Table of Contents

Chapter 1

Unranking Restricted Permutations

TODO

1.1 TODO

1. Introduction
2. Define **derived** complementary board B_α^c?
3. If we do a cyclic rotation of the rows of a chessboard, we get essentially the same thing.
4. Move code to Appendix.
5. Define B_α and $\overline{B}_\alpha^c$.
6. Do we want to talk about parking functions?
7. Is it worthwhile to discuss prefix functions for compositions, etc.?
8. Make sure "derank" doesn't occur anywhere.
9. Gives a way of sampling uniformly at random.

1.2 Overview and History

In January 2020, Richard Arratia sent out an email announcing a talk he was going to give on de-ranking derangements.

By January 2021, he announced a $100 prize for solving the analogous problem with ménage permutations. I solved that too.

Richard was interested in a more general question, which I found contagious: Given some family of combinatorial objects that can be quickly counted (say unlabelled simple graphs on n vertices) and some total ordering on them, when is it possible to **unrank** the collection in some computationally efficient way?

Definition 1.2.1. *Let $\mathscr{C}$ be a totally ordered finite set, and let $\{c_i\}_{i=1}^{|\mathscr{C}|}$ be the unique sequence of elements in $\mathscr{C}$ such that $c_i < c_{i+1}$ for all $1 \leq i < |\mathscr{C}|$.*

*The **ranking map** is the map $\operatorname{rank}_{\mathscr{C}} \colon \mathscr{C} \to \mathbb{N}_{>0}$ which sends $c_i \mapsto i$.*

*The **unranking map** is the inverse map $\operatorname{unrank}_{\mathscr{C}} \colon \mathbb{N}_{>0} \to \mathscr{C}$ which sends $i \mapsto c_i$.*

In abstract terms, these maps are not particularly interesting, but in practical terms it can be quite difficult to efficiently compute a given ranking or unranking from a given totally ordered set. After all, when these sets grow in exponential time or worse, explicitly constructing the sequence and doing a search is not computationally feasible.

In Appendix TODO, we will provide examples of total orders on combinatorial objects $\mathscr{C}$ for which constructing efficient

In this chapter we're going to explore that idea. We're going to show a general theory that allows us to de-rank permutations in lexicographic order, derangements in lexicographic order, partitions and compositions of n in lexicographic order, labeled trees by lexicographic order of Prüfer code, Lyndon words [1] (de Bruijn Sequences?), Dyck path in lexicographic order?

1.3 Prefix Counting and Word Ranking

Lemma 1.3.1. *If we have an efficient way to compute the unranking map, an efficient way to compare two elements in the total order, and an efficient way of computing the number of objects at hand, $|\mathscr{C}|$, then we can efficiently compute the ranking map.*

Proof. We can do a binary search. (TODO: write pseudo-code algorithm?) $\square$

1.3.1 Counting Words With a Given Prefix

In both the case of unranking derangements and menage permutations (and in many other applications) our combinatorial objects are words and our total order is lexicographic order.

Definition 1.3.2. ***Lexicographic order*** *is a total ordering on words where $w < v$ ... TODO*

Definition 1.3.3. *A finite* **word** *w over an alphabet $\mathscr{A}$ is a finite sequence $\{w_i \in \mathscr{A}\}_{i=1}^{N}$.*

The collection of finite words over the alphabet $\mathscr{A}$ is denoted by $\mathscr{W}_{\mathscr{A}}$, or just $\mathscr{W}$ when the alphabet is implicit from context.

Definition 1.3.4. *A word $w = \{w_i \in \mathscr{A}\}_{i=1}^{N}$ is said to begin with a* ***prefix*** *$\alpha = \{\alpha_i \in \mathscr{A}\}_{i=1}^{M}$ if $M \leq N$ and $w_i = \alpha_i$ for all $i \leq M$.*

Lemma 1.3.5. *Let $\mathscr{W}$ be the set of words of any length on the alphabet $[n]$, and let $\mathscr{C} \subsetneq \mathscr{W}$ be a finite subset of words on this alphabet, with a total order equal to its lexicographic order.*

Then let $\#\text{prefix}_{\mathscr{C}} : \mathscr{W} \to \mathscr{C}$ be the function that counts the number of words in $\mathscr{C}$ that begin with a given prefix.

Then the unranking function can be computed recursively by

$$\text{unrank}_{\mathscr{C}}(i) = f_i^{\mathscr{C}}((1), 0)$$

where

$$f_i^{\mathscr{C}}(\alpha, j) = \begin{cases} \alpha & i \in (j, j + \#\text{prefix}_{\mathscr{C}}(\alpha)] \textit{ and } \alpha \in \mathscr{C} \\ f_i^{\mathscr{C}}(\alpha', j) & i \in (j, j + \#\text{prefix}_{\mathscr{C}}(\alpha)] \textit{ and } \alpha \notin \mathscr{C} \\ f_i^{\mathscr{C}}(\alpha'', j + \#\text{prefix}_{\mathscr{C}}(\alpha)) & \textit{otherwise}, \end{cases} \tag{1.1}$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$, $\alpha' = (\alpha_1, \alpha_2, \ldots, \alpha_\ell, 1)$, $\alpha'' = (\alpha_1, \alpha_2, \ldots, \alpha_{\ell-1}, 1 + \alpha_\ell)$, *and* j *denotes the number of words in* $\mathscr{C}$ *that occur strictly before* α.

