# Automatic methods for enumerating permutation classes

Christian Bean

June 13 - 15, 2019

**Abstract**

In this mini-course we will overview some of the automatic methods used for enumerating permutations classes. We will focus on three topics. The insertion encoding is a language theoretic approach that encodes how a permutation is built up by iteratively adding a new maximum element. We will consider the case when this forms a regular language, and show in this case how to compute the rational generating functions. Secondly, every permutation can be thought of as the inflation of a simple permutations. We will show how to encode this and for permutation classes with finitely many simples compute the algebraic generating function. Finally, we will end by reviewing the result that all geometric grid classes have a rational generating function.

For the latest version go to `https://github.com/christianbean/pp2019` which also includes the most updated version of the template files for the exercises. The template files can also be found and used online (without installing python) at `https://repl.it/@christianbean/PP2019-Enumerating-Permutation-Classes`.

# Contents

# 1 Insertion encoding

In this section, we will review the *insertion encoding* introduced by [ALR05]. The underlying idea is that every length $n$ permutation can be created by taking a length $n-1$ permutation and inserting a new maximum. This is also true for the permutations in a permutation class however not all positions are allowed. Tracing how to build a permutation in this way starting from the empty permutation is called the *evolution* of a permutation. In the insertion encoding, as we map the evolution, we keep track of where future points will be inserted. For example, Figure 1 has the evolution of the permutation $325146$ where the $\diamond$ is viewed as the promise of a future point. A $\diamond$ is called a *slot* and a string such as $32\diamond1\diamond$ is called a *configuration*. We can not continue to evolve from a configuration with no slots. The idea is to model the evolution using a language.

$$\begin{gathered}\diamond\\ \diamond\,1\,\diamond\\ \diamond\,2\,\diamond\,1\,\diamond\\ 3\,2\,\diamond\,1\,\diamond\\ 3\,2\,\diamond\,1\,4\,\diamond\\ 3\,2\,5\,1\,4\,\diamond\\ 3\,2\,5\,1\,4\,6\end{gathered}$$

Figure 1: The evolution of the permutation $325146$.

The slots of a configuration are labeled from left to right, starting with $1$. At each step of the evolution, an insertion is determined by which slot it is inserted into, and how it is inserted. There are four ways to insert a new maximum element $n$ into a slot:

| | |
|---|---|
| $\diamond \mapsto \diamond n \diamond$ | represented by **m** (for middle) |
| $\diamond \mapsto n \diamond$ | represented by $\boldsymbol{\ell}$ (for left) |
| $\diamond \mapsto \diamond n$ | represented by **r** (for right) |
| $\diamond \mapsto n$ | represented by **f** (for fill) |

By subscripting the insertion type by which slot we insert into, this encoding is then unique on the set of all permutations. For example, the permutation $325146$ considered in Figure 1 has the encoding $\mathbf{m}_1\,\mathbf{m}_1\,\mathbf{f}_1\,\boldsymbol{\ell}_2\,\mathbf{f}_1\,\mathbf{f}_1$. This is called the *insertion encoding*.

For a permutation class $\mathcal{C}$, let $\mathcal{L}(\mathcal{C})$ be the language that is formed by the insertion encodings of the permutations in $\mathcal{C}$. We are going to look particularly at the case when this language is regular.

The prefix $p$ of a word in $\mathcal{L}(\mathcal{C})$ corresponds to some configuration. If it has $k$ slots then, as slots are viewed as promises, the shortest word in $\mathcal{L}(\mathcal{C})$ with this prefix will be of length $|p|+k$ by choosing to fill the promises in some order. Moreover, any permutation corresponding to a word with this prefix will contain a permutation of length $|p|+k$ whose insertion encoding has the same prefix. We say a configuration is *acceptable* if it corresponds to a prefix of some word in $\mathcal{L}(\mathcal{C})$.

For a language to be regular, it must be accepted by some deterministic finite automata (DFA). In particular, if this DFA has $k$ states, then for any prefix $p$ of any word in $\mathcal{L}$ there exists a word $w$ of length at most $k + |p|$ since we can always choose a path that does not revisit a state. In particular, there is a maximum number of slots allowed on any configuration which can be completed to form a word if $\mathcal{L}$ is regular. The work of [ALR05] tells us that this condition is sufficient, and moreover, they gave a linear time check for a basis as to whether or not this language is regular. This result was rephrased nicely by [Vat12].

