Sorting with Pop Stacks

Lara Pudwell Valparaiso University
faculty.valpo.edu/lpudwell

joint work with
Rebecca Smith (SUNY - Brockport)

Valparaiso MST Faculty Seminar
March 21, 2017

Stack sorting

Pop stack sorting

1-pop-stack sortability

2-pop-stack sortability

Polyominoes on a helix

Width 2

Width 3

Wrapping up

Permutations

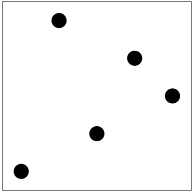
Valparaiso University

Definitions

- *permutation* – an ordering of the members of a set
- $\mathcal{S}_n$ – the set of all permutations of $\{1, 2, \ldots, n\}$.

Example: $\mathcal{S}_3 = \{123, 132, 213, 231, 312, 321\}$.

$\pi = 15243$



Permutations

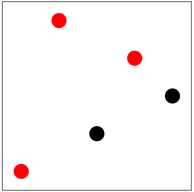
Valparaiso University

Definitions

- *permutation* – an ordering of the members of a set
- $\mathcal{S}_n$ – the set of all permutations of $\{1, 2, \ldots, n\}$.

Example: $\mathcal{S}_3 = \{123, 132, 213, 231, 312, 321\}$.

$\pi = 15243$



$\pi = 15243$ contains 132.

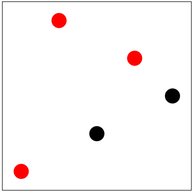
Permutations

Definitions

- *permutation* – an ordering of the members of a set
- $\mathcal{S}_n$ – the set of all permutations of $\{1, 2, \ldots, n\}$.

Example: $\mathcal{S}_3 = \{123, 132, 213, 231, 312, 321\}$.

$\pi = 15243$



$\pi = 15243$ contains 132, but avoids 231.

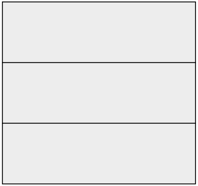
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 15243



Output: $S(15243) = \dots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 5243

Output: $S(15243) = \ldots$

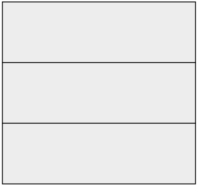
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 5243



Output: $S(15243) = 1\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 243

Output: $S(15243) = 1\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 43

Output: $S(15243) = 1\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 43

Output: $S(15243) = 12\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 3

4
5

Output: $S(15243) = 12\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input:

3
4
5

Output: $S(15243) = 12\ldots$

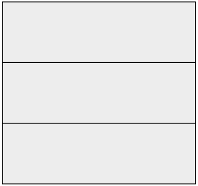
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input:



15243 is 1-stack sortable.

Output: $S(15243) = 12345$

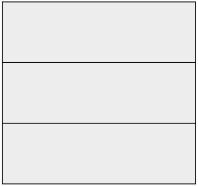
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 35142



Output:

$S(35142) = \ldots$

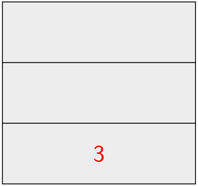
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 5142



Output:

$S(35142) = \ldots$

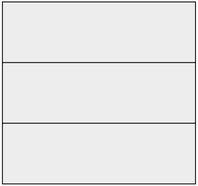
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 5142



35142 is *not* 1-stack sortable.

Output:

$S(35142) = 3\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 142

35142 is *not* 1-stack sortable.

Output:

$S(35142) = 3\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 42

35142 is *not* 1-stack sortable.

Output:

$S(35142) = 3\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 42

35142 is *not* 1-stack sortable.

Output:

$S(35142) = 31\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input: 2

35142 is *not* 1-stack sortable.

Output:

$S(35142) = 31\ldots$

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input:

2
4
5

35142 is *not* 1-stack sortable.

Output:

$S(35142) = 31\ldots$

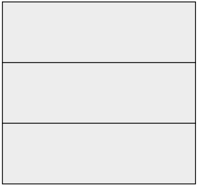
Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

Input:



Output:

$S(35142) = 31245$

35142 is *not* 1-stack sortable.
But
$S(S(35142)) = S(31245) = 12345$,
so 35142 is 2-stack sortable.

