

Automatic discovery of generating trees for pattern-avoiding inversion sequences

Abigail Ollson

Joint work with Christian Bean and Jay Pantone

Keele University
School of Computer Science
and Mathematics

Permutation Patterns 2026

Inversion sequences

Inversion sequence

A length n inversion sequence is a word $w_1 w_2 \dots w_n \in \mathbb{N}^*$ such that any value w_i at index i satisfies $w_i < i$.

In other words, $w_1 \in \{0\}$, $w_2 \in \{0, 1\}$, $w_3 \in \{0, 1, 2\}$, and so on.

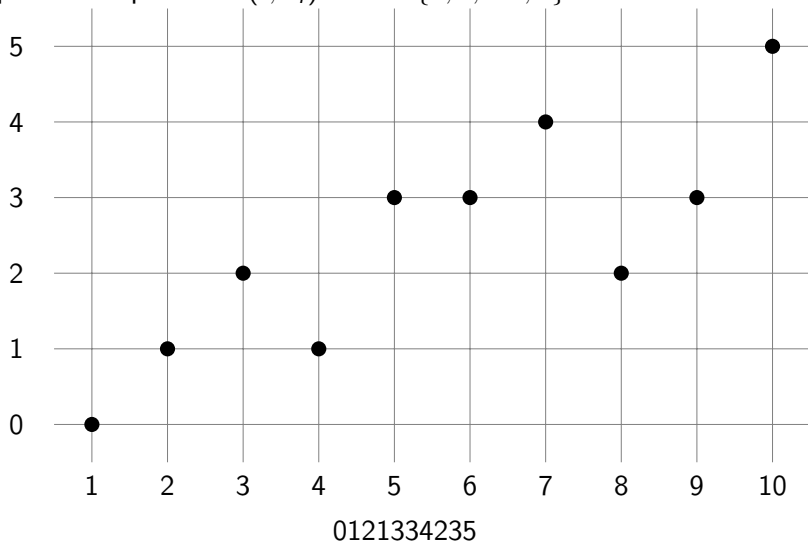
Inversion sequences:

- 0023
- 00031
- 0000
- ϵ (the empty word)

Not inversion sequences:

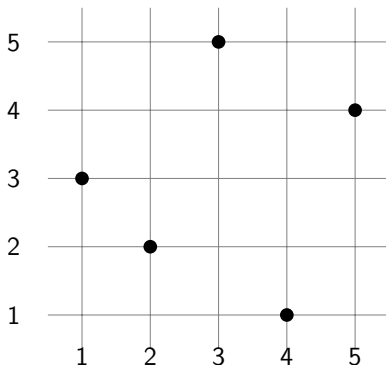
- 004
- 123
- 265
- 0246

Inversion sequences can be represented on a plot on the Cartesian plane with points at (i, σ_i) for $i \in \{1, 2, \dots, n\}$.



Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

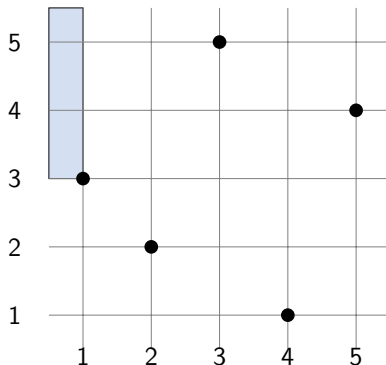
For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

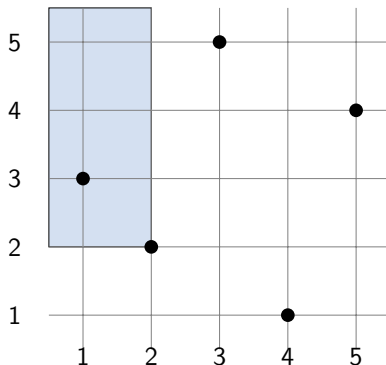
For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

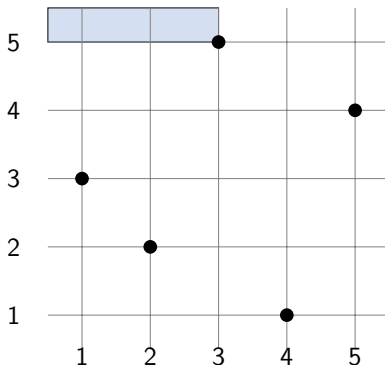
For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

