Permutations, Statistics, and Switches

by

Peter Kagey

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(MATHEMATICS)

June 2022

To my brother Luke.

Miss you, bud.

Acknowledgements

To my advisor, Sami Assaf—thank you not just for your support and excitement about all of mathematical interests (recreational and otherwise), but also for your friendship, encouragement, and generosity since even before I started at USC. My time here has been a joy, and you're largely to blame for that. The way that you supported and advocated for me is an inspiration for how I want to treat my students.

To my committee member, Richard Arratia—the sort of chance encounters I regularly had with you is what I dreamed about before coming to graduate school. You've always been generous to me with your time, with your ideas, with your bounties, with your jokes, and with your great taste in problems. It was a real pleasure getting to TA for you and get to know you these past five years. I'm really going to miss having my office across the hall from yours.

To my committee member, David Kempe—It has been a blessing having your thoughtfulness and interest throughout this process. I've left every interaction with you full of new ideas and possible directions, and I hope our paths continue to cross.

I would like to thank all of my colleagues in the program, especially those teamed up to do the important work in KAP 500 week after week. I'd like to acknowledge the support that I got from my family and friends outside of the program. It's a blessing to have such an engaged and caring group of folks supporting me.

Finally, Sierra—getting to spend half of a decade with you in graduate school has been the blessing of a lifetime. You make the world a richer place, and I couldn't be more thankful to have you in my corner. You inspire me as a student, as an educator, and as a person. I can't wait to see what the rest of our lives have in store together!

Table of Contents

List of Tables

List of Figures

Abstract

This dissertation considers four distinct problems. The first topic is motivated by generalizing a puzzles called "Spinning Switches". It looks at a family of two-player games on finite wreath products, and classifies several families of wreath products where the first player has a winning strategy and several families where the second player has a winning strategy, including a complete classification for when the base of the wreath product is abelian. The second topic attempts to understand a correspondence between k-cycles in permutations and derangements of the generalized symmetric group. In this chapter, we also develop a novel insertion algorithm that preserves the number of k-cycles in the image whenever possible. The third topic is a "proof without words" that illustrates a bijection between equilateral triangles on the triangular grid and four-element subsets of integers. The fourth topic looks at efficient ways to compute the i-th permutation in for ordered lists of both derangements and ménage permutations. Here we use rook theory to compute the number of such permutations with given prefixes, and use these computations to unrank these sets.

Chapter 1

Introduction

For me, the best problem is one that could be understood by a high school student, but which requires ideas from the last few centuries of mathematics to solve. Broadly speaking, the problems in the following chapters have their roots in recreational mathematics and are reinforced and made richer by ideas in algebra, geometry, probability and combinatorics. It is my goal that this dissertation, informed by its roots, is more interesting and accessible to a broad audience than a typical dissertation in mathematics.

1.1 Elevator pitches

This first section of the introduction is aimed at doing just that: describing the three main chapters of this dissertation in informal, accessible terms, aimed at general audiences. My target audience for this section is a friend, a family member, a student, or a curious stranger.

In this dissertation—as the title promises—we will look at permutations, statistics, and switches, although not in that order. Here we give three elevator pitches, one for each of the three subsequent chapters.

1.1.1 Generalized spinning switches, informally

Chapter 2, titled "Generalized spinning switches," is about generalizing an old puzzle popularized by Marin Gardner in the late 1970's. My favorite version of this puzzle comes from Peter Winkler. We include it here, but repeat it at the beginning of the next chapter for convenience.

> Four identical, unlabeled switches are wired in series to a light bulb. The switches are simple buttons whose state cannot be directly observed, but can be changed by pushing; they are mounted on the corners of a rotatable square. At any point, you may push, simultaneously, any subset of the buttons, but then an adversary spins the square. Show that there is a deterministic algorithm that will enable you to turn on the bulb in at most some fixed number of steps. [1]

(This is a fun puzzle to work out on your own, but you can also skip to the proof of Proposition 2.1.3 to see the solution.)

We can generalize this puzzle in at least two ways. The first way we can generalize the puzzle is by asking about different kinds of switches. For instance, one could imagine a dial with three states, and only one of them is "on." On any turn we could rotate it one or two clicks to put it in one of the other states. (If we rotate it zero or three clicks, it will be in the same state as before.) You can flip ahead to Example 2.1.4 to see examples of two other kinds of switches, one with 6 states and the other with 8 states. In general, we will want to solve the problem any switch imaginable, which we claim can be modeled by arbitrary *finite groups*.

The second way we can generalize the puzzle is by looking at the number of switches and how they "spin." For example, instead of having four switches on the corners of a square table, we could have six switches on the corners of a rotating hexagon, or perhaps we could have three switches that can be scrambled however the adversary pleases. As such, we will model the adversary's scrambling as a *faithful group action* on a finite set of switches.

We want to know what combinations of "switches" and "spinning" result in a puzzle that our puzzle-solver can solve within a finite number of steps. And when such strategies exist, we want to be able to find them and understand their structure.

1.1.2 Permutations with a given number of k-cycles, informally

The next chapter, Chapter 3, titled "Permutations with a given number of k-cycles" concerns permutations. We can think of permutations as being ways of scrambling the numbers $1, 2, \ldots, n$. We're interested in certain kinds of permutations that are subject to certain restrictions, such as those scrambles where no number stays in its same place. One kind of permutation we focus on in this chapter are permutations with a certain number of *transpositions*, which means the number of unordered pairs $\{i, j\}$ such that i moves to position j and j moves to position i (i.e., i and j "swap places.")

If we look at all of the permutations on n letters with k transpositions, and we average over all of the first letters, we see something unusual: this number is closely related to squares, cubes, and higher-dimensional *hypercubes*. In particular, it relates to the number of ways of rotating or reflecting these objects in such a way that each of their sides, faces, or *facets* move to new positions.

A priori, we should be surprised that permutations with a given number of transpositions have anything to do with moving the facets of high-dimensional cubes. This chapter seeks to explore why these objects are connected, and to use this correspondence to understand both permutations and hypercubes a little bit better.

1.1.3 Triangles in triangles, informally

The following chapter, Chapter 4, is a very short chapter that demonstrates a proof technique called "a proof without words," where a concept is proven using only an illustration.

In this case, the proof without words illustrates the correspondence between equilateral triangles in the equilateral triangular grid of size n, and ways of choosing four numbers from the

list $1, 2, \ldots, n+2$. In particular, it shows that every collection of four numbers corresponds to a triangle, and vice versa.

1.1.4 Unranking restricted permutations, informally

The final chapter, Chapter 5, titled "Unranking restricted permutations," also explores permutations. Here we look at two kinds of restricted permutations: *derangements*, where the letter i is never in position i, and *ménage* permutations, where i is never in position i or in position $i-1$.

Without too much trouble, we could start writing these down in *lexicographic* (essentially, alphabetical) order. The problem is that the number of these permutations grows very quickly! For $n = 20$ there are more than 895 quadrillion derangements and 312 quadrillion ménage permutations.

In 2020 and 2021, Richard Arratia was wondering if there was a quick way of determining which derangement or ménage permutation was at a particular position in this list. As such, he posed two problems, each with $100 bounties attached: the first asking for the 500 quadrillionth derangement, and the second for the 100 quadrillionth ménage permutation. With numbers this large, it's not feasible for a computer to simply write them all down and pick the permutation in the desired position. Ultimately, finding these restricted permutations in a reasonable amount of time requires some insight into their structure.

This chapter explains the techniques that I developed in order to develop an algorithm that can find these desired permutations in a few milliseconds. These techniques are described using *rook theory*, which is a mathematical way of analyzing the number of ways rooks can be placed on a chessboard, subject to certain restrictions.

1.2 Motivations and overview

With the elevator pitches made, I will now provide an overview targeted at readers who are familiar with the jargon, and are looking for a more formal preview of what comes next.

1.2.1 Overview of spinning switches

In Chapter 2, we discuss a broad generalization of a puzzle popularized by Martin Gardner. The puzzles that I call "generalized spinning switches puzzles" are variations on the following theme: a puzzle-solver has four indistinguishable switches (or coins, or pint glasses) on the corners of a rotatable square table. Without being able to see the state of the switches, the puzzle-solver attempts to get them all in a given state. However, an adversary spins the table between each move, which means that the puzzle-solver is playing with imperfect information.

In generalized spinning switches, we have switches that behave like arbitrary finite groups G, and spinning that allows the adversary to permute the switches in ways prescribed by an arbitrary finite group H. We use the abstraction of a wreath product to neatly package these two groups into a model of a puzzle. Other authors have considered certain families of such switches, such as cyclic groups, but in this chapter we construct solutions for all groups of prime order, fully classify puzzles with switches that behave like abelian groups, and resolve the questions for large families of puzzles with nonabelian switches—including for switches that behave like the monster group, the largest of the sporadic finite simple groups.

Along the way, we provide reductions for proving that certain kinds of puzzles do not have solutions, and decompositions for constructing winning strategies from simpler puzzles.

1.2.2 Overview of permutations with a given number of k-cycles

The topic in Chapter 3 came out of a discussion that my advisor and I had about a delightful theorem of Mark Conger, where he proved that the expected value of $\pi(1)$ over all permutations π with k descents is $k+1$—and that this doesn't depend on the number of letters in the permutation.

Curious, I generated pages of tables of numerical data about the expected value of the first letter of permutations $\pi \in S_n$ such that $\text{stat}(\pi) = k$ for various other permutation statistics. Most didn't appear to have much structure, making conjectures hard to form; others had *too much* structure, making conjectures too easy to prove. But the permutation statistic that counted the number of transpositions in a permutation was in the Goldilocks zone: the denominators of these average

values appeared to be consistent as I scanned diagonally across the table. When I looked these up in the On-Line Encyclopedia of Integer Sequences, they were found to correspond to OEIS sequence A000354, which describes "$(n-1)$-dimensional facet derangements for the n-dimensional hypercube."

When I looked at the permutation statistic that counted k-cycles instead of transpositions, the tables presented a similar structure. This time, the denominators corresponded to the number of derangements of the generalized symmetric group. Ultimately I developed a full characterization of the four-parameter family: the expected value of the ℓ-th letter over all permutations $\pi \in S_n$ with exactly m k-cycles. A formula for this expected value is given both in terms of a sum, and—more satisfyingly—in terms of the number of derangements of the generalized symmetric group.

Along the way, I found two pairs of sets that were begging for a bijection. In particular, the cardinality of sets implied that there was a family of reversible insertion algorithms that preserved the number of k-cycles in most cases. A simple bijection proved to be elusive, but the recursive insertion algorithm (and its inverse) are developed in Section 3.4 of that chapter.

1.2.3 Overview of triangles in triangles

In Chapter 4, we provide a novel "proof without words" that illustrates a bijection between equilateral triangles with vertices in the n-vertices-per-side triangular grid and 4-element subsets of $[n+2]$.

1.2.4 Overview of unranking restricted permutations

In Chapter 5, we develop the solution of two related problems posed by Richard Arratia, each with a bounty attached. These problems are particular examples of a broader class of so-called *unranking* problems: given a total order on a set of combinatorial objects (that can be counted efficiently), when is it possible to efficiently compute the k-th smallest element?

In this chapter, we look at two families of restricted permutations: derangements and ménage permutations on n letters, both ordered lexicographically as words. We start by developing a more

general theory that allows us to unrank a set of words whenever we can count the number of words with a given prefix. Then we use ideas from rook theory to develop methods for counting the number of permutations in each of these two families that have a given prefix. Finally, we put these techniques together and provide an explicit, efficient algorithms for unranking derangements and ménage permutations.

Chapter 2

Generalized spinning switches

In this chapter, we explore puzzles about switches on the corners of a spinning table. Such puzzles have been written about and generalized since they were first popularized by Martin Gardner. In this chapter, we provide perhaps the fullest generalization yet, modeling both the switches and the spinning table as arbitrary finite groups combined via a wreath product. We classify large families of wreath products depending on whether or not they correspond to a solvable puzzle, completely classifying the puzzle in the case when the switches behave like abelian groups, constructing winning strategies for all wreath product that are p-groups, and providing novel examples for other puzzles where the switches behave like nonabelian groups, including the puzzle consisting of two interchangeable copies of the monster group M. Lastly, we provide a number of open questions and conjectures, and provide other suggestions of how to generalize some of these ideas further.

2.1 Overview and preliminaries

This chapter is organized into six sections. This section, Section 2.1 provides a brief history of this genre of puzzles and introduces some of the first approaches to generalizing the puzzle further. Section 2.2 models these generalizations in the context of the wreath product, and formalizes the notation of puzzles being solvable. Section 2.3 explores situations where the puzzle does not have a winning strategy, and provides reductions that allow us to prove that entire families of puzzles are not solvable. Section 2.4 constructs a strategy for switches that behave like p-groups, and gives

us ways of building strategies from smaller parts. Section 2.5 provides novel examples of puzzles that do not behave like p-groups, but still have winning strategies. Lastly, Section 2.6 provides further generalizations, and contains dozens of conjectures, open questions, and further directions.

2.1.1 History

Generalized spinning switches puzzles are a family of closely related puzzles that were first popularized by Martin Gardner in a puzzle called "The Rotating Table" in the February 1979 edition of his column "Mathematical Games" [2]. Gardner writes that he learned of the puzzle from Robert Tappay of Toronto who "believes it comes from the U.S.S.R.," a history that is not especially forthcoming.

My preferred version of the puzzle appears in Peter Winkler's 2004 book *Mathematical Puzzles A Connoisseur's Collection* [1]

> Four identical, unlabeled switches are wired in series to a light bulb. The switches are simple buttons whose state cannot be directly observed, but can be changed by pushing; they are mounted on the corners of a rotatable square. At any point, you may push, simultaneously, any subset of the buttons, but then an adversary spins the square. Show that there is a deterministic algorithm that will enable you to turn on the bulb in at most some fixed number of steps.

(Winkler's version will be a working example in many parts of this chapter, so it is worth keeping in mind. An illustration can be found in Figure 2.1.)

Over the last three decades, various authors have considered generalizations of this puzzle. Here, we build on those results and go further. The first place authors looked to generalize was suggested by Gardner himself. In his March 1979 column the following month, he provided the answer to the original puzzle and wrote

> The problem can also be generalized by replacing glasses with objects that have more than two positions. Hence the rotating table leads into deep combinatorial questions that as far as I know have not yet been explored. [3]

In 1993, Bar Yehuda, Etzion, and Moran [4]. took on the challenge and developed a theory of the spinning switches puzzle where the switches behave like roulettes with a single "on" state. In this chapter we take Gardner's charge to its logical conclusion and consider switches that behave like arbitrary "objects that have more than two positions".

Another generalization of this puzzle could look at other ways of "spinning" the switches. In 1995, Ehrenborg and Skinner [5] did this in a puzzle they call "Blind Bartender with Boxing Gloves", that analyzed this puzzle while allowing the adversary to use an arbitrary, faithful group action to "scramble" the switches. We analyze our generalized switches within this same context.

This puzzle was re-popularized in 2019 when it appeared in "The Riddler" column from the publication FiveThirtyEight [6]. Shortly after this, in 2022, Yuri Rabinovich synthesized Bar Yehuda and Ehrenborg's results in a paper that modeled the collection of switches as a vector space over a finite field, and modeled the "spinning" or "scrambling" as a faithful, linear group action on this vector space.

For more background, see Sidana's [7] detailed overview of the history of this and related problems.

2.1.2 A solution to Winkler's Spinning Switches puzzle

We will start by discussing the solution to Winkler's version of the puzzle because the solution provides some insights and intuition for the techniques that we use later. Before solving the four-switch version of the puzzle, we will make Peter Winkler proud by beginning with a simpler, two-switch version.

Example 2.1.1. *Suppose that we have two identical unlabeled switches on opposite corners of a square table, as in Figure 2.1*

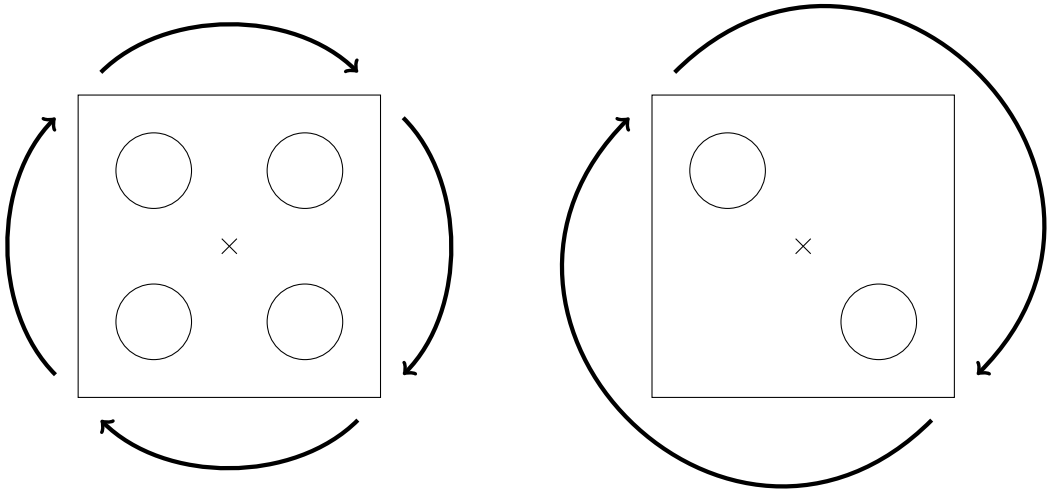


Figure 2.1: An illustration of Winkler's Spinning Switches puzzle and a two-switch analog.

Then we have a three-step solution for solving the problem. We start by toggling both switches simultaneously, and allow the adversary to spin the table. If this does not turn on the light, this means that the switches were (and still) are in different states.

Next, we toggle one of the two switches to ensure that the switches are both in the same state. If the light has not turned on, both must be in the off state.

The adversary spins the table once more, but to no avail. We know both switches are in the off state, so we toggle them both simultaneously, turning on the lightbulb.

In order to bootstrap the two-switch solution into a four-switch solution, we must notice two things:

1. First, if we can get two switches along each diagonal into the same state respectively, then we can solve the puzzle by toggling both diagonals (all four switches), followed by both switches in a single diagonal, and lastly both diagonals again. In this (sub-)strategy, toggling

both switches along a diagonal is equivalent to toggling a single switch in the two-switch analog.

2. Second, we can get both diagonals into the same state at some point by toggling a switch from each diagonal (two switches on any side of the square), followed by a single switch from one diagonal, followed by again toggling a switch from each diagonal.

We will interleave these strategies in a particular way, following the notation of Rabinovich [8].

Definition 2.1.2. *Given two sequences $A = \{a_i\}_{i=1}^{N}$ and $B = \{b_i\}_{i=1}^{M}$, we can define the **interleave** operation as*

$$A \circledast B = (A, b_1, A, b_2, A, \ldots, b_M, A) \tag{2.1}$$

$$= (\underbrace{a_1, a_2, \ldots, a_N}_{A}, b_1, \underbrace{a_1, a_2, \ldots, a_N}_{A}, b_2, \underbrace{a_1, a_2, \ldots, a_N}_{A}, \ldots, b_M, \underbrace{a_1, a_2, \ldots, a_N}_{A}). \tag{2.2}$$

which has length $(M+1)N + M = MN + M + N$.

Typically it is useful to interleave two strategies when A solves the puzzle given that the switches are in a particular state, and B gets the switches into that particular state. We also need A not to "interrupt" what B is doing. In the problem of four switches on a square table, B will ensure that the switches are in the same state within each diagonal, and A will turn on the light when that is the case. Moreover, A does not change the state *within* either diagonal.

Proposition 2.1.3. *There exists a fifteen-move strategy that guarantees that the light in Winkler's puzzle turns on.*

Proof. We begin by formalizing the two strategies. We will say that the first strategy S_1 where we toggle the two switches in a diagonal together will consist of the following three moves:

1. Switch **a**ll of the bulbs (A).

2. Switch the **d**iagonal consisting of the upper-left and lower-right bulbs (D).

3. Switch **a**ll of the bulbs (A).

We will say that the second strategy S_2 where we get the two switches within each diagonal into the same state consists of the following three moves:

1. Switch both switches on the left **s**ide (S).

2. Switch **one** switch (1).

3. Switch both switches on the left **s**ide (S).

Then the 15 move strategy is

$$S_1 \circledast S_2 = (A,D,A,S,A,D,A,1,A,D,A,S,A,D,A) \tag{2.3}$$

□

We will generalize this construction in Theorem 2.4.1, which offers a formal proof that this strategy works.

(It is worth briefly noting that $S_1 \circledast S_2$ is the fourth *Zimin word* (also called a *sequipower*), an idea that comes up in the study of combinatorics on words.)

2.1.3 Generalizing switches

Two kinds of switches are considered by Bar Yehuda, Etzion, and Moran in 1993 [4]: switches with a single "on" position that behave like n-state roulettes ($\mathbb{Z}_n$) and switches that behave like the finite field $\mathbb{F}_q$, both on a rotating k-gonal table. We generalize this notion further by considering switches that behave like arbitrary finite groups.

