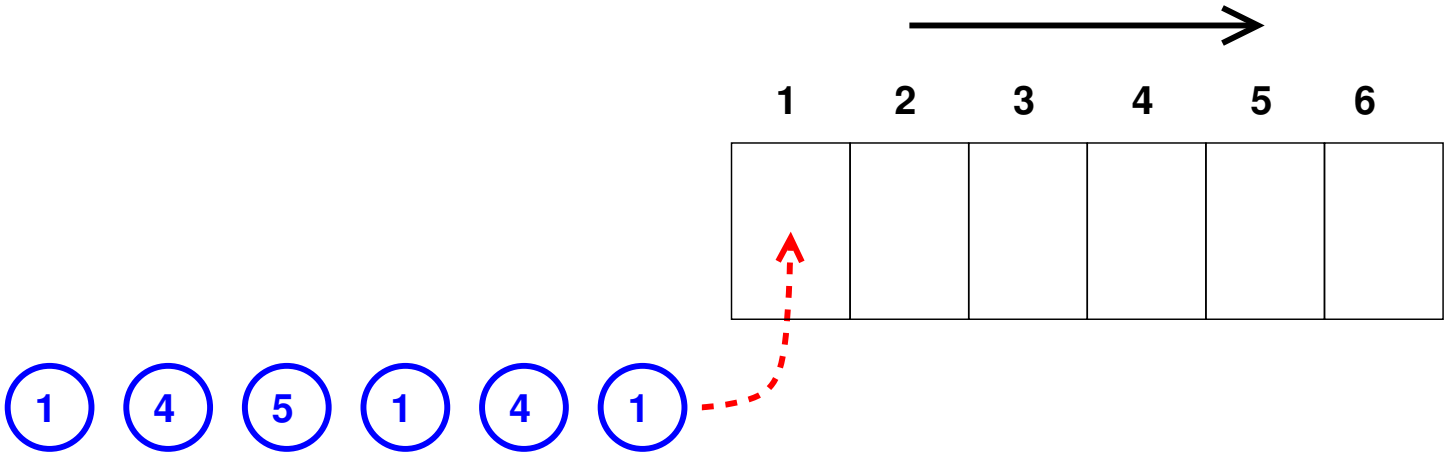
- A group of cars enter a parking lot, one by one.
- # of parking spaces = # of cars (say n).

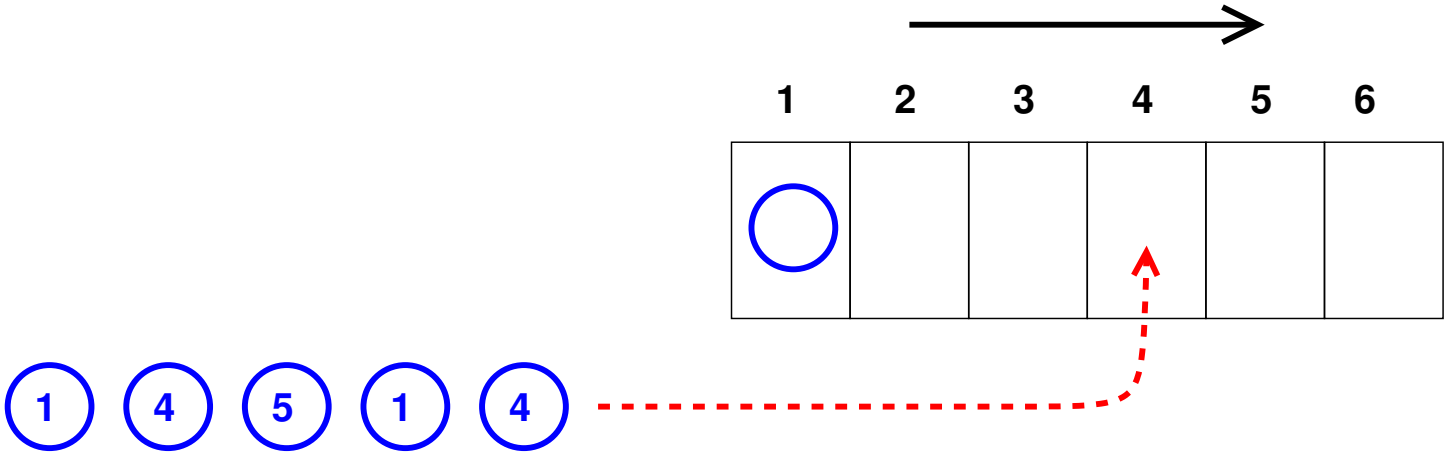
- The parking spaces are arranged along a one-way road.
- Each car has a preferred parking space that it drives to first. If that spot is not available, it continues to the first empty space.

- A "parking function" is a list of preferences that allows all cars to park.