**Theorem 1.1** ([ALR05, Vat12])**.** For a permutation class $\mathcal{C} = \mathrm{Av}(\Pi)$ the following are equivalent:

1. the set $\Pi$ contains at least one permutation from each of $\mathrm{Av}(132, 312)$, $\mathrm{Av}(213, 231)$, $\mathrm{Av}(123, 3142, 3412)$, and $\mathrm{Av}(321, 2143, 2413)$,
2. there exists an integer $k$ such that there are at most $k$ slots in any acceptable configuration for $\mathcal{C}$,
3. the language $\mathcal{L}(\mathcal{C})$ is regular.

The remainder of this section will be focused on showing how to find the insertion encoding for those whose language is regular.

## 1.1 Finding regular insertion encodings

Let $\mathcal{C} = \mathrm{Av}(\Pi)$ be a class with a regular insertion encoding, for which we want to find a DFA accepting $\mathcal{L}(\mathcal{C})$. Imagine the infinite automoton where the states are all of the acceptable configurations and the initial state is $\diamond$ together with the obvious transitions of inserting into the slots. If the slotless configurations are accepting then this clearly will accept $\mathcal{L}(\mathcal{C})$. To find the DFA accepting the insertion encoding of $\mathcal{C}$ we will minimize this finite automoton.

We say that a word $w$ *distinguishes* two configurations corresponding to the prefixes $p_1$ and $p_2$ if $p_1w$ is in $\mathcal{L}(\mathcal{C})$ but $p_2w$ is not. If there is such a $w$ then we say the two configurations are *distinguishable*.

For a configuration $c = c_1 \dots c_k$ define $c - c_i$ to be the configuration obtained by removing $c_i$ and standardising the underlying permutation. The following theorem then says that it is decidable to check if some entry of a configuration can be removed without affecting the suffixes that lead to accepting states.

**Theorem 1.2** ([Vat12])**.** Let $c = c_1 \dots c_k$ be an acceptable configuration for $\mathcal{C} = \mathrm{Av}(\Pi)$ and suppose that the longest element of $\Pi$ has length $b$. If $c_i$ is not a $\diamond$ nor adjacent to two $\diamond$s then $c$ and $c - c_i$ are distinguishable if and only if there is a word of length at most $b - 1$ that distinguishes $c$ and $c - c_i$.

*Proof.* If $c$ and $c - c_i$ are not distiguishable, then there are no words that distinguish them.

Let $p$ and $p'$ be the prefixes corresponding to $c$ and $c - c_i$. Suppose that $c$ and $c - c_i$ are distiguished by some word $w$, then if $pw$ corresponds to a permutation in $\mathcal{C}$ then so must $p'w$. Therefore, we can assume that $p'w$ corresponds to a permutation in $\mathcal{C}$. The configuration $pw$

must correspond to a permutation since the configurations have the same number of slots, as $c_i$ is neither a $\diamond$ nor adjacent to two $\diamond$s, it must follow that the permutation $\pi$ corresponding to $pw$ contains some element of the basis $B$. Choose an occurrence of $\sigma \in B$ in $\pi$. By ignoring all the entries of $\pi$ that are neither in the underlying permutation of $c$ nor in the occurrence of $\sigma$ we find a word of length at most $b-1$ that distiguishes $c$ and $c - c_i$.

□

The final piece to our puzzle is the following theorem.

**Theorem 1.3** ([Vat12])**.** The insertion encoding of $\mathcal{C} = \mathrm{Av}(\Pi)$ is regular if and only if every sufficiently long acceptable configuration $c = c_1 \dots c_k$ contains some entry $c_i$ such that $c$ and $c - c_i$ are not distinguishable.

*Proof.* sketch

( $\Longleftarrow$ ) The assumption implies that the insertion encoding has a finite accepting automoton, so is regular.

( $\Longrightarrow$ ) From Theorem 1.1 we know that there is a bound on the maximum number of slots for acceptable configurations, say $k$. Therefore, by extending the arguments of the proof of Theorem 1.2 if $c$ and $c - c_i$ are distiguishable then there must be a word $w$ of length at most $b + k - 1$ that distiguishes $c$ and $c - c_i$, where $b$ is the length of the longest element in $\Pi$. We call $w$ a *witness* for $c_i$. Each word can witness at most $b$ entries of $c$ and so for a sufficiently long configuration there must be at least one entry of $c$ which has no witnesses, say $c_j$, so clearly $c$ and $c - c_j$ are not distingushable. □

The proofs of Theorem 1.2 and the backwards direction of Theorem 1.3 in fact describe an algorithm for that can be used to find the DFA accepting any regular insertion encoding. Both of these theorems highlight the need to keep track of occurrences of the patterns. The TileScope does exactly this and leads to a far faster implementation [Bea18], where you instead consider gridded permutations.