Example 2.1.4. *In Figure 2.2, we provide a schematic for a switch that behaves like the symmetric group S_3. It consists of three identical-looking parts that need to be arranged in a particular order in order for the switch to be on.*

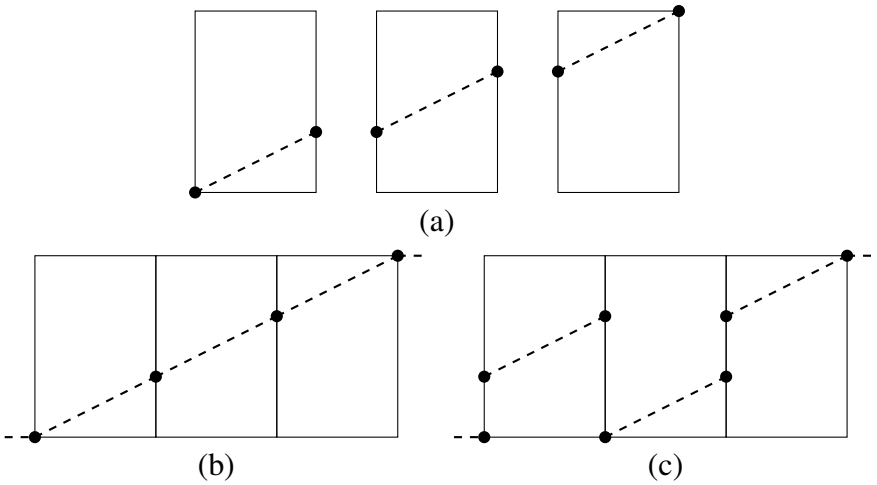


Figure 2.2: Part (a) shows a simple schematic for the components of a switch that behaves like S_3, the symmetric group on three letters. The three rectangles can be permuted arbitrarily, but only configuration (b) completes the circuit. All other configurations fail to complete the circuit (e.g., (c)).

We could also construct a switch that behaves like the dihedral group of the square, D_8. This switch might look like a flat, square prism that can slot into a square hole, such that only one of the $|D_8| = 8$ rotations of the prism completes the circuit.

Note 2.1.5. *One subtlety of using a group G to model a switch is that both the "internal state" of a switch itself and the set of "moves" or changes are modeled by G. Therefore it may be useful to think of the state as the underlying set of G where the moves act via a right group action of G on itself.*

The reason that it is appropriate to use a group to model a switch is because groups have many of the properties we would expect in a desirable switch.

Note 2.1.6. *The axioms for a group $(G,\cdot)$ closely follow what we would expect from a switch.*

1. (Closure) The group $(G,\cdot)$ is equipped with a binary operation, $\cdot\colon G\times G\to G$. That is, for all pairs of elements $g_1, g_2 \in G$ their product is in G

$$g_1 \cdot g_2 \in G. \tag{2.4}$$

In the context of switches, this means that if the switch is in some state $g_1 \in G$ and the puzzle-solver applies the move $g_2 \in G$ to it, then the resulting state $g_1 \cdot g_2 \in G$ is in the set of possible states.

2. (Identity) There exists an element $\mathrm{id}_G \in G$ such that for all $g \in G$,

$$\mathrm{id}_G \cdot g = g \cdot \mathrm{id}_G = g. \tag{2.5}$$

This axiom is useful because it means that the puzzle-solver can "do nothing" to a switch and leave it in whatever state it is in. Because the identity is a distinguished element in G, we will also use the convention that id_G is the "on" or "winning" state for a given switch. (It is worth noting that all of the arguments work basically the same way regardless of which element is designated as the on state.)

3. (Inverses) For each element $g \in G$ there exists an inverse element $g^{-1} \in G$ such that

$$g \cdot g^{-1} = g^{-1} \cdot g = \mathrm{id}_G\,. \tag{2.6}$$

This axiom states that no matter what state a switch is in, there is a move that will transition it into the on state.

4. (Associativity) Given three elements $g_1, g_2, g_3 \in G$,

$$(g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3) \tag{2.7}$$

This axiom is not strictly necessary for modeling switches, but as we will see in a later definition, it gives us a convenient way to describe the conditions for a winning strategy. (In Subsection 2.6.3, we briefly discuss dropping the associativity axiom by considering switches that behave like quasigroups with identity.)

2.1.4 Generalizing spinning

We can also consider generalizations of "spinning" the switches. In particular, we will adopt the generalization from Ehrenborg and Skinner's [5] 1995 paper, which use arbitrary faithful group actions to permute the switches. In particular, they provide a criterion that determines which group actions yield a winning strategy in the case of a given number of "ordinary" switches (those that behave like $\mathbb{Z}_2$). Rabinovich [8] stretches these results a bit further and looks at faithful linear group actions on collections of switches that are modeled as a finite-dimensional vector space over a finite field. We build on this result in the context of more general switches.

2.2 A wreath product model

Recall that Peter Winkler's Spinning Switches puzzle consists of four two-way switches on the corners of a rotating square table. The behavior of the switches are naturally modeled as $\mathbb{Z}_2$, and the rotating table is modeled as the cyclic group C_4. The abstraction that takes these two groups and creates a model for the underlying puzzle is the wreath product: that is, Winkler's puzzle behaves like the wreath product of $\mathbb{Z}_2$ by C_4.

2.2.1 Modeling generalized spinning switches puzzles

We do not evoke wreath products arbitrarily: we use them because they are the right abstraction to model a generalized spinning switches puzzle where G describes the behavior of the switches, Ω describes the positions of the switches, and the action of H on Ω models the ways the adversary can permute the switches.

Definition 2.2.1 ([9])**.** *Let G and H be groups, let Ω be a finite H-set, and let $K = \prod_{\omega\in\Omega} G_\omega$, where $G_\omega \cong G$ for all $\omega \in \Omega$. Then the **wreath product** of G by H denoted by $G \wr H$, is the semidirect product of K by H, where H acts on K by $h \cdot (g_\omega) = g_{h^{-1}\omega}$ for $h \in H$ and $g_\omega \in G_\omega$. The normal subgroup K of $G \wr H$ is called the **base** of the wreath product.*

The group operation is $(k,h)\cdot(k',h') = (k(h\cdot k'), hh')$

An element $(k,h) \in G \wr H$ represents a turn of the game: The puzzle-solver chooses an element of the base $k \in K$ to indicate how they want to modify each of their switches and then the adversary chooses $h \in H$ and acts with h on Ω to permute the switches.

Example 2.2.2. *Consider the setup in the Winkler's Spinning Switches the puzzle, which consists of two-way switches ($\mathbb{Z}_2$) on the corners of a rotating square, $C_4 \cong \langle 0^\circ, 90^\circ, 180^\circ, 270^\circ \rangle$. The game itself corresponds to the wreath product $\mathbb{Z}_2 \wr C_4$. We will use the convention that the base of the wreath product, K, is ordered upper-left, upper-right, lower-right, lower-left, and that the group action is specified by degrees in the clockwise direction.*

Consider the following two turns:

1. *During the first turn, the puzzle-solver toggles the upper-left and lower-right switches, and the adversary rotates the table 90° clockwise. This is represented by the element*

$$((1,0,1,0), 90^\circ) \in \mathbb{Z}_2 \wr C_4. \tag{2.8}$$

2. *During the second turn, the puzzle-solver toggles the upper-left switch, and the adversary rotates the table 90° clockwise. This is represented by the element*

$$((1,0,0,0), 180^\circ) \in \mathbb{Z}_2 \wr C_4. \tag{2.9}$$

As illustrated in Figure 2.3, the net result of these two turns is the same as a single turn where the puzzle-solver toggles the upper-left, upper-right, and lower-left switches and the adversary rotates the board 270° clockwise.

The multiplication under the wreath product agrees with this:

$$\begin{aligned}((1,0,1,0), 90^\circ) \cdot ((1,0,0,0), 180^\circ) &= ((1,0,1,0) + \underbrace{90^\circ \cdot (1,0,0,0)}_{(0,0,0,1)}, 90^\circ + 180^\circ) \\ &= ((1,0,1,1), 270^\circ)\end{aligned}$$

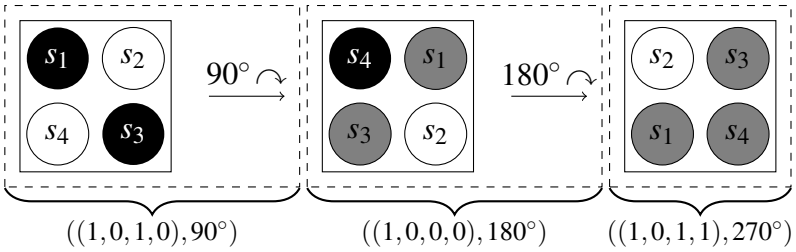


Figure 2.3: An illustration of two turns in Winkler's Spinning Switches puzzle, modeled as elements of a wreath product.

As suggested earlier, it is occasionally useful to designate a particular state of the switches as the winning state. We will use the convention that the lightbulb turns on when all of the switches are equal to the identity, that is, $\mathrm{id}_K \in K$. It is worth noting, however, that the existence of a winning strategy does not depend on a particular choice of the winning state. Instead, we will see that a winning strategy is equivalent to a choice of moves that will walk over all of the possible configuration states, regardless of the choice of the adversary's spin.

2.2.2 Switching strategy

We will begin by formalizing the notation of a winning strategy in a generalized spinning switches puzzle. Informally, this is a sequence of moves that the puzzle-solver can make that will put the switches into every possible state, which ensures that the winning state is reached regardless of the initial (hidden) state of the switches.

Definition 2.2.3. *A **switching strategy** for $G \wr H$ is a finite sequence of elements in the base K, $\{k_i \in K\}_{i=1}^{N}$, such that for every sequence of elements in H, $\{h_i \in H\}_{i=1}^{N}$,*

$$p(\{\underbrace{\mathrm{id}_{G\wr H}}_{m_0}, \underbrace{(k_1,h_1)}_{m_1}, \underbrace{(k_1,h_1)\cdot(k_2,h_2)}_{m_2}, \ldots, \underbrace{(k_1,h_1)\cdot(k_2,h_2)\cdots(k_N,h_N)}_{m_N}\}) = K. \tag{2.10}$$

where $p\colon G \wr H \to K$ is the projection map from the wreath product onto its base.

This definition is useful because it puts the problem into purely algebraic terms. It is also useful because it abstracts away the initial state of the switches: regardless of the initial state $k \in K$, the

existence of a switching strategy means that its inverse $k^{-1} \in K$ appears in the sequence. (This follows the convention that $\mathrm{id}_K \in K$ is designated as the winning state. If k' is chosen to be the winning state, then the sequence must contain $k^{-1}k'$.)

Proposition 2.2.4. *A finite sequence of moves is guaranteed to reach the winning state if and only if it is a switching strategy.*

Proof. Without loss of generality, we will say that the winning state for the switches is id_K.

We will begin by assuming that $\{k_i \in K\}_{i=1}^N$ sequence of moves is guaranteed to reach the winning state, id_K. In the puzzle, we have an initial (hidden) state, $k \neq \mathrm{id}_K$. Therefore, after the i-th move, the wreath product element that represents the state of the switches is

$$p\left((k, \mathrm{id}_H) \cdot (k_1, h_1) \cdot (k_2, h_2) \cdots (k_i, h_i)\right) = k \cdot p\left((k_1, h_1) \cdot (k_2, h_2) \cdots (k_i, h_i)\right), \tag{2.11}$$

where the equality is due to associativity of the wreath product. We can factor out the first term because the "spin" is id_H, which acts trivially: $(k, \mathrm{id}_H) \cdot (k', h') = (kk', h')$.

Say that j is the index at which the puzzle-solver gets the switches into the winning state. Then

$$k \cdot p\left((k_1, h_1) \cdot (k_2, h_2) \cdots (k_j, h_j)\right) = \mathrm{id}_K \tag{2.12}$$

$$p\left((k_1, h_1) \cdot (k_2, h_2) \cdots (k_j, h_j)\right) = k^{-1}. \tag{2.13}$$

Since $k \neq$ is arbitrary, there must exist such a j for every $k \in K$ and adversarial sequence $\{h_i \in H\}_{i=1}^N$. This means that the projection of the sequence of partial products of $\{k_i \in K\}_{i=1}^N$ is a surjection onto K and therefore is a switching strategy.

Conversely, if K is a switching strategy, then for any initial state k and sequence of adversarial moves $\{h_i \in H\}_{i=1}^N$, there exists some j that satisfies Equation (2.13), and therefore reaches a winning state. □

It is also worth noting that this model can be thought of as a random model or an adversarial model: the sequence $\{h_i \in H\}$ can be chosen randomly at any point, or it can be chosen deterministically after the sequence $\{k_i \in K\}$ is specified.

2.2.3 Bounds on the length of switching strategies

One useful consequence of this definition is that it is quite straightforward to prove certain propositions. For example, the minimum length for a switching strategy has a simple lower bound.

Proposition 2.2.5. *Every switching strategy $\{k_i \in K\}_{i=1}^{N}$ is a sequence of length at least $|K|-1$.*

Proof. This follows from an application of the Pigeonhole Principle, because the set

$$\{\mathrm{id}_{G\wr H}, (k_1,h_1), (k_1,h_1)\cdot(k_2,h_2), \ldots, (k_1,h_1)\cdot(k_2,h_2)\cdots(k_N,h_N)\} \tag{2.14}$$

has at most $N+1$ elements. In order for the projection to be equal to K,

$$p(\{\mathrm{id}_{G\wr H}, (k_1,h_1), (k_1,h_1)\cdot(k_2,h_2), \cdots, (k_1,h_1)\cdot(k_2,h_2)\cdots(k_N,h_N)\}) = K, \tag{2.15}$$

it must be the case that $N+1 \geq |K|$. Therefore $N \geq |K|-1$. □

Minimal length switching strategies are common, so we give them a name.

Definition 2.2.6. *A **minimal switching strategy** for $G \wr H$ is a switching strategy of length $N = |K|-1$.*

In practice, every wreath product known to the author to have a switching strategy also has a known minimal switching strategy. In Section 2.6, we ask whether this property always holds.

2.3 Reductions

In this section, we use three techniques to develop examples of generalized spinning switches puzzles that do not have switching strategies: directly, by a reduction on switches, or by a reduction on spinning.

2.3.1 Puzzles known to have no switching strategies

Our richest collection of known puzzles without switching strategies comes from a theorem of Rabinovich, which models switches as a vector space over a finite field.

Theorem 2.3.1. *[8] Assume that a finite "spinning" group H acts linearly and faithfully on a collection of switches that behave like a vector space V over a finite field $\mathbb{F}_q$ of characteristic p. Then the resulting puzzle has a switching strategy if and only if H is a p-group.*

It is worth noting that Rabinovich's switches are less general than arbitrary finite groups, but the "spinning" is more general: in addition to permuting the switches, the group action might add linear combinations of them as well.

Example 2.3.2. *By the theorem of Rabinovich [8], the game $\mathbb{Z}_2 \wr C_3$ does not have a switching strategy. In Rabinovich's notation, the vector space of switches is $\mathbb{Z}_2^3$ over the field $\mathbb{F}_2 = \mathbb{Z}_2$, and C_3 has 3 elements and therefore is not a 2-group.*

The wreath product $\mathbb{Z}_2 \wr C_3$ is perhaps the simplest example of a generalized spinning switches puzzle without a switching strategy, so we will continue to use it as a basis of future examples.

2.3.2 Reductions on switches

With Theorem 2.3.1 providing a family of wreath products without switching strategies, we now introduce a theorem that allows us to describe large families of wreath products that also do not have switching strategies.

Theorem 2.3.3. *If $G \wr H$ does not have a switching strategy and there exists a group G' and a surjective homomorphism $\varphi\colon G' \to G$, then $G' \wr H$ does not have a switching strategy.*

Proof. We will prove the contrapositive, and suppose that $G' \wr H$ has base K' and a switching strategy $\{k'_i \in K'\}_{i=1}^{N}$.

The homomorphism $\varphi\colon G' \to G$ extends coordinatewise to $\varphi\colon K' \to K$, which further extends in the first coordinate to $G \wr H$: $\varphi(k,h) := (\varphi(k),h)$.

It is necessary to verify that $\varphi\colon G' \wr H \to G \wr H$ is indeed a homomorphism.

$$\begin{aligned}
\varphi((k'_\alpha,h_\alpha)) \cdot \varphi((k'_\beta,h_\beta)) &= (\varphi(k'_\alpha),h_\alpha) \cdot (\varphi(k'_\beta),h_\beta) \\
&= (\varphi(k'_\alpha)(h_\alpha \cdot \varphi(k'_\beta)),h_\alpha h_\beta) \\
&= (\varphi(k'_\alpha)\varphi(h_\alpha \cdot k'_\beta),h_\alpha h_\beta) \\
&= (\varphi(k'_\alpha(h_\alpha \cdot k'_\beta)),h_\alpha h_\beta) \\
&= \varphi((k'_\alpha(h_\alpha \cdot k'_\beta),h_\alpha h_\beta)) \\
&= \varphi((k'_\alpha,h_\alpha) \cdot (k'_\beta,h_\beta))
\end{aligned}$$

Therefore the sequence $\{\varphi(k'_i) \in K\}_{i=1}^{N}$ is a switching strategy on $G \wr H$, because the quotient map $\varphi\colon G' \to G$ (and thus $\varphi\colon K' \to K$) is injective. □

Example 2.3.4. *We know that $\mathbb{Z}_2 \wr C_3$ does not have a switching strategy. This means that $\mathbb{Z}_6 \wr C_3$ does not have a switching strategy either, as illustrated in Figure 2.4.*

2.3.3 Reductions on spinning

We can do two similar reductions on the "spinning" group of a wreath product. These theorems say that if a given wreath product $G \wr H$ does not have a switching strategy, then a similar wreath product $G \wr H'$ with a "more complicated" spinning group H' will not have a switching strategy either.

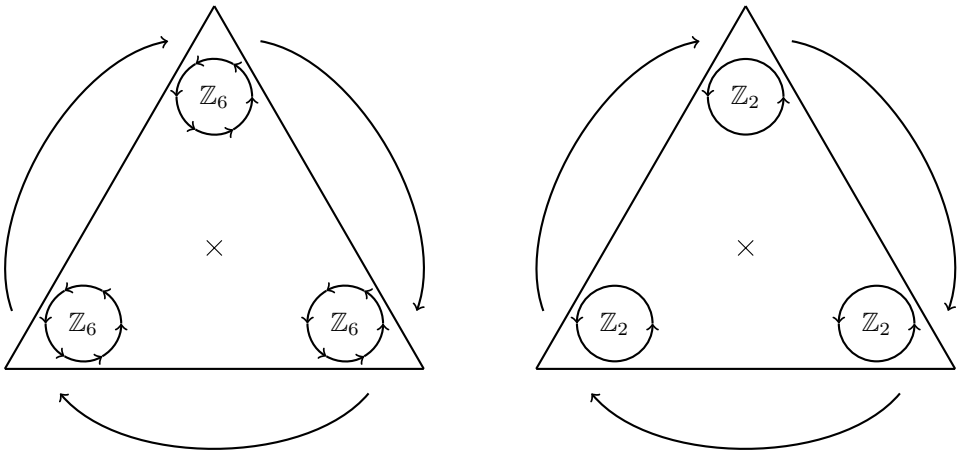


Figure 2.4: A reduction on switches: $\mathbb{Z}_6 \wr C_3$ reduces to $\mathbb{Z}_2 \wr C_3$, which is known not to have a switching strategy.

Theorem 2.3.5. *If $G \wr H$ does not have a switching strategy, and $\varphi\colon H \hookrightarrow H'$ is an embedding of H into H', then $G \wr H'$ does not have a switching strategy.*

Proof. Again we will prove the contrapositive. Assume that $G \wr H'$ does have a switching strategy, $\{k_i\}_{i=1}^N$. Then by definition, for any sequence $\{h_i'\}_{i=1}^N$, the projection of the sequence

$$p(\{(k_1,h_1')\cdot(k_2,h_2')\cdots(k_i,h_i')\}_{i=1}^N) = \left\{p\left((k_1,h_1')\cdot(k_2,h_2')\cdots(k_i,h_i')\right)\right\}_{i=1}^N = K, \tag{2.16}$$

and in particular this is true when h_i' is restricted to be in the subgroup $\mathrm{Im}(\varphi) \leq H'$. Thus a switching strategy for $G \wr H'$ is also a valid switching strategy for $G \wr H$. □

Example 2.3.6. *Consider the wreath product $\mathbb{Z}_2 \wr_{\Omega_6} C_6$ where Ω_6 consists of six switches on the corners of a hexagon as illustrated in Figure 2.5. We know that $\mathbb{Z}_2 \wr_{\Omega_6} C_3$ does not have a switching strategy, by Theorem 2.3.1, therefore $\mathbb{Z}_2 \wr_{\Omega_6} C_6$ cannot have a switching strategy either, by Theorem 2.3.5.*

In the above example, we noted that $\mathbb{Z}_2 \wr_{\Omega_6} C_6$ does not have a switching strategy because $\mathbb{Z}_2 \wr_{\Omega_6} C_3$ is known not to have one. However, there is another obstruction: the fact that the generalized spinning switches puzzle $\mathbb{Z}_2 \wr C_3$ on a triangle is known not to have a switching strategy.