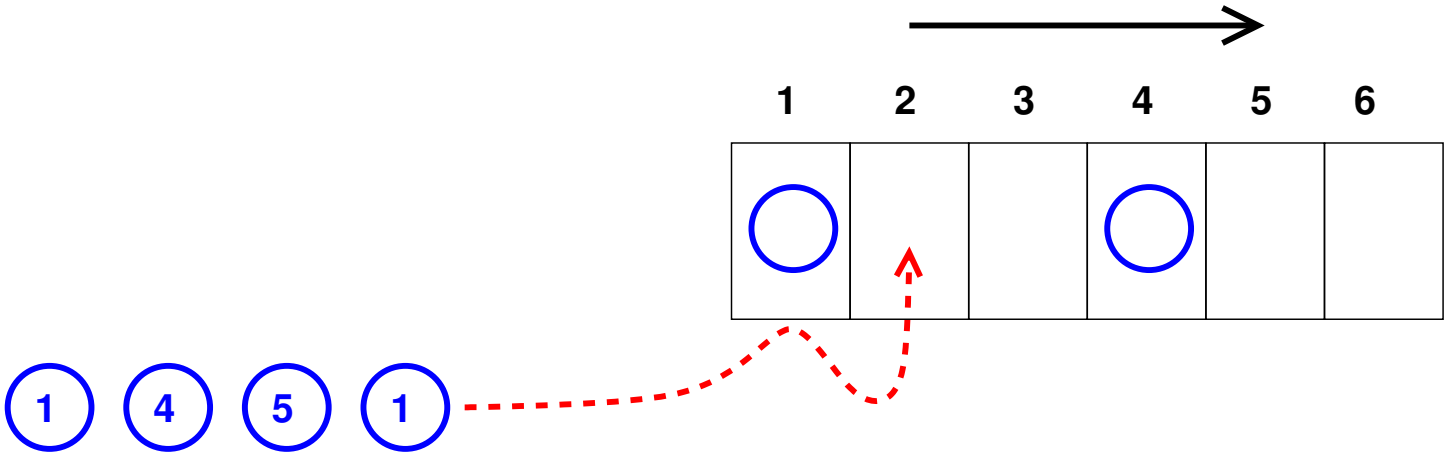
- Application: database indexing, hash tables)