Proof. TODO (sketch) The second line appends a letter, which can happen at most n times. The third line increments the last letter, which can happen at most k times per position. □

This shows that if we can construct a function $\#\text{prefix}_{\mathscr{C}}$ that efficiently counts the number of elements of $\mathscr{C}$ with a given prefix, then we can efficiently rank and unrank the elements of $\mathscr{C}$ in lexicographic order.

1.3.2 Ranking words

In Lemma 1.3.1, we showed that given an efficient algorithm to compute $\text{unrank}_{\mathscr{C}}$, we can derive an efficient algorithm to compute $\text{rank}_{\mathscr{C}}$, on the order of $O(\log(|\mathscr{C}|\,\text{unrank}_{\mathscr{C}}(n)))$ (TODO: make the size of the input explicit.)

However, we can provide a faster algorithm via another recursive function: $\text{rank}_{\mathscr{C}}(w) = g_w(1,1,0)$ where

$$g_w(i, \ell, c) = \begin{cases} c+1 & \ell = w_i \text{ and } i = |w| \\ g_w(i+1, 1, c) & \ell = w_i \text{ and } i < |w| \\ g_w(i, \ell+1, c + \#\text{prefix}_{\mathscr{C}}(w')) & \ell < w_i, \end{cases} \tag{1.2}$$

where $w' = (w_1, w_2, \ldots, w_{i-1}, \ell)$.

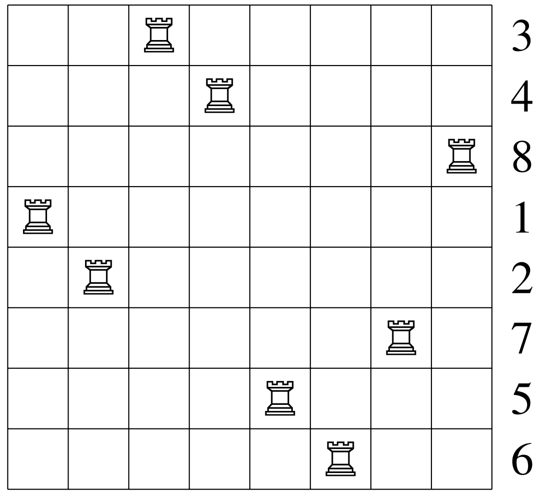


Figure 1.1: An illustration of the rook placement corresponding to the permutation $34812756 \in S_8$. A rook is placed in square $(i, \pi(i))$ for each i.

1.4 Basic Notions of Rook Theory

Because we have showed that we can ... TODO In the case of unranking derangements and permutations, it is useful to use ideas from rook theory, which provides a theory for understanding position-restricted permutations. Rook Theory was introduced by Kaplansky and Riordan [2] in their 1946 paper *The Problem of the Rooks and its Applications*. In it, they discuss problems of restricted permutations in the language of rooks placed on a chessboard.

1.4.1 Definitions in rook theory

We begin by introducing some preliminary ideas from rook theory.

Definition 1.4.1. *A board B is a subset of $[n] \times [n]$ which represents the squares of a $n \times n$ chessboard that rooks are allowed to be placed on. Every board B has a complementary board $B^c = ([n] \times [n]) \setminus B$, which consists of all of the squares of B that a rook cannot be placed on.*

To each board, we can associate a generating polynomial that keeps track of the number of ways to place a given number of rooks on the valid squares in such a way that no two rooks are in the same row or column.

Definition 1.4.2. *The rook polynomial associated with a board B,*

$$p_B(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_n x^n,$$

is a generating polynomial where r_k denotes the number of k-element subsets of B such that no two elements share an x-coordinate or a y-coordinate.

In the context of permutations, we're typically interested in r_n, the number of ways to place n rooks on a restricted $n \times n$ board. However, it turns out that a naive application of the techniques from rook theory do not immediately allow us to count the number of restricted permutations with a given prefix. Computing the number of such permutations is known to be computationally hard for a board with arbitrary restrictions. We can see this by encoding a board B as a $(0,1)$-matrix and computing the matrix permanent. (In fact, Shevelev [3] claims that "the theory of enumerating the permutations with restricted positions stimulated the development of the theory of the permanent.")

Lemma 1.4.3. *Let $M_B = \{a_{ij}\}$ be an $n \times n$ matrix where*

$$a_{ij} = \begin{cases} 1 & (i,j) \in B \\ 0 & (i,j) \notin B \end{cases}.$$

Then the coefficient of x^n in $p_B(x)$ is given by the matrix permanent

$$\operatorname{perm}(M_B) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i\sigma(i)}.$$

Now is an appropriate time to recall Valiant's Theorem.

Theorem 1.4.4 (Valiant's Theorem [4])**.** *Computing the permanent of a (0,1)-matrix is #P-complete.*

Corollary 1.4.5. *Computing the number of rook placements on an arbitrary $n \times n$ board is #P-hard.*

Therefore, in order to compute the number of permutations, we must exploit some additional structure of the restrictions.

