

Substitution decomposition for permutation classes with infinitely many simple permutations

Permutation Patterns 2019

Arnar Bjarni Arnarson

Joint work with Christian Bean, Jay Pantone and Henning Ulfarsson

Reykjavik University

arnara14@ru.is

June 21, 2019

Intervals

An *interval* in a permutation σ of size n is a set of consecutive indices $\{i, \dots, i + k - 1\}$ such that $\{\sigma(i), \dots, \sigma(i + k - 1)\}$ is a set of consecutive integers. The intervals of size 1 and n are called *trivial*, the other intervals are called *proper*.

Intervals

An *interval* in a permutation σ of size n is a set of consecutive indices $\{i, \dots, i + k - 1\}$ such that $\{\sigma(i), \dots, \sigma(i + k - 1)\}$ is a set of consecutive integers. The intervals of size 1 and n are called *trivial*, the other intervals are called *proper*.

A proper interval is *maximal* if no larger proper interval contains it.

Intervals

An *interval* in a permutation σ of size n is a set of consecutive indices $\{i, \dots, i + k - 1\}$ such that $\{\sigma(i), \dots, \sigma(i + k - 1)\}$ is a set of consecutive integers. The intervals of size 1 and n are called *trivial*, the other intervals are called *proper*.

A proper interval is *maximal* if no larger proper interval contains it.

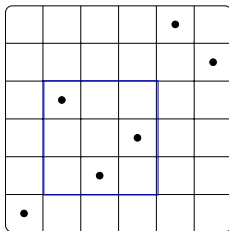


Figure: An example of an interval in a permutation.

Simple permutations

We say that a permutation is simple when it is of size at least 4 and does not have any proper interval.

Simple permutations

We say that a permutation is simple when it is of size at least 4 and does not have any proper interval.

The simple permutations of size 4 are 3142 and 2413.

Simple permutations

We say that a permutation is simple when it is of size at least 4 and does not have any proper interval.

The simple permutations of size 4 are 3142 and 2413.

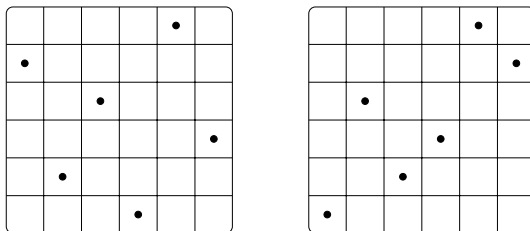


Figure: 524163 is simple, but 142365 is not

Direct sums and skew sums

Given a permutation π of size n and a permutation σ of size m , the *direct sum* $\pi \oplus \sigma$ is defined as

$$(\pi \oplus \sigma)(i) = \begin{cases} \pi(i) & \text{for } 1 \leq i \leq n \\ \sigma(i - n) + n & \text{for } n + 1 \leq i \leq n + m \end{cases}$$

and the *skew sum* $\pi \ominus \sigma$ is defined as

$$(\pi \ominus \sigma)(i) = \begin{cases} \pi(i) + m & \text{for } 1 \leq i \leq n \\ \sigma(i - n) & \text{for } n + 1 \leq i \leq n + m \end{cases}$$

Direct sums and skew sums

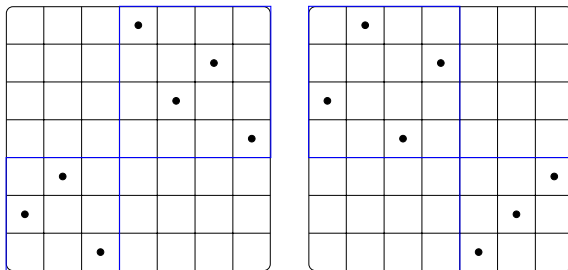


Figure: $231 \oplus 4231 = 2317564$, $2413 \ominus 123 = 5746123$

Inflating permutations

Suppose we have a permutation σ of size n .

Inflating permutations

Suppose we have a permutation σ of size n .

We can *inflate* σ with n permutations $\pi_1, \dots, \pi_n$, replacing each element $\sigma(i)$ with the permutation $\pi_{\sigma(i)}$.

Inflating permutations

Suppose we have a permutation σ of size n .

We can *inflate* σ with n permutations $\pi_1, \dots, \pi_n$, replacing each element $\sigma(i)$ with the permutation $\pi_{\sigma(i)}$.