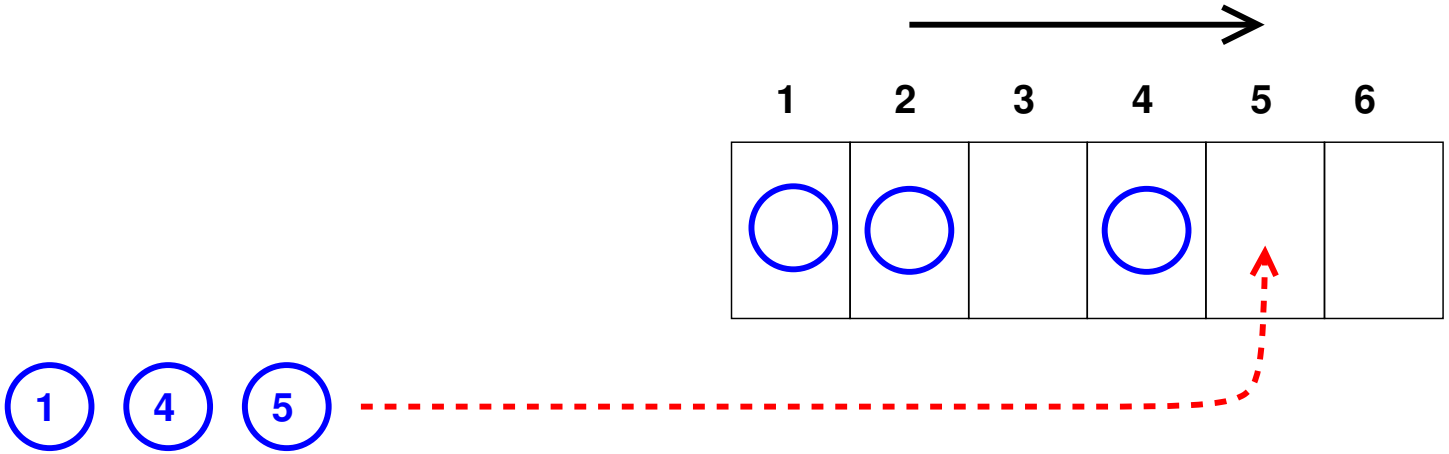
Parking Functions



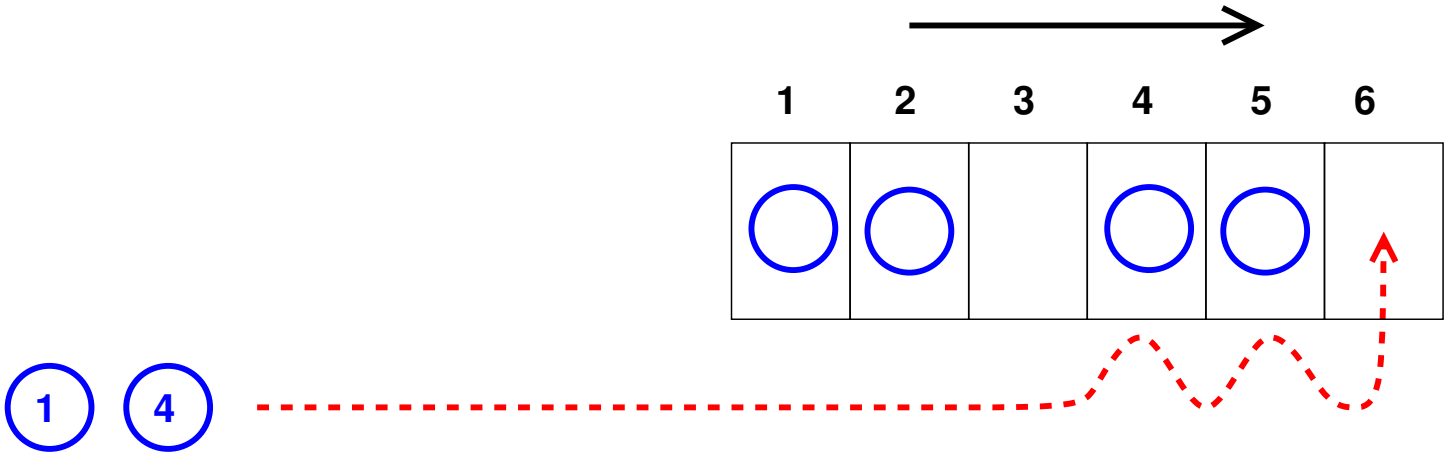


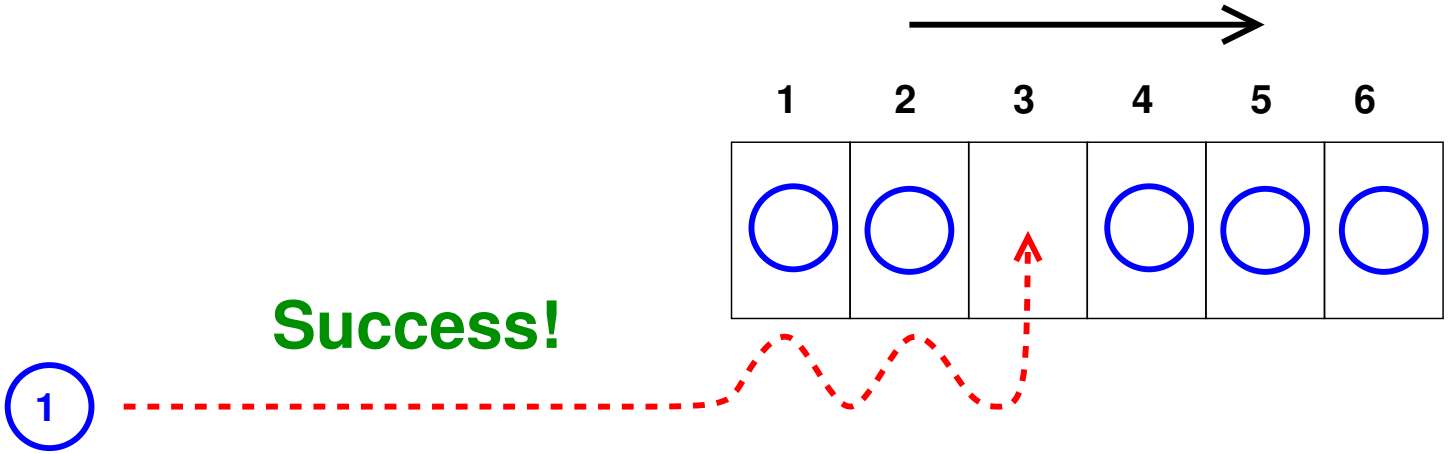
Parking Functions



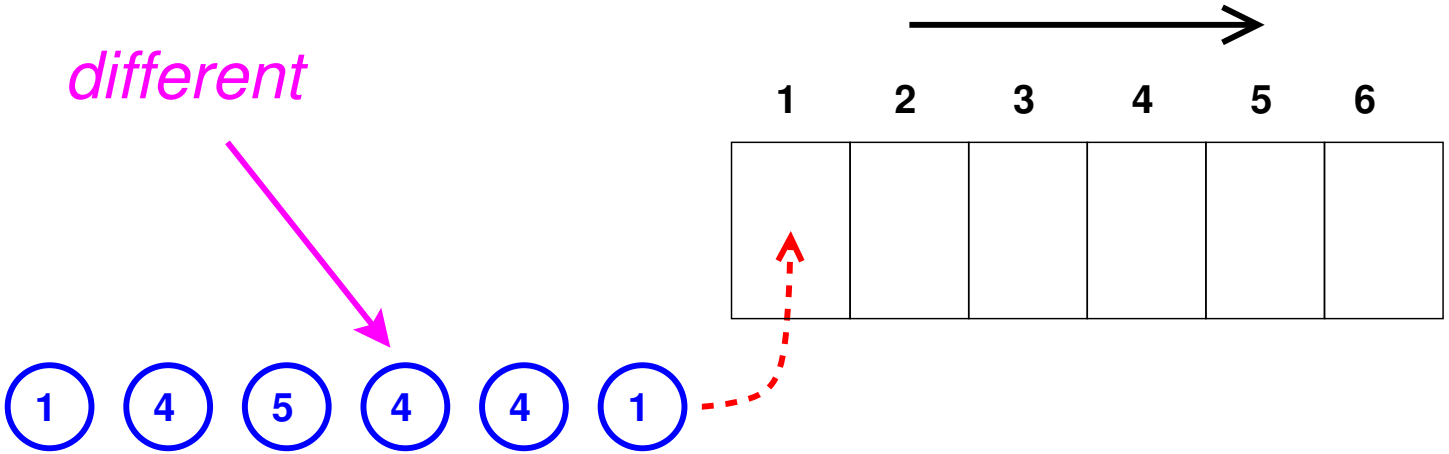


Parking Functions

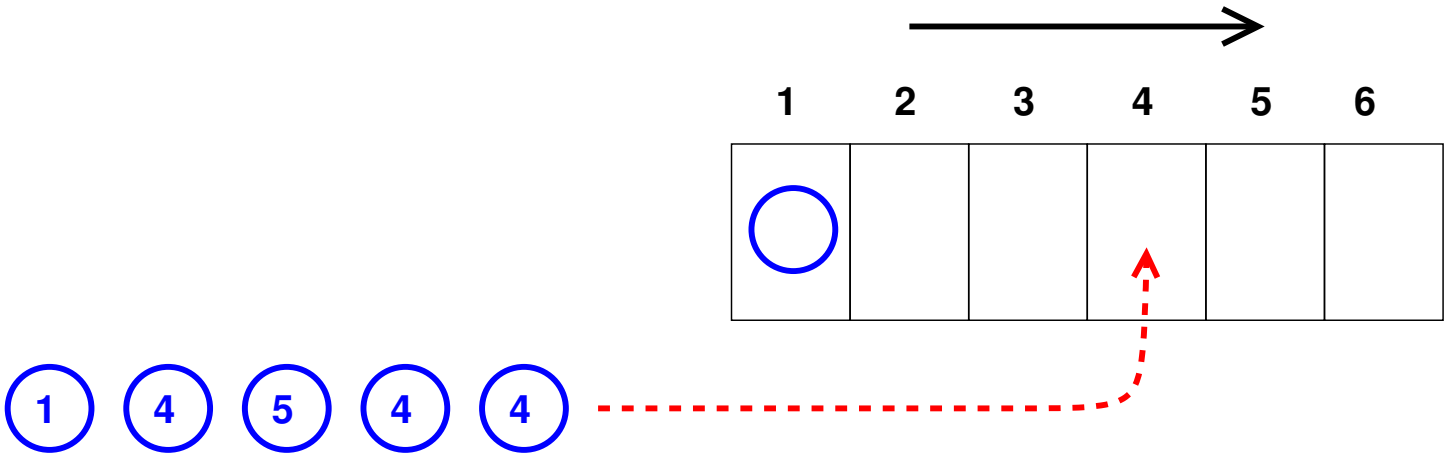


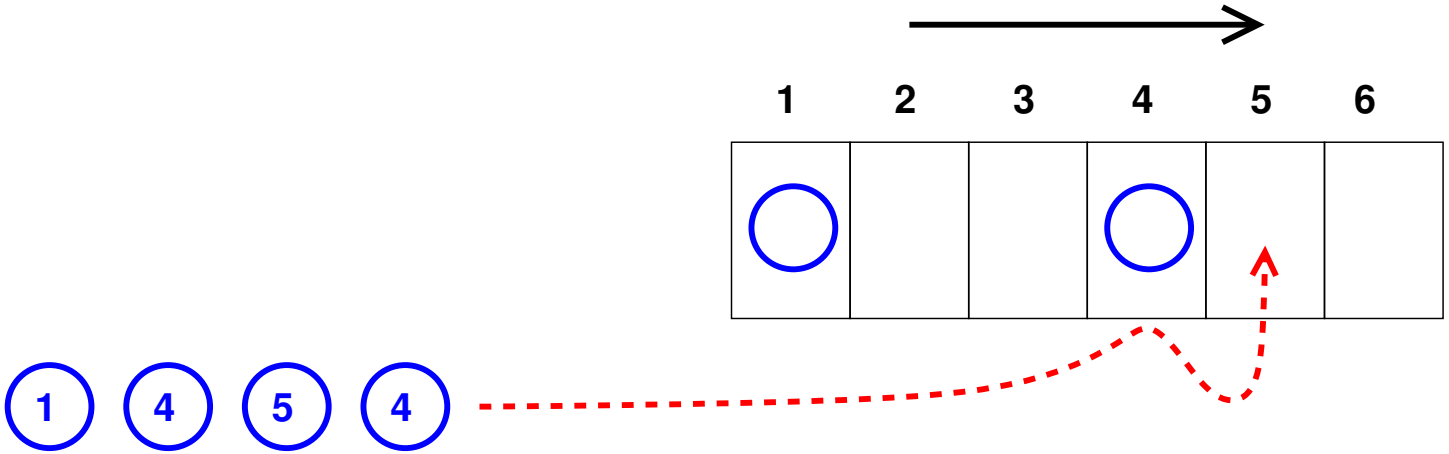


Parking Functions

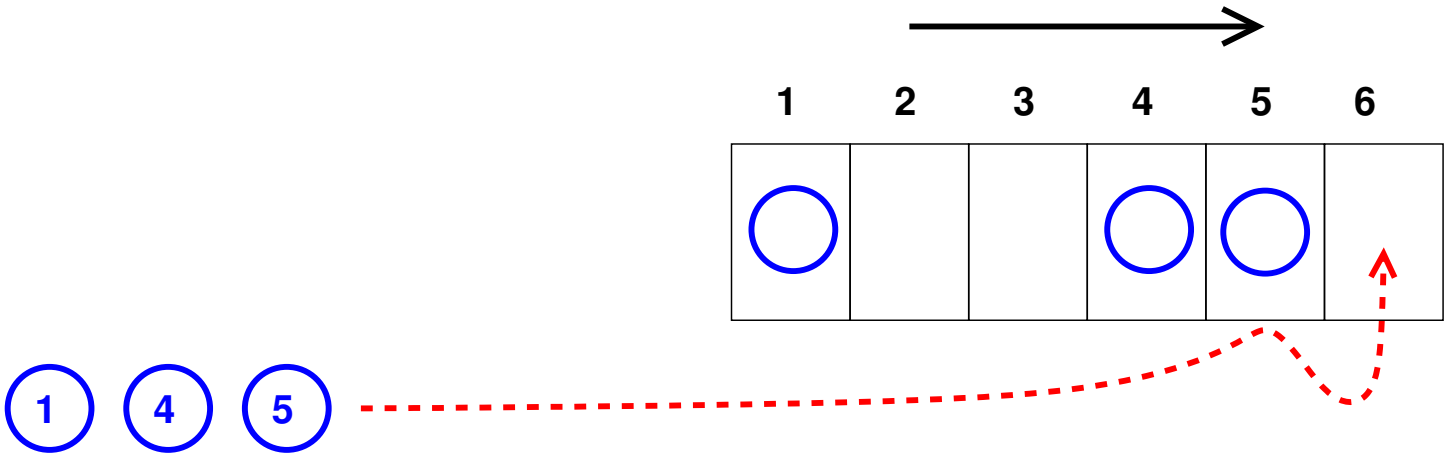


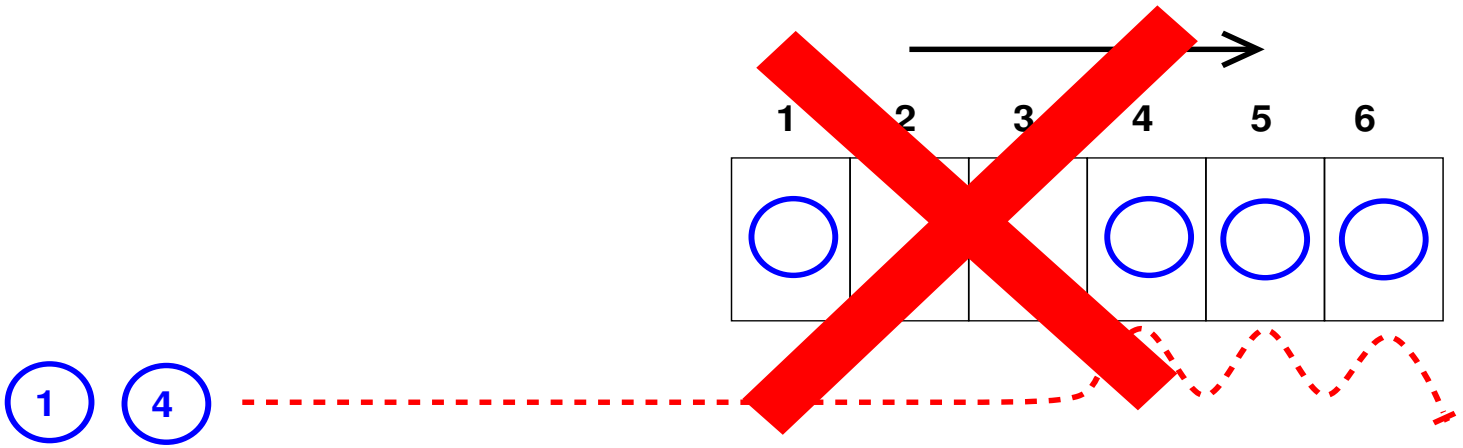
Parking Functions





Parking Functions





Parking Two Cars

There are $4 = 2^2$ possible lists of preferred spots.
