Permutations arising from the Collatz-conjecture and automatic discovery of patterns

MIT Combinatorics Seminar

Henning Ulfarsson

Reykjavik University

October 23, 2013

Table of Contents

First four sections are joint work with Michael Albert (Otago) and Bjarki Gudmundsson (Reykjavik)

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d
8	4	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d
8	4	d
4	2	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod 2 \\ 3x+1 & \text{if } x \equiv 1 \pmod 2 \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d
8	4	d
4	2	d
2	1	d

The Collatz process

$$f(x) = \begin{cases} x/2 & \text{if } x \equiv 0 \pmod{2} \\ 3x+1 & \text{if } x \equiv 1 \pmod{2} \end{cases}$$

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d
8	4	d
4	2	d
2	1	d
1		

The Collatz conjecture

The conjecture

The Collatz process ends in 1 for any starting number

The Collatz conjecture

The conjecture

The Collatz process ends in 1 for any starting number

- First stated by Lothar Collatz in 1937
- Many have tried, but the conjecture remains open
- Starting numbers up to $5 \times 2^{60} \approx 5.764 \times 10^{18}$ have been verified

The initial permutation

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5	16	u
16	8	d
8	4	d
4	2	d
2	1	d
1		

Discard powers of 2.

The initial permutation

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5		

Discard powers of 2.

The initial permutation

x	$f(x)$	step
12	6	d
6	3	d
3	10	u
10	5	d
5		

Discard powers of 2. The remaining numbers are distinct so we can flatten them to a permutation

The initial permutation

$$12 \quad 6 \quad 3 \quad 10 \quad 5$$

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

The initial permutation

$$\begin{matrix} 12 & 6 & 3 & 10 & 5 \\ & & 1 & & \end{matrix}$$

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

The initial permutation

$$\begin{array}{ccccc} 12 & 6 & 3 & 10 & 5 \\ & & 1 & & 2 \end{array}$$

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

The initial permutation

12	6	3	10	5
	3	1		2

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

The initial permutation

$$\begin{array}{ccccc} 12 & 6 & 3 & 10 & 5 \\ & 3 & 1 & 4 & 2 \end{array}$$

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

The initial permutation

$$\begin{array}{ccccc} 12 & 6 & 3 & 10 & 5 \\ 5 & 3 & 1 & 4 & 2 \end{array}$$

Put 1 instead of the smallest number (3), 2 instead of the next smallest (5), etc.

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8
7	13

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34
10	55

How many?

By running computer tests we get the following data, which we can think of as a lower bound on the correct enumeration

length	#perms
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34
10	55
11	89

Possible operation sequences

Why Fibonacci numbers?

Possible operation sequences

Why Fibonacci numbers?

- A typical operation sequence $duddudud$

Possible operation sequences

Why Fibonacci numbers?

- A typical operation sequence $duddudud$
- They never contain uu, since after u we get an even number, so next comes a d-step

Possible operation sequences

Why Fibonacci numbers?

- A typical operation sequence $duddudud$
- They never contain uu, since after u we get an even number, so next comes a d-step
- They always end with d because we cut the tail off (the last u-step into the tail)

Possible operation sequences

Why Fibonacci numbers?

- A typical operation sequence $duddudud$
- They never contain uu, since after u we get an even number, so next comes a d-step
- They always end with d because we cut the tail off (the last u-step into the tail)
- So there are $\mathrm{fib}(n+1)$ many possible operation sequences of length n

Possible operation sequences

Why Fibonacci numbers?

- A typical operation sequence $duddudud$
- They never contain uu, since after u we get an even number, so next comes a d-step
- They always end with d because we cut the tail off (the last u-step into the tail)
- So there are $\mathrm{fib}(n+1)$ many possible operation sequences of length n

Now we need to show that each operation sequence is witnessed by some starting number

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

$$U(X) = (X - 1)/3$$

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

$$U(X) = (X - 1)/3$$
$$DU(X) = (2X - 2)/3$$

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

$$U(X) = (X - 1)/3$$
$$DU(X) = (2X - 2)/3$$
$$UDU(X) = (2X - 5)/9$$

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

$$U(X) = (X - 1)/3$$
$$DU(X) = (2X - 2)/3$$
$$UDU(X) = (2X - 5)/9$$
$$DUDU(X) = (4X - 10)/9$$
$$UDUDU(X) = (4X - 19)/27$$
$$DUDUDU(X) = (8X - 38)/27$$
$$DDUDUDU(X) = (16X - 76)/27$$
$$UDDUDUDU(X) = (16X - 103)/81$$
$$DUDDUDUDU(X) = (32X - 206)/81$$