1.4.2 Techniques of Rook Theory

Rook polynomials can be computed recursively. The base case is that for an empty board $B = \emptyset$, the corresponding rook polynomial is $p_\emptyset(x) = 1$, because there is one way to place no rooks, and no way to place one or more rooks.

Lemma 1.4.6 ([5]). *Given a board, B, then for any square $(x, y) \in B$, we can define the resulting boards if we include or exclude the square respectively*

$$B_i = \{(x', y') \in B : x \neq x' \text{ and } y \neq y'\} \tag{1.3}$$

$$B_e = B \setminus (x, y). \tag{1.4}$$

Then we can write the rook polynomial for B in terms of this decomposition.

$$p_B(x) = x p_{B_i}(x) + p_{B_e}(x).$$

If we want to compute a rook polynomial using this construction, we can end up adding up lots of smaller rook polynomials—a number that is exponential in the size of B. However, when the number of squares in B^c is small in some sense, it can be easier to compute the rook polynomial p_{B^c} and use the principle of inclusion/exclusion on it's coefficients to determine the rook polynomial for the original board, B.

In the case of derangements and ménage permutations, this is the strategy we'll use. Start by finding the resulting board from a given prefix, find the rook polynomial of the complementary board, and use the principle of inclusion/exclusion to determine the number of ways to place rooks in the resulting board.

1.5 Unranking Derangements

In January 2020, Richard Arratia sent out an email proposing a seminar talk. The title describes the first “$100 problem”:

$100 Problem. *“For* 100 *dollars, what is the* 500 *quadrillion-th derangement on* $n = 20$*?”*

$100 Answer. *The computer program in Appendix TODO computed the answer in less than ten milliseconds. When written as words in lexicographic order, the derangement in* S_{20} *with rank* 5×10^{17} *is*

$$12\ 14\ 2\ 9\ 13\ 20\ 6\ 3\ 1\ 17\ 5\ 11\ 19\ 15\ 10\ 18\ 8\ 7\ 4\ 16.$$

Arratia’s question focused on unranking derangements where the rank was based on the total ordering that comes from writing the permutations as words in lexicographic order. Other authors have looked at unranking derangements based on other total orderings. In particular, Mikawa and Tanaka [6] give an algorithm to rank/unrank derangements with respect to *lexicographic ordering in cycle notation.*

In this section we will develop an algorithm for ranking and unranking with respect to their lexicographic ordering as words. The technique that we use will broadly be re-used in the next section. It is worthwhile to begin by recalling the definition of a derangement.

Definition 1.5.1. *A derangement is a permutation* $\pi \in S_n$ *such that* π *has no fixed points. That is, the set of derangements is*

$$\{\pi \in S_n : \pi(i) \neq i\ \forall i \in [n]\}.$$

1.5.1 The complementary board.

In order to compute the number of derangements with a given prefix, it is useful to look at the board that results after placing k rooks according to these positions, as illustrated in Figure 1.2.

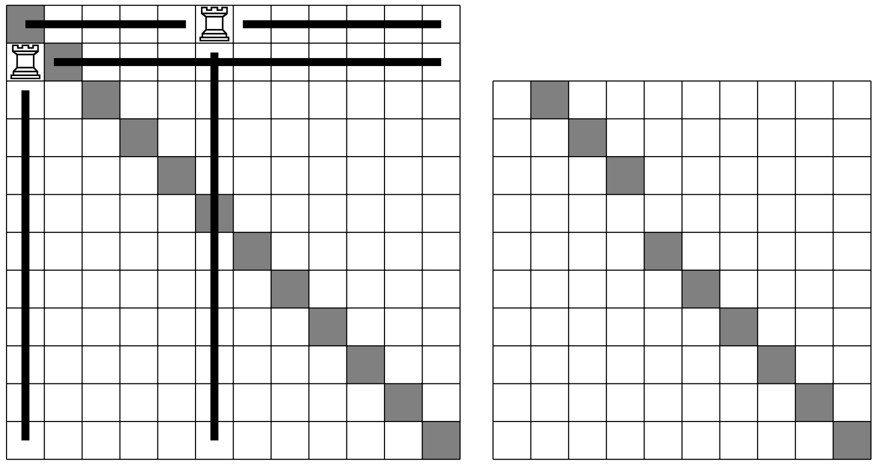


Figure 1.2: An example of a prefix $\alpha = (6,1)$, and the board that results from deleting the first $\ell = 2$ rows and columns 6 and 1. The derived complementary board of B from α is $B^c_\alpha = \{(1,2),(2,3),(3,4),(5,5),\ldots,(10,10)\}$.

Definition 1.5.2. *If B is an $n \times n$ board, and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ is a valid prefix of length ℓ, then **derived complementary board** of B from α, denoted B^c_α, is B with the appropriate rows and columns removed and reindexed in such a way that $B^c_\alpha \subseteq [n-\ell] \times [n-\ell]$.*

Lemma 1.5.3. *Given a valid ℓ letter prefix $(\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ of a word on n letters, the number of squares in the resulting complementary board is*

$$|B^c_\alpha| = n - \ell - |\{\ell+1, \ell+2, \ldots, n\} \cap \{\alpha_1, \alpha_2, \ldots, \alpha_\ell\}|,$$

and no two of these squares are in the same row or column.