3 of them successfully park both cars.

Car #1 preference	Car #2 preference	Success?
Space 1	Space 1	Yes
Space 1	Space 2	Yes
Space 2	Space 1	Yes
Space 2	Space 2	No

Parking Three Cars

There are $27 = 3^3$ possible lists of preferred spots.
16 of them successfully park all three cars.

Parking functions (the ones that work):

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

Non-parking functions (the ones that don't work):

133	222	223	233	333
313		232	323	
331		322	332	

Parking *n* Cars

Observation #1: Whether or not all the cars can park depends on what their preferred spaces are, but not on the order in which they enter the parking lot.

For example, if there are 6 cars and the preference list includes two 5's and one 6, not all cars will be able to park.

Also, every parking function must include at least one 1. (What are some other conditions that must be satisfied?)

Parking n Cars

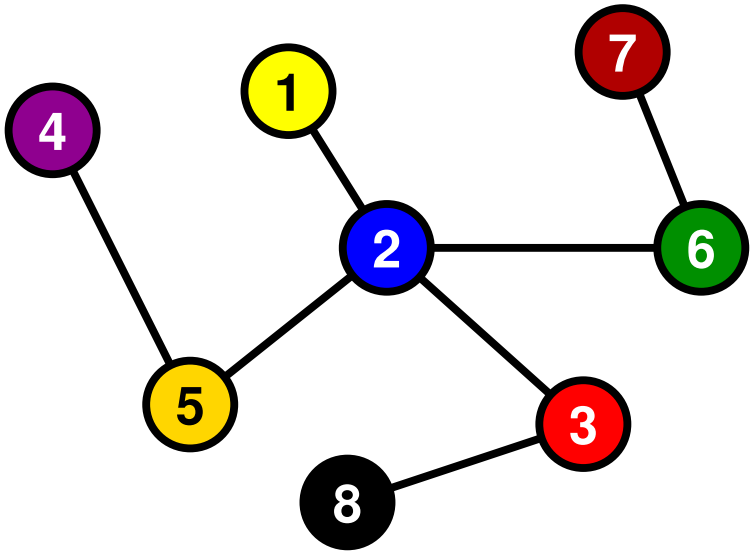
Observation #2: 3 cars $\Longrightarrow$ 16 parking functions.