Looking from the tail

Let $X = 2^x$ be the number we hit in the tail. Let $D = d^{-1}$, $U = u^{-1}$ and consider the operation sequence $duddudud$:

$$U(X) = (X - 1)/3$$
$$DU(X) = (2X - 2)/3$$
$$UDU(X) = (2X - 5)/9$$
$$DUDU(X) = (4X - 10)/9$$
$$UDUDU(X) = (4X - 19)/27$$
$$DUDUDU(X) = (8X - 38)/27$$
$$DDUDUDU(X) = (16X - 76)/27$$
$$UDDUDUDU(X) = (16X - 103)/81$$
$$DUDDUDUDU(X) = (32X - 206)/81$$

X needs to satisfy $32X = 206 \bmod 81$, or $X = 52 \bmod 81$

The modular requirement

- We had $X = 2^x$ so we get $2^x = 52 \bmod 81$

The modular requirement

- We had $X = 2^x$ so we get $2^x = 52 \mod 81$
- In general we get an equation of the form $2^x = c \mod 3^k$ which always has a solution since 2 is a primitive root modulo 3^k for all k (k is the number of u's in the operation sequence)

The modular requirement

- We had $X = 2^x$ so we get $2^x = 52 \mod 81$
- In general we get an equation of the form $2^x = c \mod 3^k$ which always has a solution since 2 is a primitive root modulo 3^k for all k (k is the number of u's in the operation sequence)

Therefore every possible operation sequence is witnessed by some starting number so each one gives rise to one permutation

The modular requirement

- We had $X = 2^x$ so we get $2^x = 52 \bmod 81$
- In general we get an equation of the form $2^x = c \bmod 3^k$ which always has a solution since 2 is a primitive root modulo 3^k for all k (k is the number of u's in the operation sequence)

Therefore every possible operation sequence is witnessed by some starting number so each one gives rise to one permutation, ...

Longer permutations

length n	#perms	$\text{fib}(n)$
10	55	55
11	89	89

Longer permutations

length n	#perms	$\text{fib}(n)$
10	55	55
11	89	89
12	144	144

Longer permutations

length n	#perms	$\mathrm{fib}(n)$
10	55	55
11	89	89
12	144	144
13	233	233

Longer permutations

length n	#perms	$\text{fib}(n)$
10	55	55
11	89	89
12	144	144
13	233	233
14	377	377

Longer permutations

length n	#perms	$\text{fib}(n)$
10	55	55
11	89	89
12	144	144
13	233	233
14	377	377
15	611	610

Longer permutations

length n	#perms	$\text{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1

Longer permutations

length n	#perms	$\text{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2

Longer permutations

length n	#perms	$\text{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3

Longer permutations

length n	#perms	$\mathrm{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3

Longer permutations

length n	#perms	$\mathrm{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3
19	4185	4181	4

Longer permutations

length n	#perms	$\mathrm{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3
19	4185	4181	4
20	6771	6765	6

Longer permutations

length n	#perms	$\mathrm{fib}(n)$	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3
19	4185	4181	4
20	6771	6765	6
21	10953	10946	7

Longer permutations

length n	#perms	fib(n)	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3
19	4185	4181	4
20	6771	6765	6
21	10953	10946	7
22	17720	17711	9

Longer permutations

length n	#perms	fib(n)	excess
10	55	55	
11	89	89	
12	144	144	
13	233	233	
14	377	377	
15	611	610	1
16	989	987	2
17	1600	1597	3
18	2587	2584	3
19	4185	4181	4
20	6771	6765	6
21	10953	10946	7
22	17720	17711	9
23	28669	28657	12