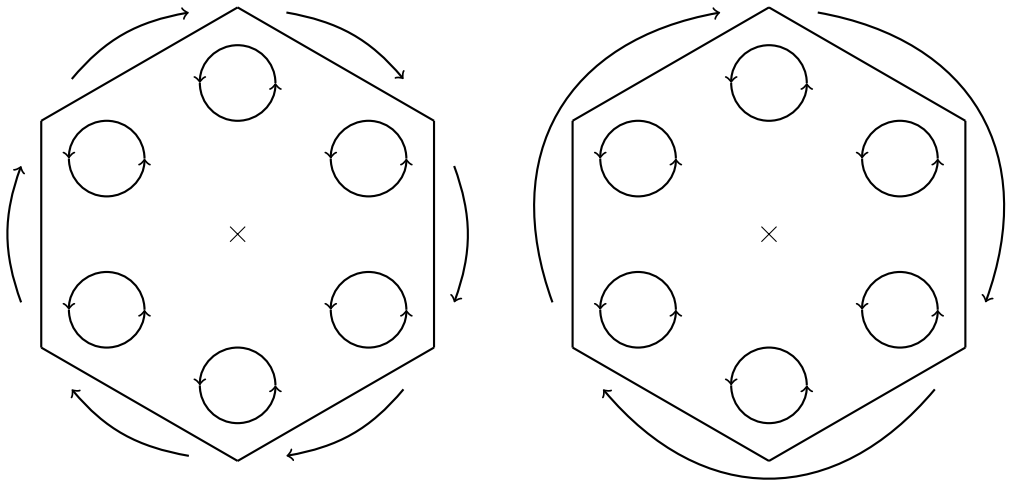


Figure 2.5: If there were a solution to $\mathbb{Z}_2 \wr_{\Omega_6} C_6$, then there would be a solution to $\mathbb{Z}_2 \wr_{\Omega_6} C_3$.

That is, we cannot guarantee that the upright triangle (which is formed by taking "every other" switch) is ever in the all "on" state.

The following theorem abstracts this idea.

Theorem 2.3.7. *Suppose that H and H' are groups, $\varphi\colon H \hookrightarrow H'$ is an embedding of H into H', and let*

$$\mathrm{Orb}(\omega) = \{\omega \cdot a : a \in \mathrm{Im}(\varphi)\} \subseteq \Omega \tag{2.17}$$

be the (right) orbit of $\omega \in \Omega$ under $\mathrm{Im}(\varphi)$. If $G \wr_{\mathrm{Orb}(\omega)} H$ does not have a switching strategy, then $G \wr_\Omega H'$ cannot have a switching strategy either.

Proof. We start by making the contrapositive assumption that $G \wr_\Omega H'$ has a switching strategy $\{k_i \in K\}_{i=1}^N$, and we consider the projection $p_\omega\colon K \to K_\omega$ where

$$K = \prod_{\omega' \in \Omega} G_{\omega'} \qquad \text{and} \qquad K_\omega = \prod_{\omega' \in \mathrm{Orb}(\omega)} G_{\omega'}. \tag{2.18}$$

Then $\{p_\omega(k_i) \in K_\omega\}_{i=1}^N$ is a switching strategy for $G \wr_{\mathrm{Orb}(\omega)} H$, since the projection is a surjective map. □

Example 2.3.8. *We know that $\mathbb{Z}_2 \wr C_3$ does not have a switching strategy. This means that $\mathbb{Z}_2 \wr C_6$ does not have a switching strategy either, as illustrated in Figure 2.6.*

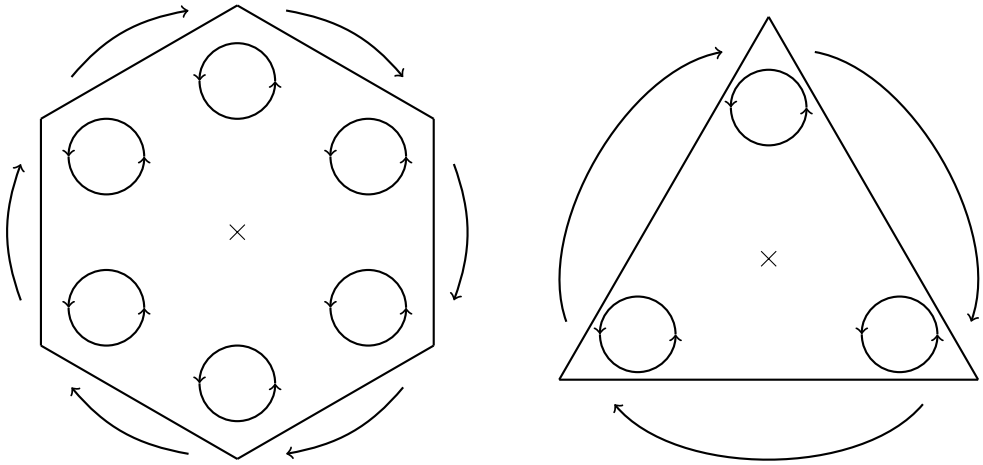


Figure 2.6: We know that $\mathbb{Z}_2 \wr_\Omega C_6$ cannot have a switching strategy, because that would imply a switching strategy for $\mathbb{Z}_2 \wr_{\Omega'} C_3$, where Ω' is the orbit of the top switch rotations of multiples of 120°.

Now that we have proven that large families of wreath products do not have switching strategies, it is worthwhile to construct families of wreath products that do have switching strategies.

2.4 Switching strategies on p-groups

In this section, we will develop a broad family of switching strategies, namely those where G and H (and thus $G \wr H$) are p-groups.

2.4.1 Switching strategy decomposition

Our first constructive theorem provides a technique that can be used to construct switching strategies for switches that behave like a group G in terms of a normal group and its corresponding quotient group.

Theorem 2.4.1. *The wreath product $G \wr H$ has a switching strategy if there exists a normal subgroup $N \trianglelefteq G$ such that both $N \wr H$ and $G/N \wr H$ have switching strategies.*

Proof. Let

$$S_{G/N} = \{k_i^{G/N} \in K_{G/N}\} \qquad \text{and} \qquad S_N = \{k_i^N \in K_N\}$$

denote the switching strategies for $G/N \wr H$ and $N \wr H$ respectively.

We ultimately would like to interleave these two strategies, but $k_i^{G/N} \notin K_G$. To find the appropriate analog, we partition G into $[G:N] = m$ right cosets of N,

$$G = Ng_1 \sqcup Ng_2 \sqcup \cdots \sqcup Ng_m, \tag{2.19}$$

each with a chosen representative in G. Now we define a map $r\colon G/N \to G$ that chooses the chosen representative of the coset, and extends coordinatewise. We use this map to define a sequence $S = \{r(k_i^{G/N}) \in K_G\}$.

We claim that these two sequences interleaved, $S_N \circledast S$, forms a switching strategy for $G \wr H$. To prove this claim, we observe two facts:

1. Multiplying by elements of S_N will not change cosets and will walk through every element of the current coset.

2. Multiplying by elements of S will walk through all cosets.

Therefore the interleaved sequence will walk through all elements of each coset, and thus is surjective onto K. □

2.4.2 Construction of switching strategies on p-groups

With the decomposition from Theorem 2.4.1 established, we can now construct a switching strategy on all finite p-groups.

Theorem 2.4.2. *If H is a finite p-group that acts faithfully on Ω, then the wreath product $G \wr H$ has a switching strategy whenever $|G| = p^n$ for some n.*

Proof. We will use the fact that if $|G| = p^n$, then either $G \cong \mathbb{Z}_p$ or G is not simple.

If $n = 1$, then $G \cong \mathbb{Z}_p \cong \mathbb{F}_p$, so there exists a switching strategy by Theorem 2.3.1. This is because H permutes the coordinates of $V = \mathbb{F}_p^{|\Omega|} \cong K$, and so it is a linear action on the vector space, and so it has a switching strategy by Theorem 2.3.1.

Otherwise, G is not simple. This means that G has a proper normal subgroup N of order $|N| = p^t$ (with $0 < t < n$) and a quotient G/N with order $|G/N| = p^{n-t}$. Therefore, by induction on the exponent, whenever G and H (and thus $G \wr H$) are p-groups, $G \wr H$ has a switching strategy. □

This resolves the situation for switches that behave like p-groups on spinning groups that act faithfully and behave like p-groups. This allows us to fully classify the case of abelian switches.

2.4.3 A classification of puzzles with abelian switches

Theorem 2.4.3. *If G is an abelian group, and H acts faithfully on Ω, then $G \wr_\Omega H$ has a switching strategy if and only if G and H are both p-groups for the same prime p.*

Proof. One direction is clear: if G and H are both p-groups for the same prime p, then $G \wr H$ has a switching strategy by Theorem 2.4.2.

For the other direction, we will consider the contrapositive assumption, split into three cases, each time assuming that G and H are not both p-groups for the same prime p.

First, assume that G is a p-group, but H is not a p-group. Then by Sylow's first theorem, there exists a subgroup $\hat{H} \leq H$ that is a q-group for some prime $q \neq p$. By Theorem 2.3.1, since $q \neq p$, $G \wr \hat{H}$ does not have a switching strategy, thus it follows from the reduction on spinning given in Theorem 2.3.5 that $G \wr H$ does not have a switching strategy either.

Second, assume that H is a p-group, but G is not a p-group. Similarly, by Sylow's first theorem, there exists a subgroup $\hat{G} \leq G$ that is a q-group for some prime $q \neq p$. Since G is abelian, $\hat{G}$ is a normal subgroup. By Theorem 2.3.1, since $q \neq p$, $\hat{G} \wr H$ does not have a switching strategy, thus it follows from the reduction on switches given in Theorem 2.3.3 that $G \wr H$ does not have a switching strategy either.

Lastly, assume that there does not exist any p such that G is a p-group or H is a p-group. Then $|G|$ and $|H|$ must have multiple prime divisors, so G has a normal subgroup, $\overline{G}$, that is a q-group for some prime q, and H has a normal subgroup $\overline{H}$ that is an r-group for some prime $r \neq q$. $\overline{G} \wr \overline{H}$

does not have a switching strategy by Theorem 2.3.1, so $G \wr \overline{H}$ does not have a switching strategy by Theorem 2.3.3, so $G \wr H$ does not have a switching strategy by Theorem 2.3.5. □

2.4.4 A folklore conjecture

Here we note a conjecture from folklore, which—if true—implies that we have *almost* solved the problem in its full generality.

Conjecture 2.4.4 (Folklore)**.** *Almost all groups are* 2*-groups.*

There are both computational and theoretical bases for this conjecture. According to the On-Line Encyclopedia of Integer Sequences [10], there are $A000001(2^{10}) = 49487367289$ groups of order 2^{10} and there are $A063756(2^{11}-1) = 49910536613$ groups of order less than 2^{11}. This means that more than 99.15% of the groups of order less than 2^{11} are of order 2^{10}.

If this conjecture is true, then most types of switches have switching strategies on most kinds of faithful finite group actions. Of course, while most finite groups may be 2-groups, most mathematicians are more interested in the other finite groups. This next section develops two families of examples of switching strategies where the switches do not behave like p-groups.

2.5 Switching strategies on other wreath products

Other authors have given switching strategies for various configurations of generalized spinning switches strategies, but in all of these examples, the wreath products themselves are p-groups: that is, $|G \wr_\Omega H| = |G|^{|\Omega|} \cdot |H|$, where H acts faithfully. In this section we introduce two families of wreath products that have switching strategies, but are not p-groups.

2.5.1 The trivial wreath product, $G \wr 1$

The simplest—and least interesting—way to construct a wreath product with a switching strategy is to remove the adversary (or randomness) altogether, by letting the spinning group be the trivial

group $H = \mathbf{1}$. Additionally we will consider the case where $|\Omega| = 1$, so there is only one switch. Because the adversary cannot "spin" the switches at all, the puzzle-solver has perfect information the entire time. We will see that whenever G is a finite group, $G \wr \mathbf{1} \cong G$ has not just one switching strategy, but many.

Proposition 2.5.1. *The wreath product $G \wr \mathbf{1}$ has $(|G|-1)!$ minimal switching strategies.*

Proof. There are $(|G|-1)!$ permutations of $G \setminus \{\mathrm{id}_G\}$, and we claim each one corresponds to a minimal switching strategy. Namely, if $(k_1, k_2, \ldots, k_{|G|-1})$ is such a permutation, then the sequence $\{k'_i\}_{i=1}^{|G|-1}$ where $k'_1 = k_1$ and $k'_i = k_{i-1}^{-1} k_i$ is a switching strategy on $G \wr \mathbf{1}$.

Then we claim by induction that $(k'_1, \mathrm{id}) \cdot (k'_2, \mathrm{id}) \cdots (k'_j, \mathrm{id}) = (k_j, \mathrm{id})$. By construction, the base case is true when $j = 1$. If the claim holds up to $j-1$, then

$$\underbrace{(k'_1, \mathrm{id}) \cdot (k'_2, \mathrm{id}) \cdots (k'_{j-1}, \mathrm{id})}_{(k_{j-1}, \mathrm{id})} (k'_j, \mathrm{id}) = (k_{j-1}, \mathrm{id})(k_{j-1}^{-1} k_j, \mathrm{id}) = (k_j, \mathrm{id}), \tag{2.20}$$

as desired. Thus the projection of the partial products is

$$p(\{\underbrace{\mathrm{id}_{G \wr \mathbf{1}}}_{m_0}, \underbrace{(k'_1, h_1)}_{m_1}, \underbrace{(k'_1, h_1) \cdot (k'_2, h_2)}_{m_2}, \cdots, \underbrace{(k'_1, h_1) \cdot (k'_2, h_2) \cdots (k'_{|G|-1}, h_{|G|-1})}_{m_{|G|-1}}\}) \tag{2.21}$$

$$= p(\{\mathrm{id}_{G \wr \mathbf{1}}, (k_1, \mathrm{id}), (k_2, \mathrm{id}), \cdots, (k'_{|G|-1}, \mathrm{id})\}) \tag{2.22}$$

$$= \{\mathrm{id}_{G \wr \mathbf{1}}, k_1, k_2, \cdots, k_{|G|-1}\} = K, \tag{2.23}$$

where $\{k_1, k_2, \ldots, k_{|G|-1}\}$ spans $G \setminus \{\mathrm{id}_G\} \cong K \setminus \{\mathrm{id}_K\}$ by assumption. □

While the trivial wreath product is a useful example to keep in mind for generating counterexamples, we are generally more interested in the situation where the adversary permutes the switches to create uncertainty for the puzzle-solver.

2.5.2 Two interchangeable groups generated by involutions

In this section, we will construct a switching strategy for the generalized spinning switches puzzle $G \wr H$ that consists of two switches each behaving like a group G, generated by involutions, together with a nontrivial spinning action H. (See, for example, Figure 2.2 which illustrates switches that behave like $S_3 = \langle (12), (13) \rangle$)

This strategy relies on the fact that because each generator is its own inverse, applying a generator to the first switch or the second switch has no effect on their difference.

This switching strategy has two parts. The first part ensures that the two switches have every possible difference. The second part shows that we can get the first switch (with respect to the projection onto K) to take on every possible value without changing the difference between the switches.

Theorem 2.5.2. *Suppose that G is a finite group that can be generated by involutions. Then the generalized spinning switches puzzle $G \wr C_2$ consisting of two interchangeable copies of G has a switching strategy.*

Proof. We start by writing G in terms of its generators: $G = \langle t_1, t_2, \ldots, t_N \rangle$, where $t_i^{-1} = t_i$. Because this is the generating set, there exists a finite sequence of transpositions $(t_{i_1}, t_{i_2}, \ldots, t_{i_M})$ such that the partial products of the sequence generate G:

$$G = \{\mathrm{id}_G, t_{i_1}, t_{i_1} t_{i_2}, \ldots, t_{i_1} t_{i_2} \cdots t_{i_M}\}. \tag{2.24}$$

We develop the strategy in two parts. First, we provide a strategy $A = \{\alpha_i \in K\}$ such that for any adversarial sequence $\{h_i \in H\}$ and element $g \in G$, there exists an $i \geq 0$ such that the i-th partial product, $(g_{i,1}, g_{i,2}) = p((\mathrm{id}_K, \mathrm{id}_H) \cdot (\alpha_1, h_1) \cdot (\alpha_2, h_2) \ldots (\alpha_i, h_i))$, has a difference of g. That is, $g_{i,1} g_{i,2}^{-1} = g$.

To do this, we define $\alpha_j = (t_{i_j}, \mathrm{id}_G) \in K$, and notice that the difference of the coordinates is the same whether we add α_j or $(180^\circ) \cdot \alpha_j$ to an element $(g_1, g_2) \in K$:

$$g_1(g_2 t_{i_j})^{-1} = g_1 t_{i_j}^{-1} g_2^{-1} = (g_1 t_{i_j}) g_2^{-1} \tag{2.25}$$

Because the partial products of t_{i_j} cover G, there exists some $t_{i_1} t_{i_2} \ldots t_{i_k} = g_1^{-1} g g_2$, so that

$$g_1(t_{i_1} t_{i_2} \ldots t_{i_k}) g_2^{-1} = g_1(g_1^{-1} g g_2) g_2^{-1} = g, \tag{2.26}$$

as desired.

Next, we give a strategy $B = \{\beta_i\}$ such that for any adversarial sequence $\{h_i \in H\}$ and element $g \in G$, there exists an $i \geq 0$ such that the i-th partial product,

$$p((\mathrm{id}_K, \mathrm{id}_H) \cdot (\alpha_1, h_1) \cdot (\alpha_2, h_2) \cdots (\alpha_i, h_i)) \tag{2.27}$$

has a first coordinate equal to g.

To do this, we define $\beta_j = (t_{i_j}, t_{i_j})$. This strategy is invariant up to actions of H, so we can see that regardless of the initial state $(g_1, g_2) \in K$, there exists some k such that

$$\beta_1 \beta_2 \cdots \beta_k = (g_1^{-1} g, g_1^{-1} g) \tag{2.28}$$

and therefore $(g_1, g_2)\beta_1 \beta_2 \cdots \beta_j = (g, g_2 g_1^{-1} g)$, as desired.

It is important to note that applying β_j does not affect the difference: $(g_1, g_2)\beta_j = (g_1 t_{i_j}, g_2 t_{i_j})$ has a difference of $(g_1 t_{i_j})(g_2 t_{i_j})^{-1} = g_1 t_{i_j} t_{i_j}^{-1} g_2^{-1} = g_1 g_2^{-1}$.

Now by interleaving these two strategies, we see that the partial products of $B \circledast A$ hit every possible first letter and every possible difference for every first letter, therefore the projection of the set partial products of $B \circledast A$ covers K for all adversarial sequences, so $B \circledast A$ is a switching strategy. □

We will illustrate this idea explicitly letting G be the smallest nonabelian group of composite order, the symmetric group on three letters S_3, which is isomorphic to the dihedral group of the triangle, D_6.

Example 2.5.3. *Note that $S_3 = \langle (12), (13) \rangle$ is generated by involutions, and that the partial products of the sequence $((12), (13), (12), (13), (12))$ cover S_3.*

As a bit of notation, for a permutation $\pi \in S_3$, let $\pi_1 = (\pi, \mathrm{id}_{S_3}) \in K$ and $\pi_2 = (\pi, \pi) \in K$, corresponding to sequences B and A respectively in the above theorem. Then the following is a (minimal) switching strategy on $S_3 \wr C_2$:

$$
\begin{aligned}
&(12)_2(13)_2(12)_2(13)_2(12)_2 \\
&\qquad\qquad\qquad\qquad (12)_1 \\
&(12)_2(13)_2(12)_2(13)_2(12)_2 \\
&\qquad\qquad\qquad\qquad (13)_1 \\
&(12)_2(13)_2(12)_2(13)_2(12)_2 \\
&\qquad\qquad\qquad\qquad (12)_1 \\
&(12)_2(13)_2(12)_2(13)_2(12)_2 \\
&\qquad\qquad\qquad\qquad (13)_1 \\
&(12)_2(13)_2(12)_2(13)_2(12)_2 \\
&\qquad\qquad\qquad\qquad (12)_1 \\
&(12)_2(13)_2(12)_2(13)_2(12)_2
\end{aligned}
$$

It is natural to ask which spinning groups H correspond to to a generalized spinning switches puzzle $S_n \wr H$ with a switching strategy. We can use a reduction on switches to rule out a large number of these.

Proposition 2.5.4. *For $n > 1$, $S_n \wr H$ does not have a switching strategy whenever $|H|$ is not a power of 2.*

Proof. The alternating group A_n is an index 2 subgroup of S_n, so A_n is normal, and $S_n/A_n \cong \mathbb{Z}_2$. Since we know that $\mathbb{Z}_2 \wr H$ has no switching strategy when $|H|$ is not a power of 2, by the reduction in Theorem 2.3.3, $S_n \wr H$ does not have a switching strategy. □

Many groups are generated by transpositions, including 22 of the 26 sporadic simple groups and the alternating groups A_5 and A_n for $n > 9$, as shown by Mazurov and Nuzhin respectively.

Theorem 2.5.5. *[11] Let G be one of the 26 sporadic simple groups. The group G cannot be generated by three involutions two of which commute if and only if G is isomorphic to M_{11}, M_{22}, M_{23}, or M_L^c.*

Theorem 2.5.6. *[12] The alternating group A_n is generated by three involutions, two of which commute, if and only if $n \geq 9$ or $n = 5$.*

Nuzhin provides other families of groups that are generated by three involutions, two of which commute, in subsequent papers [13–15].

Thus, we have characterized finite wreath products with switching strategies in many cases: wreath products that are p-groups, trivial wreath products, and wreath products $G \wr C_2$ where G is generated by involutions. In the next section, we provide even more general constructions and ask more specific questions.

2.6 Generalizations and open questions

In this section, we provide conjectures and suggest open questions about the structure of switching strategies when they exist, further generalizations of spinning switches puzzles, and, lastly, introduce a notion of an infinite switching strategy for infinite wreath products.

The ultimate open question is a full classification of finite wreath products with switching strategies.