Stacks

Stack Operations

A *stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove the top element from the stack and put it on the end of the output.

In general,
if $\pi_i = n$,
write $L = \pi_1 \cdots \pi_{i-1}$ and $R = \pi_{i+1} \cdots \pi_n$
so that $\pi = LnR$.

$$S(LnR) = S(L)S(R)n.$$

Stack sortable permutations

Theorem (Knuth, 1973)

$\pi \in \mathcal{S}_n$ is 1-stack sortable iff π avoids 231. There are $C_n = \frac{\binom{2n}{n}}{n+1}$ such permutations.

Stack sortable permutations

Theorem (Knuth, 1973)

$\pi \in \mathcal{S}_n$ is 1-stack sortable iff π avoids 231. There are $C_n = \frac{\binom{2n}{n}}{n+1}$ such permutations.

Theorem (West, 1990)

$\pi \in \mathcal{S}_n$ is 2-stack sortable iff π avoids 2341 and $3\bar{5}241$.

$$S(S(2341)) = S(2314) = 2134$$
$$S(S(3241)) = S(2314) = 2134$$
$$S(S(35241)) = S(32145) = 12345$$

Stack sortable permutations

Valparaiso University

Theorem (Knuth, 1973)

$\pi \in \mathcal{S}_n$ is 1-stack sortable iff π avoids 231. There are $C_n = \frac{\binom{2n}{n}}{n+1}$ such permutations.

Theorem (West, 1990)

$\pi \in \mathcal{S}_n$ is 2-stack sortable iff π avoids 2341 and $3\bar{5}241$.

Theorem (Zeilberger, 1992)

There are $\frac{2(3n)!}{(n+1)!(2n+1)!}$ 2-stack sortable permutations of length n.

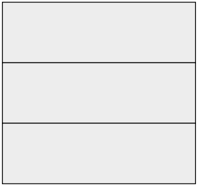
Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 15243



Output: $P(15243) = \ldots$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 5243

Output: $P(15243) = \ldots$

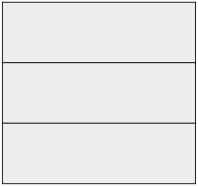
Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 5243



Output: $P(15243) = 1\ldots$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 243

Output: $P(15243) = 1\ldots$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 43

Output: $P(15243) = 1\ldots$

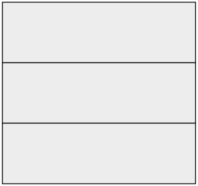
Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 43



15243 is *not* 1-pop-stack sortable.

Output: $P(15243) = 125\ldots$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input: 3

15243 is *not* 1-pop-stack sortable.

Output: $P(15243) = 125\ldots$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input:

15243 is *not* 1-pop-stack sortable.

Output: $P(15243) = 125\ldots$

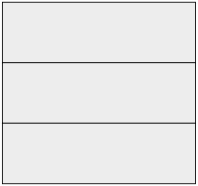
Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

Input:



15243 is *not*
1-pop-stack sortable.

Output: $P(15243) = 12534$

Pop stacks

Pop Stack Operations

A *pop stack* is a last-in first-out data-structure with two operations:

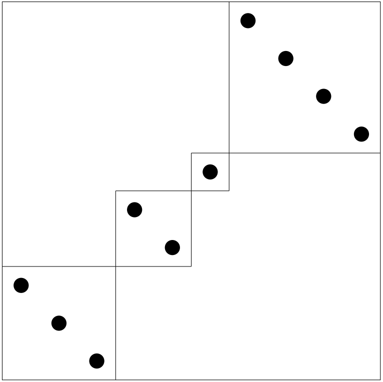
- *push* – remove the first element from input and put it on the top of the stack
- *pop* – remove all elements from the stack and put them on the end of the output.