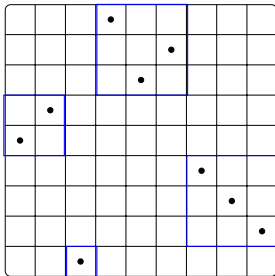


Figure: $3142[12, 1, 312, 321] = 561978432$

Theorem (Albert and Atkinson [1])

Every permutation π of size n with $n \geq 2$ can be uniquely decomposed as either:

Theorem (Albert and Atkinson [1])

Every permutation π of size n with $n \geq 2$ can be uniquely decomposed as either:

- $\pi_1 \oplus \pi_2$, with π_1 *sum-indecomposable*.

Theorem (Albert and Atkinson [1])

Every permutation π of size n with $n \geq 2$ can be uniquely decomposed as either:

- $\pi_1 \oplus \pi_2$, with π_1 *sum-indecomposable*.
- $\pi_1 \ominus \pi_2$, with π_1 *skew-indecomposable*.

Theorem (Albert and Atkinson [1])

Every permutation π of size n with $n \geq 2$ can be uniquely decomposed as either:

- $\pi_1 \oplus \pi_2$, with π_1 *sum-indecomposable*.
- $\pi_1 \ominus \pi_2$, with π_1 *skew-indecomposable*.
- $\sigma[\pi_1, \pi_2, \dots, \pi_k]$ with σ a simple permutation of size k .

Bassino et al. [3] introduced an algorithm for classes with finitely many simple permutations.

Bassino et al. [3] introduced an algorithm for classes with finitely many simple permutations.

It uses the substitution decomposition to automatically enumerate permutation classes.

Bassino et al. [3] introduced an algorithm for classes with finitely many simple permutations.

It uses the substitution decomposition to automatically enumerate permutation classes.

We implemented the algorithm and a more general approach.

We use a framework known as the CombSpecSearcher. (Bean [5])

We use a framework known as the CombSpecSearcher. (Bean [5])

The CombSpecSearcher applies different types of strategies to discover the structure of combinatorial classes.

We use a framework known as the CombSpecSearcher. (Bean [5])

The CombSpecSearcher applies different types of strategies to discover the structure of combinatorial classes.

- Batch strategies

We use a framework known as the CombSpecSearcher. (Bean [5])

The CombSpecSearcher applies different types of strategies to discover the structure of combinatorial classes.

- Batch strategies
- Decomposition strategies

We use a framework known as the CombSpecSearcher. (Bean [5])

The CombSpecSearcher applies different types of strategies to discover the structure of combinatorial classes.

- Batch strategies
- Decomposition strategies
- Equivalence strategies

We use a framework known as the CombSpecSearcher. (Bean [5])

The CombSpecSearcher applies different types of strategies to discover the structure of combinatorial classes.

- Batch strategies
- Decomposition strategies
- Equivalence strategies
- Verification strategies

Gridded permutations

A *gridded permutation* is a permutation where each value has been assigned to a cell in a grid.

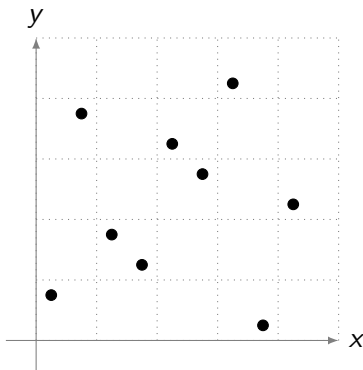


Figure: (284376915, ((0, 0), (0, 3), (1, 1), (1, 1), (2, 3), (2, 2), (3, 4), (3, 0), (4, 2)))

Tilings are a way to represent sets of gridded permutations.

Tilings are a way to represent sets of gridded permutations.

Tiling have obstructions, gridded patterns that the permutations avoid.

Tilings are a way to represent sets of gridded permutations.

Tiling have obstructions, gridded patterns that the permutations avoid.

Tilings also have requirements, gridded patterns that the permutations contain.

Sum decomposition

Any sum-decomposable permutation π can be written as $\pi = \pi_1 \oplus \pi_2$.

Sum decomposition

Any sum-decomposable permutation π can be written as $\pi = \pi_1 \oplus \pi_2$.

To preserve uniqueness π_1 must be sum-indecomposable.

Sum decomposition

Any sum-decomposable permutation π can be written as $\pi = \pi_1 \oplus \pi_2$.

To preserve uniqueness π_1 must be sum-indecomposable.



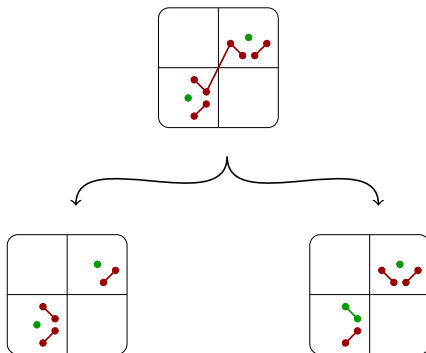
The skew case is similar.