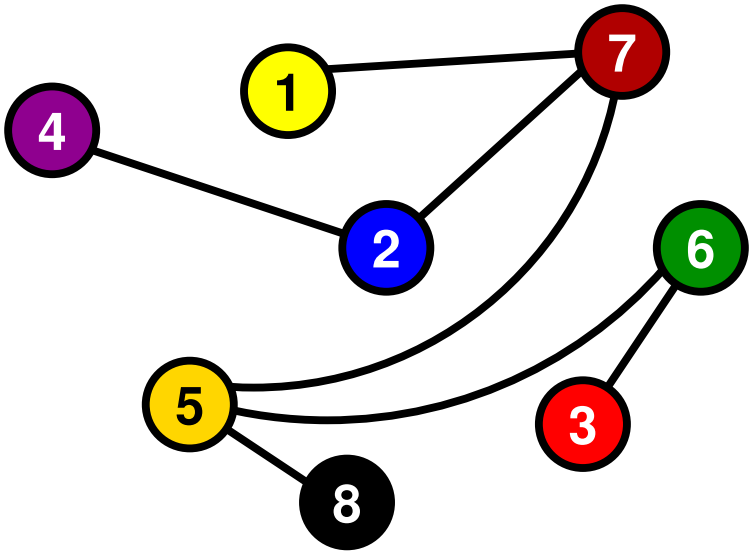
Number of cars (n)	Number of parking functions	
1	1	$= 2^0$
2	3	$= 3^1$
3	16	$= 4^2$
4	125	$= 5^3$
5	1296	$= 6^4$

Conjecture: n cars $\Longrightarrow$ $(n+1)^{n-1}$ parking functions.

Connecting Points

Problem: Connect n points with as few links as possible.

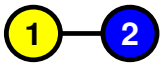




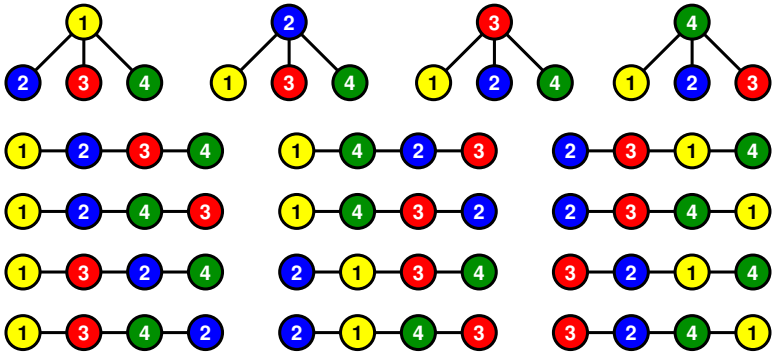
- It doesn't matter where the points are or how you draw the links — just which pairs of points are linked.
- These structures are called trees.

How Many Trees?

1 point:  1 tree

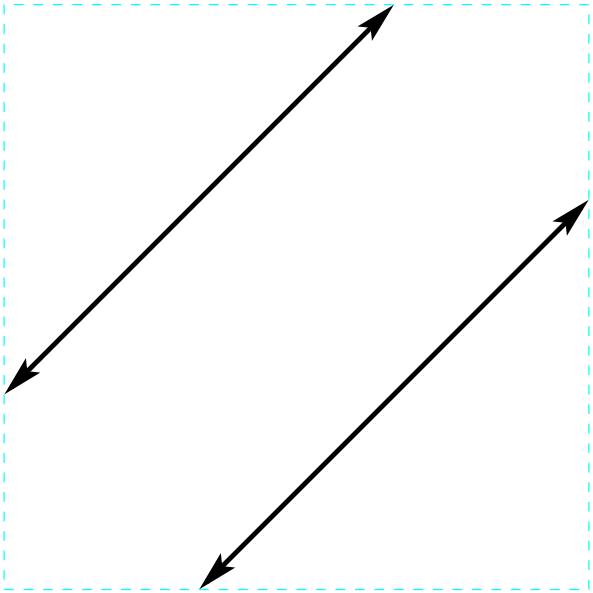
2 points:  1 tree

3 points:  3 trees

4 points:  16 trees

The Shi Arrangement

- Draw two parallel lines in 2-dimensional space.



- There are three regions ("above", "between", "below").
- Equations of the lines: $x = y$ and $x = y + 1$.