In general,
if $\pi_1 \cdots \pi_i$ is the longest decreasing prefix of π,
write $R = \pi_{i+1} \cdots \pi_n$
so that $\pi = \pi_1 \cdots \pi_i R$.

$$P(\pi_1 \cdots \pi_i R) = \pi_i \cdots \pi_1 P(R).$$

Pop-stack sortable permutations

If $P(\pi) = 12\cdots n$, then π is layered.



Theorem (Avis and Newborn, 1981)

$\pi \in \mathcal{S}_n$ is 1-pop-stack sortable iff π avoids 231 and 312.
There are 2^{n-1} such permutations.

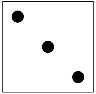
Compositions

Definition

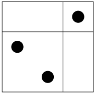
Composition of n – an ordered arrangement of positive integers whose sum is n

Example: $n = 3$

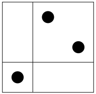
3



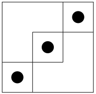
2+1



1+2



1+1+1

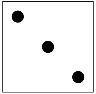


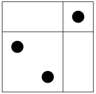
Compositions

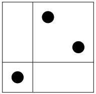
Definition

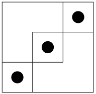
Composition of n – an ordered arrangement of positive integers whose sum is n

Example: $n = 3$

3


2+1


1+2


1+1+1


$$\sum_{\pi \in 1PSS} x^{|\pi|} y^{\mathrm{asc}(\pi)} = \sum_{n=1}^{\infty} \sum_{k=0}^{n-1} \binom{n-1}{k} x^n y^k = \frac{x}{1-x-xy}$$

2-Pop-stack sortability

Definition

A *block* of a permutation is a maximal contiguous decreasing subsequence.

Example: $\pi = 15243$ has blocks $B_1 = 1$, $B_2 = 52$, $B_3 = 43$

2-Pop-stack sortability

Definition

A *block* of a permutation is a maximal contiguous decreasing subsequence.

Example: $\pi = 15243$ has blocks $B_1 = 1$, $B_2 = 52$, $B_3 = 43$

Claim

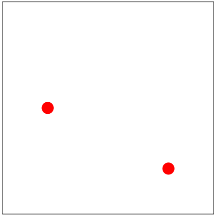
π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

Notice: $P(P(15243)) = P(12534) = 12354$.

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

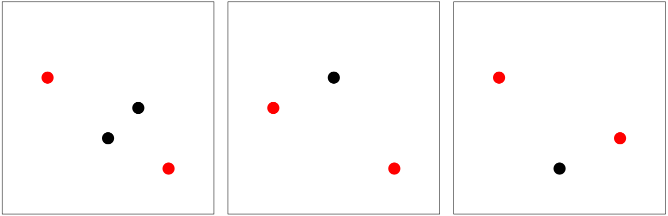


2-Pop-stack sortability

Valparaiso University

Claim

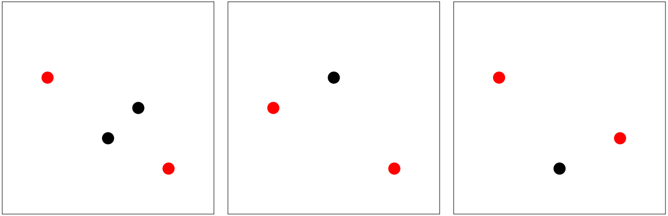
π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.



2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

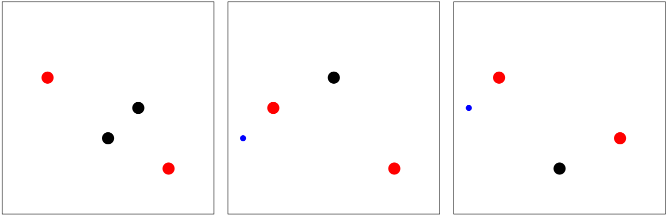


π avoids **4231**.

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.