Crossing obstructions

Sometimes obstructions cross between cells.

Crossing obstructions

Sometimes obstructions cross between cells.



Classes with finitely many simple permutations

The strategies presented so far can enumerate any class that does not contain any simple permutation.

Classes with finitely many simple permutations

The strategies presented so far can enumerate any class that does not contain any simple permutation.

Only inflations of simple permutations remain.

Classes with finitely many simple permutations

The strategies presented so far can enumerate any class that does not contain any simple permutation.

Only inflations of simple permutations remain.

First we determine whether the class has finitely many simple permutations. (Bassino et al. [4])

Inflations using tilings

Create a tiling with a point requirement in cell $(i - 1, \sigma(i) - 1)$.

Inflations using tilings

Create a tiling with a point requirement in cell $(i-1, \sigma(i)-1)$.

Then we make sure the inflation avoids the patterns in the class.

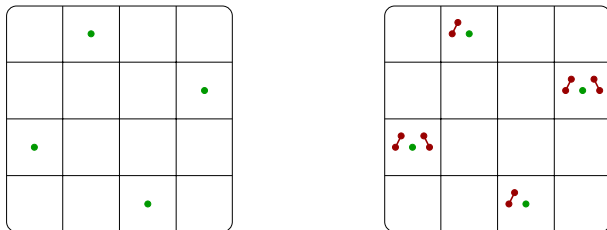
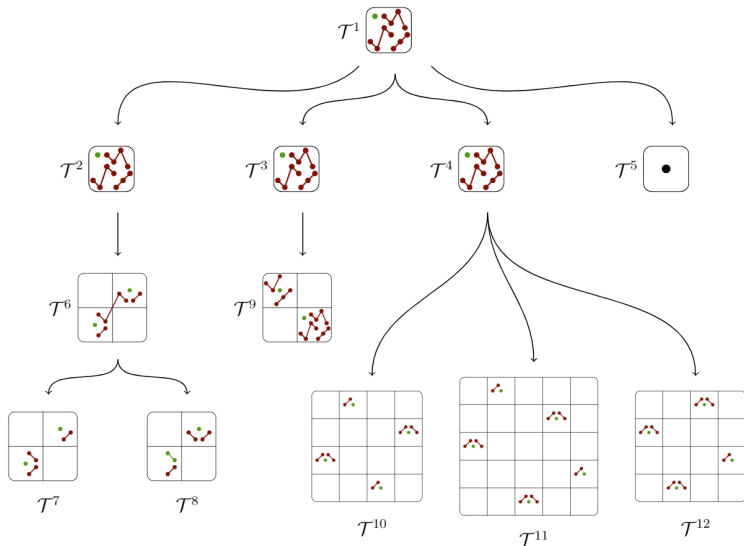


Figure: Inflations of 2413 in $\text{Av}(123, 2143, 3241)$

A combinatorial specification for $\text{Av}(123, 2143, 3241)$



Classes with infinitely many simple permutations

Iterating through the simple permutations would never stop.

Classes with infinitely many simple permutations

Iterating through the simple permutations would never stop.

We want to discover a recursive structure for the simple permutations in the class and inflate them.

Interval placement

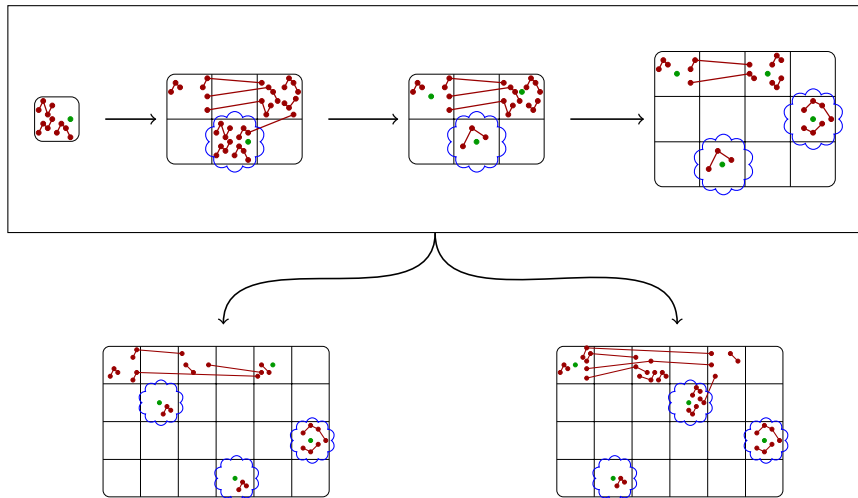
Suppose we have a tiling which only has permutations which are inflations of simple permutations.