Proof. TODO $\square$

1.5.2 Derangements with a given prefix

Now that we have a way of quickly computing $|B^c_\alpha|$, we can compute the number of ways to place j rooks on the complementary board. We can use this to compute the number of derangements that begin with the prefix α.

Lemma 1.5.4. *The rook polynomial for the complementary board B^c_α is*

$$p_{B^c_\alpha}(x) = \sum_{j=0}^{|B^c_\alpha|} \binom{|B^c_\alpha|}{j} x^j. \tag{1.5}$$

Proof. No two squares in B^c (and thus B^c_α) are in the same row or column. Thus the number of ways to place j rooks is equivalent to selecting j cells from $|B^c_\alpha|$. $\square$

Now we introduce a lemma of Stanley [7] to compute the number of TODO from a complementary board.

Lemma 1.5.5 ([7])**.** *The number of ways, N_0, of placing n nonattacking rooks on a board $B \subseteq [n] \times [n]$ is given by*

$$N_0 = \sum_{k=0}^{n} (-1)^k r_k (n-k)!,$$

where $P_{B^c}(x) = \sum_{k=0}^{n} r_k x^k$.

Corollary 1.5.6. *The number of derangements with prefix $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ is given by*

$$\#\text{prefix}(\alpha) = \sum_{j=0}^{|B^c_\alpha|} (-1)^j \binom{|B^c_\alpha|}{j} (n-\ell-j)!,$$

which is A047920$(n-\ell, |B^c_\alpha|)$ *in the On-Line Encyclopedia of Integer Sequences [8].*

Example 1.5.7. *For example, for $N = 14$, we wish to count the number of derangements that start with the prefix* $6\,1$*. Since the prefix has two letters, $p = 2$ and $n = 14-2 = 12$. The only crossed-out cell that is deleted by the prefix in the remaining board is the cell that was in position* 6*: in particular, $\{3,4,\ldots,14\} \cap \{6,1\} = 6$. Thus $k = 12-1 = 11$. Thus there are* A047920$(12,11) =$ $190\,899\,411$ *derangements that start with* $6\,1$.

α (prefix)	#prefix(α)	index range	$\lvert B_\alpha^c \rvert$	$f_i^{\mathscr{D}}(\alpha,\ell)$
1	0	$(0,0]$	–	$f_{1000}^{\mathscr{D}}(1,0)$
2	2119	$(0,2119]$	6	$f_{1000}^{\mathscr{D}}(2,0)$
21	265	$(0,265]$	6	$f_{1000}^{\mathscr{D}}(21,0)$
22	0	$(265,265]$	–	$f_{1000}^{\mathscr{D}}(22,265)$
23	309	$(265,574]$	5	$f_{1000}^{\mathscr{D}}(23,265)$
24	309	$(574,883]$	5	$f_{1000}^{\mathscr{D}}(24,574)$
25	309	$(883,1192]$	5	$f_{1000}^{\mathscr{D}}(25,883)$
251	53	$(883,936]$	4	$f_{1000}^{\mathscr{D}}(251,883)$
253	0	$(936,936]$	–	$f_{1000}^{\mathscr{D}}(253,936)$
254	64	$(936,1000]$	3	$f_{1000}^{\mathscr{D}}(254,936)$
2541	11	$(936,947]$	3	$f_{1000}^{\mathscr{D}}(2541,936)$
2543	11	$(947,958]$	3	$f_{1000}^{\mathscr{D}}(2543,947)$
2546	14	$(958,972]$	2	$f_{1000}^{\mathscr{D}}(2546,958)$
2547	14	$(972,986]$	2	$f_{1000}^{\mathscr{D}}(2547,972)$
2548	14	$(986,1000]$	2	$f_{1000}^{\mathscr{D}}(2548,986)$
25481	3	$(986,989]$	2	$f_{1000}^{\mathscr{D}}(25481,986)$
25483	3	$(989,992]$	2	$f_{1000}^{\mathscr{D}}(25483,989)$
25486	4	$(992,996]$	1	$f_{1000}^{\mathscr{D}}(25486,992)$
25487	4	$(996,1000]$	1	$f_{1000}^{\mathscr{D}}(25487,996)$
254871	2	$(996,998]$	0	$f_{1000}^{\mathscr{D}}(254871,996)$
254873	2	$(998,1000]$	0	$f_{1000}^{\mathscr{D}}(254873,998)$
2548731	1	$(998,999]$	0	$f_{1000}^{\mathscr{D}}(2548731,998)$
2548736	1	$(999,1000]$	0	$f_{1000}^{\mathscr{D}}(2548736,999)$
25487361	1	$(999,1000]$	0	$f_{1000}^{\mathscr{D}}(25487361,999)$

Table 1.1: There are $A000166(8) = 14833$ derangements on 8 letters. This algorithm finds the derangement at index 1000.