## 1.2 Exercises

**Exercise 1.1.** Find a DFA accepting the regular insertion encoding for $\mathrm{Av}(123, 231)$. Show that the generating function is

$$\frac{-x^3 + x^2 - x}{x^3 - 3x^2 + 3x - 1}.$$

**Exercise 1.2.** Use the template provided to implement your own algorithm for finding regular insertion encodings that uses the comb_spec_searcher and permuta modules. The template files can be accessed at `https://github.com/christianbean/pp2019` or (without installation) at `https://repl.it/@christianbean/PP2019-Enumerating-Permutation-Classes`.

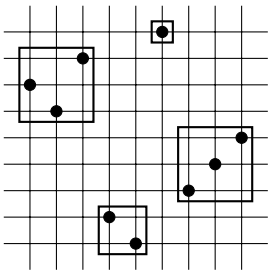

Figure 2: A geometric view of the inflation $3142[213, 21, 1, 123] = 768\ 21\ 9\ 345$.

# 2 Simple permutations

An *interval* in a permutation $\pi$ is a set of consecutive indices $I = [i, j] = \{i, i+1, \ldots, j\}$ in $\pi$ such that the values are consecutive, i.e. $\pi(I) = \{\pi(i) | i \in I\} = [\min(\pi(I)), \max(\pi(I))]$. Every permutation with length $n \geq 1$ has *trivial* intervals of lengths $1$ and $n$. Every other interval is called *proper*. A permutation of length at least two is *simple* if has no proper intervals.

For a permutation $\sigma$ of length $m$ and non-empty permutations $\alpha_1, \ldots, \alpha_m$, the *inflation* of $\sigma$ by $\alpha_1, \ldots, \alpha_m$ is the permutation $\pi[\alpha_1, \ldots, \alpha_m]$ given by replacing each entry $\sigma(i)$ with an interval order-isomorphic to $\alpha_i$. For example,

$$3142[213, 21, 1, 123] = 768\ 21\ 9\ 345$$

seen geometrically in Figure 2.

The inflations of the length two simple permutations need special attention. The *(direct) sum* of the permutations $\alpha_1$ and $\alpha_2$ is $\alpha_1 \oplus \alpha_2 = 12[\alpha_1, \alpha_2]$. A permutation $\pi$ is *sum decomposable* if it is an inflation of $12$, else it is *sum indecomposable*. Every permutation $\pi$ can be written $\pi = \alpha_1 \oplus \alpha_2 \oplus \ldots \oplus \alpha_k$ with unique sum indecomposable permutations $\alpha_1, \ldots, \alpha_k$ called the *sum components*. Similarly, the *skew sum* of the permutations $\alpha_1$ and $\alpha_2$ is $\alpha_1 \ominus \alpha_2 = 21[\alpha_1, \alpha_2]$, with the notions of skew decomposable, skew indecomposable, and skew components defined analagously.

**Theorem 2.1** ([AA05])**.** Every permutation of length at least $2$ can be written as the inflation $\pi = \sigma[\alpha_1, \ldots, \alpha_m]$ where $\sigma$ is a unique simple permutation. If $m \geq 4$ then the $\alpha_i$ are unique. If $\sigma = 12$ ($\sigma = 21$ resp.), and $\alpha_1$ is sum indecomposable (skew indecomposable resp.) then this decomposition is unique.

For a permutation class $\mathcal{C}$, let $\mathcal{C}^{\oplus}$ and $\mathcal{C}^{\ominus}$ be the set of sum and skew decomposable permutations, respectively, and $\mathcal{C}^{S}$ be the set of permutations that are a inflation of a simple permutation of length at least $4$. Using Theorem 2.1, we have the disjoint union

$$\mathcal{C} = \{1\} \sqcup \mathcal{C}^{\oplus} \sqcup \mathcal{C}^{\ominus} \sqcup \mathcal{C}^{S}. \tag{1}$$

This equation is at the heart of the *substitution decomposition*, which has been used to enumerate many permutation classes, e.g. [AAB12, AB14, AAV14]. Permutation classes with finitely many simple permutations are finitely based and have an algebraic generating function,