The Lines

Consider again

$$U(X) = (X-1)/3$$
$$DU(X) = (2X-2)/3$$
$$UDU(X) = (2X-5)/9$$
$$DUDU(X) = (4X-10)/9$$
$$UDUDU(X) = (4X-19)/27$$
$$DUDUDU(X) = (8X-38)/27$$
$$DDUDUDU(X) = (16X-76)/27$$
$$UDDUDUDU(X) = (16X-103)/81$$
$$DUDDUDUDU(X) = (32X-206)/81$$

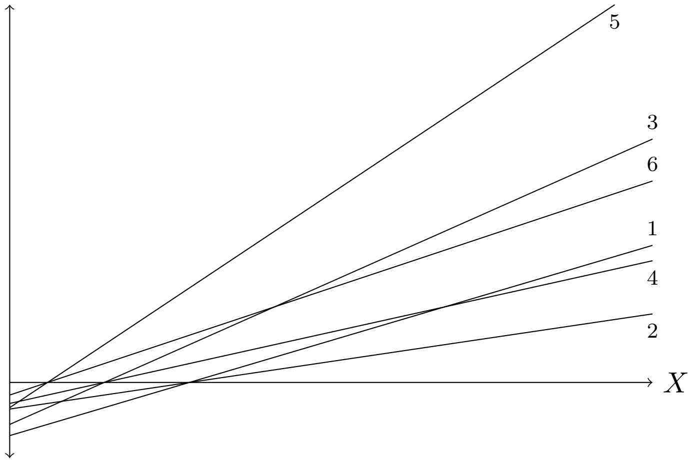
The Lines

Consider again

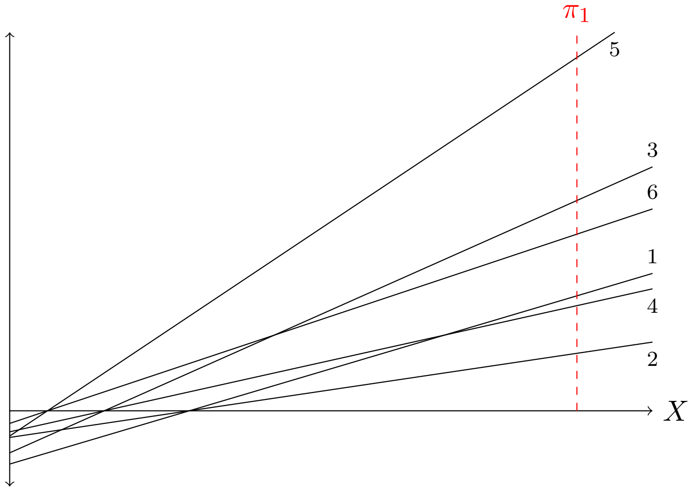
$$U(X) = (X-1)/3$$
$$DU(X) = (2X-2)/3$$
$$UDU(X) = (2X-5)/9$$
$$DUDU(X) = (4X-10)/9$$
$$UDUDU(X) = (4X-19)/27$$
$$DUDUDU(X) = (8X-38)/27$$
$$DDUDUDU(X) = (16X-76)/27$$
$$UDDUDUDU(X) = (16X-103)/81$$
$$DUDDUDUDU(X) = (32X-206)/81$$