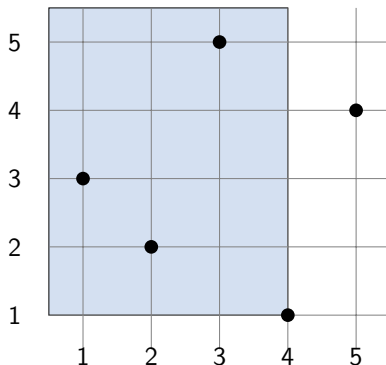
For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

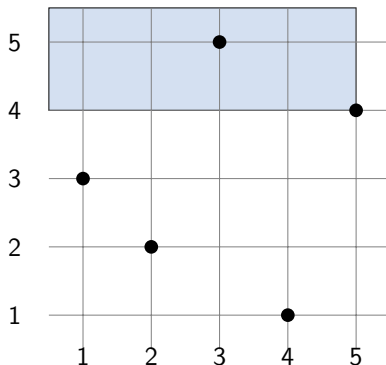
For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Inversion sequences are in bijection with permutations, so there are $n!$ inversion sequences of length n .

For a permutation π , the corresponding inversion sequence is $\sigma_1\sigma_2\dots\sigma_n$ where σ_i is the number of inversions ending at index i in π .



The permutation 32514 corresponds to 01031.

Cayley permutations

A *Cayley permutation* (or a *pattern*) is a word $p \in \mathbb{N}^*$ which contains at least one occurrence of every value between 0 and the maximum value in p .

Cayley permutations:

- 3210
- 1010
- 0123
- 0001

Not Cayley permutations:

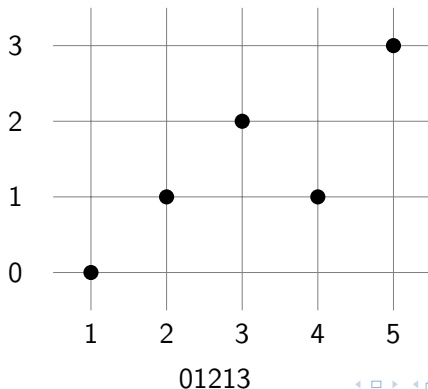
- 0002
- 0134
- 1234
- 2000

Cayley permutations are not necessarily inversion sequences.

Pattern containment

An inversion sequence σ contains a Cayley permutation π if there is a subsequence of σ that is order isomorphic to π , not necessarily at consecutive indices.

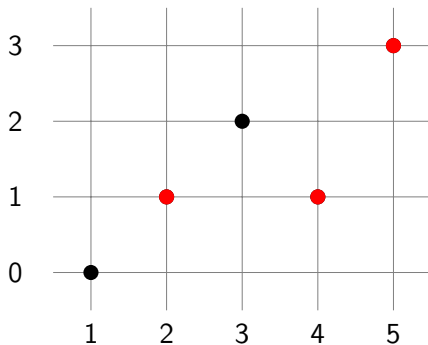
For example, in the inversion sequence 01213 the subsequence 113 implies an occurrence of the Cayley permutation 001.



Pattern containment

An inversion sequence σ contains a Cayley permutation π if there is a subsequence of σ that is order isomorphic to π , not necessarily at consecutive indices.

For example, in the inversion sequence 01213 the subsequence 113 implies an occurrence of the Cayley permutation 001.

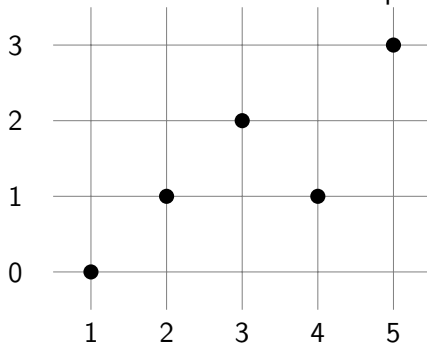


01213 occurrence of 001

Pattern avoidance

An inversion sequence σ avoids a Cayley permutation π if there is no subsequence of σ that is order isomorphic to π .