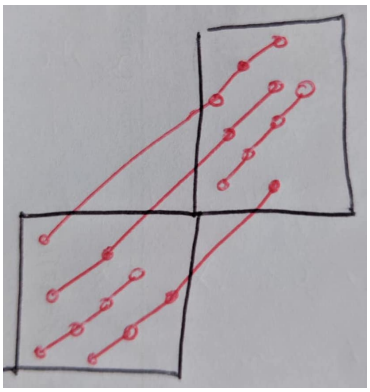

Figure 3: All of the sum embeddings $(\sigma_1, \sigma_2)$ of $1234$ drawn such that $\sigma_1$ is in the left box, and $\sigma_2$ is in the right box.

first shown by [AA05]. The proof of this statement is constructive and we will review this in order to describe an algorithm for computing the generating function. This was done by [BBP+17] in Section 5, who also give a more involved algorithm (in Section 6) that results in a combinatorial specification which can then be used for random samplers.

## 2.1 The sum and skew decomposable permutations

Let $\mathrm{Av}(\Pi)$ be a permutation class, then Theorem 2.1 says that every permutation in $\mathrm{Av}^{\oplus}(\Pi)$ can be written uniquely as $\alpha_1 \oplus \alpha_2$ such that $\alpha_1 \in \mathrm{Av}^{\not\oplus}(\Pi)$ and $\alpha_2 \in \mathrm{Av}(\Pi)$. If all of the patterns in $\Pi$ are sum indecomposable then every choice for $\alpha_1$ and $\alpha_2$ will lead to a permutation in $\mathrm{Av}^{\oplus}(\Pi)$. If, however, there is a sum decomposable pattern $\sigma = \sigma_1 \oplus \sigma_2$ in $\Pi$, if $\alpha_2$ contains $\sigma_2$ then $\alpha_1$ must avoid $\sigma_1$.

For a permutation $\sigma$ we define the sum embeddings of $\sigma$, as all of the pairs of permutations such that $\sigma = \sigma_1 \oplus \sigma_2$ where we, for the first time, allow $\sigma_1$ and $\sigma_2$ to be the empty permutation $\epsilon$. For example, the sum embeddings of $1234$ are

$$\{(\epsilon, 1234), (1, 123), (12, 12), (123, 1), (1234, \epsilon)\},$$

shown pictorially in Figure 3.

For $\mathrm{Av}^{\oplus}(\Pi)$ after (recursively) considering whether $\alpha_2$ contains $\sigma_2$ for every $12$-embedding of a pattern $\sigma$ in $\Pi$, we will have that

$$\mathrm{Av}^{\oplus}(\Pi) = \bigsqcup_{i=0}^{k} \mathrm{Av}^{\not\oplus}(\Pi_i) \oplus \left( \mathrm{Av}(\Pi_i') \cap \bigcap_{j=0}^{\ell} \mathrm{Co}(\alpha_j) \right) \tag{2}$$

where each $\Pi_i$ and $\Pi_j'$ are bases obtained by adding subpatterns of the patterns in $\Pi$ to itself.

[BBP+17] discuss how to write this as a disjoint union, and thus give a combinatorial specification, but we will instead use inclusion/exclusion

$$\mathrm{Av}(\Pi_i') \cap \bigcap_{j=0}^{\ell} \mathrm{Co}(\alpha_j) = \left( \mathrm{Av}(\Pi_i') \cap \bigcap_{j=0}^{\ell-1} \mathrm{Co}(\alpha_j) \right) \setminus \left( \mathrm{Av}(\Pi_i' \cup \{\alpha_\ell\}) \cap \bigcap_{j=0}^{\ell-1} \mathrm{Co}(\alpha_j) \right)$$

to get the enumeration of these.

From Equation 1, the sets of sum and skew indecomposable permutations, $\mathcal{C}^{\not\oplus}$ and $\mathcal{C}^{\not\ominus}$ satisfy

$$\mathcal{C}^{\not\oplus} = \{1\} \sqcup \mathcal{C}^{\ominus} \sqcup \mathcal{C}^{S} \quad \text{and} \quad \mathcal{C}^{\not\ominus} = \{1\} \sqcup \mathcal{C}^{\oplus} \sqcup \mathcal{C}^{S}. \tag{3}$$

which can be used for enumerating the left hand sides of each term in Equation 2.

A symmetric approach can then be taken for the skew decomposable permutations.