These are equations for lines

The Lines

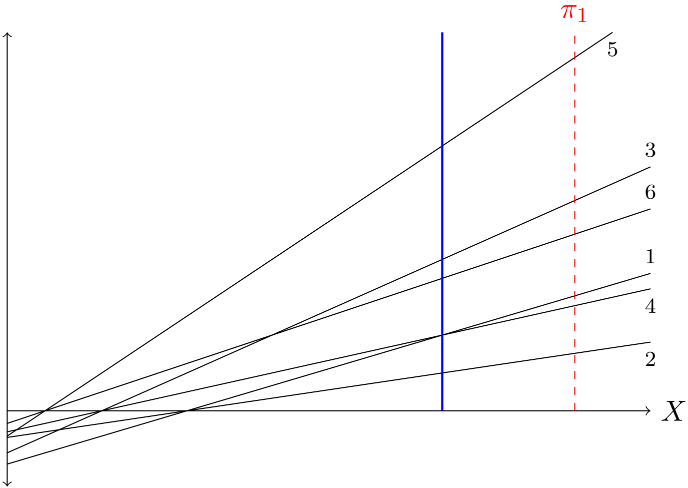


The Lines



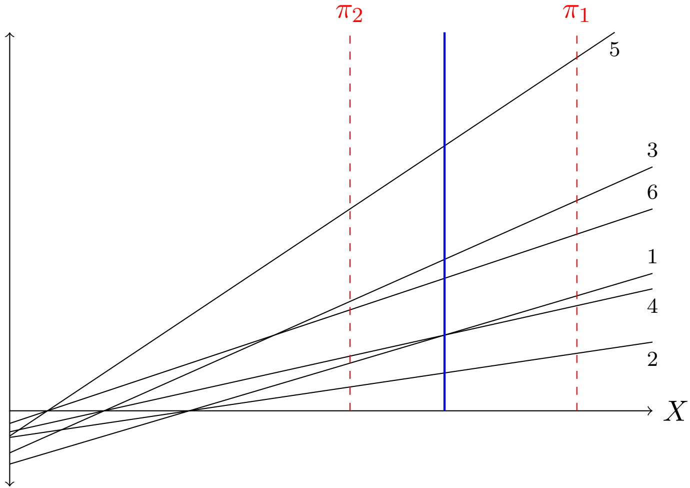
The order of the lines determines the permutation.

The Lines



The order of the lines determines the permutation. Intersection points change that order.

The Lines



The order of the lines determines the permutation. Intersection points change that order.

Solutions within the largest intersection

Solutions to $2^x = c \mod 3^k$, for a fixed operation sequence, are what give us permutations.

- There is always a solutions outside the largest intersection point
- Between adjacent intersection points there could be a solution, giving us a different permutation
- This first occurs for an operation sequence of length 14

The first excess permutation

- The operation sequence $uddudududdu ddd$ gives us the modular requirement

$$2^x = 16 \bmod 729$$

The first two solutions are $x = 4$ and $x = 490$

The first excess permutation

- The operation sequence $uddudududdud dd$ gives us the modular requirement

$$2^x = 16 \bmod 729$$

  The first two solutions are $x = 4$ and $x = 490$
  - The solution $x = 4$ corresponds to a process ending in 2^4

The first excess permutation

- The operation sequence $uddudududdduddd$ gives us the modular requirement

$$2^x = 16 \bmod 729$$

The first two solutions are $x = 4$ and $x = 490$

- The solution $x = 4$ corresponds to a process ending in 2^4
- The solution $x = 490$ corresponds to a process ending in 2^{490}

The first excess permutation

- The operation sequence $uddudududdudd d$ gives us the modular requirement

$$2^x = 16 \bmod 729$$

  The first two solutions are $x = 4$ and $x = 490$
  - The solution $x = 4$ corresponds to a process ending in 2^4
  - The solution $x = 490$ corresponds to a process ending in 2^{490}
- The largest intersection point is ≈ 44.05 so these solutions give us different permutations

1	4	9	14	6	11	15	8	13	5	10	2	7	12	3
1	3	9	14	6	11	15	8	13	5	10	2	7	12	4

How many excess permutations?

- Current data points to the excess being close to

$$\sqrt{\operatorname{fib}(n-11)} \text{ for } n \geq 15$$

How many excess permutations?

- Current data points to the excess being close to

$$\sqrt{\mathrm{fib}(n-11)} \text{ for } n \geq 15$$

- When an excess permutation appears it becomes a root of a tree of longer excess permutations

How many excess permutations?

- Current data points to the excess being close to

$$\sqrt{\mathrm{fib}(n-11)} \text{ for } n \geq 15$$

- When an excess permutation appears it becomes a root of a tree of longer excess permutations
- We can get a very rough upper bound of $n^2\mathrm{fib}(n)$, since n lines can have at most n^2 intersection points

How many excess permutations?

- Current data points to the excess being close to

$$\sqrt{\mathrm{fib}(n-11)} \text{ for } n \geq 15$$

- When an excess permutation appears it becomes a root of a tree of longer excess permutations
- We can get a very rough upper bound of $n^2\mathrm{fib}(n)$, since n lines can have at most n^2 intersection points
- But we can do better

An upper bound

It is not difficult to show that for an operation sequence of length $n-1$ with k u-steps the largest intersection point is bounded by $X = 2 \cdot 3^{k-1}$.