1.6 Unranking Ménage Permutations

1.6.1 Proof of concept (The $100 answer!)

In February 2020, Richard Arratia offered

$100 Problem. *For $n = 20$ there are $A000179(20) = 312\,400\,218\,671\,253\,762 > 3.1 \cdot 10^{17}$ ménage permutations. Determine the 10^{17}-th such permutation when listed in lexicographic order.*

$100 Answer. *The desired permutation is*

$$7\ 16\ 19\ 12\ 2\ 8\ 15\ 1\ 18\ 14\ 3\ 9\ 20\ 10\ 5\ 17\ 13\ 4\ 11\ 6. \tag{1.6}$$

A Ménage permutation comes from the *problème des ménages*. Here we will define it as

Definition 1.6.1. *A ménage permutation is a permutation $\pi \in S_n$ such that for all $i \in [n]$, $\pi(i) \neq i$ and $\pi(i) + 1 \not\equiv i \bmod n$.*

We can use the prefix to get a new board, which is block diagonal (whenever the prefix is non-empty), if we know the number of cells in each block, we can compute the number of valid boards. This gives us the number of ménage permutations with a given prefix.

Prefix $\Rightarrow$ grouped columns $\Rightarrow$ partition/multiset $\Rightarrow$ complementary polynomial $\Rightarrow$ count

1.6.2 Block diagonal decomposition

When we look at Figure TODO, it appears that placing rooks according to a prefix results in a derived complementary board where the squares can be grouped into sub-boards that don't share any rows or columns. We will see that this property holds more generally, and we can exploit this in order to describe the number of ménage permutations with a given prefix.

It is useful to begin by formalizing this notion of grouping squares.

Definition 1.6.2. *Two boards B and B' are called **disjoint** if no squares of B are in the same row or column as any square in B'.*

The reason that we care about decomposing a board into disjoint parts is because that perspective allows us to factor the rook polynomial.

Theorem 1.6.3 ([2])**.** *If B can be partitioned into disjoint boards $b_1, b_2, \ldots, b_m$, then the rook polynomial of B is the product of the rook polynomials of the b_is*

$$p_B(x) = \prod_{i=1}^{m} p_{b_i}(x).$$

This disjoint decomposition is useful for this context because, as Figure 1.3 suggests, the derived complementary board for a nonempty prefix is block-diagonal, with well-understood blocks.

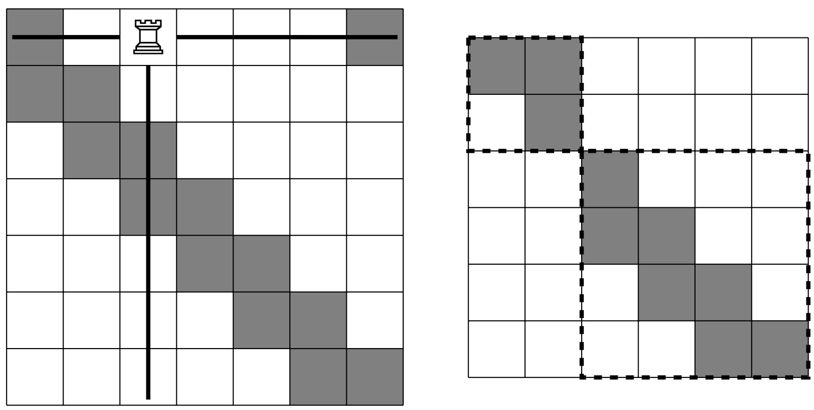


Figure 1.3: The first chessboard shows a placement of a rook at position 3, the second shows how the derived complementary board can be partitioned into two disjoint boards with 3 and 7 squares respectively.

Now we will give a name to these blocks, which are illustrated in Figure 1.4.

Definition 1.6.4. *A board is called* ***staircase-shaped*** *if it matches one of the following four shapes:*

$$\begin{aligned}
\mathcal{O}_{2n-1} &= \{(i,i) : i \in [n]\} &&\cup \{(i,i+1) : i \in [n-1]\} \\
\mathcal{O}^{\intercal}_{2n-1} &= \{(i,i) : i \in [n]\} &&\cup \{(i+1,i) : i \in [n-1]\} \\
\mathcal{E}_{2n-2} &= \{(i,i) : i \in [n-1]\} &&\cup \{(i+1,i) : i \in [n-1]\} \\
\mathcal{E}^{\intercal}_{2n-2} &= \{(i,i) : i \in [n-1]\} &&\cup \{(i,i+1) : i \in [n-1]\},
\end{aligned}$$

the subscripts represent the number of cells, and the names represent their parity.

Lemma 1.6.5. *For $\ell \geq 1$, and prefix $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_\ell)$ the derived complementary board B^c_α can be partitioned into disjoint staircase-shaped boards.*

Proof. The proof proceeds by induction on the length of the prefix.