Interval placement

Suppose we have a tiling which only has permutations which are inflations of simple permutations.

We consider all the ways we can place the bottom-most maximal interval in the top row of the tiling.

Interval placement



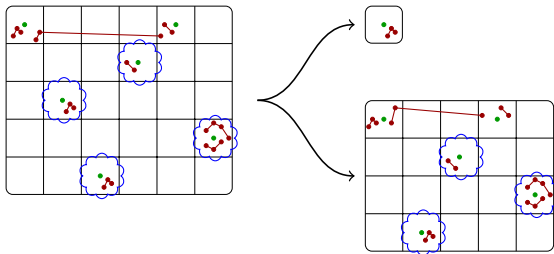
Deletable intervals

We can delete a subset of interval cells if simplicity is preserved.

Deletable intervals

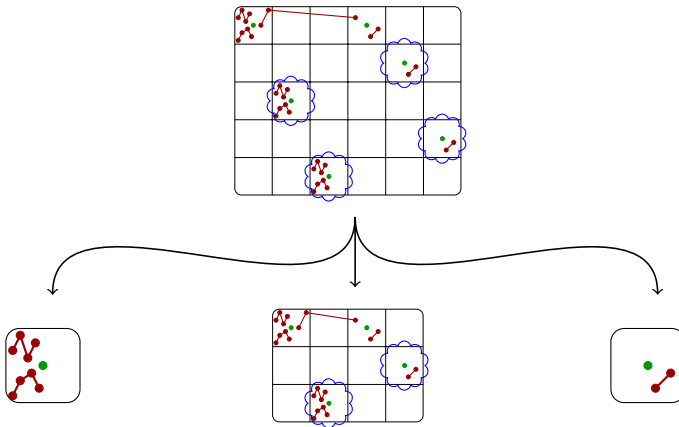
We can delete a subset of interval cells if simplicity is preserved.

After delete the cells we obtain a tiling which we may have seen before.

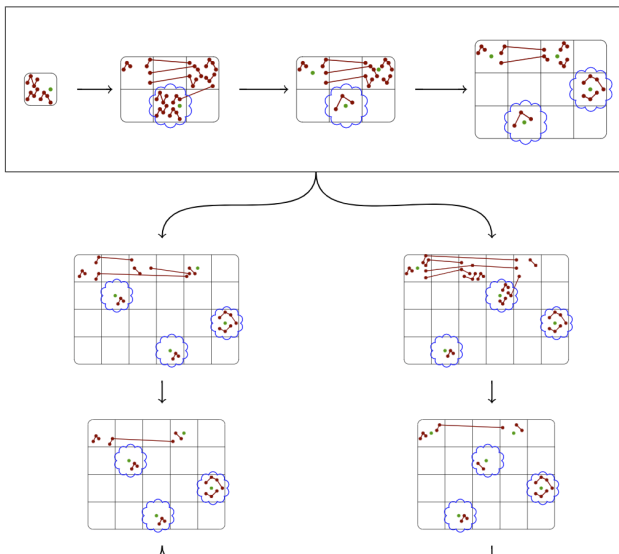


Deletable intervals

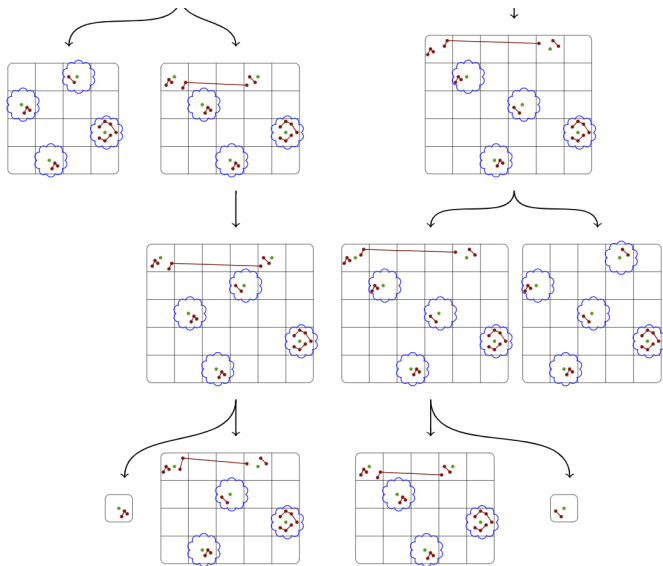
We may need to remove more than one cell at the same time.