An upper bound

It is not difficult to show that for an operation sequence of length $n-1$ with k u-steps the largest intersection point is bounded by $X = 2 \cdot 3^{k-1}$. Since $X = 2^x$ this gives us

$$x = 1 + \log_2 3(k-1)$$

An upper bound

It is not difficult to show that for an operation sequence of length $n-1$ with k u-steps the largest intersection point is bounded by $X = 2 \cdot 3^{k-1}$. Since $X = 2^x$ this gives us

$$x = 1 + \log_2 3(k-1)$$

The modular requirement has modulus $m = 3^k$ so the distance between two consecutive solutions is $\phi(m) = 2 \cdot 3^{k-1}$

An upper bound

It is not difficult to show that for an operation sequence of length $n-1$ with k u-steps the largest intersection point is bounded by $X = 2 \cdot 3^{k-1}$. Since $X = 2^x$ this gives us

$$x = 1 + \log_2 3(k-1)$$

The modular requirement has modulus $m = 3^k$ so the distance between two consecutive solutions is $\phi(m) = 2 \cdot 3^{k-1}$, so there can be at most one solution in the interval $[0, x]$, giving us $\mathrm{fib}(n)$ as an upper bound on the excess

Structure

- We do know a little about the structure, e.g., how a permutation of length n can be used to create permutations of length $n+1$ and $n+2$

Structure

- We do know a little about the structure, e.g., how a permutation of length n can be used to create permutations of length $n+1$ and $n+2$
- A popular way to describe the structure of a class of permutations is to describe the patterns that the class avoids

Characterizing by what's not there

This is similar to other fields of mathematics

- planar graphs are the graphs that avoid K_5 and $K_{3,3}$

Characterizing by what's not there

This is similar to other fields of mathematics

- planar graphs are the graphs that avoid K_5 and $K_{3,3}$
- simply connected topological spaces are the ones that avoid holes

Characterizing by what's not there

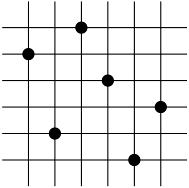
This is similar to other fields of mathematics

- planar graphs are the graphs that avoid K_5 and $K_{3,3}$
- simply connected topological spaces are the ones that avoid holes

For a given class of permutations we want to find the patterns being avoided

Drawing permutations

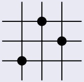
We can draw the graph of a permutation by placing dots on a grid

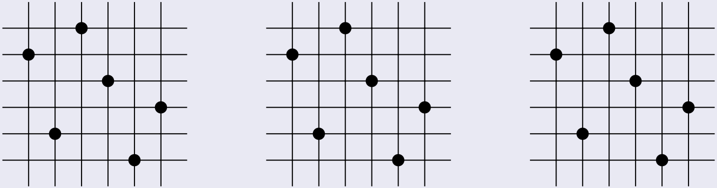
$526413 =$ 

Classical patterns

Patterns are permutations inside other permutations ...

Example

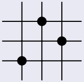
The pattern $132 =$  occurs in the permutation 526413

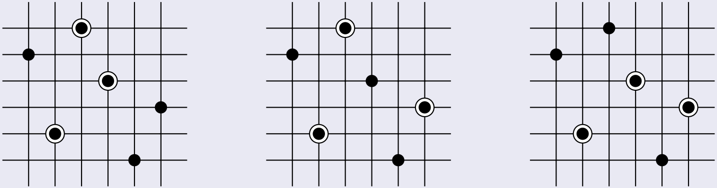


Classical patterns

Patterns are permutations inside other permutations ...

Example

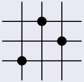
The pattern $132 =$  occurs in the permutation 526413

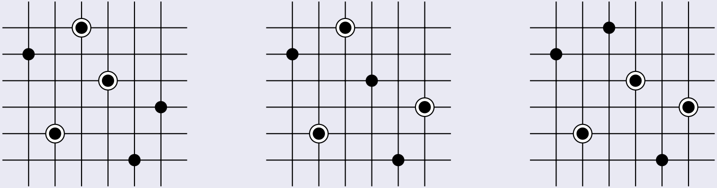


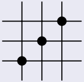
Classical patterns

Patterns are permutations inside other permutations . . .