0121 avoids both 210 and 000 for example.



01213

Inversion class

An inversion class $I(B)$ is a set of inversion sequences that avoid a set of Cayley permutations B , called a basis.

For example, the inversion class $I(10)$ is all inversion sequences which do not contain the Cayley permutation 10.

Inversion sequences in $I(10)$:

- 0000
- 0111
- 0122
- 0123

Some not in $I(10)$:

- 0010
- 0110
- 0020
- 0121

This is not necessarily unique, for example $I(10) = I(010, 0021)$.

Inversion class

An inversion class $I(B)$ is a set of inversion sequences that avoid a set of Cayley permutations B , called a basis.

For example, the inversion class $I(10)$ is all inversion sequences which do not contain the Cayley permutation 10.

Inversion sequences in $I(10)$:

- 0000
- 0111
- 0122
- 0123

Some not in $I(10)$:

- 0010
- 0110
- 0020
- 0121

This is not necessarily unique, for example $I(10) = I(010, 0021)$.

Aim: Enumerate inversion classes by finding a generating function for the class.

This has already been done for some classes:

- All bases with 1 size 3 Cayley permutation have been systematically studied and enumerated.
- All bases with 2 size 3 Cayley permutations have been systematically studied and enumerated.
- Each across lots of different articles¹.

¹[Benjamin Testart](#). “Completing the enumeration of inversion sequences avoiding one or two patterns of length 3”. In: (2024). URL: <https://arxiv.org/pdf/2407.07701>.

This has already been done for some classes:

- All bases with 1 size 3 Cayley permutation have been systematically studied and enumerated.
- All bases with 2 size 3 Cayley permutations have been systematically studied and enumerated.
- Each across lots of different articles¹.
- Most using **generating trees** - we present an automatic method.

¹[Benjamin Testart](#). “Completing the enumeration of inversion sequences avoiding one or two patterns of length 3”. In: (2024). URL: <https://arxiv.org/pdf/2407.07701>.

Combinatorial generating tree

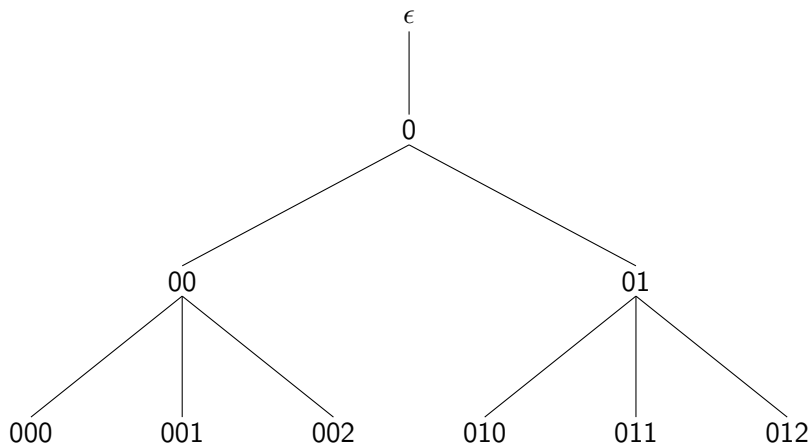


Figure: The first 3 levels of the combinatorial generating tree growing on the right for inversion sequences.

Generating tree

A generating tree is a rooted, labelled tree such that the label of each node determines the labels of the children of that node.

- A *combinatorial generating tree* is a generating tree such that each label of a node is an object in the combinatorial class.
- Each inversion sequence of size n is the label of a node at level n .
- Testart² defined the combinatorial generating tree for inversion sequences obtained by adding new rightmost values as a generating tree *growing on the right*.

²Benjamin Testart. “Completing the enumeration of inversion sequences avoiding one or two patterns of length 3”. In: (2024). URL: <https://arxiv.org/pdf/2407.07701>.

Generating tree

A generating tree is a rooted, labelled tree such that the label of each node determines the labels of the children of that node.