**Example 2.1.** Consider $\mathcal{C} = \mathrm{Av}(1234, 2413, 3142)$. The set $\mathcal{C}^{S} = \emptyset$, since every simple permutation of length at least $4$ contains one of $3142$ and $2413$. Then after considering the sum embeddings we get that

$$\begin{aligned}\mathrm{Av}^{\oplus}(1234, 2413, 3142) = &\left(\mathrm{Av}^{\not\oplus}(123, 2413, 3142) \oplus \mathrm{Av}(12)\right) \\ &\sqcup \left(\mathrm{Av}^{\not\oplus}(12) \oplus (\mathrm{Av}(123, 2413, 3142) \cap \mathrm{Co}(12))\right).\end{aligned}$$

As all of the patterns are skew indecomposable we get that

$$\mathrm{Av}^{\ominus}(1234, 2413, 3142) = \mathrm{Av}^{\not\oplus}(1234, 2413, 3142) \oplus \mathrm{Av}(1234, 2413, 3142).$$

If we let $F(x)$ be the generating function for $\mathrm{Av}(1234, 2413, 3142)$ and $F_{\oplus}(x)$, $F_{\ominus}(x)$, $F_{\not\oplus}(x)$, $F_{\not\ominus}(x)$ be the generating function for the obvious relevant permutations in the class we then get the following system of equations

$$\begin{aligned}F(x) &= x + F_{\oplus}(x) + F_{\ominus}(x) \\ F_{\ominus}(x) &= F_{\not\ominus}(x)F(x) \\ F_{\oplus}(x) &= G_{\not\oplus}(x)\frac{x}{1-x} + x\left(G(x) - \frac{x}{1-x}\right)\end{aligned}$$

where $G(x)$ and $G_{\not\oplus}(x)$ refer to the subclass $\mathrm{Av}(123, 2413, 3142)$.

## 2.2 Inflating simple permutations

For permutation $\pi$ and length $n$ simple permutation $\sigma$ the set of $\pi$-embeddings of $\sigma$ is the set of all $n$-tuples $(\alpha_1, \ldots, \alpha_n)$ of possibly empty permutations such that $\pi = \sigma[\alpha_1, \ldots, \alpha_n]$. For example, the $2413$-embeddings of $123$ are

$$\begin{aligned}\{&(123, \epsilon, \epsilon, \epsilon), (\epsilon, 123, \epsilon, \epsilon), (\epsilon, \epsilon, 123, \epsilon), (\epsilon, \epsilon, \epsilon, 123), (12, 1, \epsilon, \epsilon), \\ &(12, \epsilon, \epsilon, 1), (1, 12, \epsilon, \epsilon), (\epsilon, \epsilon, 12, 1), (1, \epsilon, \epsilon, 12), (\epsilon, 1, \epsilon, 12)\}.\end{aligned}$$

As in the sum and skew decomposable cases we may have some $\pi$-embeddings if a pattern $\sigma$ is not simple. In this case, we handle the embeddings in the same way, by considering whether or not the interval it represents contains or avoids the subpattern of the embedding. In the case of the $2413$ embeddings, as each interval contains at least a point it follows that each of the intervals must be decreasing.

**Example 2.2.** Consider $\mathrm{Av}(123, 3142, 362514)$ whose simple permutations are $\{12, 21, 2413\}$. Then, if we let $F(x)$, $F_\oplus(x)$, $F_\ominus(x)$ and $F_S(x)$ be the obvious generating function for $\mathrm{Av}(123, 3142, 362514)$ then it follows that they satisfy the following system of equations

$$\begin{aligned} F(x) &= F_\oplus(x) + F_\ominus(x) + F_S(x) + x \\ F_\oplus(x) &= \frac{x^2}{(1-x)^2} \\ F_\ominus(x) &= (F_\oplus(x) + F_S(x) + x)\, F(x) \\ F_S(x) &= \frac{x^4}{(1-x)^4} \end{aligned}$$

which can be solved to find the rational generating function

$$F(x) = \frac{-x^5 + 2x^4 - 4x^3 + 3x^2 - x}{x^5 - 3x^4 + 8x^3 - 9x^2 + 5x - 1}$$

for the class.

The crux of the substitutuion decomposition algorithm is to keep track of the occurrences of the patterns to be avoided. This can be done in the language of gridded permutations and has also been implemented in the TileScope in an effective manner. As an exercise, you can use this to implement your own version.