Example

The pattern $132 =$  occurs in the permutation 526413



The same permutation avoids the pattern $123 =$ 

Some avoidance theorems

We use $\mathrm{Av}(P)$ to denote the perms avoiding the pattern(s) in P.
The class of

- increasing perms $= \mathrm{Av}(21)$ (anonymous cavemen)

Some avoidance theorems

We use $\mathrm{Av}(P)$ to denote the perms avoiding the pattern(s) in P.
The class of

- increasing perms $= \mathrm{Av}(21)$ (anonymous cavemen)
- stack-sortable perms $= \mathrm{Av}(231)$ (Knuth)

Some avoidance theorems

We use $\mathrm{Av}\,(P)$ to denote the perms avoiding the pattern(s) in P. The class of

- increasing perms $= \mathrm{Av}\,(21)$ (anonymous cavemen)
- stack-sortable perms $= \mathrm{Av}\,(231)$ (Knuth)
- smooth Schubert varieties $= \mathrm{Av}\,(1324, 2143)$ (Lakshmibai+Sandhya)

Some avoidance theorems

We use $\operatorname{Av}(P)$ to denote the perms avoiding the pattern(s) in P. The class of

- increasing perms $= \operatorname{Av}(21)$ (anonymous cavemen)
- stack-sortable perms $= \operatorname{Av}(231)$ (Knuth)
- smooth Schubert varieties $= \operatorname{Av}(1324, 2143)$ (Lakshmibai+Sandhya)
- dihedral subgroup $= \operatorname{Av}(16 \text{ patterns})$ (U)

Some avoidance theorems

We use $\mathrm{Av}(P)$ to denote the perms avoiding the pattern(s) in P. The class of

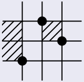
- increasing perms $= \mathrm{Av}(21)$ (anonymous cavemen)
- stack-sortable perms $= \mathrm{Av}(231)$ (Knuth)
- smooth Schubert varieties $= \mathrm{Av}(1324, 2143)$ (Lakshmibai+Sandhya)
- dihedral subgroup $= \mathrm{Av}(16 \text{ patterns})$ (U)

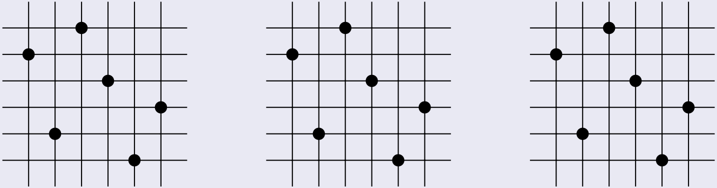
But this vocabulary is not powerful enough to describe West-2-stack-sortable perms, factorial Schubert varieties, alternating subgroup of perms, simsun perms, and others

Mesh patterns patterns

Mesh patterns, introduced by Claesson and Brändén, can restrict regions in the permutation

Example

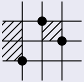
The mesh pattern  occurs in the permutation 526413

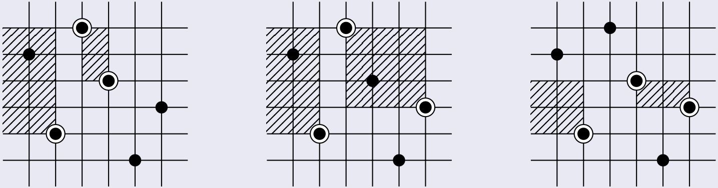


Mesh patterns patterns

Mesh patterns, introduced by Claesson and Brändén, can restrict regions in the permutation

Example

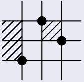
The mesh pattern  occurs in the permutation 526413

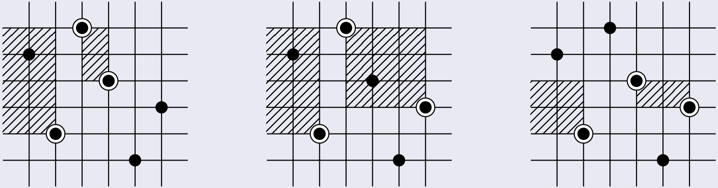


Mesh patterns patterns

Mesh patterns, introduced by Claesson and Brändén, can restrict regions in the permutation

Example

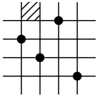
The mesh pattern  occurs in the permutation 526413



Only the last occurrence is valid.