- A *combinatorial generating tree* is a generating tree such that each label of a node is an object in the combinatorial class.
- Each inversion sequence of size n is the label of a node at level n .
- Testart² defined the combinatorial generating tree for inversion sequences obtained by adding new rightmost values as a generating tree *growing on the right*.

We will also make commitments about which of the values already seen will appear again in the inversion sequences.

²Benjamin Testart. “Completing the enumeration of inversion sequences avoiding one or two patterns of length 3”. In: (2024). URL: <https://arxiv.org/pdf/2407.07701>.

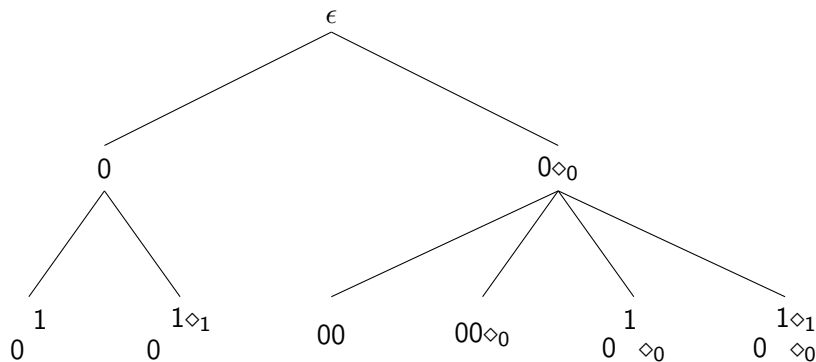


Figure: The first 2 levels of the infinite generating tree growing on the right for configurations.

Each node is called a *configuration*. To count inversion sequences, we only count configurations with no commitments.

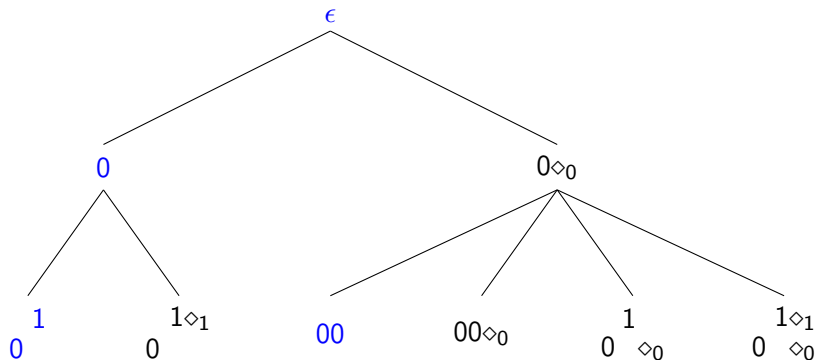


Figure: The first 2 levels of the infinite generating tree growing on the right for configurations.

Each node is called a *configuration*. To count inversion sequences, we only count configurations with no commitments.

Configuration

A configuration is a pair (π, S) such that π is the prefix of an inversion sequence and S is a set of values which are all less than or equal to the largest value in π .

The set S is called the *slots* and each value in S represents a commitment to at least one more occurrence of that value.

2 $\diamond_2$
 $\diamond_1$
00

Figure: The configuration $(002, \{1, 2\})$

Configuration

A configuration is a pair (π, S) such that π is the prefix of an inversion sequence and S is a set of values which are all less than or equal to the largest value in π .

The set S is called the *slots* and each value in S represents a commitment to at least one more occurrence of that value.



Figure: The configuration $(002, \{1, 2\})$

The next value inserted in a configuration can be either a value in the slots or a new maximum.

Inserting into an existing slot is called a *point placement*.
Placing the 1 in $(002, \{1, 2\})$ gives:

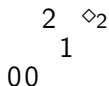


Figure: $(0021, \{2\})$

No more of 1 to come.

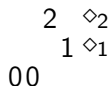


Figure: $(0021, \{1, 2\})$

More of 1 to come.