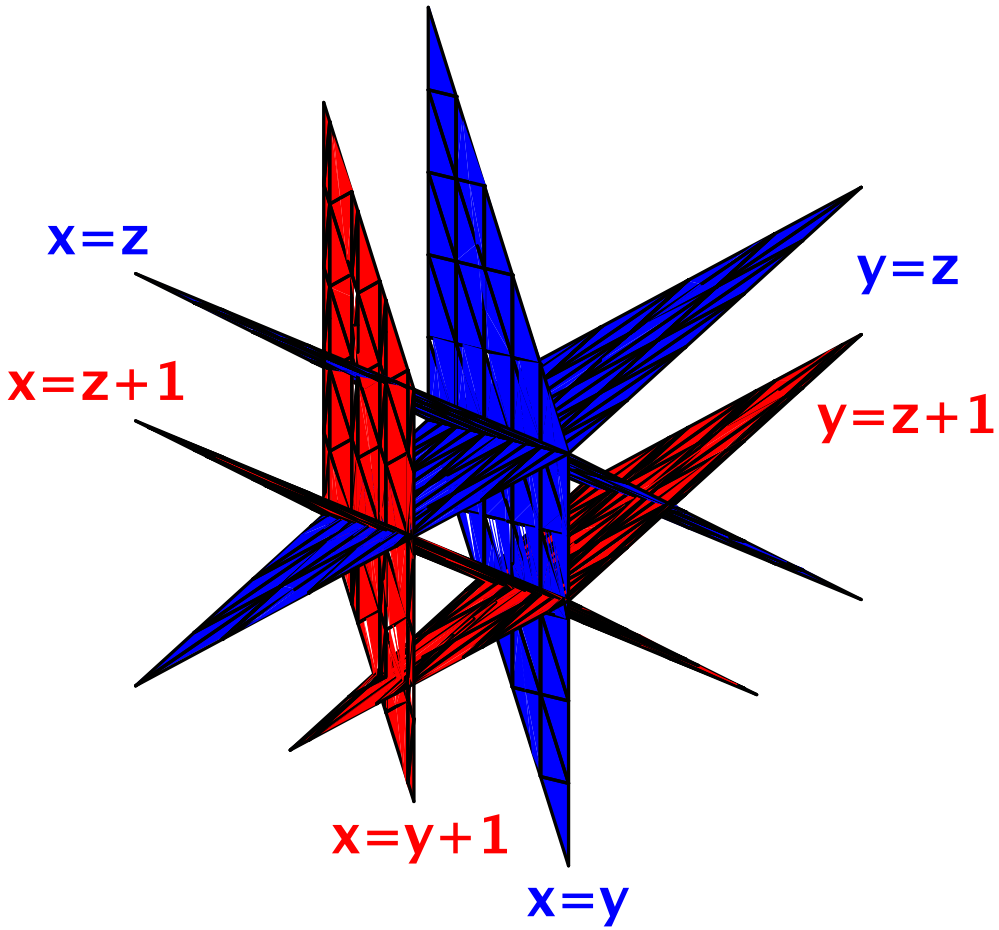
The Shi Arrangement

The *n*-dimensional Shi arrangement consists of the $n(n-1)$ hyperplanes defined by the equations

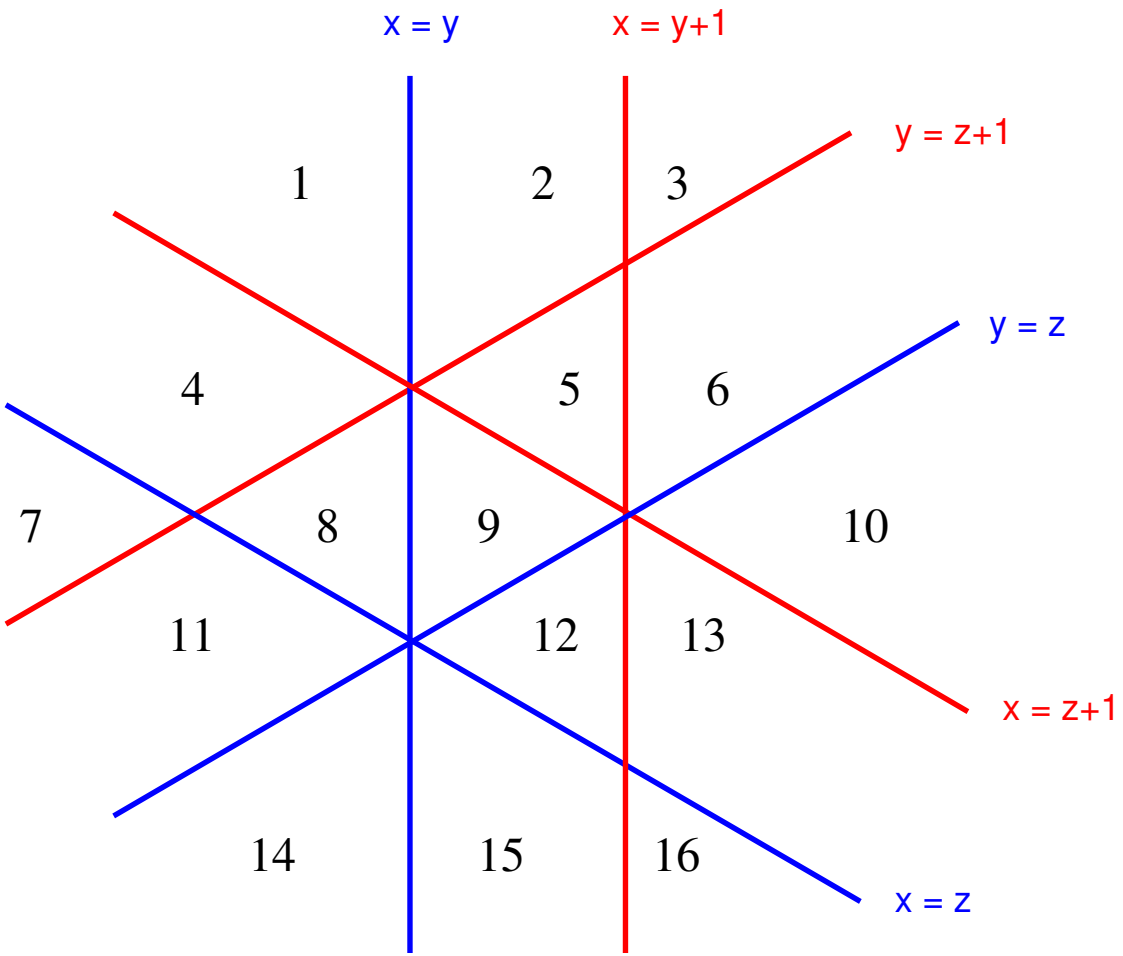
$$\begin{aligned} x_1 &= x_2, & x_1 &= x_2 + 1, \\ x_1 &= x_3, & x_1 &= x_3 + 1, \\ &\dots \\ x_{n-1} &= x_n, & x_{n-1} &= x_n. \end{aligned}$$

("Take the braid arrangement, make a copy of it, and push the copy a little bit.")