Open Question 2.6.1. *What finite wreath products, $G \wr H$, have a switching strategy?*

2.6.1 Switches generated by elements of prime power order

In Theorem 2.5.2, we constructed a strategy for $G \wr C_2$, when G can be generated by elements of order 2. We conjecture that that there is a broader construction for the case where the adversary can act on the switches with a group of order 2^ℓ.

Conjecture 2.6.2. *When G is a finite group generated by involutions, and H is 2-group that acts faithfully on the set of switches, there exists a switching strategy for $G \wr H$.*

We can also ask about three switches that are generated by elements of order 3 on the corners of a triangular table. In particular, the alternating group is such a group, and we conjecture that it has a switching strategy.

Conjecture 2.6.3. *There exists a switching strategy for $A_n \wr C_3$.*

Putting these two conjectures together, we boldly predict a large family of wreath products with switching strategies.

Conjecture 2.6.4. *If G can be generated by elements of order p^n, and H is a p-group acting faithfully on the set of switches Ω, then $G \wr_\Omega H$ has a switching strategy.*

2.6.2 Palindromic switching strategies

In all known examples, when there exists a switching strategy S, we also know of a *palindromic* switching strategy $S' = \{k'_i \in K\}_{i=1}^N$ such that $k'_i = k'_{N-i+1}$ for all i.

Conjecture 2.6.5. *Whenever $G \wr H$ has a switching strategy, it also has a palindromic switching strategy.*

If this conjecture is false, we suspect a counterexample can be found in the case of the trivial wreath product, $G \wr 1 \equiv G$.

2.6.3 Quasigroup switches

In Section 2.1.3, we argued for modeling switches as finite groups because of some desirable properties:

1. Closure. Regardless of which state a switch is in, every move results in a valid state.

2. Identity. We do not have to move a switch on a given turn.

3. Inverses. If a switch is off, we can always turn it on.

In the list, we also included the axiom of associativity for three reasons: switches in practice typically have associativity, groups are easier to model than quasigroups with identity, and associativity makes defining a "switching strategy" simpler.

However, one could design a non-associative switch and the puzzle would still be coherent. This is because the process of a generalized spinning switches puzzle is naturally "left associative" in the language of programming language theory: we are always "stacking" our next move onto the right. As such, it is worth noting the slightly more general way of modeling switches: as quasigroups with identity, called *loops*.

In particular, we are interested in the smallest loop that is not a group [16], which we denote as $\mathscr{L} = (\{1, a, b, c, d\}, *)$, and describe via its multiplication table, a Latin square of order 5. (Notice that $(ab)d = a \neq d = a(bd)$.)

$*$	1	a	b	c	d
1	1	a	b	c	d
a	a	1	c	d	b
b	b	d	1	a	c
c	c	b	d	1	a
d	d	c	a	b	1

(2.29)

Conjecture 2.6.6. *There exists a nontrivial adversarial group H such that the generalized spinning switches puzzle with switches that behave like $\mathscr{L}$ has a winning strategy for the puzzle-solver.*

2.6.4 Expected number of turns

In practice, a puzzle-solver can get the switches into the winning state by playing randomly. Random play will *eventually* turn on the lightbulb with probability 1, due to the finite number of configurations and the law of large numbers.

Thus we drop the requirement of a finite strategy and ask about the expected value of the number of turns given various sequences of moves. Notice that this is an interesting question even (perhaps especially) in the context of generalized spinning switches puzzles that do not have a switching strategy.

Winkler [17] notes in the solution "Spinning Switches":

> Although no fixed number of steps can guarantee turning the bulb on in the three-switch version [with two-way switches], a smart randomized algorithm can get the bulb on in at most $5\frac{5}{7}$ steps on average, against any strategy by an adversary who sets the initial configuration and turns the platform [17].

A basic model for computing the expected number of turns assumes that the initial hidden state $k \in K$ is not the winning state id_K, and that the adversary's "spins" are independent and identically distributed uniformly random elements $h_j \in H$.

Proposition 2.6.7. *If the puzzle-solver chooses $k_j \in K \setminus \{\mathrm{id}_K\}$ uniformly at random (that is, never choosing the "do nothing" move) then the distribution of the resulting state after each turn will be uniformly distributed among the $|K|-1$ different states. The probability of the resulting state being the winning state is*

$$\mathbb{P}(p((k_1,h_1)(k_2,h_2)\dots(k_j,h_j)) = k^{-1} \mid p((k_1,h_1)(k_2,h_2)\dots(k_{j-1},h_{j-1}) \neq k^{-1})) = \frac{1}{|K|-1}, \tag{2.30}$$

and the expected number of moves is $|K|-1$.

Proof. Because the new states are in 1-to-1 correspondence with the elements of $K \setminus \{\mathrm{id}_K\}$, and since $k_j \in K \setminus \{\mathrm{id}_K\}$ is chosen uniformly at random, $p((k_1,h_1)(k_2,h_2)\cdots(k_j,h_j))$ is uniformly

distributed among all elements of $K \setminus \{p((k_1,h_1)(k_2,h_2)\cdots(k_{j-1},h_{j-1}))\}$. The expected value is $|K|-1$ because the number of turns follows a geometric distribution with parameter $(|K|-1)^{-1}$.

□

When a generalized spinning switches puzzle has a *minimal* switching strategy, we can *guarantee* that we turn on the light within $|K|-1$ moves, so we can certainly solve in fewer than $|K|-1$ moves on average.

Proposition 2.6.8. *If the generalized spinning switches puzzle, $G \wr H$, has a minimal switching strategy, then the expected number of moves is $|K|/2$.*

Proof. If there exists a switching strategy of length $|K|-1$, then for each adversarial strategy $\{h_i \in H\}_{i=1}^{|K|-1}$ the projection of the sequence of partial products of moves induces a permutation of $K \setminus \{\mathrm{id}_K\}$.

If the initial hidden state k is chosen uniformly at random, then k^{-1} is equally likely to occur at any position in this permutation, so the index of the winning state is uniform on $\{1,2,\ldots,|K|-1\}$ and the expected number of moves is

$$\frac{1+2+\cdots+|K|-1}{|K|-1} = |K|/2. \tag{2.31}$$

□

As this proposition suggests, we can always come up with a strategy that does better than the uniformly random strategy in Proposition 2.6.7.

Proposition 2.6.9. *For every generalized spinning switches puzzle, $G \wr H$ such that $|K| > 2$, there always exists a (perhaps infinite) sequence whose expected number of moves is strictly less than $|K|-1$.*

Proof. When $|K| > 2$, we can always improve on the random strategy in Proposition 2.6.7, by avoiding the move of $(g,g,...,g) \in K$ followed by $(g^{-1},g^{-1},...,g^{-1}) \in K$, because the second move will put us into a previous state and so will turn on the lightbulb with probability 0. □

We conjecture that this technique can be extended, and that the puzzle-solver can always do asymptotically better than randomly guessing.

Conjecture 2.6.10. *There exists a constant $\frac{1}{2} < c < 1$ such that for all (finite) wreath products $G \wr H$ with sufficiently large $|K|$, the expected number of moves is less than $c|K|$.*

2.6.5 Shortest switching strategies

Based on all of the examples that we know of, we conjecture that there exist minimal switching strategies whenever there exists a switching strategy of any length.

Conjecture 2.6.11. *Whenever $G \wr H$ has a switching strategy, it also has a minimal switching strategy $\{k_i \in K\}_{i=1}^{|K|-1}$.*

On the other extreme, we have a weaker conjecture: whenever a wreath product has a switching strategy, we can provide an upper bound for its shortest switching strategy.

Conjecture 2.6.12. *Let K/H be the set of equivalence classes of K up to the action of H. Then if $G \wr H$ has a switching strategy, it always has a switching strategy of length $N < 2^{|K/H|-1}$.*

2.6.6 Counting switching strategies

The counting problem analog to the decision problem "does $G \wr H$ have a switching strategy" is obviously of interest to this combinatorialist. It is interesting to count both the number of switching strategies of length N, and the number of such switching strategies *up to the action of H*. We might also be interested in the number of palindromic switching strategies, or the number of switching strategies satisfying another desirable criterion.

In the case of the trivial wreath product, $G \wr \mathbf{1}$, we saw in Proposition 2.5.1 that there are $(|G|-1)!$ switching strategies. (And since the group action is trivial, there are also this many strategies up to group action.)

Open Question 2.6.13. *Given a wreath product $G \wr H$ how many switching strategies of length N does it have? How many up to the action of H? How many are palindromic?*

Proposition 2.6.14. *The wreath product $S_3 \wr \mathbf{1}$ has* 12 *palindromic switching sequences:*

(1 2),	(1 3),	(1 2),	(1 3),	(1 2).
(1 2),	(2 3),	(1 2),	(2 3),	(1 2).
(1 3),	(1 2),	(1 3),	(1 2),	(1 3).
(1 3),	(2 3),	(1 3),	(2 3),	(1 3).
(1 2 3),	(1 2 3),	(1 2),	(1 2 3),	(1 2 3).
(1 2 3),	(1 2 3),	(1 3),	(1 2 3),	(1 2 3).
(1 2 3),	(1 2 3),	(2 3),	(1 2 3),	(1 2 3).
(1 3 2),	(1 3 2),	(1 2),	(1 3 2),	(1 3 2).
(1 3 2),	(1 3 2),	(1 3),	(1 3 2),	(1 3 2).
(1 3 2),	(1 3 2),	(2 3),	(1 3 2),	(1 3 2).
(2 3),	(1 2),	(2 3),	(1 2),	(2 3).
(2 3),	(1 3),	(2 3),	(1 3),	(2 3).

Table 2.1: The 12 palindromic switching sequences on $S_3 \wr \mathbf{1}$.

Proof. The search space is small here, so this was computed naively by brute force. □

2.6.7 Restricted spinning

Another way that we could generalize a spinning switches puzzle is by restricting the adversary's moves. For instance, we could modify the puzzle in such a way that the adversary can only spin the switches every k turns. For every finite setup $G \wr H$, there exists $k \in \mathbb{N}$ such that the puzzle-solver can win. (For example, we can always take $k > |K|$ so that the puzzle-solver can do a walk of K.)

Open Question 2.6.15. *How does one compute the minimum k such that the puzzle solver has a switching strategy of $G \wr H$ given that the adversary's sequence $\{h_i \in H\}$ is constrained so that $h_i = \mathrm{id}_H$ whenever $i \not\equiv 0 \bmod k$?*

2.6.8 Multiple winning states

One assumption that we made about our switches is that they have a single "on" state. Of course, we might conceive of a switch that has m possible states and $k \leq m$ "on" states.

Open Question 2.6.16. *Given a generalized spinning switches puzzle $G \wr H$ where each switch has a set of "on" states $\mathcal{O}_G \subseteq G$, when is it possible for the puzzle-solver to have a finite strategy that guarantees the switches will get to a winning state?*

2.6.9 Nonhomogeneous switches

Another way to generalize the spinning switches puzzle is by allowing different sorts of switches together in the same puzzle. For instance, we could imagine a square board containing four buttons: two 2-way switches, a 3-way switch, and a 5-way switch. Can a puzzle like this be solved? It is important that all of these buttons have the same "shape", that is there is some group G with a surjective homomorphism onto each of them.

One way to formalize this generalization is as follows:

Definition 2.6.17. *A **nonhomogeneous generalized spinning switches puzzle** consists of a triple*

- *A wreath product $\mathbb{F}_k \wr_\Omega H$ of the free group on k generators by a "rotation" group with a base denoted $K = \prod_{\omega \in \Omega} F_{k,\omega}$,*
- *a product of finite groups denoted $\hat{G} = \prod_{\omega \in \Omega} G_\omega$ where each group is specified by a presentation with k generators and any number of relations: $G_\omega = \langle g_1^\omega, g_2^\omega, \ldots, g_k^\omega \mid R_\omega \rangle$, and*
- *a corresponding sequence of evaluation maps $e_\omega \colon \mathbb{F}_{k,\omega} \to G_\omega$, that send generators in $F_{k,\omega}$ to the corresponding generators in G_ω. This can be induced coordinatewise to a map $e_\Omega \colon K \to \hat{G}$.*

When all of the copies of G_ω are isomorphic, this essentially simplifies to the original definition.

The analogous definition of a switching strategy becomes more complicated.

Definition 2.6.18. *Let $(\mathbb{F}_k \wr_\Omega H, \hat{G}, e_\Omega)$ be a nonhomogeneous generalized spinning switches puzzle.*

*Then a **nonhomogeneous switching strategy** is a sequence $\{k_i \in K\}_{i=1}^N$ such that for each adversarial sequence $\{h_i \in H\}_{i=1}^N$ the induced projection/evaluation map $e_\Omega \circ p \colon \mathbb{F}_k \wr_\Omega H \to \hat{G}$ on the partial products of $\{(k_i, h_i) \in \mathbb{F}_k\}$ covers $\hat{G}$.*

Proposition 2.6.19. *In the specific case that $\Omega = [n]$, $H \subseteq S_n$, and $\hat{G} = \prod_{\omega\in\Omega} G_\omega$ is a product of cyclic groups of pairwise coprime order, the nonhomogeneous generalized spinning switches puzzle has a switching strategy, namely $\{(1,1,\ldots,1) \in K\}_{i=1}^{|K'|-1}$.*

Proof. By the fundamental theorem of abelian groups, $\hat{G}$ is cyclic and is generated by

$$\hat{G} = \langle (1_{C_{k_1}}, 1_{C_{k_2}}, \ldots, 1_{C_{k_{|\Omega|}}}) \rangle. \tag{2.32}$$

Thus, the sequence $\{(1,1,\ldots,1) \in K\}_{i=1}^{|K'|-1}$ is a switching strategy because it is a fixed point under H, and its image is a generator of $\hat{G}$. □

Open Question 2.6.20. *Which nonhomogeneous generalized spinning switches puzzles have a nonhomogeneous switching strategy?*

2.6.10 Infinite switching strategies

When we first introduced the notion of a switching strategy in Definition 2.2.3, we defined it to be a finite sequence on finite wreath products. However, we can expand the definition to (countably) infinite wreath products by allowing for infinite sequences of moves in K. In particular, we can extend this definition to settings where switches have a countably infinite number of states, where there is a countably infinite number of switches, or both. To keep K countable in the latter cases, we use the restricted wreath product, where $K \cong \bigoplus_{\omega\in\Omega} G_\omega$ is defined to be a direct sum instead of a direct product.

Definition 2.6.21. *A **infinite switching strategy** on an infinite wreath product $G \wr H$ is a sequence $\{k_i \in K\}_{i=1}^{\infty}$ such that for all $k \in K$ and all infinite sequences $\{h_i \in H\}_{i=1}^{\infty}$, there exists some $N \geq 0$ such that the projection*

$$p((k_1, h_1) \cdot (k_2, h_2) \cdots (k_N, h_N)) = k^{-1}. \tag{2.33}$$

We claim, but do not prove that $G \wr_{\Omega} C_2$ has an infinite switching strategy in the following three settings:

1. $|\Omega| = 2$ and $G \cong (\mathbb{N}_{\geq 0}, \wedge)$ where $\wedge$ is the bitwise XOR operator.
2. $\Omega \cong \mathbb{N}_{\geq 0}$ as a set, and $K = \bigoplus_{i=0}^{\infty} \mathbb{Z}_2^{(i)}$ where $C_2^{(i)} \cong \mathbb{Z}_2$.
3. $\Omega \cong \mathbb{N}_{\geq 0}$ as a set, and $K = \bigoplus_{i=0}^{\infty} (\mathbb{N}_{\geq 0}, \wedge)$ for all i.

Chapter 3

Permutations with a given number of k-cycles

In this chapter, we study permutations $\pi \in S_n$ with exactly m transpositions. In particular, we are interested in the expected value of $\pi(1)$ when such permutations are chosen uniformly at random. When n is even, this expected value is approximated closely by $(n+1)/2$, with an error term that is related to the number isometries of the $(n/2-m)$-dimensional hypercube that move every face. Furthermore, when $k \mid n$, this construction generalizes to allow us to compute the expected value of $\pi(1)$ for permutations with exactly m k-cycles. In this case, the expected value has an error term which is related instead to the number derangements of the generalized symmetric group $S(k, n/k-m)$.

When k does not divide n, the expected value of $\pi(1)$ is precisely $(n+1)/2$. Indirectly, this suggests the existence of a reversible algorithm to insert a letter into a permutation which preserves the number of k-cycles, which we construct.

3.1 Overview and preliminaries

In 2010, Mark Conger [18] proved that a permutation with k descents has an expected first letter of $\pi(1) = k+1$, independent of n. This chapter has the same premise, but with a different permutation statistic: the number of k-cycles of a permutation.

This section (Section 3.1) provides an overview of where we're headed, and includes a critical example that will hopefully spark the reader's curiosity and motivate the remainder of the chapter.

Section 3.2 establishes some recurrence relations for the number of permutations in S_n with a given number of k-cycles. It also contains a theorem that gives an explicit way to compute the expected value of the first letter based on these counts.

Section 3.3 describes an explicit correspondence between k-cycles of permutations in S_{kn} and fixed points of elements of the generalized symmetric group $(\mathbb{Z}/k\mathbb{Z}) \wr S_n$. Using generating functions and results from the previous section, this shows that the expected value of $\pi(1)$ of a permutation with a given number of k-cycles is intimately connected to the number of derangements of a generalized symmetric group.

While Section 3.3 emphasizes the case of S_{kn}, Section 3.4 looks at S_N where $k \nmid N$. Here, the expected value of $\pi(1)$ is simply $(N+1)/2$, which agrees with the expected value of the first letter of a uniformly chosen N-letter permutation with no additional restrictions. This fact together with the main theorem from Section 3.2 implies the existence of a bijection $\varphi_k \colon S_{N-1} \times [N] \to S_N$ that preserves the number of k-cycles whenever $k \nmid N$. Section 3.4 constructs such a bijection explicitly, and proves that it has the desired properties.

3.1.1 Motivating examples

In support of the first examples, we start by defining the first bit of notation.

Definition 3.1.1. *Let $C_k(n,m)$ denote the number of permutations $\pi \in S_n$ such that π has exactly m k-cycles.*

The ideas in this chapter—and many of the following lemmas and theorems—were motivated by looking at examples such as the following, written in both one-line and cycle notation:

Example 3.1.2. *There are $C_2(4,0) = 15$ permutations in S_4 with no 2-cycles:*

$$\begin{array}{llll} 1234 = (1)(2)(3)(4) & 2314 = (312)(4) & 3124 = (321)(4) & 4123 = (4321) \\ 1342 = (1)(423) & 2341 = (4123) & 3142 = (4213) & 4132 = (421)(3) \\ 1423 = (1)(432) & 2413 = (4312) & 3241 = (2)(413) & 4213 = (2)(431) \\ & 2431 = (3)(412) & 3421 = (4132) & 4312 = (4231) \end{array}$$

There are $C_2(4,1) = 6$ permutations in S_4 with exactly one 2-cycle:

$$\begin{array}{ll} 1243 = (1)(2)(43) & 2134 = (21)(3)(4) \\ 1324 = (1)(32)(4) & 3214 = (2)(31)(4) \\ 1432 = (1)(3)(42) & 4231 = (2)(3)(41) \end{array}$$

And there are $C_2(4,2) = 3$ permutations in S_4 with exactly two 2-cycles,

$$2143 = (21)(43) \qquad 3412 = (31)(42) \qquad 4321 = (32)(41).$$

By averaging the first letter over these examples, we can compute that

$$\begin{aligned} \mathbb{E}[\pi(1) \,|\, \pi \in S_4 \textit{ has no 2-cycles}] &= \frac{3(1)+4(2+3+4)}{15} = \frac{13}{5}, \\ \mathbb{E}[\pi(1) \,|\, \pi \in S_4 \textit{ has exactly 1 2-cycle}] &= \frac{3(1)+(2+3+4)}{6} = 2, \textit{ and} \\ \mathbb{E}[\pi(1) \,|\, \pi \in S_4 \textit{ has exactly 2 2-cycles}] &= \frac{2+3+4}{3} = 3. \end{aligned}$$

Table 3.1 gives the expected value of $\pi(1)$ given that $\pi \in S_n$ and has exactly m 2-cycles in its cycle decomposition. Notice that when i is odd, row i has a constant value of $(i+1)/2$. Also notice

		m						
		0	1	2	3	4	5	6
n	1	$1/1$						
	2	$1/1$	$2/1$					
	3	$2/1$	$2/1$					
	4	$13/5$	$2/1$	$3/1$				
	5	$3/1$	$3/1$	$3/1$				
	6	$101/29$	$18/5$	$3/1$	$4/1$			
	7	$4/1$	$4/1$	$4/1$	$4/1$			
	8	$1049/233$	$130/29$	$23/5$	$4/1$	$5/1$		
	9	$5/1$	$5/1$	$5/1$	$5/1$	$5/1$		
	10	$12809/2329$	$1282/233$	$159/29$	$28/5$	$5/1$	$6/1$	
	11	$6/1$	$6/1$	$6/1$	$6/1$	$6/1$	$6/1$	
	12	$181669/27949$	$15138/2329$	$1515/233$	$188/29$	$33/5$	$6/1$	$7/1$
	13	$7/1$	$7/1$	$7/1$	$7/1$	$7/1$	$7/1$	$7/1$

(3.2)

Table 3.1: A table of the expected value of the first letter of $\pi \in S_n$ with exactly m 2-cycles, $\mathbb{E}[\pi(1) \,|\, \pi \in S_n \text{ has exactly } m \text{ 2-cycles}]$.

that the number in position (i, j) has the same denominator as the number in position $(i+2, j+1)$, and that these denominators increase with n. The sequence of denominators begins

$$1, 5, 29, 233, 2329, 27949, \ldots, \tag{3.1}$$

which agrees with the type B derangement numbers, sequence A000354 in the On-Line Encyclopedia of Integer Sequences (OEIS) [10]. In other words, the denominators in the table appear to be related to the symmetries of the hypercube that move every facet.