Inserting a new maximum is called a *bounce placement*. Bouncing up by 2 in $(0021, \{1, 2\})$ gives:

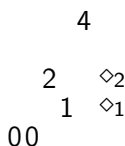


Figure: $(00214, \{1, 2\})$

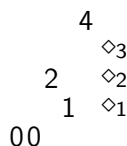


Figure: $(00214, \{1, 2, 3\})$

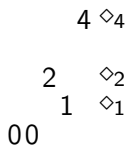


Figure: $(00214, \{1, 2, 4\})$

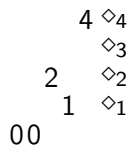


Figure: $(00214, \{1, 2, 3, 4\})$

For any given configuration there is a bound on how far it can bounce up by.

For the previous configuration, bouncing up by 1 would also be okay, but bouncing up by 3 would create a sequence which is not an inversion sequence.

For any given configuration there is a bound on how far it can bounce up by.

For the previous configuration, bouncing up by 1 would also be okay, but bouncing up by 3 would create a sequence which is not an inversion sequence.

Bounce value

The *bounce value*, β , of a configuration is the maximum amount that a bounce placement can bounce up by.

$(0021, \{1, 2\})$ has bounce value 2.

Every configuration has a bounce value of at least 1. The configuration with just a slot has $\beta = 1$.

For any given configuration there is a bound on how far it can bounce up by.

For the previous configuration, bouncing up by 1 would also be okay, but bouncing up by 3 would create a sequence which is not an inversion sequence.

Bounce value

The *bounce value*, β , of a configuration is the maximum amount that a bounce placement can bounce up by.

(0021, $\{1, 2\}$) has bounce value 2.

Every configuration has a bounce value of at least 1. The configuration with just a slot has $\beta = 1$.

Point placement: β increases by 1.

Bounce up by n : β decreases by $n - 1$.

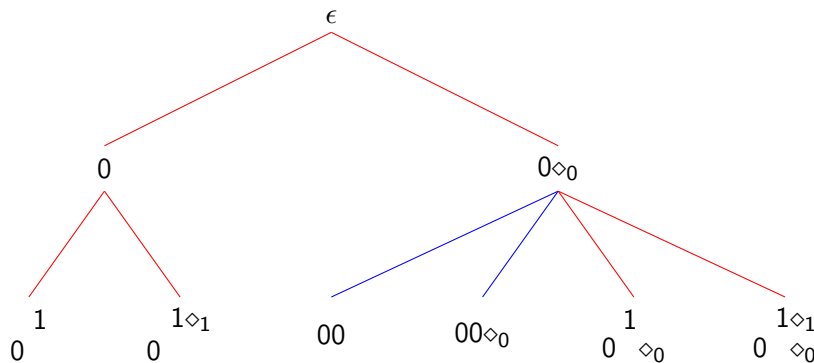


Figure: The first 2 levels of the infinite generating tree growing on the right for configurations.

Point placements are highlighted in blue and bounce placements in red.

Finite generating trees

If a combinatorial generating tree is not finite then we find a finite generating tree which is isomorphic to it, from which we can find an algebraic generating function for the class³.

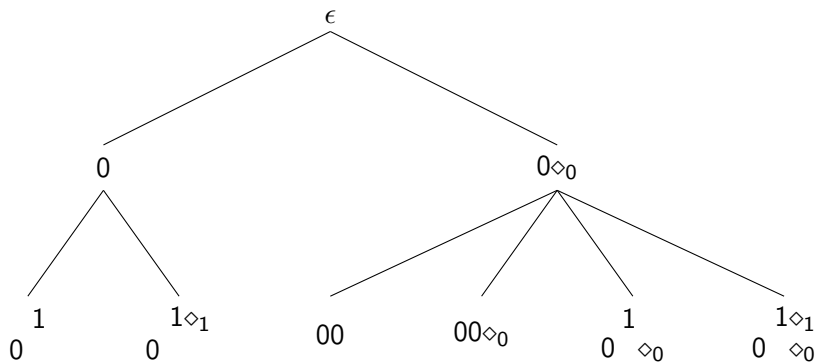
1. Remove any configuration from which all completed inversion sequences contain basis patterns to create a generating tree for the class.