## 2.3 Deciding finiteness for simple permutations

The automatic methods for the substitution decomposition we have described apply only to the case where the permutation class contains finitely many simple permutations. Therefore, it is natural to ask whether or not a class contain finitely many simple permutations. [BRV08] indeed show that this is question is decidable, by showing that every sufficiently long simple permutation contains at least one of three diferent types of simple permutations. An efficient algorithm was given by [BBPR15] and in order to generate the simple permutations one can use the approach outlined by [PR12].

## 2.4 Exercises

**Exercise 2.1.** Use the substitution decomposition to compute the generating function for $\mathrm{Av}(123, 2413, 3142)$, which contains no simple permutations of length 4 or greater, and complete Example 2.1.

**Exercise 2.2.** Use the substitution decomposition to compute the generating function for $\mathrm{Av}(123, 3142)$. Note, this class does not have finitely many simple permutations.

**Exercise 2.3.** Use the template provided to implement your own substitution decomposition algorithm for permutation classes with finitely many simple permutations that uses the comb_spec_searcher, permuta and tilings modules. The template files can be accessed at `https://repl.it/@christianbean/PP2019-Enumerating-Permutation-Classes` or `https://github.com/christianbean/pp2019`.

# 3 Grid classes

Before introducing *grid classes* we introduce the notion of *gridded permutations*. In order to do this, we borrow some definitions from Section 2 of [AAB$^+$13] altering them slightly to suit our thinking. The *grid position* of a point $(x, y)$ in $\mathbb{R}^2$ is defined as the integer point $(a, b)$ such that $(x, y) \in [a, a + 1) \times [b, b + 1)$.

We say that a *figure* $\mathcal{F} \subseteq \mathbb{R}^2$ is *grid involved* in the figure $\mathcal{G}$, denoted $\mathcal{F} \preceq \mathcal{G}$ if there are subsets $A, B \subseteq \mathbb{R}$ and increasing injections $\phi_x : A \mapsto \mathbb{R}$ and $\phi_y : B \mapsto \mathbb{R}$ such that

$$\mathcal{F} \subseteq A \times B \text{ and } \phi(\mathcal{F}) = \{(\phi_x(a), \phi_y(b)) : (a, b) \in \mathcal{F}\} \subseteq \mathcal{G}$$

and $\phi$ preserves the grid position of a point $(x, y)$ in $\mathbb{R}^2$. Ignoring the grid position recovers the definition of involvement as in [AAB$^+$13].

As in [AAB$^+$13] this relation forms a preorder on the collection of all figures. If $\mathcal{F} \preceq \mathcal{G}$ and $\mathcal{G} \preceq \mathcal{F}$ we say they are *grid equivalent* (this means two figures are equivalent if and only if one can be transformed to the other by stretching and shrinking the axes without ever crossing the integer lattice).

We say a figure is *independent* if no two points lie on the same horizontal or vertical line. The set of all *gridded permutations*, $\mathcal{G}$, is then defined as the set of all equivalence classes of finite independent figures with respect to grid equivalence. We define containment of gridded permutations with respect to the involvement relation, and we refer to $\sigma$ as a gridded pattern when $\sigma \preceq \pi$ for some gridded permutation $\pi$. Define the *length* of a gridded permutation as the number of points.

For convenience to represent an equivalence class we write the pair, $(\pi, P)$, where $\pi$ is the underlying permutation and $P$ is a tuple of the grid position of each point, for example the figure $\left\{\left(\frac{1}{2}, \frac{1}{2}\right), \left(\frac{3}{2}, \frac{3}{2}\right)\right\}$ is in the equivalence class $(12, ((0, 0), (1, 1))$. A larger example is given in Figure 4 where we have drawn the figure in Equation (4).

$$\mathcal{F} = \left\{\left(\tfrac{1}{4}, \tfrac{3}{4}\right), \left(\tfrac{3}{4}, \tfrac{15}{4}\right), \left(\tfrac{5}{4}, \tfrac{7}{4}\right), \left(\tfrac{7}{4}, \tfrac{5}{4}\right), \left(\tfrac{9}{4}, \tfrac{13}{4}\right), \left(\tfrac{11}{4}, \tfrac{11}{4}\right), \left(\tfrac{13}{4}, \tfrac{17}{4}\right), \left(\tfrac{15}{4}, \tfrac{1}{4}\right), \left(\tfrac{15}{4}, \tfrac{9}{4}\right)\right\}. \tag{4}$$