3.2 Structure of permutations with m k-cycles

This section is about connecting the number of permutations with a given number of k-cycles to the expected value of the first letter. Saying this, it is appropriate to start with a 1944 theorem of Goncharov that, by the principle of inclusion/exclusion, gives an explicit formula that counts the number of such permutations.

3.2.1 Counting permutations based on cycles

Theorem 3.2.1 ([19], [20])**.** *The number of permutations in S_n with exactly m k-cycles is given by the following sum, via the principle of inclusion/exclusion:*

$$C_k(n,m) = \frac{n!}{m!k^m} \sum_{i=0}^{\lfloor n/k \rfloor - m} \frac{(-1)^i}{i!\,k^i}. \tag{3.3}$$

Corollary 3.2.2. *For $k \nmid n$, there are exactly n times as many permutations in S_n with exactly m k-cycles as there are in S_{n-1}. When $k \mid n$, there is an explicit formula for the difference.*

$$C_k(n,m) - nC_k(n-1,m) = \begin{cases} 0 & k \nmid n \quad \text{(3.4a)} \\ \dfrac{n!(-1)^{\frac{n}{k}-m}}{(n/k)!\,k^{\frac{n}{k}}} \dbinom{n/k}{m} & k \mid n \quad \text{(3.4b)} \end{cases}$$

Proof. When $k \nmid n$, $\left\lfloor \dfrac{n}{k} \right\rfloor = \left\lfloor \dfrac{n-1}{k} \right\rfloor$, so the bounds on the sums are identical and the result follows directly:

$$\frac{n!}{m!k^m} \sum_{i=0}^{\lfloor n/k \rfloor - m} \frac{(-1)^i}{i!\,k^i} - n\left(\frac{(n-1)!}{m!k^m} \sum_{i=0}^{\lfloor (n-1)/k \rfloor - m} \frac{(-1)^i}{i!\,k^i} \right) = 0. \tag{3.5}$$

Otherwise, when $k \mid n$, $\left\lfloor \frac{n-1}{k} \right\rfloor = \frac{n}{k} - 1$, so

$$\begin{aligned}
&\frac{n!}{m!k^m} \sum_{i=0}^{n/k-m} \frac{(-1)^i}{i!\,k^i} - n\left(\frac{(n-1)!}{m!k^m} \sum_{i=0}^{n/k-1-m} \frac{(-1)^i}{i!\,k^i} \right) \\
&\qquad = \frac{n!}{m!k^m} \left(\frac{(-1)^{n/k-m}}{(n/k-m)!k^{n/k-m}} \right) \\
&\qquad = \frac{n!(-1)^{n/k-m}}{(n/k-m)!m!k^{n/k}} \\
&\qquad = \frac{n!(-1)^{\frac{n}{k}-m}}{(n/k)!\,k^{n/k}} \binom{n/k}{m}. \qquad (3.6)
\end{aligned}$$

□

See Section 3.4 for a bijective proof of Equation (3.4a).

3.2.2 Permutations by first letter

In order to compute the expected value of the first letter of a permutation, it is useful to be able to compute the number of permutations that have a given number of k-cycles *and* a given first letter.

Definition 3.2.3. *Let $C_k^{(a)}(n,m)$ be the number of permutations $\pi \in S_n$ that have exactly m k-cycles and $\pi(1) = a$.*

The expected value of $\pi(1)$ with a given number of k-cycles is

$$\mathbb{E}[\pi(1) \mid \pi \in S_n \text{ has exactly } m \text{ } k\text{-cycles}] = \frac{1}{C_k(n,m)} \sum_{a=1}^{n} a C_k^{(a)}(n,m). \qquad (3.7)$$

The following three lemmas compute $C_k^{(a)}(n,m)$ from $C_k(n,m)$.

Proposition 3.2.4. *For all $k > 1$, the number of permutations in S_n starting with 1 and having m k-cycles is equal to the number of permutations in S_{n-1} with m k-cycles:*

$$C_k^{(1)}(n,m) = C_k(n-1,m). \tag{3.8}$$

Proof. The straightforward bijection from $\{\pi \in S_n : \pi(1) = 1\}$ to S_{n-1} given by deleting 1 and relabeling preserves the number of k-cycles for $k > 1$. □

Proposition 3.2.5. *For all $a, b \geq 2$, the number of permutations having m k-cycles and starting with a is the same as the number of those starting with b:*

$$C_k^{(2)}(n,m) = \cdots = C_k^{(a)}(n,m) = \cdots = C_k^{(b)}(n,m) = \cdots = C_k^{(n)}(n,m). \tag{3.9}$$

Proof. Since the permutations under consideration do not fix 1, conjugation by (ab) is an isomorphism which takes all words starting with a to words starting with b without changing the cycle structure. □

Lemma 3.2.6. *For all $2 \leq a \leq n$,*

$$C_k^{(a)}(n,m) = \frac{C_k(n,m) - C_k(n-1,m)}{n-1}. \tag{3.10}$$

Proof. Since

$$C_k(n,m) = C_k^{(1)}(n,m) + C_k^{(2)}(n,m) + \cdots + C_k^{(n)}(n,m), \tag{3.11}$$

using Proposition 3.2.5 for the last $(n-1)$ terms, this can be rewritten as

$$C_k(n,m) = C_k^{(1)}(n,m) + (n-1)C_k^{(a)}(n,m). \tag{3.12}$$

Solving for $C_k^{(a)}(n,m)$ and using the substitution from Proposition 3.2.4 gives the desired result.

□

Now, equipped with explicit formulas for $C_k^{(a)}(n,m)$ and $C_k(n,m)$, we can compute the expected value of $\pi(1)$ for $\pi \in S_n$ with exactly m k-cycles.

3.2.3 Expected value of first letter

Theorem 3.2.7. *For $k > 1$, the expected value of the first letter of a permutation*

$$\mathbb{E}[\pi(1) \mid \pi \in S_n \text{ has exactly } m \text{ } k\text{-cycles}] = \frac{n}{2}\left(1 - \frac{C_k(n-1,m)}{C_k(n,m)}\right) + 1. \tag{3.13}$$

Proof. Using Proposition 3.2.5, we can consolidate all but the first term of the sum in Equation (3.7)

$$\begin{aligned}
\sum_{a=1}^{n} aC_k^{(a)}(n,m) &= C_k^{(1)}(n,m) + \sum_{a=2}^{n} aC_k^{(n)}(n,m) && (3.14)\\
&= C_k^{(1)}(n,m) + \frac{(n-1)(n+2)}{2} C_k^{(n)}(n,m) && (3.15)\\
&= C_k(n-1,m) + \frac{(n-1)(n+2)}{2}\left(\frac{C_k(n,m) - C_k(n-1,m)}{n-1}\right) && (3.16)\\
&= \left(\frac{n}{2}+1\right) C_k(n,m) - \frac{n}{2} C_k(n-1,m). && (3.17)
\end{aligned}$$

Dividing by $C_k(n,m)$ yields the result. □

Corollary 3.2.8. *When $k \nmid n$, $C_k(n,m) = nC_k(n-1,m)$ by Equation* (3.4a)*, so*

$$\mathbb{E}[\pi(1) \mid \pi \in S_n \text{ has exactly } m \text{ } k\text{-cycles}] = \frac{n}{2}\left(1 - \frac{1}{n}\right) + 1 = \frac{n+1}{2}. \tag{3.18}$$

Together with Theorem 3.2.1, this theorem and its corollary provides our first formula for the expected value of $\pi(1)$ with an associated algorithm that performs exponentially better than brute force.

3.2.4 Identities for counting permutations with given cycle conditions

Both in practical terms (if computing the expected value of $\pi(1)$ by hand or optimizing an algorithm) and in a theoretical sense, the following recurrence is simple and useful.

Lemma 3.2.9. *For $n < mk$ or $m < 0$, $C_k(n,m) = 0$. Otherwise, for all $k, m \geq 1$*

$$mC_k(n,m) = (k-1)!\binom{n}{k}C_k(n-k,m-1). \tag{3.19}$$

While this can be proven directly by the algebraic manipulation of the identity in Theorem 3.2.1, a bijective proof has been included here because it is natural and may be of interest.

Proof. Let

$$\mathscr{C}_k(n,m) = \{\pi \in S_n \mid \pi \text{ has exactly } m \text{ } k\text{-cycles}\}. \tag{3.20}$$

Then consider the following two sets, whose cardinalities match the left- and right-hand sides of the equation above:

$$X^L_{n,m,k} = \{(\pi, c) \mid \pi \in \mathscr{C}_k(n,m), c \text{ a distinguished } k\text{-cycle of } \pi\}. \tag{3.21}$$

$$X^R_{n,m,k} = \{(\sigma, d) \mid \pi \in \mathscr{C}_k(n-k,m-1), d \text{ an } n\text{-ary necklace of length } k\}. \tag{3.22}$$

The first set, $X^L_{n,m,k}$, is constructed by taking a permutation in $\mathscr{C}_k(n,m)$ and choosing one of its m k-cycles to be distinguished, so $\#X^L_{n,m,k} = mC_k(n,m)$.

In the second set, $X^R_{n,m,k}$, the two parts of the tuple are independent. There are $C_k(n-k,m-1)$ choices for the permutation σ and $(k-1)!\binom{n}{k}$ choices for the necklace d. Thus $\#X^R_{n,m,k} = (k-1)!\binom{n}{k}C_k(n-k,m-1)$.

Now, consider the map $\varphi\colon X^L_{n,m,k} \to X^R_{n,m,k}$ which removes the distinguished k-cycle and relabels the remaining $n-k$ letters as $\{1,2,\ldots,n-k\}$, preserving the relative order:

$$(\pi_1\pi_2\cdots\pi_\ell, \pi_i) \overset{\varphi}{\mapsto} (\pi'_1\pi'_2\cdots\pi'_{i-1}\pi'_{i+1}\cdots\pi'_\ell, \pi_i) \tag{3.23}$$

where π_i' is π_i after relabeling.

By construction, σ has one fewer k-cycle and k fewer letters than π.

The inverse map is similar. To recover π, increment the letters of σ appropriately and add the necklace d back in as the distinguished cycle. Thus φ is a bijection and $\#X^L_{n,m,k} = \#X^R_{n,m,k}$. □

Example 3.2.10. *Suppose* $\pi = (423)(\mathbf{61})(75)$ *in cycle notation with* (61) *distinguished. Then*

$$\varphi((423)(61)(75),(61)) = ((312)(54),(61)) \tag{3.24}$$

under the bijection φ *described in the proof of Lemma 3.2.9.*

The recurrence in Lemma 3.2.9 suggests that understanding $C_k(n,m)$ is related to understanding $C_k(n-km,0)$, the permutations of S_{n-km} with no k-cycles. On the other hand, Corollary 3.2.2 suggests that the case where $k \mid n$ has some of the most intricate structure. We can, of course, combine these two observations and analyze the case of $C_k(kn,0)$, which has a particularly simple generating function, which will show up again in a different guise.

Lemma 3.2.11. *For* $k \geq 2$*,*

$$\sum_{n=0}^{\infty} \frac{C_k(kn,0)k^n}{(kn)!}x^n = \frac{\exp(-x)}{1-kx}. \tag{3.25}$$

Proof. By substitution of $C_k(kn,0)$ via the identity in Theorem 3.2.1,

$$\begin{aligned}
\sum_{n=0}^{\infty} \frac{C_k(kn,0)k^n}{(kn)!}x^n &= \sum_{n=0}^{\infty}\sum_{i=0}^{n} \frac{(-1)^i}{k^i i!}k^n x^n && (3.26)\\
&= \sum_{n=0}^{\infty}\sum_{i=0}^{n} \frac{(-x)^i}{i!}(kx)^{n-i} && (3.27)\\
&= \left(\sum_{n=0}^{\infty}\frac{(-x)^n}{n!}\right)\left(\sum_{n=0}^{\infty}(kx)^n\right) && (3.28)\\
&= \frac{\exp(-x)}{1-kx}. && (3.29)
\end{aligned}$$

□

This section allowed for the practical computation of the expected value of $\pi(1)$ with a given number of k-cycles, but leaves the observation about Table 3.1 unexplained. The following section will explain the connection between the expected values of $\pi(1)$ and the facet-derangements of the hypercube.

3.3 Connection with the generalized symmetric group

This section explains the connection between the expected value of $\pi(1)$ given that π has exactly m 2-cycles and the facet-derangements of the hypercube, by telling the more general story of derangements of the generalized symmetric group. Thus it is appropriate to start this section by defining both the generalized symmetric group and its derangements.

3.3.1 Derangements of the generalized symmetric group

Definition 3.3.1. *The **generalized symmetric group** $S(k,n)$ is the wreath product $(\mathbb{Z}/k\mathbb{Z}) \wr S_n$, which in turn is a semidirect product $(\mathbb{Z}/k\mathbb{Z})^n \rtimes S_n$.*

A natural way of thinking about the symmetric group S_n is by considering how the elements act on length-n sequences by permuting the indices. Informally, we can think about the generalized symmetric group $S(k,n)$ in an essentially similar way: each element consists of an ordered pair in $(\mathbb{Z}/k\mathbb{Z})^n \rtimes S_n$, where $(\mathbb{Z}/k\mathbb{Z})^n$ gives information about what to add componentwise, and S_n gives information about how to rearrange afterward.

Example 3.3.2. *Consider the generalized permutation*

$$(\underbrace{(1,3,0)}_{\in(\mathbb{Z}/4\mathbb{Z})^3}, \underbrace{(23)}_{\in S_3}) \in S(4,3). \tag{3.30}$$

It acts on the sequence $(0,1,1) \in (\mathbb{Z}/2\mathbb{Z})^3$ *first by adding element-wise, and then permuting:*

$$\underbrace{((1,3,0),(23))}_{\in S(k,n)} \cdot (0,1,1) = \underbrace{(23)}_{\in S_3} \cdot (1+0,3+1,0+1) = (23) \cdot (1,0,1) = (1,1,0). \tag{3.31}$$

When $k = 1$, the sequence $(\mathbb{Z}/1\mathbb{Z})^n$ is trivially the zero sequence, so $S(1,n) \cong S_n$. When $k = 2$, $S(2,n)$ is the hyperoctahedral group that we brushed up against in Table 3.1: the group of symmetries of the n-dimensional hypercube. When $k \geq 3$, $S(k,n)$ does not have such an immediate geometric interpretation, but it is precisely the right analog for the expected value of $\pi(1)$ when π has a given number of k-cycles.

Definition 3.3.3. *A* ***derangement*** *or fixed-point-free element of the generalized symmetric group is an element* $((x_1,\ldots,x_n),\pi) \in S(k,n)$ *such that for all i, either* $\pi(i) \neq i$ *or* $x_i \neq 0$.

That is, when a derangement acts on a sequence in the manner described above, it changes the position or the value of every term in the sequence. When $k = 1$ and $S(1,n) \cong S_n$, this recovers the usual sense of a derangement in S_n: a permutation with no fixed points. In terms of the hyperoctahedral group, $S(2,n)$, a derangement is a symmetry of the n-cube that moves each $(n-1)$-dimensional face.

Example 3.3.4. *The element* $((1,3,0),(23)) \in S(4,3)$ *is a derangement because it increments the first term and swaps the second and third terms—thus changing the position or value for each term.*

The number of derangements of the generalized symmetric group can be described by an explicit sum via the principle of inclusion/exclusion, and it has a particularly elegant exponential generating function.

Theorem 3.3.5 ([21])**.** *For* $k > 1$*, the number of derangements of the generalized symmetric group* $S(k,n)$ *is*

$$D(k,n) = k^n n! \sum_{i=0}^{n} \frac{(-1)^i}{k^i i!}. \tag{3.32}$$

which has exponential generating function

$$\sum_{n=0}^{\infty} \frac{D(k,n)}{n!} x^n = \frac{\exp(-x)}{1-kx}. \tag{3.33}$$

Notice that this agrees identically with the generating function in Lemma 3.2.11, which is our first hint in explaining the connection between k-cycles in permutations and fixed points in elements of the generalized symmetric group.

3.3.2 Permutation cycles and derangements

Lemma 3.3.6. *For $k \geq 1$, the number of permutations with $kn+km$ letters and m k-cycles is*

$$C_k(k(n+m),m) = \binom{kn+km}{kn} C_k(kn,0) \frac{(km)!}{k^m m!}. \tag{3.34}$$

Algebraic proof. This will proceed by induction on m. The base case is clear when $m = 0$, so suppose that the lemma is true up to $m-1$, that is

$$C_k(k(n+m-1),m-1) = \frac{(km-k)!}{k^{m-1}(m-1)!} \binom{kn+km-k}{kn} C_k(kn,0). \tag{3.35}$$

$$= \frac{(kn+km-k)!}{k^{m-1}(m-1)!(kn)!} C_k(kn,0). \tag{3.36}$$

Rearranging Lemma 3.2.9,

$$C_k(k(n+m),m) = \frac{(k-1)!}{m} \binom{k(n+m)}{k} C_k(k(n+m-1),m-1) \tag{3.37}$$

$$= \frac{(kn+km)!}{km(kn+km-k)!} C_k(k(n+m-1),m-1). \tag{3.38}$$

Now, notice there is a $(kn+km-k)!$ term in the numerator of Equation (3.36) and the denominator of Equation (3.38), so substituting and simplifying yields

$$C_k(k(n+m),m) = \frac{(kn+km)!}{k^m m!(kn)!} C_k(kn,0), \tag{3.39}$$

as desired. □

Combinatorial proof. This lemma lends itself to a combinatorial proof. The left-hand side of the equation counts the number of permutations in S_{kn+km} with exactly m k-cycles. The right-hand side of the equation says that this is the number of ways to choose kn letters in the permutation that will not be in k-cycles, and for each of these, there are $C_k(kn,0)$ ways to arrange these such that they have no k-cycles. This leaves over km letters, for which there are $(km)!/(k^m m!)$ ways to write them as products of m disjoint k-cycles. □

The following lemma uses the above identities to establish that the proportion of permutations in the symmetric group S_{kn} with exactly m k-cycles is equal to the proportion of elements in the generalized symmetric group $S(k,n)$ with exactly m fixed points.

Lemma 3.3.7. *For $k \geq 2$,*

$$\frac{C_k(kn,m)}{(kn)!} = \binom{n}{m} \frac{D(k,n-m)}{k^n n!}. \tag{3.40}$$

Proof. By solving for $D(k,n-m)$ on the right-hand side and substituting $n+m$ for n, it is enough to show that the exponential generating function for $D(k,n)$ (as shown in Theorem 3.3.5) is also the exponential generating function for

$$C_k(kn+km,m) \frac{m!n!k^{n+m}}{(kn+km)!}. \tag{3.41}$$

By the identity in Lemma 3.3.6,

$$\sum_{n=0}^{\infty} C_k(kn+km,m)\frac{m!n!k^{n+m}}{(kn+km)!}\frac{x^n}{n!} \tag{3.42}$$

$$= \sum_{n=0}^{\infty} \frac{(km)!}{m!k^m}\binom{kn+km}{kn} C_k(kn,0)\frac{m!n!k^{n+m}}{(kn+km)!}\frac{x^n}{n!} \tag{3.43}$$

$$= \sum_{n=0}^{\infty} C_k(kn,0)\frac{k^n x^n}{(kn)!} \tag{3.44}$$

$$= \frac{\exp(-x)}{1-kx}, \tag{3.45}$$

with the final equality being the identity in Lemma 3.2.11. □

3.3.3 Expected value of letters of permutations

We now have the ingredients we need to prove the pattern that we observed in Table 3.1 that purported to show a relationship between permutations with a given number of 2-cycles and derangements of the hyperoctahedral group. These ingredients come together in the following theorem, which establishes the more general relationship between permutations with a given number of k-cycles and derangements of the generalized symmetric group, $S(k,n)$.

Theorem 3.3.8. *The expected value of the first letter of a permutation $\pi \in S_{kn}$ with exactly m k-cycles, where $k > 1$ and $0 \leq m \leq n$, is*

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_{kn} \text{ has exactly } m \text{ } k\text{-cycles}] = \frac{kn+1}{2} + \frac{(-1)^{n-m}}{2D(k,n-m)}, \tag{3.46}$$

where $D(k,n)$ denotes the number of derangements of the generalized symmetric group, $S(k,n)$.

Proof. Inverting the identity in Lemma 3.3.7, yields

$$\frac{\frac{(kn)!}{n!k^n}\binom{n}{m}}{C_k(kn,m)} = \frac{1}{D(k,n-m)}. \tag{3.47}$$

Multiplying through by $(-1)^{n-m}$ to match the right-hand side of Equation (3.46) together with some small manipulations yields

$$1 - \frac{C_k(kn,m) - (-1)^{n-m}\frac{(kn)!}{n!k^n}\binom{n}{m}}{C_k(kn,m)} = \frac{(-1)^{n-m}}{D(k,n-m)}. \tag{3.48}$$

Now adding $kn+1$ and dividing by 2 yields

$$\frac{kn}{2}\left(1 - \frac{C_k(kn,m) - (-1)^{n-m}\frac{(kn)!}{n!k^n}\binom{n}{m}}{knC_k(kn,m)}\right) + 1 = \frac{kn+1}{2} + \frac{(-1)^{n-m}}{2D(k,n-m)}, \tag{3.49}$$

which gives the right-hand side as desired. Since the numerator on the left-hand side is equal to $knC_k(kn-1,m)$ by Equation (3.2.2), the proof then follows from Theorem 3.2.7. □

With the expected value of the first letter found, we can generalize this one more step to find the expected value of the i-th letter of these permutations.