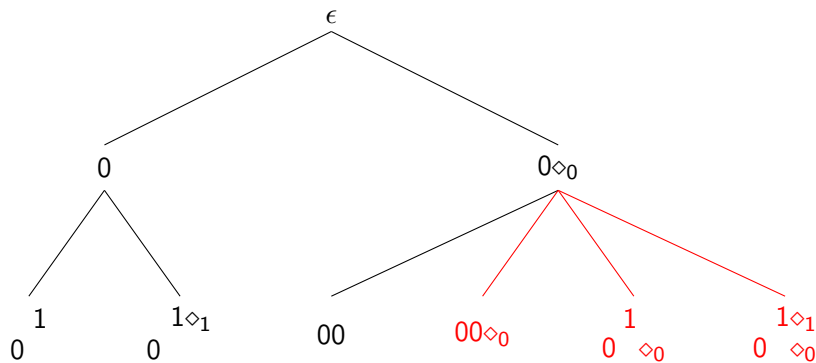
E.g. $(00, \{0\})$ for $I(000)$.

2. Find equivalences between nodes to create a generating tree which is isomorphic to the combinatorial generating tree for the class.

E.g. $(0, \{\})$ and $(01, \{\})$ for $I(000)$.

³Mireille Bousquet-Mélou and Arnaud Jehanne. “Polynomial equations with one catalytic variable, algebraic series and map enumeration”. In: *Journal of Combinatorial Theory, Series B* 96 (Sept. 2006), pp. 623–672. DOI: 10.1016/j.jctb.2005.12.003.

$I(000, 010, 021)$ 

$I(000, 010, 021)$ 

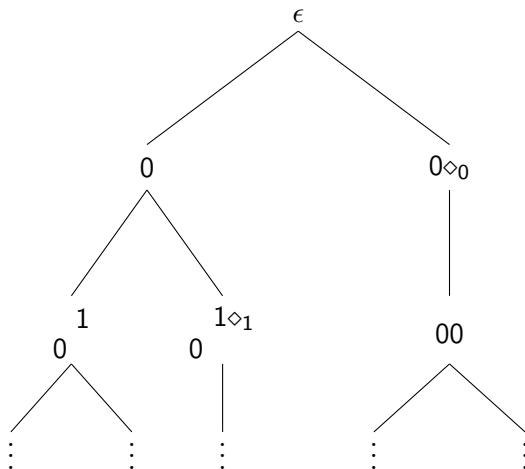
$I(000, 010, 021)$ 

Figure: The infinite generating tree for the class $I(000, 010, 021)$

Equivalent nodes

Two nodes of a generating tree $\mathcal{T}$ are $\mathcal{T}$ -equivalent if they are roots of isomorphic subtrees of $\mathcal{T}$.

When two nodes are equivalent, we can give them the same label.

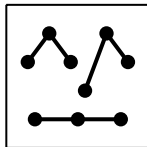
For a configuration c , the label representing c consists of a pair (Π, β) such that β is the bounce of c and Π is an *avoidance set*, also called *tilings* by Albert, Bean, Claesson, Nadeau, Pantone, and Ulfarsson⁴ in the context of permutations.

⁴[Michael Albert et al.](#) “Combinatorial Exploration: An Algorithmic Framework for Enumeration”. In: *Memoirs of the American Mathematical Society* 317 (Feb. 2026). DOI: [10.1090/memo/1611](https://doi.org/10.1090/memo/1611).

Our running example will be the class $I(000, 010, 021)$, which is represented by the avoidance set A .

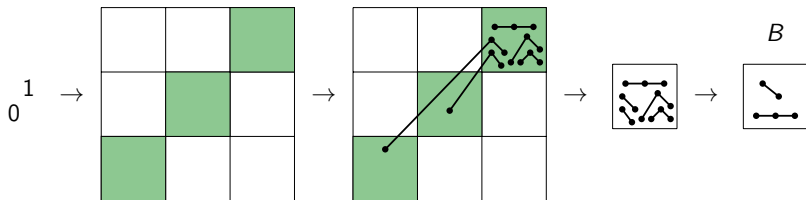
All patterns $p \in \{000, 010, 021\}$ are represented on A .