Suppose $M$ is a $t \times u$ matrix (indexed from left to right and bottom to top) whose entries are permutation classes. We say that a gridded permutation $(\pi, P)$ is in $\mathrm{Grid}^{\#}(M)$ if for each $(k, l)$ in $[0, t) \times [0, u)$, the restriction of $(\pi, P)$ to the permutation using only the points in cell $(k, l)$ is in $M_{k,l}$. The *grid class* of $M$, denoted $\mathrm{Grid}(M)$ consists of all the permutations $\pi$ such that there is some gridded permutation $(\pi, P)$ in $\mathrm{Grid}^{\#}(M)$.

There are not many automatic methods for enumerating grid classes except in special cases such as peg permutations where an effective enumeration was given by [HV16]. In order to use the previous methods, a basis for the permutation class is required. In general, this problem is hard but in the case of juxtapositions (1x2 grid classes), one can compute the basis [Atk99]. It is not even obvious when this basis is finite, for example the grid class $\mathrm{Av}(321654) \oplus \mathrm{Av}(321654)$ has an infinite basis [Mur02].

## 3.1 Monotone grid classes

In the case of automatically enumerating grid classes not a whole lot is known, although there are some interesting results for computing growth rates [Bev15, AV16, APV16]. For $1 \times N$

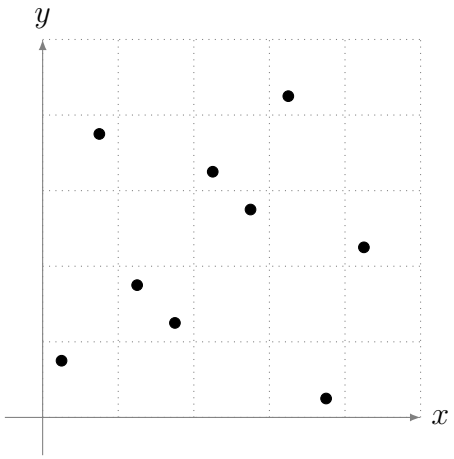


Figure 4: A drawing of the figure $\mathcal{F}$ from Equation (4) that is in the equivalence class represented by $(284376915, ((0,0),(0,3),(1,1),(1,1),(2,3),(2,2),(3,4),(3,0),(4,2)))$.

grid classes, [BS19] outline an algorithm for computing the generating function when at most one cell is not monotone i.e., $\mathrm{Av}(12)$ or $\mathrm{Av}(21)$.

A *monotone grid classes* is any grid class whose entries are all monotone (or empty). The *cell-graph* of $M$ is the graph with vertices being the non-empty cells of $M$ and there is an edge between $(i,j)$ and $(k,\ell)$ if the cells are on the same row or column and there are no non-empty cells between them in this row or column. When the cell-graph is a forest, the grid class and all its subclasses are finitely based, partially well ordered and have a rational generating function [MV03, AAB$^+$13].

This is part of a more general result of [AAB$^+$13] on geometric grid classes. Let $M$ contain only monotone cells, then the *standard figure* of $M$ is the point set in $\mathbb{R}^2$ made up from

- the line segment from $(k-1,\ell-1)$ to $(k,\ell)$ if $M_{k,\ell} = \mathrm{Av}(21)$ or
- the line segment from $(k-1,\ell)$ to $(k,\ell-1)$ if $M_{k,\ell} = \mathrm{Av}(12)$.

The set $\mathrm{Geom}^{\#}(M)$ is the set consisting of all gridded permutations $(\pi, P)$ such that there is some figure in the equivalence class defined by $(\pi, P)$ that is a subset of the standard figure of $M$. That is, a gridded permutation is in $\mathrm{Geom}^{\#}(M)$ if it can be drawn on the standard figure. The *geometric grid class* of $M$, denoted $\mathrm{Geom}(M)$ consists of all the permutations $\pi$ such that there is some gridded permutation $(\pi, P)$ in $\mathrm{Geom}^{\#}(M)$. A permutation class is said to be geometrically griddable if it is contained in a geometric grid class.

**Theorem 3.1** ([AAB$^+$13])**.** Every geometrically griddable class is finitely based, partially well ordered and has a rational generating function.

This result holds only for monotone grid classes that are forests due to the fact that $\mathrm{Grid}(M) = \mathrm{Geom}(M)$ if and only if $M$ is a forest [MV03]. Unfortunately, the proof of this theorem is not constructive, and does not lead to an automatic method for finding the rational generating function.