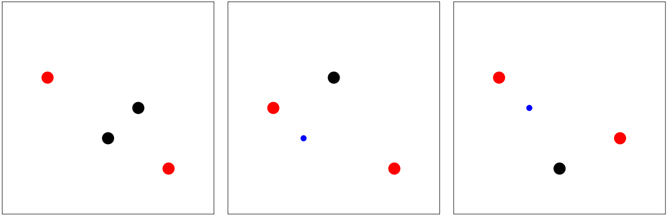
π avoids 4231, **2341**, **3412**

2-Pop-stack sortability

Valparaiso University

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

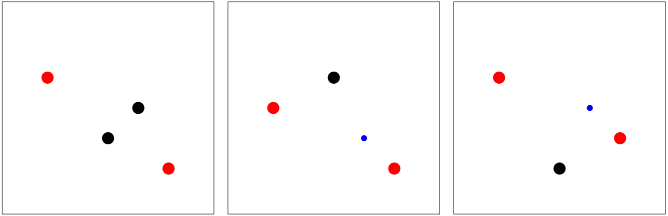


π avoids 4231, 2341, 3412, **3241**, **4312**

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

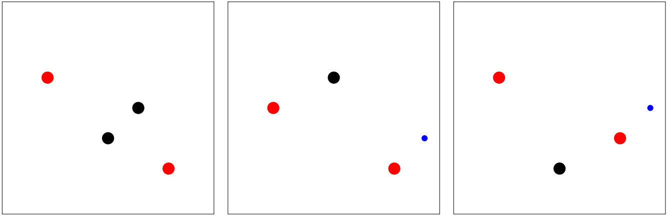


π avoids 4231, 2341, 3412, 3241, 4312, **3421**, **4132**

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

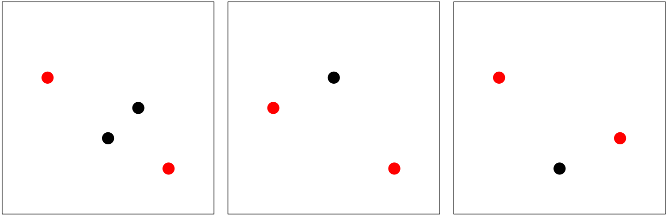


π avoids 4231, 2341, **3412**, 3241, 4312, 3421, 4132, **4123**

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \le i \le \ell - 1$.



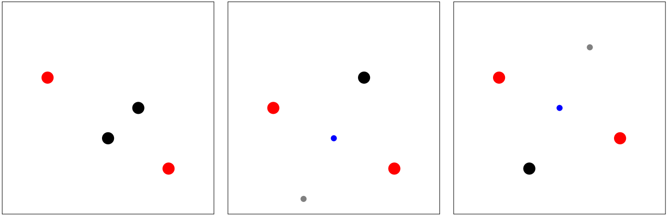
π avoids 4231, 2341, 3412, 3241, 4312, 3421, 4132, 4123

2-Pop-stack sortability

Valparaiso University

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.



π avoids 4231, 2341, 3412, ~~3241~~, 4312, 3421, ~~4132~~, 4123, $4\bar{1}352$, $413\bar{5}2$

2-Pop-stack sortability

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

Theorem (P and Smith, 2017+)

π is 2-pop-stack sortable iff π avoids 2341, 3412, 3421, 4123, 4231, 4312, $4\bar{1}352$, and $413\bar{5}2$.

2-Pop-stack sortability

Valparaiso University

Claim

π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

- Given a composition c, let $f(c)$ be the number of pairs of adjacent summands that aren't both 1.

Example: $f(1+2+1+1) = 2$.

2-Pop-stack sortability

Claim

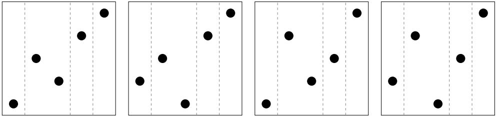
π with blocks $B_1, \ldots, B_\ell$ is 2-pop-stack sortable iff either $\max(B_i) < \min(B_{i+1})$ or $\max(B_i) = \min(B_{i+1}) + 1$ for $1 \leq i \leq \ell - 1$.

- Given a composition c, let $f(c)$ be the number of pairs of adjacent summands that aren't both 1.

  Example: $f(1+2+1+1) = 2$.

- There are $2^{f(c)}$ 2-pop-stack sortable permutations with block structure given by c.