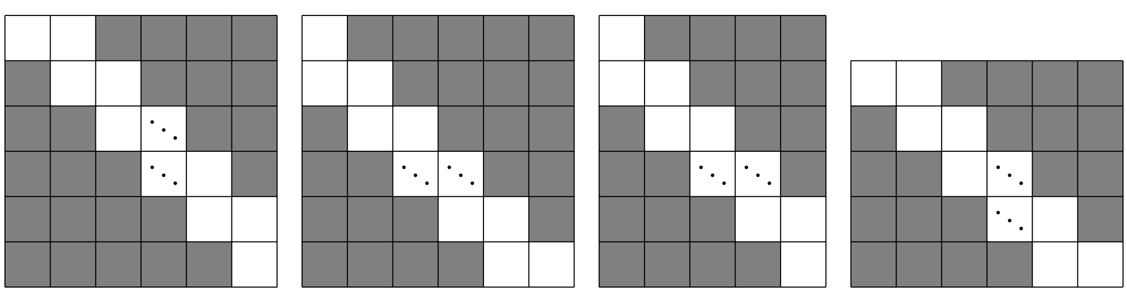
To establish the base case, consider a prefix of length $\ell = 1$. Because of the ménage restriction, $\alpha_1 \in \{2, 3, \dots, n-1\}$, and the derived complimentary board $B^c_{(\alpha_1)}$ can be partitioned into two disjoint blocks with shapes $A_{\alpha_1 - 1}$ and $A^\intercal_{n-\alpha_1}$. (This is illustrated for the case of $n = 7$ and α_1 in Figure 1.3.)

The inductive hypothesis is that the derived complimentary board for a prefix of length $\ell - 1$ consists of blocks with shape A_{n_1}, $A^\intercal_{n_2}$, B_{n_3}, or $B^\intercal_{n_4}$. Placing a rook in row ℓ can remove a top row or a column or both in a given block. Table 1.2 below shows the resulting blocks after placing a rook in ℓ-th row of B, which may be in the top row or the i-th column of a given block.

Rook placement	$\mathscr{O}_{2n-1}$	$\mathscr{O}^\intercal_{2n-1}$	$\mathscr{E}_{2n-2}$	$\mathscr{E}_{2n-2}$
Row 1	$\mathscr{O}_{2n-3}$	$\mathscr{E}^\intercal_{2n-2}$	$\mathscr{O}_{2n-3}$	$\mathscr{E}^\intercal_{2n-4}$
Column i	$\mathscr{O}_{2i-3}$, $\mathscr{E}_{2n-2i}$	$\mathscr{E}_{2i-2}$, $\mathscr{O}^\intercal_{2n-2i-1}$	$\mathscr{E}_{2i-2}$, $\mathscr{E}_{2n-2i-2}$	$\mathscr{O}_{2i-3}$, $\mathscr{O}^\intercal_{2n-2i-1}$
Row 1, column i	$\mathscr{O}_{2i-5}$, $\mathscr{E}_{2n-2i}$	$\mathscr{O}_{2i-3}$, $\mathscr{O}^\intercal_{2n-2i-1}$	$\mathscr{O}_{2i-3}$, $\mathscr{E}_{2n-2i-2}$	$\mathscr{O}_{2i-5}$, $\mathscr{O}^\intercal_{2n-2i-1}$

Table 1.2: The results of placing a rook in the first row, i-th column, or both for all staircase-shaped boards.

Therefore placing any number of rooks in the first ℓ results in a board of one of the four described shapes. □



$\mathscr{O}_{2n-1}$ $\mathscr{O}^\intercal_{2n-1}$ $\mathscr{E}_{2n-2}$ $\mathscr{E}^\intercal_{2n-2}$

Figure 1.4: Examples of each of the four staircase-shaped boards. The first two boards are on grids of size $n \times n$, the third is on a grid of size $n \times (n-1)$ and the fourth is on a grid of size $(n-1) \times n$.

1.6.3 Rook polynomials of blocks

Recall that the goal of partitioning B into disjoint boards $b_1, b_2, \ldots, b_m$ is so that we can factor $p_B(x)$ in terms of $p_{b_i}(x)$. Of course, this is only helpful if we can describe $p_{b_i}(x)$, which is the goal of this section. Thankfully, the rook polynomial of each b_i will turn out to depend only on the number of squares, $|b_i|$, which can be computed easily because of its structure.

We will begin by defining a family of polynomials that, suggestively, will turn out to be the rook polynomials that we are looking for. This family is nearly described by OEIS sequence A011973 [8].

Definition 1.6.6. *For $j \geq 0$, the jth* ***Fibonacci polynomial*** *$F_j(x)$ is defined recursively as:*

$$F_0(x) = 1 \tag{1.7}$$

$$F_1(x) = 1 + x \tag{1.8}$$

$$F_n(x) = F_{n-1}(x) + xF_{n-2}(x). \tag{1.9}$$

Lemma 1.6.7. *If B is a staircase-shaped board with k squares, then B has rook polynomial $p_B(x) = F_k(x)$ which agrees with the k-th Fibonacci polynomial.*

Proof. We will recall the recursive construction of rook polynomials from Lemma 1.4.6, and proceed by induction on the number of squares, always choosing to include or exclude the upper-left square.

Since the reflections of board has the same rook polynomial as the unreflected board, without loss of generality, we will compute the rook polynomials for $\mathscr{O}_{2n-1}$ and $\mathscr{E}_{2n-2}$ respectively.