A



For configurations:

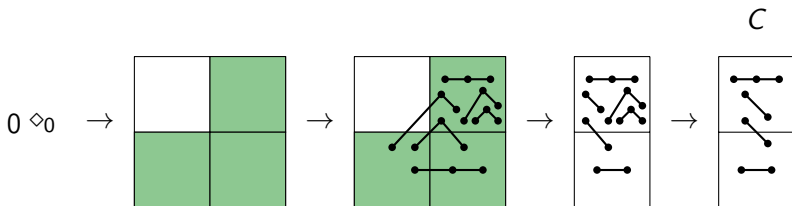
1. Create the *active cells* representing each point in the configuration so far, each slot, and the possibility of new values.
2. Add patterns from $\{000, 010, 021\}$ to active cells such that there is
 - at most 1 index in all but the rightmost column and
 - at most 1 value in all but the topmost row.
3. Take a subset of patterns in the rightmost column.
4. Simplify with respect to containment.



The avoidance set representing $(01\{\})$ is $(B, 1)$.

For configurations:

1. Create the *active cells* representing each point in the configuration so far, each slot, and the possibility of new values.
2. Add patterns from $\{000, 010, 021\}$ to active cells such that there is
 - at most 1 index in all but the rightmost column and
 - at most 1 value in all but the topmost row.
3. Take a subset of patterns in the rightmost column.
4. Simplify with respect to containment.



The avoidance set representing $(0 \diamond 0, \{0\})$ is $(C, 1)$.

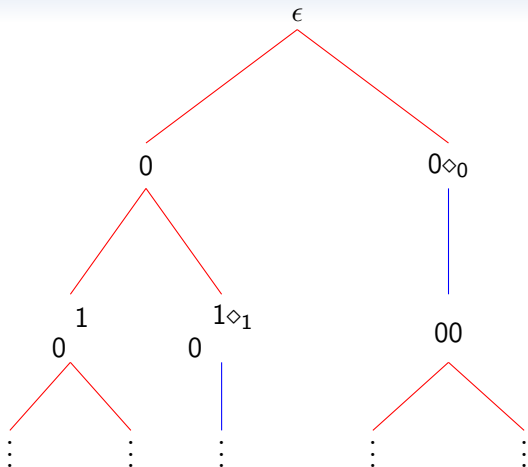


Figure: The infinite generating tree for the class $I(000, 010, 021)$ with bounce placements and point placements marked red and blue respectively.

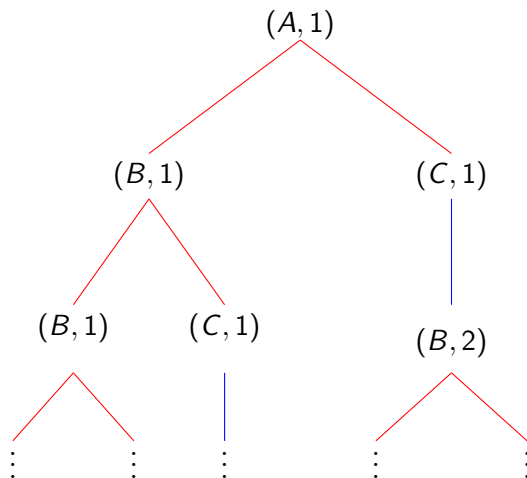


Figure: Replacing configurations with avoidance sets.

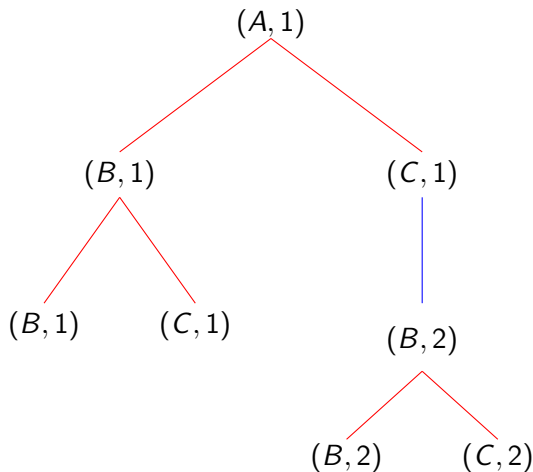


Figure: Replacing configurations with avoidance sets.