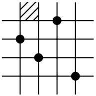
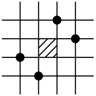
Some more avoidance theorems

The class of

- West-2-stack-sortable perms $= \mathrm{Av}\Big(2341,$  $\Big)$ (West)

Some more avoidance theorems

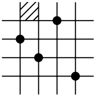
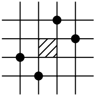
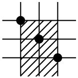
The class of

- West-2-stack-sortable perms $= \mathrm{Av}\left(2341, \right.$  $\left.\right)$ (West)
- factorial Schubert varieties $= \mathrm{Av}\left(1324, \right.$  $\left.\right)$

  (Bousquet-Mélou+Butler)

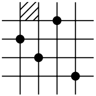
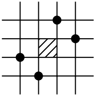
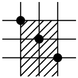
Some more avoidance theorems

The class of

- West-2-stack-sortable perms $= \mathrm{Av}\left(2341,\right.$  $\left.\right)$ (West)
- factorial Schubert varieties $= \mathrm{Av}\left(1324,\right.$  $\left.\right)$ (Bousquet-Mélou+Butler)
- alternating subgroup $= \mathrm{Av}\,($an infinite list of mesh patts$)$ (U)
- simsun perms $= \mathrm{Av}\left(\right.$  $\left.\right)$ (Claesson+Brändén & U)

Some more avoidance theorems

The class of

- West-2-stack-sortable perms $= \mathrm{Av}\left(2341,\right.$  $\left.\right)$ (West)
- factorial Schubert varieties $= \mathrm{Av}\left(1324,\right.$  $\left.\right)$ (Bousquet-Mélou+Butler)
- alternating subgroup $= \mathrm{Av}$ (an infinite list of mesh patts) (U)
- simsun perms $= \mathrm{Av}\left(\right.$  $\left.\right)$ (Claesson+Brändén & U)

It is easy to see that any class of permutations can be described as avoiding a (possibly infinite) list of mesh patterns

Kid-in-the-candy-store problem

There are so many interesting classes of permutations! Which ones have nice descriptions with mesh patterns?

Kid-in-the-candy-store problem

There are so many interesting classes of permutations! Which ones have nice descriptions with mesh patterns?
In 2011 at the AWM Anniversary Conference Sara Billey posed an open problem:

Find a method to learn marked mesh patterns by computer

Kid-in-the-candy-store problem

There are so many interesting classes of permutations! Which ones have nice descriptions with mesh patterns?
In 2011 at the AWM Anniversary Conference Sara Billey posed an open problem:

Find a method to learn marked mesh patterns by computer

More precicely: Construct an algorithm that inputs a (finite piece of a) class of perms and outputs a conjectural description in terms of mesh patterns

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$

Output: $\mathrm{Av}\left(\quad \right)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$123,$$

Output: $\mathrm{Av}\left(123, \qquad\qquad\right)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$\color{red}{123}, 132, 213, 231, 312, 321,$$

Output: $\text{Av}\,(123, \qquad\qquad\quad)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$123, 132, 213, 231, 312, 321,$$
$$1234, 1243, 1324, 1324, 1423,$$

Output: $\mathrm{Av}\,(123, \qquad\qquad)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$123, 132, 213, 231, 312, 321,$$
$$1234, 1243, 1324, 1324, 1423, 1432,$$

Output: $\mathrm{Av}\,(123, 1432, \quad)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$\color{red}{123}, 132, 213, 231, 312, 321,$$
$$\color{red}{1234, 1243, 1324, 1324, 1423}, \color{red}{1432}, \color{red}{2134}, 2143,$$

Output: $\mathrm{Av}\,(123, 1432, \quad)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$123, 132, 213, 231, 312, 321,$$
$$1234, 1243, 1324, 1324, 1423, 1432, 2134, 2143, \ldots$$

Output: $\mathrm{Av}(123, 1432, \ldots)$

A classical algorithm

There is a folklore algorithm that works when the class only avoids classical patterns: Just add non-redundant patterns as you read in the class.