  Example: The 2-pop-stack sortable permutations with block structure $1+2+1+1$ are:



Enumeration via ascents

Valparaiso University

Notation

- $a(n,k)$ is the number of 2-pop-stack sortable permutations of length n with k ascents.
- $b(n,k)$ is the number permutations counted by $a(n,k)$ with last block of size 1.

$$a(n,k) = \begin{cases} 0 & k < 0 \text{ or } k \geq n \\ 1 & k = 0 \text{ or } k = n-1 \\ 2\sum_{i=1}^{n-1} a(i,k-1) - b(n-1,k-1) & \text{otherwise} \end{cases}$$

$$b(n,k) = \begin{cases} 0 & k < 1 \text{ or } k \geq n \\ 1 & k = n-1 \\ 2a(n-1,k-1) - b(n-1,k-1) & \text{otherwise} \end{cases}$$

Enumeration via ascents

Valparaiso University

Theorem (P and Smith, 2017+)

$$\sum_{\pi \in 2PSS} x^{|\pi|} y^{\mathrm{asc}(\pi)} = \sum_{n=1}^{\infty} \sum_{k=0}^{n-1} a(n,k) x^n y^k = \frac{x(x^2y+1)}{1 - x - xy - x^2y - 2x^3y^2}$$

$$collect\left(expand\left(series\left(\frac{x\cdot(x^2\cdot y+1)}{1 - x - x\cdot y - x^2\cdot y - 2\cdot x^3\cdot y^2}, x, 10\right)\right), x\right)$$

$$x + (y+1)x^2 + (y^2+4y+1)x^3 + (y^3+8y^2+6y+1)x^4 + (y^4+12y^3+20y^2+8y+1)x^5 + (y^5+16y^4+48y^3 + 36y^2+10y+1)x^6 + (y^6+20y^5+92y^4+116y^3+56y^2 + 12y+1)x^7 + (y^7+24y^6+152y^5+296y^4+224y^3 + 80y^2+14y+1)x^8 + (y^8+28y^7+228y^6+636y^5 + 708y^4+380y^3+108y^2+16y+1)x^9 + O(x^{10})$$

Enumeration via ascents

Theorem (P and Smith, 2017+)

$$\sum_{\pi \in 2PSS} x^{|\pi|} y^{\mathrm{asc}(\pi)} = \sum_{n=1}^{\infty} \sum_{k=0}^{n-1} a(n,k) x^n y^k = \frac{x(x^2y+1)}{1-x-xy-x^2y-2x^3y^2}$$

Corollary

$$\sum_{\pi \in 2PSS} x^{|\pi|} = \frac{x(x^2+1)}{1-2x-x^2-2x^3}$$

OEIS A224232: $1, 2, 6, 16, 42, 112, 298, 792, \ldots$

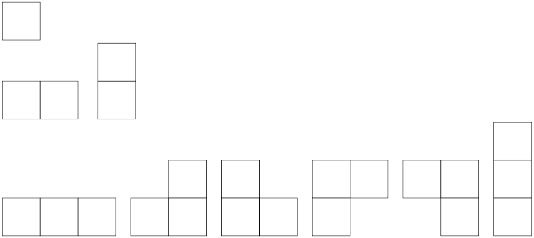
So $|2PSS_n| = 2\,|2PSS_{n-1}| + |2PSS_{n-2}| + 2\,|2PSS_{n-3}|$.

Polyominoes

$n = 1$:

$n = 2$:

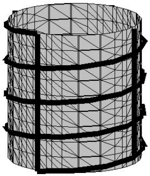
$n = 3$:

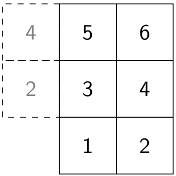


OEIS A001168: $1, 2, 6, 19, 63, 216, 760, \ldots$

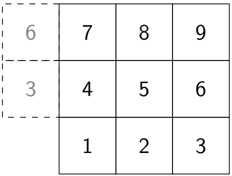
(General formula is open)