Corollary 3.3.9. *The expected value of the i-th letter of a permutation in S_{kn} with exactly m k-cycles, where $n \in \mathbb{N}_{>0}$, $k > 1$, $1 \leq i \leq kn$, and $0 \leq m \leq n$, is*

$$\mathbb{E}[\pi(i) \,|\, \pi \in S_{kn} \text{ has exactly } m \text{ } k\text{-cycles}] = \frac{kn+1}{2} + \frac{(-1)^{n-m}}{2D(k,n-m)}\frac{kn+1-2i}{kn-1}. \tag{3.50}$$

Proof. Denote by N the number of permutations in S_{kn} with m k-cycles where 1 is a fixed point; denote by M the number of permutations in S_{kn} with m k-cycles where $\pi(1) = a \neq 1$. Note that while N and M implicitly depend on m, n, and k, M does not depend on a by Proposition 3.2.5.

Thus

$$\begin{aligned}
&\mathbb{E}[\pi(1) \,|\, \pi \in S_{kn} \text{ has exactly } m \text{ } k\text{-cycles}] \\
&\quad = \frac{1}{N + (kn-1)M}\left(N + \sum_{a=2}^{kn} aM\right) \\
&\quad = \frac{1}{N + (kn-1)M}\left(N + \left(\frac{kn(kn+1)}{2} - 1\right)M\right).
\end{aligned} \tag{3.51}$$

More generally, if we conjugate with $(1i)$ then

$$\begin{aligned}
&\mathbb{E}[\pi(i) \mid \pi \in S_{kn} \text{ has exactly } m \text{ } k\text{-cycles}] \\
&\quad = \frac{1}{N+(kn-1)M}\left(N + \sum_{a \neq i} aM\right) \\
&\quad = \frac{1}{N+(kn-1)M}\left(iN + \left(\frac{kn(kn+1)}{2} - i\right)M\right). \qquad (3.52)
\end{aligned}$$

We can extend the function $\mathbb{E}[\pi(i) \mid \pi \in S_{kn} \text{ has exactly } m \text{ } k\text{-cycles}]$ to a function $f(n,k,m,i)$ where $i \in \mathbb{Q}$ is not necessarily an integer. As can be seen in Equation (3.52), f is an affine function in i. By Theorem 3.3.8, when $i = 1$,

$$f(n,k,m,1) = \frac{kn+1}{2} + \frac{(-1)^{n-m}}{2D(k,n-m)}. \qquad (3.53)$$

When $i = (kn+1)/2$, this implies that

$$f(n,k,m,(kn+1)/2) = \frac{kn+1}{2}. \qquad (3.54)$$

Because $f(n,k,m,i)$ is affine in i, it is enough to use linear interpolation and extrapolation to compute f for arbitrary i. This can be done by scaling the $\dfrac{(-1)^{n-m}}{2D(k,n-m)}$ term by an affine function of i which is 1 when $i = 1$ and which vanishes when $i = (kn+1)/2$, namely $\dfrac{kn+1-2i}{kn-1}$, as desired. □

Example 3.3.10. *For $n = 2$, $k = 2$, and $m = 0$ the expected value of the first letter in a permutation in $S_{nk} = S_4$ with no $k = 2$-cycles is $\dfrac{13}{5}$, as shown in Example 3.1.2. This agrees with Theorem 3.3.8:*

$$\frac{kn+1}{2} + \frac{(-1)^{n-m}}{2D(k,n-m)} = \frac{4+1}{2} + \frac{(-1)^{2-0}}{2D(2,2-0)} = \frac{5}{2} + \frac{1}{10} = \frac{13}{5}, \qquad (3.55)$$

since $D(2,2) = 5$ as illustrated in Figure 3.1.

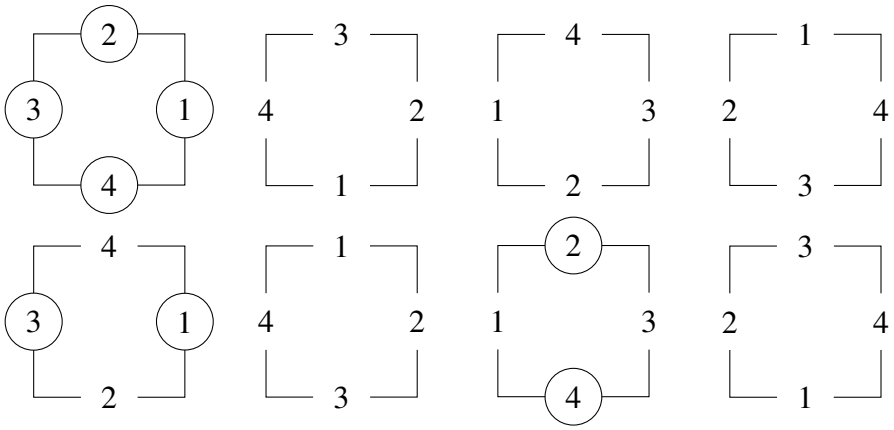


Figure 3.1: The $2^2 2! = 8$ symmetries of a square with fixed sides circled. The square (2-dimensional hypercube) has symmetry group $S(2,2) = (\mathbb{Z}/2\mathbb{Z}) \wr \mathbb{S}_2$ and $D(2,2) = 5$ of these symmetries are derangements, meaning that they do not fix any sides.

While Theorem 3.2.7 gave us our first way to efficiently compute the expected value of the first letter of a permutation on kn letters with a given number of k-cycles, we can also compute this efficiently with Theorem 3.3.8 by using the formulas for $D(k,n)$ in Theorem 3.3.5. But this is not the only reason that Theorem 3.3.8 is of interest; because of the structure of the formula it provides, this theorem suggests other quantitative and qualitative insights.

Recall that when there are no restrictions on a permutation $\pi \in S_{kn}$, the first letter is equally likely to take on any value, so $\mathbb{E}[\pi(1) \mid \pi \in S_{kn}] = (kn+1)/2$. The first insight given by Theorem 3.3.8 is that the expected value of $\pi(1)$ with a given number of k-cycles differs from $(kn+1)/2$ by at most $1/2$, because $D(k,N) \geq 1$ for all $N \geq 0$ and $k \geq 2$. Secondly, since $D(k,N)$ increases as a function of N, the expected value gets closer to $(kn+1)/2$ as the number of k-cycles decreases. Lastly, the numerator of $(-1)^{n-m}$ in the second summand of Equation (3.46) shows that the expected value of the first letter is larger than $(kn+1)/2$ if and only if n and m have the same parity.

3.4 A k-cycle preserving bijection

Motivated by Equation (3.4a), this section describes a family of bijections,

$$\phi_k \colon S_{n-1} \times [n] \to S_n, \tag{3.56}$$

each of which preserves the number of k-cycles when $k \nmid n$. Of course, there is no map that preserves the number of k-cycles when $k \mid n$. For example, a permutation in S_n consisting entirely of k-cycles contains n/k k-cycles, while a permutation in S_{n-1} can contain at most $n/k-1$ k-cycles by the pigeonhole principle.

Informally, these maps are defined by writing down a permutation $\sigma \in S_{n-1}$ in *canonical cycle notation*, incrementing all letters in σ that are greater than or equal to $x \in [n]$, inserting x into the rightmost cycle, and then recursively moving letters into or out of subsequent cycles, whenever a k-cycle is turned into a $(k+1)$-cycle or a $(k-1)$-cycle is turned into a k-cycle.

3.4.1 Example of recursive structure

The definition of the map can look complicated, so it's worthwhile to start with an example to give some sense of the overarching idea.

Example 3.4.1. *This example illustrates how the map ϕ_3 inserts a letter into a permutation while preserving the number of 3-cycles. The maps ϕ_k and ψ_k are the result of moving letters according to the arrows and are applied from right to left. (This example uses the convention that $1 < 2 < \cdots < 9 < A < B < \cdots < N$.)*

$$\phi_3((D76)(E\text{␣})(F\underline{3}2\text{␣})(G\underline{9}1\underline{C})(K\text{␣}5\underline{4})(L\text{␣}J\underline{8})(M\text{␣}B\text{␣})(N\underline{A}H\text{␣}),\underline{I})$$

$$= (D76)(E3)(F29)(G1)(KC5)(L4J)(M8BA)(NHI)$$

$$\psi_3((D76)(E\underline{3})(F\llcorner 2\underline{9})(G\llcorner 1\llcorner)(K\underline{C}5\llcorner)(L\underline{4}J\llcorner)(M\underline{8}B\underline{A})(N\llcorner H\underline{I}))\llcorner$$

$$= ((D76)(E)(F32)(G91C)(K54)(LJ8)(MB)(NAH),I)$$

Again, it is worth reemphasizing that the following definitions will follow the convention that permutations are written in canonical cycle notation,

$$\pi = \underbrace{(c_1^{(t)} \cdots c_{\ell_t}^{(t)})}_{c^{(t)}} \cdots \underbrace{(c_1^{(1)} \cdots c_{\ell_1}^{(1)})}_{c^{(1)}}, \tag{3.57}$$

where cycle $c^{(i)} = (c_1^{(i)} \cdots c_{\ell_i}^{(i)})$ has ℓ_i letters. This means that the first letter in each cycle, $c_1^{(i)}$, is the largest letter in that cycle, and that the cycles are ordered in decreasing order by first letter when read from right to left: $c_1^{(i+1)} < c_1^{(i)}$ for all i.

3.4.2 Formal definition and properties

We will start by defining two functions, ϕ_k and ψ_k which allow us to insert a letter and extract a letter respectively. We will later see in Lemma 3.4.7 that these functions are inverse to each other.

Definition 3.4.2. *Let $\phi_k \colon S_{n-1} \times [n] \to S_n$ be defined recursively as follows:*

$$\phi_k(\emptyset, 1) = (1), \tag{3.58}$$

and for $n > 1$, $\pi \in S_{n-1}$, and $x \in [n]$,

$$\phi_k(\pi, x) = \begin{cases} c^{(t)} \cdots c^{(1)}(x) & x > c_1^{(1)} \quad (3.59a) \\ \phi_k(c^{(t)} \cdots c^{(2)}, c_2^{(1)})(c_1^{(1)} c_3^{(1)} \cdots c_k^{(1)} x) & \ell_1 = k \quad (3.59b) \\ \pi'(c_1^{(1)} x' c_2^{(1)} \cdots c_{k-1}^{(1)} x) & \ell_1 = k-1, t > 1 \quad (3.59c) \\ c^{(t)} \cdots c^{(2)}(c_1^{(1)} \cdots c_{\ell_1}^{(1)} x) & \textit{otherwise.} \quad (3.59d) \end{cases}$$

Here, ϕ_k depends on the auxillary function $\psi_k \colon S_n \to S_{n-1} \times [n]$,

$$\psi_k(\pi) = \begin{cases} \left(c^{(t)} \ldots c^{(2)}, c_1^{(1)}\right) & \ell_1 = 1 \quad (3.60a) \\ \left(\phi_k(c^{(t)} \cdots c^{(2)}, c_2^{(1)})(c_1^{(1)} c_3^{(1)} \ldots c_k^{(1)}), c_{k+1}^{(1)}\right) & \ell_1 = k+1 \quad (3.60b) \\ \left(\pi'(c_1^{(1)} x' c_2^{(1)} \ldots c_{k-1}^{(1)}), c_k^{(1)}\right) & \ell_1 = k, t > 1 \quad (3.60c) \\ \left(c^{(t)} \cdots c^{(2)}(c_1^{(1)} \cdots c_{\ell_1 - 1}^{(1)}), c_{\ell_1}^{(1)}\right) & \textit{otherwise,} \quad (3.60d) \end{cases}$$

and in both functions, $(\pi', x') = \psi(c^{(t)} \cdots c^{(2)})$.

Note 3.4.3. *Strictly speaking, ϕ_k and ψ_k have an additional implicit parameter n, which indicates the size of permutation that these functions act on. Since the constructions of these functions do not depend on n, this is suppressed in the notation.*

The following theorem motivates this map, and together with Lemma 3.4.7, it implies Equation (3.4a).

Theorem 3.4.4. *If $k \nmid n$, the number of k-cycles of $\pi \in S_{n-1}$ is equal to the number of k-cycles in $\phi_k(\pi, x)$.*

Proof. By construction, the maps ϕ_k and ψ_k change the rightmost cycle into a (different) k-cycle if it was previously a k-cycle, and they change non-k-cycles into non-k-cycles, except for the case

where there is one cycle remaining with length $k-1$ (in the case of ϕ) or length k (in the case of ψ). These cases can only be achieved when $k \mid n$, by the following lemma. □

Lemma 3.4.5. *The number of letters in π in (recursive) applications of ϕ_k and ψ_k are congruent to $n-1 \pmod{k}$ and $n \pmod{k}$, respectively. Therefore, the only time that the input to ϕ_k can be a single cycle of length $k-1$ or the input to ψ_k can be a single cycle of length k is when $n \equiv 0 \pmod{k}$.*

Proof. The proof proceeds by induction on the number of recursive iterations of ϕ_k and ψ_k. The base case is clear: on the first application of the map $\phi_k \colon S_{n-1} \times [n] \to S_n$, the input permutation has $n-1$ letters by definition.

Now, either we're finished, or we recurse (Equations (3.59b), (3.59c), (3.60b), or (3.60c)), which we look at case-by-case.

Case 1. In Equation (3.59b), the map ϕ_k sets aside k letters from the input, so the number of letters in the recursive input to ϕ_k is also congruent to $n-1 \pmod{k}$.

Case 2. In Equation (3.59c), the map ϕ_k sets aside $k-1$ letters from the leftmost cycle of the input. Since the number of letters in the original permutation was congruent to $n-1 \pmod{k}$, the number of letters in the permutation being input to ψ_k is congruent to $n \pmod{k}$.

Case 3. In Equation (3.60b), the map ψ_k sets aside $k+1$ letters from the leftmost cycle of the input. Since the number of letters in the original permutation was congruent to $n \pmod{k}$, the number of letters in the permutation being input to ϕ_k is congruent to $n-1 \pmod{k}$.

Case 4. In Equation (3.60c), the map ψ_k sets aside k letters from the input, so the number of letters in the recursive input to ψ_k is also congruent to $n \pmod{k}$.

□

The following lemma provides a certain "niceness" property of the map, which allows us to analyze it. In particular, all recursive inputs in both ϕ_k and ψ_k are written in canonical cycle notation.

Lemma 3.4.6. *The output of ϕ_k is in canonical cycle notation.*

Proof. Canonical cycle notation is preserved by construction. In particular, ϕ_k moves the first letter in any cycle, and Equation (3.59a) guards against inserting a number into a cycle that is bigger than the largest number already in the cycle. Similarly, ψ_k only moves the first letter in the case of Equation (3.60a), but in this case, the cycle only has one letter, so this is equivalent to deleting the cycle. □

3.4.3 Inverting the bijection

Lemma 3.4.7. *The maps $\phi_k \colon S_{n-1} \times [n] \to S_n$ and $\psi_k \colon S_n \to S_{n-1} \times [n]$ are inverse to one another.*

Proof. To prove this lemma, it suffices to show that $\psi_k \circ \phi_k = \mathrm{id}$ by induction on the number of cycles of π. This will simultaneously prove that $\phi_k \circ \psi_k = \mathrm{id}$, because $S_{n-1} \times [n]$ and S_n, both having $n!$ elements, have the same cardinality.

When π has no cycles, the base case is clear: $\psi_k(\phi_k(\emptyset, x)) = \psi_k((x)) = (\emptyset, x)$.

Now there are five remaining cases to check, corresponding to each of the cases in the definition of $\phi_k(\pi, x)$:

Case 1. Assume $x > c_1^{(1)}$, so that $\phi_k(\pi, x)$ is evaluated via Equation (3.59a):

$$\psi_k(\phi_k(\pi, x)) = \psi_k(c^{(t)} \cdots c^{(1)}(x)) \tag{3.61}$$

$$= (c^{(t)} \cdots c^{(1)}, x) \tag{3.62}$$

$$= (\pi, x). \tag{3.63}$$

Case 2. Assume $\ell_1 = k$, so that $\phi_k(\pi, x)$ is evaluated via Equation (3.59b):

$$\psi_k(\phi_k(\pi, x)) = \psi_k(\phi_k(c^{(t)} \cdots c^{(2)}, c_2^{(1)}) \underbrace{(c_1^{(1)} c_3^{(1)} \cdots c_k^{(1)} x)}_{\text{length } k}) \tag{3.64}$$

$$= (\pi'(c_1^{(1)} x' c_3^{(1)} \ldots c_k^{(1)}), x) \tag{3.65}$$

Case 3. Assume $\ell_1 = k-1$ and $t > 1$, so that $\phi_k(\pi, x)$ is evaluated via Equation (3.59c):

$$\psi_k(\phi_k(\pi,x)) = \psi_k(\pi' \underbrace{(c_1^{(1)} x' c_2^{(1)} \cdots c_{k-1}^{(1)} x)}_{\text{length } k+1}) \tag{3.66}$$

where $(\pi', x') = \psi_k(c^{(t)} \dots c^{(2)})$. Therefore, this simplifies by Equation (3.60c):

$$\psi_k(\phi_k(\pi,x)) = \left(\phi_k(\pi',x')(c_1^{(1)} \cdots c_{k-1}^{(1)}), x\right) \tag{3.67}$$

$$= \left(\underbrace{\phi_k(\psi_k(c^{(t)} \dots c^{(2)}))}_{c^{(t)} \dots c^{(2)}} \underbrace{(c_1^{(1)} \cdots c_{k-1}^{(1)})}_{c^{(1)}}, x\right) \tag{3.68}$$

$$= (\pi, x), \tag{3.69}$$

because $\phi_k(\psi_k(c^{(t)} \dots c^{(2)})) = c^{(t)} \dots c^{(2)}$ by the induction hypothesis on $t-1$ letters.

Case 4. Assume that $x > c_1^{(1)}$ and $\ell_1 \notin \{k-1, k\}$, so that $\phi_k(\pi, x)$ is evaluated via Equation (3.59d):

$$\psi_k(\phi_k(\pi,x)) = \psi_k(c^{(t)} \cdots c^{(2)} (c_1^{(1)} \cdots c_{\ell_1}^{(1)} x)) \tag{3.70}$$

$$= (c^{(t)} \cdots c^{(1)}, x) \tag{3.71}$$

$$= (\pi, x). \tag{3.72}$$

Case 5. Assume that $\ell_1 = k-1$ and $t = 1$, so that $\phi_k(\pi, x)$ is evaluated via Equation (3.59d):

$$\psi_k(\phi_k(\pi,x)) = \psi_k((c_1^{(1)} \cdots c_{k-1}^{(1)} x)) \tag{3.73}$$

$$= (c^{(1)}, x) \tag{3.74}$$

$$= (\pi, x). \tag{3.75}$$

□

In this section we constructed a recursively defined map and its inverse to give a bijective proof that $C_k(n,m) = nC_k(n-1,m)$ when $k \nmid n$. This is a novel, reversible algorithm for inserting a letters into a permutation that preserves the number of k-cycles whenever possible.

3.5 Further directions

In the introduction, we mentioned Conger's paper which analyzed how the number of descents of a permutation affects the expected value of the first letter of the permutation. And similarly in the following sections, we looked at how the number of k-cycles affects the expected value of the first letter of the permutation. This section will principally look at the obvious generalization: given some permutation statistic $\text{stat}\colon S_n \to \mathbb{Z}$, does the map

$$f(n,m) = \mathbb{E}[\pi(i) \mid \pi \in S_n, \text{stat}(\pi) = m] \tag{3.76}$$

have any interesting structure?

Notice that the first letter of a permutation is itself a statistic, so we can ask a more general question. Given pairs of statistics $(\text{stat}_1, \text{stat}_2)$, does the map

$$g(n,m) = \mathbb{E}[\text{stat}_1(\pi) \mid \pi \in S_n, \text{stat}_2(\pi) = m] \tag{3.77}$$

have any interesting structure?

3.5.1 FindStat database

The result by Conger gives the expected value of $\pi(1)$ given $\text{des}(\pi)$, and this chapter gave the expected value of $\pi(1)$ given the number of k-cycles of π. Of course, it would be interesting to do analogous analysis with other statistics. In particular, the FindStat permutation statistics database [22] contains over 380 different permutation statistics, and many of these appear to have some structure with respect to the expected value of the first letter of a permutation.

Moreover, this database currently has a total of about 1800 statistics on a variety of combinatorial objects: binary trees, Dyck paths, parking functions, posets, and standard tableaux, to name a few. By choosing two statistics on a given object, it may be interesting to ask how knowing the value of one statistic changes the expected value of the other.