To establish a base case, consider the rook polynomials when $n = 1$, so the even board has $|\mathscr{E}_0| = 0$ squares and the odd board has $|\mathscr{O}_1| = 1$ square. We can see the corresponding rook polynomials directly. There is 1 way to place 0 rooks on $\mathscr{E}_0$ and no ways to place more rooks;

similarly there is 1 way to place 0 rooks on $\mathscr{O}_1$, 1 way to place 1 rooks on $\mathscr{O}_1$, and no ways to place more than one rook. Thus

$$p_{\mathscr{E}_0}(x) = 1 \qquad = F_0(x), \text{ and} \tag{1.10}$$

$$p_{\mathscr{O}_1}(x) = 1 + x = F_1(x). \tag{1.11}$$

With the base case established, our inductive hypothesis is that $p_B(x) = F_h(x)$ for whenever B is a staircase-shaped boards with $h < k$ squares.

Assume that we have k squares where k is even, so our board looks like $\mathscr{E}_k$. We can either place a rook or not in the upper-left square. If we include the square, then $(\mathscr{E}_k)_i \cong \mathscr{E}_{k-2}$, if we exclude the square, then $(\mathscr{E}_k)_e \cong \mathscr{O}_{k-1}$. Thus by Lemma 1.4.6, the rook polynomial of $\mathscr{E}_k$ is

$$p_{\mathscr{E}_k}(x) = xp_{\mathscr{E}_{k-2}}(x) + p_{\mathscr{O}_{k-1}}(x) \tag{1.12}$$

$$= xF_{k-2}(x) + F_{k-1}(x) \tag{1.13}$$

$$= F_k(x). \tag{1.14}$$

The case where k is odd proceeds in almost the same way. Here our board looks like $\mathscr{O}_k$. We can either place a rook or not in the upper-left square. If we include the square, then $(\mathscr{O}_k)_i \cong \mathscr{O}_{k-2}$, if we exclude the square, then $(\mathscr{O}_k)_e \cong \mathscr{E}_{k-1}$. Again by Lemma 1.4.6, the rook polynomial of $\mathscr{O}_k$ is

$$p_{\mathscr{O}_k}(x) = xp_{\mathscr{O}_{k-2}}(x) + p_{\mathscr{E}_{k-1}}(x) \tag{1.15}$$

$$= xF_{k-2}(x) + F_{k-1}(x) \tag{1.16}$$

$$= F_k(x). \tag{1.17}$$

$\square$

1.6.4 Prefix to block sizes

Here's the idea: we group the uncrossed columns.

Lemma 1.6.8. *Given a prefix $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ and $i \notin \alpha$, the number of cells of B^c in column i that do not have a first coordinate in $[\ell]$ is given by the rule:*

$$c_i = \begin{cases} 0 & i < \ell \\ 1 & i = k \text{ or } i = n \\ 2 & \ell < i < n \end{cases} \tag{1.18}$$

Proof. TODO: This almost follows from the description? □

Now we can put these column counts together based on the continuous blocks.

Lemma 1.6.9. *TODO: Partition $[n] \setminus \alpha$ into contiguous parts. This naturally partitions B^c_α into disjoint boards. The size of these boards is $\sum_{x \in part} c_x$. (this is what I've been calling our "composition")*

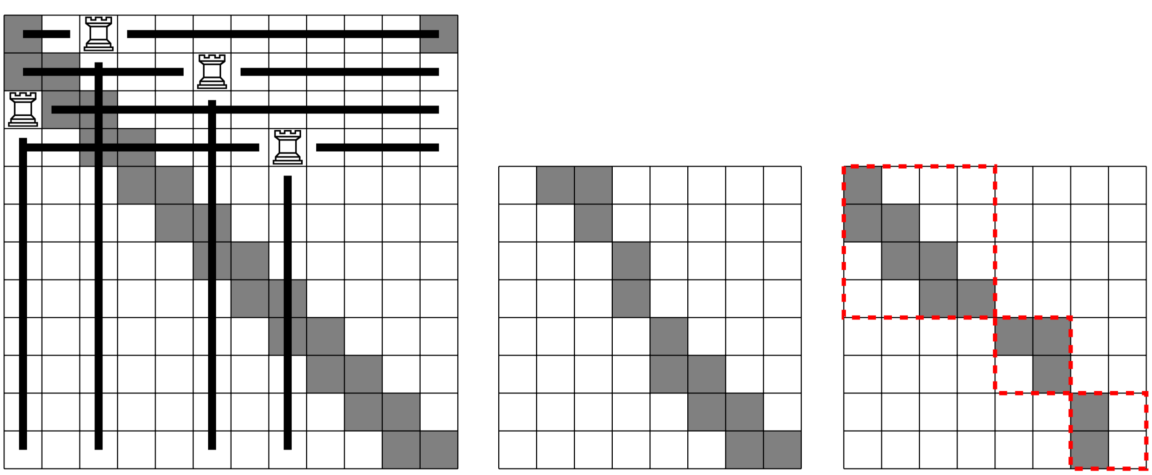


Figure 1.5: xxx

1.6.5 Complementary polynomials to ménage permutations with a given prefix

Recap: We've taken a prefix, used it to find contiguous regions, used these to find disjoint sub-boards related to B_α^c, whose rook polynomials we know. Now it's time to take these to count our number of ménage permutations with the aforementioned prefix.