Input: (red perms not in the class)

$$1,$$
$$12, 21,$$
$$123, 132, 213, 231, 312, 321,$$
$$1234, 1243, 1324, 1324, 1423, 1432, 2134, 2143, \ldots$$

Output: $\mathrm{Av}\,(123, 1432, \ldots)$

We need something more powerful if the class is avoiding mesh patterns

The idea behind BiSC

Bi = Billey, S = Steingrímsson, C = Claesson

The idea behind BiSC

Bi = Billey, S = Steingrímsson, C = Claesson

- Step 1: Look at the permutations in the class, to figure out the allowed patterns

The idea behind BiSC

Bi = Billey, S = Steingrímsson, C = Claesson

- Step 1: Look at the permutations in the class, to figure out the allowed patterns
- Step 2: Generate the forbidden patterns from the allowed patterns

The idea behind BiSC

Bi = Billey, S = Steingrímsson, C = Claesson

- Step 1: Look at the permutations in the class, to figure out the allowed patterns
- Step 2: Generate the forbidden patterns from the allowed patterns
- Step 3: Clean up redundancies

Demo

You can download BiSC from my website:
http://staff.ru.is/henningu/programs/bisc/bisc.html
Here we will test the following classes

- West-2-stack-sortable perms
- dihedral subgroup
- alternating subgroup
- simsun perms

A closer look at step 1

- Step 1: Look at the permutations in the class, to figure out the allowed patterns

A closer look at step 1

- Step 1: Look at the permutations in the class, to figure out the allowed patterns

We just keep track of the maximal allowed patterns

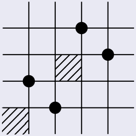
A closer look at step 1

- Step 1: Look at the permutations in the class, to figure out the allowed patterns

We just keep track of the maximal allowed patterns

Example

If we first find a permutation in our class that contains the pattern



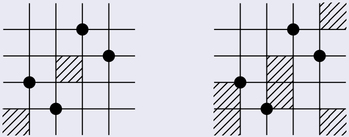
A closer look at step 1

- Step 1: Look at the permutations in the class, to figure out the allowed patterns

We just keep track of the maximal allowed patterns

Example

If we first find a permutation in our class that contains the pattern



and then later a permutation containg the second pattern, we forget the first one, since it is now redundant

A closer look at step 2

- Step 2: Generate the forbidden patterns from the allowed patterns

A closer look at step 2

- Step 2: Generate the minimal forbidden patterns from the allowed patterns

A closer look at step 2

- Step 2: Generate the minimal forbidden patterns from the allowed patterns

Example

For example, if we have the two allowed shadings of the classical pattern 12:

A closer look at step 2

- Step 2: Generate the minimal forbidden patterns from the allowed patterns

Example

For example, if we have the two allowed shadings of the classical pattern 12:

then we generate

A closer look at step 3

- Step 3: Clean up redundancies

There are several possibilities

- Try every subset of patterns in the output (bad if large output)

A closer look at step 3

- Step 3: Clean up redundancies

There are several possibilities

- Try every subset of patterns in the output (bad if large output)
- Use the perms not in the class to see which patterns must be used (bad if large output)

A closer look at step 3

- Step 3: Clean up redundancies

There are several possibilities

- Try every subset of patterns in the output (bad if large output)
- Use the perms not in the class to see which patterns must be used (bad if large output)
- Apply the shading lemma (bad if large output)

A closer look at step 3

- Step 3: Clean up redundancies

There are several possibilities

- Try every subset of patterns in the output (bad if large output)
- Use the perms not in the class to see which patterns must be used (bad if large output)
- Apply the shading lemma (bad if large output)

Natural question

When does a set of mesh patterns M make a mesh pattern m redundant, i.e., $\mathrm{Av}\,(M) = \mathrm{Av}\,(M \cup \{m\})$?

Further work on BiSC

- Find a clever way to perform the clean-up step

Further work on BiSC

- Find a clever way to perform the clean-up step
- Can we teach BiSC to find decorated patterns?

Further work on BiSC

- Find a clever way to perform the clean-up step
- Can we teach BiSC to find decorated patterns?
- Can we make it probabilistic and/or use some techniques from machine learning?

Further work on BiSC

- Find a clever way to perform the clean-up step
- Can we teach BiSC to find decorated patterns?
- Can we make it probabilistic and/or use some techniques from machine learning?
- Can BiSC prove theorems, instead of just stating conjectures?

Thanks!
Please ask questions!