The 3D Shi Arrangement



The 3D Shi Arrangement



Scoring with a Handicap

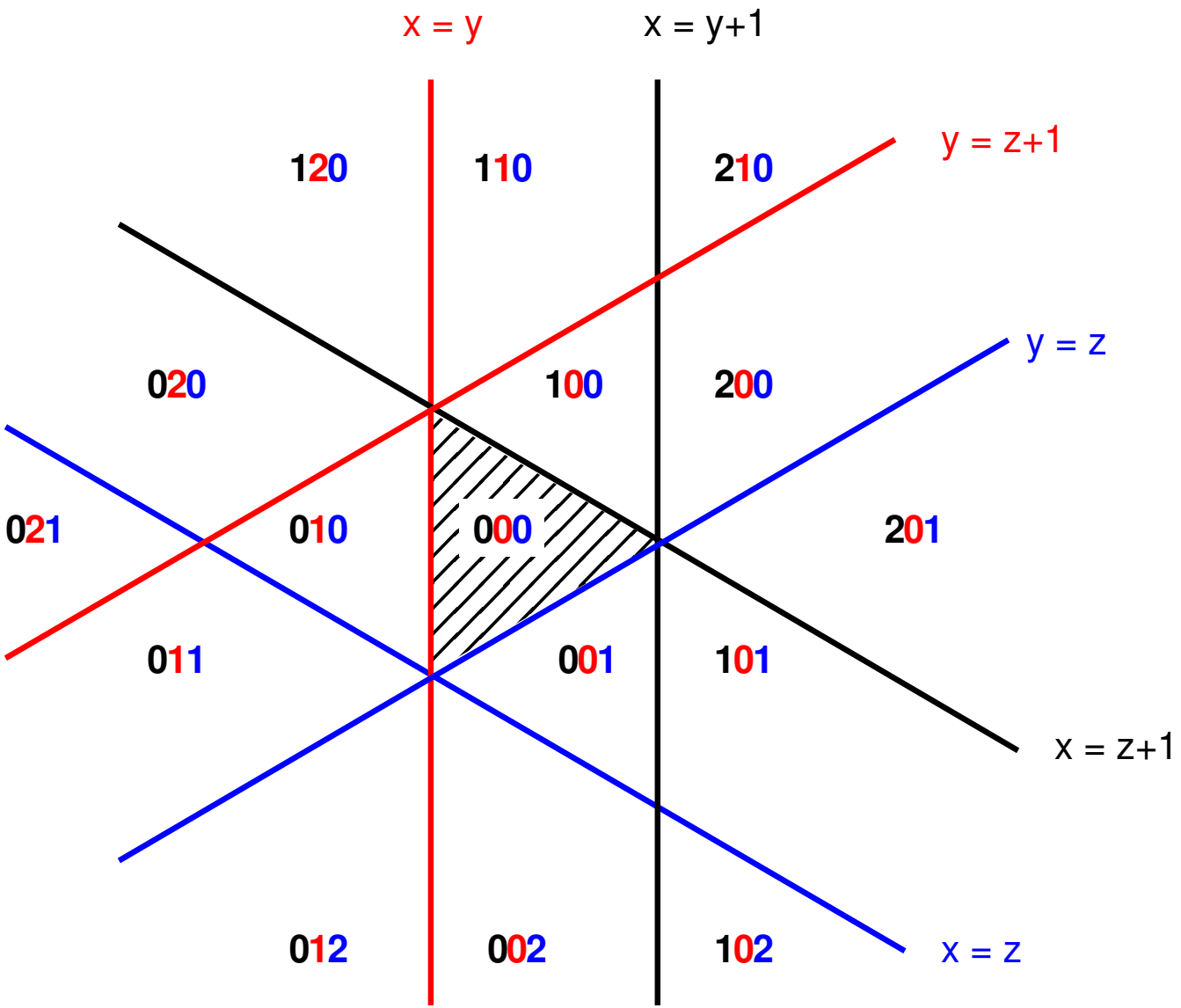
- A group of marathon runners are ranked 1 through n.
- You score one point for each other runner you beat head-to-head.
- But, in order to score a point against a lower-ranked runner, you must beat him/her by at least one minute.
- The possible outcomes correspond to regions of the Shi arrangement!

Slicing n-Dimensional Space

$$
\begin{aligned}
(n+1)^{n-1} &= \text{number of regions of the Shi arrangement} \\
&= \text{number of handicapped-scoring outcomes} \\
&= \text{number of trees on } n+1 \text{ points} \\
&= \text{number of ways to park } n \text{ cars}
\end{aligned}
$$

Why are all these numbers the same?

The next figure shows the correspondence between Shi-arrangement regions and parking functions for $n = 3$.


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