Enumerating $A_v(1324, 2413, 2431)$



Enumerating $Av(1324, 2413, 2431)$



Enumerating the simple permutations of a class

We can also enumerate the simple permutations in a permutation class.

Enumerating the simple permutations of a class

We can also enumerate the simple permutations in a permutation class.

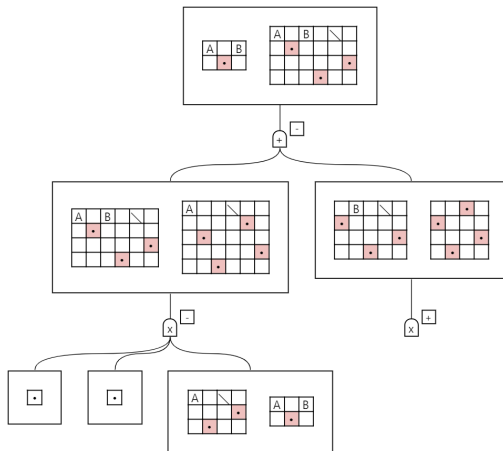


Figure: The simple permutations in $\text{Av}(1342, 2413)$

The CombSpecSearcher requires that it can be determined if a tiling is empty.

The CombSpecSearcher requires that it can be determined if a tiling is empty.

Currently our method checking emptiness is naive.

The CombSpecSearcher requires that it can be determined if a tiling is empty.

Currently our method checking emptiness is naive.

We check if the tiling contains some permutation up to a certain size.

The CombSpecSearcher requires that it can be determined if a tiling is empty.

Currently our method checking emptiness is naive.

We check if the tiling contains some permutation up to a certain size.

The bound is experimental and may cause errors.

The CombSpecSearcher requires that it can be determined if a tiling is empty.

Currently our method checking emptiness is naive.

We check if the tiling contains some permutation up to a certain size.

The bound is experimental and may cause errors.

Usually 90% or more of the running time is spent on this check.

Results for 2x4 classes

Class	Running time	Original enumeration
$A_v(1234, 2413)$	38s	Vatter (2012)
$A_v(1243, 3214)$	478s	Kremer & Shiu (2003)
$A_v(1243, 3241)$	13s	Kremer & Shiu (2003)
$A_v(1243, 3421)$	55s	Kremer & Shiu (2003)
$A_v(1324, 2413)$	7s	Bóna (1998)
$A_v(1342, 2413)$	3s	Albert, Atkinson & Vatter (2014)
$A_v(1342, 2431)$	9s	Bóna (1998)
$A_v(1342, 3214)$	58s	Kremer & Shiu (2003)
$A_v(1342, 4213)$	4s	Kremer & Shiu (2003)
$A_v(1432, 2341)$	100s	Kremer & Shiu (2003)

Table: Results with bottom-most maximal interval placements.

Multiple directions

Why limit ourselves to just bottom-most interval placements?

Multiple directions

Why limit ourselves to just bottom-most interval placements?

When placing intervals we allow all four directions.

Multiple directions

Why limit ourselves to just bottom-most interval placements?

When placing intervals we allow all four directions.

We consider placing the bottom or top-most maximal interval into each row.

More results for 2x4 classes

Class	Running time	Original enumeration
$Av(1243, 2413)$	137s	Albert, Atkinson & Vatter (2014)
$Av(1243, 4231)$	$\sim 30h$	Albert, Atkinson & Brignall (2012)
$Av(1324, 2431)$	158s	Albert, Atkinson & Vatter (2014)
$Av(1324, 3412)$	$\sim 10h$	Albert, Atkinson & Brignall (2011)
$Av(1324, 4231)$	131s	Albert, Atkinson & Vatter (2009)
$Av(1432, 2413)$	$\sim 1.5h$	Le (2005)

Table: Additional results when all direction are allowed.

A question

This method is similar to the insertion encoding method.

A question

This method is similar to the insertion encoding method.

It can be determined by the basis whether a permutation class has a regular insertion encoding.

A question

This method is similar to the insertion encoding method.

It can be determined by the basis whether a permutation class has a regular insertion encoding.

Can we determine whether the simple permutations in a permutation class have a regular insertion encoding?

Conjecture

Conjecture

Using all directions the method can enumerate all permutation classes whose simple permutations can be geometrically gridded.

Conjecture

Conjecture

Using all directions the method can enumerate all permutation classes whose simple permutations can be geometrically gridded.

Conjollary (Albert, Ruškuc, and Vatter [2])

The TileScope algorithm can enumerate all small permutation classes.