Lemma 1.6.10. *Given a board B_α^c that is partitioned into disjoint boards $b_1, b_2, \ldots, b_m$, the rook polynomial of B_α^c is*

$$p_{B_\alpha^c}(x) = \prod_{i=1}^{m} F_{b_i}(x).$$

Proof. This follows directly from Lemma (TODO: rook polynomials of blocks) and Lemma (TODO: product of blocks is whole thing). □

Now that we know $p_{B_\alpha^c}$, we can use Lemma (TODO: Complementary to original) to determine how many ménage permutations there are with a given prefix. Because of Lemma (TODO: all we need is the prefix to unrank), we have an algorithm to unrank.

Example 1.6.11. *Illustrating this particular example is too big to be of much interest, so here's a smaller example. There are* A000179(8) $= 4738$ *ménage permutations on* 8 *letters. We'll use this algorithm to find the one at index* 1000*.*

1.7 Generalizations and Open Questions

1.7.1 Other restricted permutations

Doron Zeilberger considers a more general family of restricted permutations.

Definition 1.7.1 ([9]). *Let $S \subset \mathbb{Z}$, then a S-avoiding permutation is a permutation $\pi \in S_n$ such that*

$$\pi(i) - i - s \not\equiv 0 \bmod n \textit{ for all } i \in [n] \textit{ and } s \in S.$$

α	$\#\text{prefix}(\alpha)$	index range	composition	$\text{unrank}_i(\alpha,\ell)$
1	0	$(0,0]$	–	$\text{unrank}_{1000}(1,0)$
2	787	$(0,787]$	$(1,11)$	$\text{unrank}_{1000}(2,0)$
3	791	$(787,1578]$	$(3,9)$	$\text{unrank}_{1000}(3,787)$
31	0	$(787,787]$	–	$\text{unrank}_{1000}(31,787)$
32	0	$(787,787]$	–	$\text{unrank}_{1000}(32,787)$
33	0	$(787,787]$	–	$\text{unrank}_{1000}(33,787)$
34	159	$(787,946]$	$(1,7)$	$\text{unrank}_{1000}(34,787)$
35	166	$(946,1112]$	$(1,2,5)$	$\text{unrank}_{1000}(35,946)$
351	24	$(946,970]$	$(0,2,5)$	$\text{unrank}_{1000}(351,946)$
$\cdots$	0	$(970,970]$	–	
354	34	$(970,1004]$	$(0,5)$	$\text{unrank}_{1000}(354,970)$
3541	5	$(970,975]$	$(0,5)$	$\text{unrank}_{1000}(3541,970)$
3542	5	$(975,980]$	$(0,5)$	$\text{unrank}_{1000}(3542,975)$
$\cdots$	0	$(980,980]$	–	
3546	8	$(980,988]$	$(0,3)$	$\text{unrank}_{1000}(3546,980)$
3547	10	$(988,998]$	$(0,2,1)$	$\text{unrank}_{1000}(3547,988)$
3548	6	$(998,1004]$	$(0,4)$	$\text{unrank}_{1000}(3548,998)$
35481	1	$(998,999]$	$(0,4)$	$\text{unrank}_{1000}(35481,998)$
35482	1	$(999,1000]$	$(0,4)$	$\text{unrank}_{1000}(35482,999)$
354821	0	$(999,999]$	(3)	$\text{unrank}_{1000}(354821,999)$
$\cdots$	0	$(999,999]$	–	
354827	1	$(999,1000]$	$(0,1)$	$\text{unrank}_{1000}(354827,999)$
3548271	1	$(999,1000]$	(0)	$\text{unrank}_{1000}(3548271,999)$
35482716	1	$(999,1000]$	$()$	$\text{unrank}_{1000}(35482716,999)$

Table 1.3: The recursive computation of the 1000th ménage permutation.

Example 1.7.2. *Ordinary permutations are* $\emptyset$*-avoiding permutations, derangements are* $\{0\}$*-avoiding permutations, and we've defined menagé permutations as* $\{-1,0\}$*-avoiding permutations.*

The results in this paper generalize pretty easily to $\{i,i+1\}$*-avoiding permutations for all i.*

1.7.2 Observation about Lyndon Words after? a given prefix

Definition 1.7.3. *A Lyndon word is a string that is the unique minimum with respect to all of its rotations.*

Example 1.7.4. 00101 *is a Lyndon word because* $00101 = \min\{00101, 01010, 10100, 01001, 10010\}$ *is the unique minimum of all of its rotations.*

011011 *is not a Lyndon word because while* $011011 = \min\{011011, 110110, 101101, 011011, 110110, 101101\}$ *it is not the **unique** minimum.*

Conjecture 1.7.5. *Let* $\mathscr{E}^{-1}$ *denote the inverse Euler transform. Then the number of length* $n+1$ *Lyndon words that start with a prefix* α *follows a "simple" linear recurrence for sufficiently large* n*.*