Polyominoes on a helix



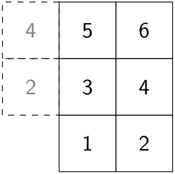


width 2 helix



width 3 helix

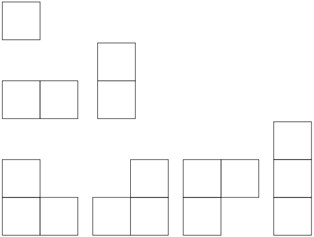
Polyominoes on a helix of width 2

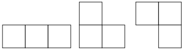


$n = 1$:

$n = 2$:

$n = 3$:



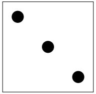
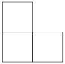
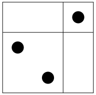
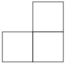
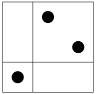
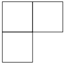
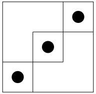
( are all the same polyomino now!)

Sequence: $1, 2, 4, \ldots, 2^{n-1}, \ldots$

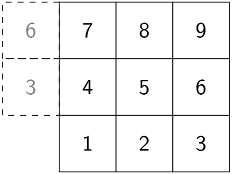
Polyominoes on a helix of width 2

The following are equinumerous:

- compositions of n
- 1-pop-stack sortable permutations of length n
- $\{a, d\}^{n-1}$
- polyominoes of size n on a helix of width 2

3		dd	
2+1		da	
1+2		ad	
1+1+1		aa	

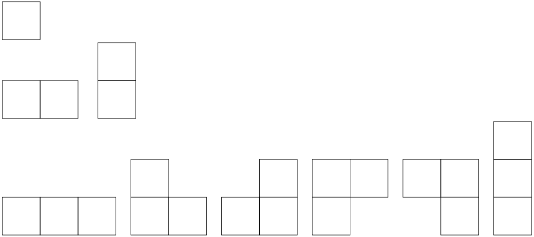
Polyominoes on a helix of width 3



$n = 1$:

$n = 2$:

$n = 3$:



Sequence: $1, 2, 6, 16, 42, 112, \ldots$

(OEIS A224232, same as 2-pop-stack sortable permutations)

Rebuilding 2-pop-stack sortable permutations

- last block has size 1
  - $\pi = \hat{\pi}|n$
  - $\pi = \hat{\pi}|n\bar{\pi}|(n-1)$
    where $\bar{\pi}$ is the longest decreasing suffix of $\hat{\pi}\bar{\pi}$
  - $\pi = \hat{\pi}|(a+1)|(n)(a)|(n-1)$ where $a < n-2$
- last block has size at least 2
  - $\pi = \hat{\pi}|n\bar{\pi}$
    where $\bar{\pi}$ is the longest decreasing suffix of $\hat{\pi}\bar{\pi}$
  - $\pi = \hat{\pi}|(n-1)|(n)(n-2)$
  - $\pi = \hat{\pi}|(n-2)|(n)(n-3)$

Rebuilding 2-pop-stack sortable permutations

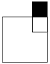
- last block has size 1
  - $\pi = \hat{\pi}|n$ $\hat{\pi} \in 2PSS_{n-1}$
  - $\pi = \hat{\pi}|n\bar{\pi}|(n-1)$
    where $\bar{\pi}$ is the longest decreasing suffix of $\hat{\pi}\bar{\pi}$
    $\hat{\pi}\bar{\pi} \in 2PSS_{n-2}$
  - $\pi = \hat{\pi}|(a+1)|(n)(a)|(n-1)$ where $a < n-2$
    $\hat{\pi}(a+1) \in 2PSS_{n-3}$, ends in ascent
- last block has size at least 2
  - $\pi = \hat{\pi}|n\bar{\pi}$
    where $\bar{\pi}$ is the longest decreasing suffix of $\hat{\pi}\bar{\pi}$
    $\hat{\pi}\bar{\pi} \in 2PSS_{n-1}$
  - $\pi = \hat{\pi}|(n-1)|(n)(n-2)$ $\hat{\pi} \in 2PSS_{n-3}$
  - $\pi = \hat{\pi}|(n-2)|(n)(n-3)$ $\hat{\pi} \in 2PSS_{n-3}$, ends in descent

$$|2PSS_n| = 2 \cdot |2PSS_{n-1}| + |2PSS_{n-2}| + 2 \cdot |2PSS_{n-3}|$$

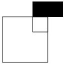
Permutations to polyominoes

Matching with recursive width 3 polyomino cases of Aleksandrowicz, et. al., 2013

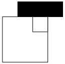
- last block has size 1
  - $\pi = \hat{\pi} | \cancel{n}$