3.5.2 Mahonian statistics

In particular, the family of Mahonian statistics may be fruitful to investigate. Below, we have given conjectures about two: the major index and the inversion number. Mahonian statistics are maps $\mathrm{mah}\colon S_n \to \mathbb{N}_{\geq 0}$ that are equidistributed with the inversion number.[23] That is,

$$\#\{w \in S_n : \mathrm{mah}(w) = k\} = \#\{w \in S_n : \mathrm{inv}(w) = k\}. \tag{3.78}$$

Naturally, all Mahonian statistics share the same generating function:

$$\sum_{\sigma \in S_n} x^{\mathrm{mah}(\sigma)} = [n]_q! = \prod_{i=0}^{n-1} \sum_{j=0}^{i} (q^j). \tag{3.79}$$

Because the expected value of the first letter is given by the weighted sum of the permutations with $\mathrm{mah}(w) = k$ divided by the number of such permutations, $\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{mah}(\pi) = k]$ has a denominator that is (a factor of) $M(n,k)$, the number of permutations of $w \in S_n$ such that $\mathrm{inv}(w) = k$. For fixed k, these satisfy a degree-k polynomial for all $n > k$. Notably, in the cases of the major index and the inversion number, the numerators appear to satisfy degree-k and degree-$k-1$ polynomials respectively.

Conjecture 3.5.1. *For fixed k and $n > k$, the expected value of the first letter of a permutation with a given number of inversions is equal to a rational function in n given by*

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{inv}(\pi) = k] = \frac{M(n+1,k)}{M(n,k)}, \tag{3.80}$$

where $M(n,k)$, as above, is the number of permutations $w \in S_n$ such that $\mathrm{inv}(w) = k$.

Conjecture 3.5.2. *For fixed $k > 0$ and $n \geq k$, $\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{maj}(\pi) = k]$ satisfies a rational function in n that is $1/(k+1)$ times the quotient of a monic degree-$(k+1)$ polynomial by a monic degree-k polynomial. Specifically,*

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{maj}(\pi) = 1] = \frac{1}{2}\left(\frac{n^2+n-2}{n-1}\right), \tag{3.81}$$

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{maj}(\pi) = 2] = \frac{1}{3}\left(\frac{n^3-n-6}{n^2-n-2}\right), \tag{3.82}$$

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{maj}(\pi) = 3] = \frac{1}{4}\left(\frac{n^4+6n^3-13n^2-18n}{n^3-7n}\right), \text{ and} \tag{3.83}$$

$$\mathbb{E}[\pi(1) \,|\, \pi \in S_n, \mathrm{maj}(\pi) = 4] = \frac{1}{5}\left(\frac{n^5+20n^4-45n^3-80n^2-16n}{n^4+2n^3-13n^2-14n}\right). \tag{3.84}$$

Note that the denominator is given by an integer multiple of $M(n,k)$, a degree-k polynomial.

3.5.3 An elusive bijection

Let $F_k(n,m)$ be the number of elements of the generalized symmetric group $S(k,n) = (\mathbb{Z}/k\mathbb{Z}) \wr S_n$ with m fixed points, and recall that $C_k(n,m)$ is the number of elements of S_{kn} with m k-cycles. For each pair of nonnegative integers (α,β) with $\alpha,\beta \leq n$, Lemma 3.3.7 suggests that there exists a bijection of sets

$$C_k(n,\alpha) \times F_k(n,\beta) \to C_k(n,\beta) \times F_k(n,\alpha). \tag{3.85}$$

This bijection has proven to be elusive to construct outside of the special cases where $n = 1$ or $k = 1$. Note that the map cannot be a group automorphism of $S_{kn} \times S(k,n)$, because the identity of this group is in $C_k(n,0) \times F_k(n,n)$, so it cannot be preserved under this map.

It would be especially interesting if there's a way to use the embedding of $(\mathbb{Z}/k\mathbb{Z}) \wr S_n$ into S_{kn} as the centralizer of an element that is the product of n disjoint k-cycles.

Chapter 4

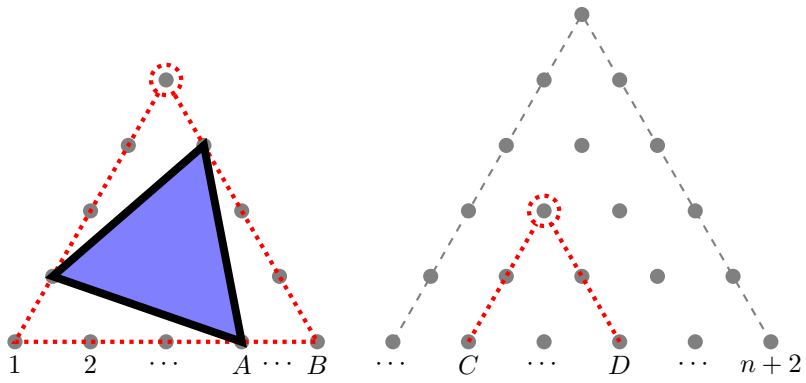
Triangles in triangles (a proof without words)

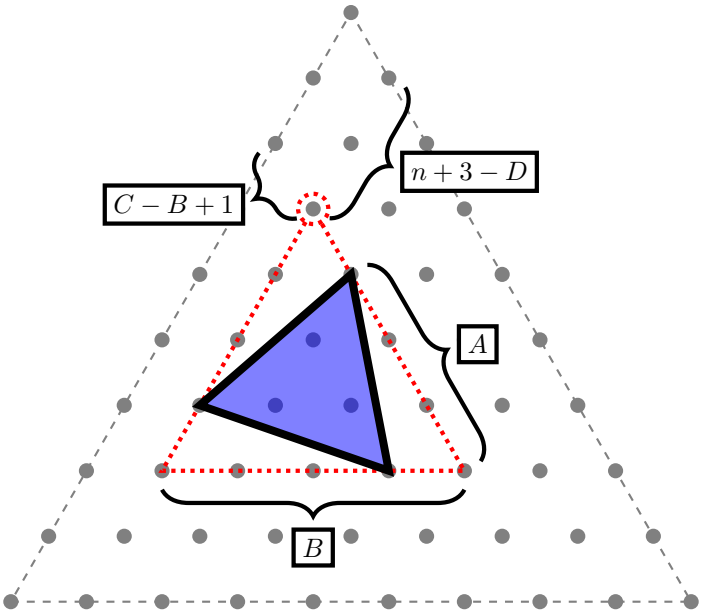
Theorem 4.0.1. *There are* $\binom{n+2}{4}$ *equilateral triangles with vertices in a triangular region of the triangular grid with n vertices on each side.*

Proof. The following is a bijection without words from a choice of four integers satisfying

$$1 \leq A < B < C < D \leq n+2$$

to equilateral triangles in the n-vertices-per-side triangular grid. (Specifically, we have illustrated the case where $n = 10$, $A = 4$, $B = 5$, $C = 6$, and $D = 10$.)




$C - B + 1$
$n + 3 - D$
A
B

□

Chapter 5

Unranking restricted permutations

In this chapter we discuss efficient methods for unranking derangements and ménage permutations. That is, given a list of these restricted permutations in lexicographic order, we will provide an algorithm to efficiently extract the k-th permutation from the list. We will show that this problem can be reduced to the problem of computing the number of restricted permutations with a given prefix, and then we will use rook theory to solve this counting problem.

5.1 Overview and preliminaries

In January 2020, Richard Arratia sent out an email announcing a talk he was going to give about "unranking derangements". He noted that one can easily enumerate derangements in both senses of the word: both counting derangements and listing them out. He asked, however, what if we want to list derangements in lexicographic order and then sample from a particular index somewhere in the middle of the list? Can one do this efficiently—in particular without writing out the entire list?

After showing him a proof of concept for unranking derangements, Richard suggested a related problem that I (wrongly) suspected was out of reach. In January 2021, he announced a $100 prize for solving the analogous problem with ménage permutations.

In the spirit of Richard's bounties, we offer a $100 bounty of our own in the final section of this chapter. Our bounty asks for an efficient way to derank the set of permutations where letters do not move exactly one space to the left or the right.

Richard was, of course, interested in an even more general question too: given some family of combinatorial objects $\mathscr{C}_n$ that can be counted efficiently (for example, derangements or Dyck paths or graphs on n vertices) and some total ordering on them (for example, lexicographic order), when is it possible to unrank the collection in some computationally efficient way?

An efficient way to unrank a collection of objects implies an efficient algorithm for sampling from the collection uniformly at random. This can potentially be of use in the case of Monte Carlo simulations and other instances where it is useful to be able to perform an unbiased sample.

Definition 5.1.1. *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 $\text{rank}_{\mathscr{C}} : \mathscr{C} \to \mathbb{N}_{>0}$ which sends $c_i \mapsto i$.*

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

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 the size of these sets grows in exponential time or worse, explicitly constructing the sequence and doing a search is not computationally feasible.

In Appendix B, we provide examples of totally ordered combinatorial objects $\mathscr{C}$ that have efficient ranking and unranking maps

In this chapter we're going to explore this idea in the context of combinatorial words in lexicographic order. In particular, we will focus on two families of restricted permutations: derangements and ménage permutations. We will show that the existence of an efficient way to count the number of such permutations with a given prefix implies that there is an efficient way to compute the ranking and unranking maps. Then we will develop some ideas from rook theory and apply them to the context of derangements and ménage permutations.

5.2 Prefix counting and word ranking

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

We begin by developing a general theory for unranking collections of words in lexicographic order by counting the number of words with a given prefix.

5.2.1 Words with a given prefix

We will start by introducing some basic definitions about words and prefixes, and to formalize the notion of lexicographic order.

Definition 5.2.1. *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 $\mathcal{W}_{\mathscr{A}}$, or just $\mathcal{W}$ when the alphabet is implicit from context.

Definition 5.2.2. *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 $i \leq M$.*

Definition 5.2.3. *A word w is said to be before w' in **lexicographic order** if either w is a proper prefix of w', or if at the first position, i, where w and w' differ, $w_i < w'_i$.*

With these definitions established, we can turn the problem of unranking words into a problem about counting words with specified prefixes.

Theorem 5.2.4. *For $k > 0$, let $\mathcal{W}$ be a set of nonempty words on the alphabet $[n]$, and let $\mathscr{C} \subsetneq \mathcal{W}$ be a finite subset of words on this alphabet, with a total order given by its lexicographic order.*

Let $\#\text{prefix}_{\mathscr{C}} : \mathcal{W} \to \mathscr{C}$ 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

$$\text{unrank}_{\mathscr{C}}(i) = f_i^{\mathscr{C}}((1), 0) \tag{5.1}$$

where

$$f_i^{\mathscr{C}}(\alpha, j) = \begin{cases} f_i^{\mathscr{C}}(\alpha', j + \#\text{prefix}_{\mathscr{C}}(\alpha)) & i > j + \#\text{prefix}_{\mathscr{C}}(\alpha) \quad (5.2a) \\ f_i^{\mathscr{C}}(\alpha'', j) & \alpha \notin \mathscr{C} \text{ and } i \leq j + \#\text{prefix}_{\mathscr{C}}(\alpha) \quad (5.2b) \\ f_i^{\mathscr{C}}(\alpha'', j+1) & \alpha \in \mathscr{C},\ i \leq j + \#\text{prefix}_{\mathscr{C}}(\alpha), \text{ and } i \neq j+1 \quad (5.2c) \\ \alpha & \alpha \in \mathscr{C} \text{ and } i = j+1, \quad (5.2d) \end{cases}$$

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

Proof. We make four claims that we will prove using induction on the recursive applications of $f_i^{\mathscr{C}}(\alpha, j)$: that j is the number of words in $\mathscr{C}$ that occur strictly before α, that $\alpha \leq w_i$, and that the sequence of αs is strictly increasing.

This final claim (that the sequence of αs is strictly increasing) follows from the observation that $\alpha < \alpha'' < \alpha'$ in lexicographic order.

Because each iteration increases either j or ℓ or both, the number of recursive applications of $f_i^{\mathscr{C}}$ required to determine w_i is at most $i + \max_{w \in \mathcal{W}} |w|$, which is finite because $\mathcal{W}$ only contains of finite words.

The base case is clear: We start with $f_i^{\mathscr{C}}((1), 0)$ because (1) is the lexicographically earliest word, so 0 nonempty words strictly precede it, and $(1) \leq w_i$.

We will repeatedly use the observation that if j words precede α, then a word has prefix α if and only if its index is in $(j, j + \#\text{prefix}_{\mathscr{C}}(\alpha)]$. Note that this range is empty whenever there are no words prefixed by α.

Case (5.2a). Because $j + \#\text{prefix}_{\mathscr{C}}(\alpha)$ is the index of the last word that begins with α, if $i > j + \#\text{prefix}_{\mathscr{C}}(\alpha)$, then w_i must begin with a length-ℓ prefix that is lexicographically later than α.

By construction, α' is the lexicographically earliest word of length ℓ that comes after α, therefore $\alpha' \leq w_i$. As such, the number of words that strictly precede α' is $j + \#\text{prefix}_{\mathscr{C}}(\alpha)$, which is

the sum of the number of words that occur strictly before α and the number of words that have prefix α.

Case (5.2b). If $\alpha \notin \mathscr{C}$ and $i \leq j + \#\text{prefix}_{\mathscr{C}}(\alpha)$, then α is a *proper* prefix of w_i, and w_i is of length at least $\ell + 1$. By construction, α'' is the lexicographically earliest word of length $\ell + 1$ that has a prefix of α, so $\alpha'' \leq w_i$, and the number of words in $\mathscr{C}$ that precede α'' is equal to j, the number of words that precede α.

Case (5.2c). If $\alpha \in \mathscr{C}$, $i \leq j + \#\text{prefix}_{\mathscr{C}}(\alpha)$, and $i \neq j + 1$ then α must be the word at index $j + 1 < i$, because α itself is the lexicographically earliest word with the prefix α. Because words cannot appear multiple times in $\mathscr{C}$, w_i must have α as a *proper* prefix. Therefore $\alpha'' \leq w_i$ and the number of words that strictly precede α'' is $j + 1$: the number of words that strictly precede α plus α itself.

Case (5.2d). If $\alpha \in \mathscr{C}$ and $i = j + 1$, then $w_i = \alpha$ because α itself is the lexicographically earliest word with the prefix α, so it must occur at index $i = j + 1$. □

Notice that each recursive call of $f_i^{\mathscr{C}}$ increases the sum of the letters of α. If we suppose that $\mathscr{C}^{(\leq k)} \subsetneq \mathcal{W}_{[n]}$ is a finite set of words on the alphabet $[n]$ of length at most k, then unranking a word from $\mathscr{C}^{(\leq k)}$ requires at most nk recursive applications of $f_i^{\mathscr{C}^{(\leq k)}}$.

Therefore if there exists a polynomial time algorithm for computing $\#\text{prefix}_{\mathscr{C}}$, then there exists an unranking algorithm that is polynomial in the size of the alphabet $\mathscr{A}$ and the length of the longest word. In the case of restricted permutations, each of these grow linearly with the number of letters in the ménage permutations.

5.2.2 Ranking words

Just as we can recursively find a word at a given index, we can also recursively find a index corresponding to a given word.

Claim 5.2.5. *If $\mathscr{C}$ is a collection of words such that no word is a prefix of another, then the rank of the word $w \in \mathscr{C}$ can be computed as the sum*

$$1 + \sum_{i=1}^{|w|} \sum_{i=1}^{w_i - 1} \#\text{prefix}_{\mathscr{C}}(w_{(i)}^{x})$$

where $w = (w_1, w_2, \ldots, w_{|w|})$ and $w_{(i)}^{x} = (w_1, w_2, \ldots, w_{i-1}, x)$.

5.3 Basic notions of rook theory

Now that we have shown that we can unrank words whenever we can compute the number of words with a given prefix, we want to develop techniques for this new counting problem. 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 [24] 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.

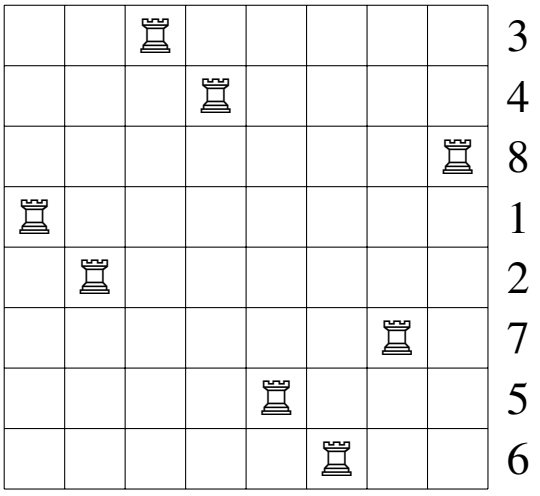


Figure 5.1: An illustration of the rook placement corresponding to the permutation $34812756 \in S_8$. A rook is placed in square $(i, \pi(i))$ for each i.

5.3.1 Definitions in rook theory

We begin by introducing some preliminary ideas from rook theory.

Definition 5.3.1. *A **board** B is a subset of $[n] \times [n]$ which represents the squares of an $n \times n$ 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 squares of B in such a way that no two rooks are in the same row or column.

Definition 5.3.2. *The **rook polynomial** associated with a board B,*

$$p_B(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_n x^n, \tag{5.3}$$

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 does 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 #P-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, since there is a bijection between boards and $(0,1)$-matrices. (In fact, Shevelev [25] claims that "the theory of enumerating the permutations with restricted positions stimulated the development of the theory of the permanent.")

Lemma 5.3.3. *Let $M_B = \{a_{ij}\}$ be an $n \times n$ matrix where*

$$a_{ij} = \begin{cases} 1 & (i,j) \in B \\ 0 & (i,j) \notin B \end{cases}. \tag{5.4}$$

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

$$\text{perm}(M_B) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i\sigma(i)}. \tag{5.5}$$

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

Theorem 5.3.4 (Valiant's Theorem [26])**.** *The counting problem of computing the permanent of a (0,1)-matrix is #P-complete.*

Corollary 5.3.5. *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.

5.3.2 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 5.3.6 ([27])**.** *Given a board B and a square $(x,y) \in B$, we can define two resulting boards from including or excluding the given square:*

$$B_i = \{(x',y') \in B : x \neq x' \text{ and } y \neq y'\} \tag{5.6}$$

$$B_e = B \setminus (x,y). \tag{5.7}$$

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

$$p_B(x) = x p_{B_i}(x) + p_{B_e}(x). \tag{5.8}$$

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. When the number of squares that are missing from B is small, it can be easier to compute the rook polynomial of the complementary board, p_{B^c}, and use the principle of inclusion/exclusion on its 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. We will 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.

5.4 Unranking derangements

5.4.1 The answer to a $100 question about derangements

In January 2020, Richard Arratia sent out an email proposing a seminar talk. The title describes his 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 A computed the answer in less than twenty milliseconds. When written as words in lexicographic order, the derangement in* S_{20} *with rank* 5×10^{17} *is*

$$12\ 14\ 2\ 9\ 13\ 20\ 6\ 3\ 1\ 17\ 5\ 11\ 19\ 15\ 10\ 18\ 8\ 7\ 4\ 16. \tag{5.9}$$

5.4.2 Overview for unranking derangements

Arratia's question focused on unranking derangements written as words in lexicographic order. Other authors have looked at unranking derangements based on other total orderings. In particular, Mikawa and Tanaka [28] give an algorithm to rank/unrank derangements with respect to *lexicographic ordering in cycle notation*.

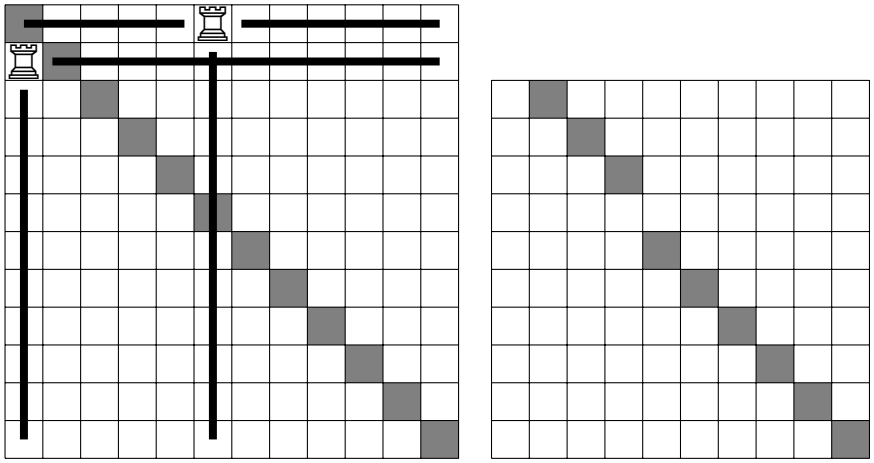


Figure 5.2: An example of a prefix $\alpha = (6,1)$, and the board that results from deleting the first $\ell = 2$ rows and columns 6 and 1. The derived complementary board of B from α is $B^c_\alpha = \{(1,2),(2,3),(3,4),(5,5),\ldots,(10,10)\}$.

In this section we will develop an algorithm for ranking and unranking derangements 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 5.4.1. *A **derangement** is a permutation $\pi \in S_n$ such that π has no fixed points. That is, the set of derangements on n letters is*

$$\mathscr{D}_n = \{\pi \in S_n : \pi(i) \neq i \; \forall i \in [n]\}. \tag{5.10}$$

5.4.3 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 ℓ rooks according to these positions, as illustrated in Figure 5.2.