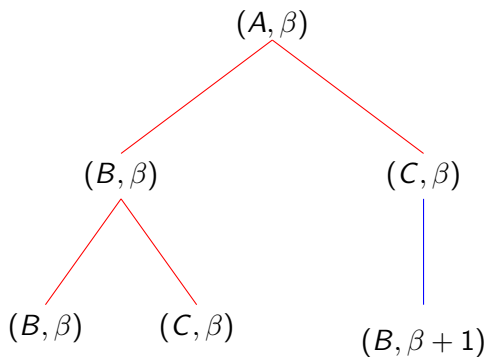


Figure: A finite generating tree for the class $I(000, 010, 021)$.

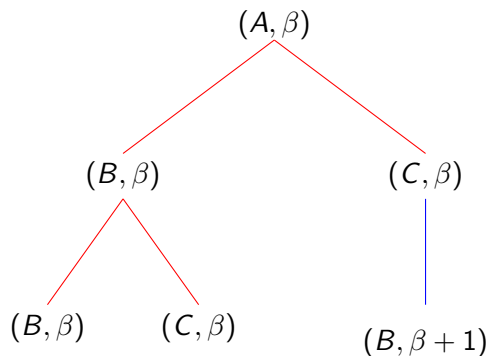


Figure: A finite generating tree for the class $I(000, 010, 021)$.

Our algorithm works on avoidance sets directly so can immediately find equivalences and recursions.

Results

From the finite generating tree we can find the system of equations:

$$S(x, y) = A(x, y) + B(x, y)$$

$$A(x, y) = y$$

$$B(x, y) = xyC(x, y) + x\Phi(A(x, y)) + x\Phi(B(x, y))$$

$$C(x, y) = x\Phi(A(x, y)) + x\Phi(B(x, y))$$

which corresponds to the sequence beginning

$$1, 1, 2, 4, 9, 21, 51, 127, 323, 835$$

which are the Motzkin numbers, A001006 in the OEIS⁵.

⁵Neil J. A. Sloane. *The Online Encyclopedia of Integer Sequences*.
<https://oeis.org>. 2025.

In general, this process is automatic and often quick, especially with lots of basis patterns, but not all classes can be enumerated with this method.

Avoidance sets have one column and the number of rows is determined by the number of slots in configurations they represent.

- Unbounded number of slots in configurations leads to infinitely many nodes (which can't be solved by our methods).

We have proven that there is a bound on the number of slots in configurations if and only if certain Cayley permutations are avoided by the class.

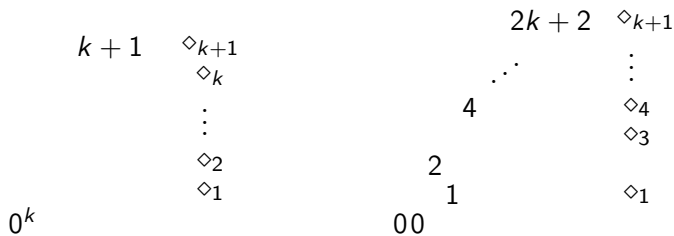


Figure: Configurations with $k + 1$ many slots.

We have proven a check on the basis which is sufficient for this to be the case. Any class which passes this check can be enumerated by our method.

Thank you for listening!

Thank you to the NSF for support in coming here.



If you are interested in working with me, I am looking for a postdoc.

-  Albert, Michael et al. “Combinatorial Exploration: An Algorithmic Framework for Enumeration”. In: *Memoirs of the American Mathematical Society* 317 (Feb. 2026). DOI: 10.1090/memo/1611.
-  Bousquet-Mélou, Mireille and Arnaud Jehanne. “Polynomial equations with one catalytic variable, algebraic series and map enumeration”. In: *Journal of Combinatorial Theory, Series B* 96 (Sept. 2006), pp. 623–672. DOI: 10.1016/j.jctb.2005.12.003.
-  Sloane, Neil J. A. *The Online Encyclopedia of Integer Sequences*. <https://oeis.org>. 2025.
-  Testart, Benjamin. “Completing the enumeration of inversion sequences avoiding one or two patterns of length 3”. In: (2024). URL: <https://arxiv.org/pdf/2407.07701>.