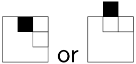
  - $\pi = \hat{\pi} | \cancel{n} \bar{\pi} | \cancel{(n-1)}$



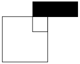
  - $\pi = \hat{\pi} | (a+1) | \cancel{(n)} \cancel{(a)} | \cancel{(n-1)}$



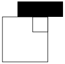
- last block has size at least 2
  - $\pi = \hat{\pi} | \cancel{n} \bar{\pi}$



  - $\pi = \hat{\pi} | \cancel{(n-1)} | \cancel{(n)} \cancel{(n-2)}$



  - $\pi = \hat{\pi} | \cancel{(n-2)} | \cancel{(n)} \cancel{(n-3)}$



Summary

- π is 2-pop-stack sortable iff π avoids 2341, 3412, 3421, 4123, 4231, 4312, $4\bar{1}352$, and $413\bar{5}2$.
- $$\sum_{\pi \in 2PSS} x^{|\pi|} = \frac{x(x^2+1)}{1-2x-x^2-2x^3}$$
- Bijections!
  - 1-pop-stack sortable permutations are in bijection with polyominoes on a helix of width 2
  - 2-pop-stack sortable permutations are in bijection with polyominoes on a helix of width 3

Summary

- π is 2-pop-stack sortable iff π avoids 2341, 3412, 3421, 4123, 4231, 4312, $4\bar{1}352$, and $413\bar{5}2$.
- $$\sum_{\pi \in 2PSS} x^{|\pi|} = \frac{x(x^2+1)}{1-2x-x^2-2x^3}$$
- Bijections!
  - 1-pop-stack sortable permutations are in bijection with polyominoes on a helix of width 2
  - 2-pop-stack sortable permutations are in bijection with polyominoes on a helix of width 3
  - ...but 3-pop-stack sortable permutations aren't in bijection with polyominoes on a helix of width 4.

References

- G. Aleksandrowich, A. Asinowski, and G. Barequet, Permutations with forbidden patterns and polyominoes on a twisted cylinder of width 3. *Discrete Math.* **313** (2013), 1078–1086.
- D. Avis and M. Newborn, On pop-stacks in series. *Utilitas Math.* **19** (1981), 129–140.
- D. E. Knuth, The art of computer programming. Volume 3. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1973. Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing.
- J. West, Permutations with forbidden subsequences and stack sortable permutations. Ph.D. thesis, MIT, 1990.
- D. Zeilberger, A proof of Julian West's conjecture that the number of two-stack sortable permutations of length n is $2(3n)!/((n+1)!(2n+1)!)$. *Disc. Math.* **102** (1992), 85–93.

References

- G. Aleksandrowich, A. Asinowski, and G. Barequet, Permutations with forbidden patterns and polyominoes on a twisted cylinder of width 3. *Discrete Math.* **313** (2013), 1078–1086.
- D. Avis and M. Newborn, On pop-stacks in series. *Utilitas Math.* **19** (1981), 129–140.
- D. E. Knuth, The art of computer programming. Volume 3. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1973. Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing.
- J. West, Permutations with forbidden subsequences and stack sortable permutations. Ph.D. thesis, MIT, 1990.
- D. Zeilberger, A proof of Julian West's conjecture that the number of two-stack sortable permutations of length n is $2(3n)!/((n+1)!(2n+1)!)$. *Disc. Math.* **102** (1992), 85–93.

Thanks for listening!

slides at `faculty.valpo.edu/lpudwell`

email: `Lara.Pudwell@valpo.edu`