## 3.2 Exercises

**Exercise 3.1.** Let $M$ be the $1 \times N$ matrix whose entries are all $\mathrm{Av}(21)$. What is the rational generating function for $\mathrm{Grid}^{\#}(M)$? What about $\mathrm{Grid}(M)$?

# References


[AA05] M. H. Albert and M. D. Atkinson, *Simple permutations and pattern restricted permutations*, Discrete Math. **300** (2005), no. 1-3, 1–15. MR 2170110

[AAB12] M. H. Albert, M. D. Atkinson, and Robert Brignall, *The enumeration of three pattern classes using monotone grid classes*, Electron. J. Combin. **19** (2012), no. 3, Paper 20, 34. MR 2967225

[AAB+13] Michael H. Albert, M. D. Atkinson, Mathilde Bouvel, Nik Ruškuc, and Vincent Vatter, *Geometric grid classes of permutations*, Trans. Amer. Math. Soc. **365** (2013), no. 11, 5859–5881. MR 3091268

[AAV14] Michael H. Albert, M. D. Atkinson, and Vincent Vatter, *Inflations of geometric grid classes: three case studies*, Australas. J. Combin. **58** (2014), 24–47. MR 3211768

[AB14] Michael H. Albert and Robert Brignall, *Enumerating indices of Schubert varieties defined by inclusions*, J. Combin. Theory Ser. A **123** (2014), 154–168. MR 3157805

[ALR05] Michael H. Albert, Steve Linton, and Nik Ruškuc, *The insertion encoding of permutations*, Electron. J. Combin. **12** (2005), Research Paper 47, 31. MR 2176523

[APV16] Michael Albert, Jay Pantone, and Vincent Vatter, *On the growth of merges and staircases of permutation classes*, arXiv:1608.06969 (2016).

[Atk99] M. D. Atkinson, *Restricted permutations*, Discrete Math. **195** (1999), no. 1-3, 27–38. MR 1663866

[AV16] Michael Albert and Vincent Vatter, *An elementary proof of bevan's theorem on the growth of grid classes of permutations*, arXiv:1608.06967 (2016).

[BBP+17] Frédérique Bassino, Mathilde Bouvel, Adeline Pierrot, Carine Pivoteau, and Dominique Rossin, *An algorithm computing combinatorial specifications of permutation classes*, Discrete Appl. Math. **224** (2017), 16–44. MR 3637994

[BBPR15] Frédérique Bassino, Mathilde Bouvel, Adeline Pierrot, and Dominique Rossin, *An algorithm for deciding the finiteness of the number of simple permutations in permutation classes*, Adv. in Appl. Math. **64** (2015), 124–200. MR 3300331

[Bea18] Christian Bean, *Finding structure in permutation sets*, Ph.D. thesis, Reykjavik University, June 2018.

[Bev15] David Bevan, *Growth rates of permutation grid classes, tours on graphs, and the spectral radius*, Trans. Amer. Math. Soc. **367** (2015), no. 8, 5863–5889. MR 3347191

[BRV08] Robert Brignall, Nik Ruškuc, and Vincent Vatter, *Simple permutations: decidability and unavoidable substructures*, Theoret. Comput. Sci. **391** (2008), no. 1-2, 150–163. MR 2381360

[BS19] R. Brignall and J. Sliačan, *Combinatorial specifications for juxtapositions of permutations classes*, arXiv:1902.02705 (2019).

[HV16] Cheyne Homberger and Vincent Vatter, *On the effective and automatic enumeration of polynomial permutation classes*, J. Symbolic Comput. **76** (2016), 84–96. MR 3461260

[Mur02] Maximillian M. Murphy, *Restricted permutations, antichains, atomic classes, and stack sorting*, Ph.D. thesis, Univ. of St Andrews, 2002.

[MV03] Maximillian M. Murphy and Vincent R. Vatter, *Profile classes and partial well-order for permutations*, Electron. J. Combin. **9** (2002/03), no. 2, Research paper 17, 30, Permutation patterns (Otago, 2003). MR 2028286

[PR12] Adeline Pierrot and Dominique Rossin, *Simple permutations poset*, arXiv:1201.3119 (2012).

[Vat12] Vincent Vatter, *Finding regular insertion encodings for permutation classes*, J. Symbolic Comput. **47** (2012), no. 3, 259–265. MR 2869320