Definition 5.4.2. *If B is an $n \times n$ board, and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ is a valid prefix of length ℓ, the **derived board** of B from α, denoted B_α, is constructed by removing rows $1,2,\ldots,\ell$ and columns $\alpha_1, \alpha_2, \ldots, \alpha_\ell$ from B, reindexing in such a way that both the row and column indexes are in $[n-\ell]$.*

*The **derived complementary board** B^c_α is the complement of B_α with respect to $[n-\ell] \times [n-\ell]$.*

Given a prefix of length ℓ, the number of ways of placing $n-\ell$ rooks on the derived board B_α is, by construction, equal to the number of words in $\mathscr{C}$ with prefix α

Lemma 5.4.3. *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 derived complementary board is*

$$|B_\alpha^c| = n - \ell - |\{\ell+1, \ell+2, \ldots, n\} \cap \{\alpha_1, \alpha_2, \ldots, \alpha_\ell\}|, \tag{5.11}$$

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

Proof. Notice that the derived complementary board can be constructed in a different order: by first taking the complement, then deleting rows and columns, and finally reindexing the squares. Because the complementary board has no two squares in the same row or column, deleting and reindexing results in a derived complementary board with the same property.

Thus, we only need to classify which squares in the complementary board are deleted to make the derived complementary board. We start by deleting ℓ squares corresponding to the deletion of the first ℓ rows, namely $(1,1),(2,2),\ldots,(\ell,\ell)$.

Some of these squares may also be in columns $\alpha_1, \alpha_2, \ldots, \alpha_\ell$, but to avoid double-counting, we only consider those letters that are greater than ℓ. These are $|\{\ell+1, \ell+2, \ldots, n\} \cap \{\alpha_1, \alpha_2, \ldots, \alpha_\ell\}|$, as desired. □

5.4.4 Derangements with a given prefix

Now that we have a way of quickly computing $|B_\alpha^c|$, we can compute the number of ways to place a given number of rooks on the complementary board. We can use this to compute the rook polynomial for the derived complementary board $p_{B_\alpha^c}(x)$. We will see later that we can use the coefficients of this polynomial to compute the number of ways of placing $n-\ell$ rooks on the derived board B_α.

Lemma 5.4.4. *The rook polynomial for the complementary board B_α^c is*

$$p_{B_\alpha^c}(x) = \sum_{j=0}^{|B_\alpha^c|} \binom{|B_\alpha^c|}{j} x^j. \tag{5.12}$$

Proof. Recall that no two squares of B_α^c are in the same row or column. Thus the number of ways to place j rooks is equivalent to selecting any j squares from the collection of $|B_\alpha^c|$ squares.

Therefore the coefficient of x^j in the rook polynomial is $\binom{|B_\alpha^c|}{j}$. □

Now we introduce a lemma of Stanley [29] to compute the number of ways of placing $n-\ell$ rooks in the derived board $B_\alpha \subseteq [n-\ell] \times [n-\ell]$.

Lemma 5.4.5 ([29])**.** *Let $B \subseteq [n] \times [n]$ be a board with complementary board B^c, and denote the rook polynomial of B^c by $P_{B^c}(x) = \sum_{k=0}^{n} r_k^c x^k$.*

Then the number of ways, N_0, of placing n nonattacking rooks on B is given by the principle of inclusion/exclusion

$$N_0 = \sum_{k=0}^{n} (-1)^k r_k^c (n-k)!. \tag{5.13}$$

This lemma allows us to compute the number of rook placements on the derived board B_α, which is the number of derangements in $\mathscr{D}$ that begin with the prefix α.

Corollary 5.4.6. *The number of derangements with prefix $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ is given by*

$$\#\text{prefix}_{\mathscr{D}}(\alpha) = \sum_{j=0}^{|B_\alpha^c|} (-1)^j \binom{|B_\alpha^c|}{j} (n-\ell-j)!, \tag{5.14}$$

which is A047920$(n-\ell, |B_\alpha^c|)$ *in the On-Line Encyclopedia of Integer Sequences [10].*

Because we can compute $|B_\alpha^c|$ from α in linear time (see Lemma 5.4.3), if we use a computational model where factorials are given by an oracle and arithmetic can be computed in constant time, then $\#\text{prefix}_{\mathscr{D}}$ can be computed in linear time with respect to ℓ, the length of the prefix.

Example 5.4.7. *For $n = 12$, we wish to count the number of derangements that start with the prefix $\alpha = (6,1)$, as illustrated in Figure 5.2. Since the prefix has two letters, $\ell = 2$ and $n - \ell = 12 - 2 = 10$. The number of squares in B_α^c is*

$$|B_\alpha^c| = 12 - 2 - \underbrace{|\{3,4,\ldots,12\} \cap \{6,1\}|}_{1} = 9. \tag{5.15}$$

Thus there are $A047920(10,9) = 1\,468\,457$ derangements in S_{12} that start with the prefix $\alpha = (6,1)$.

Now that we have an efficient algorithm for computing $\#\text{prefix}_{\mathscr{D}} \colon \mathcal{W} \to \mathbb{N}_{\geq 0}$, we can invoke the recursive formula in Theorem 5.2.4 to compute $\text{unrank}_{\mathscr{D}} \colon \mathbb{N}_{\geq 0} \to \mathscr{D}$ and unrank derangements. The sequence of recursive steps is illustrated in Table 5.1.

5.5 Unranking ménage permutations

After claiming Richard's prize for unranking derangements, the conversation shifted to how this technique could be extended. A natural next step seemed to be to look at another family of restricted permutations, namely ménage permutations.

A ménage permutation comes from the *problème des ménages*, introduced by Édouard Lucas in 1891. There are a few choices of how to define these permutations, but we will use the following definition for simplicity.

Definition 5.5.1. *A **ménage permutation** is a permutation $\pi \in S_n$ such that for all $i \in [n]$, $\pi(i) \neq i$ and $\pi(i) + 1 \not\equiv i \bmod n$. The set of ménage permutations of length n is denoted by $\mathcal{M}_n$.*

5.5.1 The answer to a $100 question about ménage permutations

By February 2020, it appeared that the techniques for unranking derangements would not directly translate to the context of ménage permutations. In response, Richard Arratia upped the stakes by offering another prize for unranking ménage permutations. Specifically, he posed the following problem.

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

Table 5.1: There are $A000166(8) = 14833$ derangements on 8 letters. The table shows the recursive steps to find that the derangement at index 1000 is 25487361.

$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.*

Using the techniques described in this section, we developed a computer program that computed the answer in less than 30 milliseconds. (By comparison, unranking the 10^{157}-th ménage permutation of the more than 1.25×10^{157} ménage permutations in S_{100} with the same program takes about 7 seconds.)

$100 Answer. *The desired permutation is*

$$7\ 16\ 19\ 12\ 2\ 8\ 15\ 1\ 18\ 14\ 3\ 9\ 20\ 10\ 5\ 17\ 13\ 4\ 11\ 6. \tag{5.16}$$

5.5.2 Overview for unranking ménage permutations

As in the section about unranking derangements, we will use the insight from Theorem 5.2.4 that if we can efficiently count the number of words with a given prefix, then we can efficiently unrank the words.

The technique exploits the following observations: after placing rooks on a board corresponding to our prefix, the remaining board has the property that its complement can be partitioned into sub-boards that do not share rows or columns. These sub-boards have a structure that we can understand, and we can leverage that understanding to compute the rook polynomials of these sub-boards and consequently of the complementary board itself. Once we have computed the rook polynomial of the complementary board, we can again use Lemma 5.4.5 to compute the number of full rook placements on the original board. This gives us the number of ménage permutations with a given prefix.

5.5.3 Disjoint board decomposition

Figure 5.3 suggestively shows a placement of rooks according to a prefix that results in a board whose complement can be partitioned into sub-boards whose squares don't share any rows or

columns. We will see that this property indeed holds in general, and we can exploit this in order to count the ménage permutations with a given prefix.

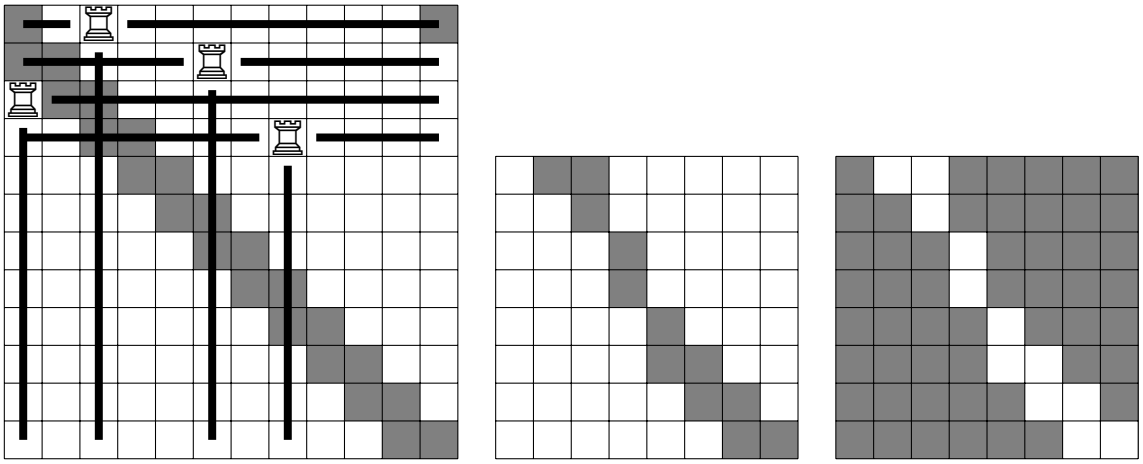


Figure 5.3: The prefix $\alpha = (3,6,1,8)$, the derived board B_α, and the derived complementary board $B_\alpha^c = \mathscr{O}_3 \sqcup \mathscr{E}_2 \sqcup \mathscr{O}_7^\intercal$. There are 8062 ways of placing eight nonattacking rooks on B_α.

The property of complements that can be partitioned into sub-boards whose squares don't share rows or columns is useful because it provides a way of factoring the rook polynomial of the bigger board into the rook polynomials of the sub-boards.

Definition 5.5.2. *Two sub-boards B and B' are called **disjoint** if no squares of B are in the same row or column as any square in B'.*

Kaplansky gives a way of computing the rook polynomial of a board in terms of its disjoint boards.

Theorem 5.5.3 ([24])**.** *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 each sub-board*

$$p_B(x) = \prod_{i=1}^{m} p_{b_i}(x). \tag{5.17}$$

We will use this disjoint board decomposition repeatedly, because the boards that result after placing a prefix can be partitioned into disjoint sub-boards whose structure is well understood. Now we will give a name to these blocks, which are illustrated in Figure 5.4.

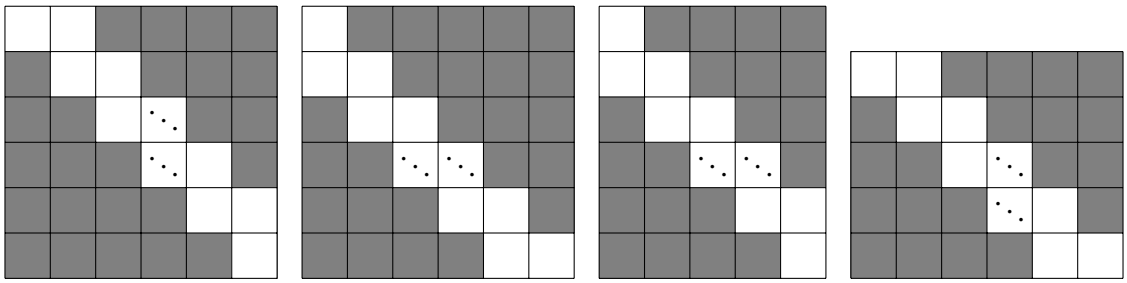


Figure 5.4: Examples of each of the four staircase-shaped boards. The first two boards are on grids of size $n \times n$, the third is on a grid of size $n \times (n-1)$ and the fourth is on a grid of size $(n-1) \times n$.

Definition 5.5.4. *A board is called **staircase-shaped** if it matches one of the following four shapes:*

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

The subscripts represent the number of squares, and the names represent their parity.

We now show that our resulting boards can be partitioned into boards of these shapes.

Lemma 5.5.5. *For $\ell \geq 1$, and prefix $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ 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, $\alpha_1 \in \{2, 3, \ldots, n-1\}$, and the derived complementary board $B^c_{(\alpha_1)}$ can be partitioned into two

disjoint sub-boards with shapes $\mathscr{O}_{2\alpha_1-3}$ and $\mathscr{O}^{\mathsf{T}}_{2n-2\alpha_1-1}$. (This is illustrated for the case of $n=7$ and α_1 in Figure 5.5.)

The inductive hypothesis is that the derived complementary board for a prefix of length $\ell-1$ consists of sub-boards with shape $\mathscr{O}_{2m-1}$, $\mathscr{O}^{\mathsf{T}}_{2m-1}$, $\mathscr{E}_{2m-2}$, or $\mathscr{E}^{\mathsf{T}}_{2m-2}$. Placing a rook in row ℓ can remove a top row or a column or both in a given sub-board. Table 5.2 below shows the resulting sub-boards after placing a rook in ℓ-th row of B, which may be in the top row, the i-th column, or both.

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

Table 5.2: 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 rows results in a board whose complementary derived board is composed of disjoint staircase-shaped sub-boards. □

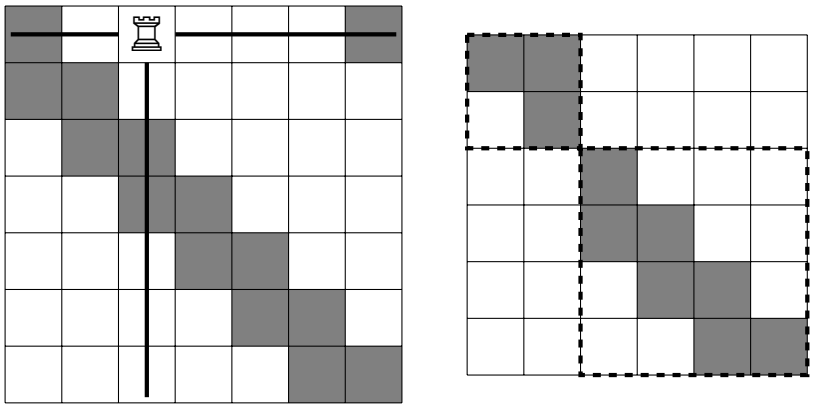


Figure 5.5: The first chessboard shows a placement of a rook at position 3, the second shows how the derived complementary board can be partitioned into two disjoint boards with 3 and 7 squares respectively.

5.5.4 Rook polynomials of blocks

Recall that the goal of partitioning B into disjoint sub-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 useful if we can describe $p_{b_i}(x)$, which is the goal of this subsection. Conveniently, the rook polynomial of each b_i will turn out to depend only on the number of squares, $|b_i|$, which can be computed recursively because of its staircase shape.

We will begin by defining a family of polynomials that, suggestively, will turn out to be the rook polynomials that we are looking for. (The coefficients of these polynomials are described by OEIS sequence A011973 [10].)

Definition 5.5.6. *For $j \geq 0$, the j-th **Fibonacci polynomial** $F_j(x)$ is defined recursively as:*

$$F_0(x) = 1 \tag{5.18}$$

$$F_1(x) = 1 + x \tag{5.19}$$

$$F_n(x) = xF_{n-2}(x) + F_{n-1}(x). \tag{5.20}$$

The rook polynomials of the staircase-shaped boards agree with these Fibonacci polynomials.

Lemma 5.5.7. *If B is a staircase-shaped board with k squares, then B has rook polynomial $p_B(x) = F_k(x)$, equal to the k-th Fibonacci polynomial.*

Proof. We will recall the recursive construction of rook polynomials from Lemma 5.3.6, and proceed by induction on the number of squares, always choosing to include or exclude the upper-left square.

Since the reflections of board have the same rook polynomial as the unreflected board, without loss of generality, we will compute the rook polynomials for $\mathscr{O}_{2m-1}$ and $\mathscr{E}_{2m-2}$, 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 rook on $\mathscr{O}_1$, and no way to place more than one rook. Thus

$$p_{\mathscr{E}_0}(x) = 1 \qquad = F_0(x), \text{ and} \tag{5.21}$$

$$p_{\mathscr{O}_1}(x) = 1 + x = F_1(x). \tag{5.22}$$

With the base case established, our inductive hypothesis is that $p_B(x) = F_h(x)$ whenever B is a staircase-shaped board 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 5.3.6, the rook polynomial of $\mathscr{E}_k$ is

$$p_{\mathscr{E}_k}(x) = x p_{\mathscr{E}_{k-2}}(x) + p_{\mathscr{O}_{k-1}}(x) \tag{5.23}$$

$$= x F_{k-2}(x) + F_{k-1}(x) \tag{5.24}$$

$$= F_k(x). \tag{5.25}$$

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 5.3.6, the rook polynomial of $\mathscr{O}_k$ is

$$p_{\mathscr{O}_k}(x) = x p_{\mathscr{O}_{k-2}}(x) + p_{\mathscr{E}_{k-1}}(x) \tag{5.26}$$

$$= x F_{k-2}(x) + F_{k-1}(x) \tag{5.27}$$

$$= F_k(x). \tag{5.28}$$

□

Therefore, we now have the ingredients to describe the rook polynomial of a derived complementary board.

Corollary 5.5.8. *Suppose that B^c_α can be partitioned into m disjoint staircase-shaped sub-boards of sizes $b_1, b_2, \ldots, b_m$. Then the rook polynomial of B^c_α is*

$$p_{B^c_\alpha}(x) = \prod_{i=1}^{m} F_{b_i}, \tag{5.29}$$

where F_j is the j-th Fibonacci polynomial.

Proof. This follows directly from Theorem 5.5.3 together with Lemma 5.5.7. □

5.5.5 Sub-boards from prefix

In this part, we discuss how to algorithmically compute the size of the sub-boards of the partition of the derived complementary board B^c_α for a given prefix α.

Lemma 5.5.9. *Given a nonempty prefix $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ and $i \notin \alpha$, the number of squares of B^c in column i that do not have a first coordinate in $[\ell]$ is given by the rule:*

$$c_i = \begin{cases} 0 & i < \ell \\ 1 & i = \ell \text{ or } i = n \\ 2 & \ell < i < n \end{cases} \tag{5.30}$$

Proof. It is helpful to recall that the complementary board B^c consists of squares on the diagonal, squares on the subdiagonal, and the square $(1,n)$:

$$B^c = \{(i,i) : i \in [n]\} \cup \{(i+1,i) : i \in [n-1]\} \cup \{(1,n)\}. \tag{5.31}$$

Now if $i < \ell$, then (i,i) and $(i+1,i)$ both have a first coordinate less than or equal to ℓ.

If $i = \ell$, then (i,i) has a first coordinate in $[\ell]$, but $(i+1,i) = (\ell+1,\ell)$ does not have its first coordinate in $[\ell]$.

If $i = n$, there are two squares of B^c in column i: (n,n) and $(1,n)$. Only $(1,n)$ has its first coordinate in $[\ell]$.

If $\ell < i < n$, then neither the square (i,i) nor $(i+1,i)$ has its first coordinate in $[\ell]$. □

Now we will go through each contiguous section of columns, and count the number of squares in each to build up the size of each of the blocks.

Lemma 5.5.10. *Partition $[n] \setminus \alpha$ into contiguous parts, $\mathcal{P}$. Each part $P_i \in \mathcal{P}$ of the partition corresponds to a staircase-shaped sub-board of size $\sum_{p \in P_i} c_p$.*

Therefore the size of the disjoint sub-boards in the derived complementary board B_α^c is given by the multiset

$$\mathcal{P}_\alpha = \left\{ \sum_{p \in P_i} c_p : P_i \in \mathcal{P} \right\}. \tag{5.32}$$

Proof. Once the first row of a complementary ménage board has been deleted, the resulting board has the property that any two nonadjacent columns do not have any squares in the same row, because column i has squares in (i,i) and $(i+1,i)$.

Within each contiguous interval between the letters of α, the columns form a staircase-shaped sub-board because each column with a square in position $(i+1,i)$ has a square to its right, in position $(i+1,i+1)$ whenever $i+1 \notin \alpha$. □

Example 5.5.11. *As illustrated in Figure 5.3, if $n = 12$ and $\alpha = (3,6,1,8)$, then the contiguous partition of*

$$[12] \setminus \{3,6,1,8\} = \{\underbrace{2}_{P_1}, \underbrace{4,5}_{P_2}, \underbrace{7}_{P_3}, \underbrace{9,10,11,12}_{P_4}\} \tag{5.33}$$

is $\{P_1, P_2, P_3, P_4\}$. The corresponding staircase-shaped sub-boards have sizes

$$\begin{aligned} k_1 &= c_2 && = 0 && = 0 \\ k_2 &= c_4 + c_5 && = 1+2 && = 3 \\ k_3 &= c_7 && = 2 && = 2 \\ k_4 &= c_9 + c_{10} + c_{11} + c_{12} && = 2+2+2+1 && = 7, \end{aligned}$$

which matches what we observe in the illustration:

$$B_\alpha^c = \mathscr{E}_0 \sqcup \mathscr{O}_3 \sqcup \mathscr{E}_2 \sqcup \mathscr{O}_7^\intercal, \tag{5.34}$$

5.5.6 Complementary polynomials

We have now established a method taking a prefix α and partitioning B_α^c into disjoint staircase-shaped sub-boards, which allow us to determine the rook polynomial of B_α^c. Using Lemma 5.4.5, this allows us to finally compute the number of ways of placing $n-\ell$ rooks on B_α, thus determining the number of derangements that begin with α.

Theorem 5.5.12. *The number of ménage permutations that begin with a valid, nonempty prefix α is*

$$\#\text{prefix}_{\mathcal{M}}(\alpha) = \sum_{k=0}^{n} (-1)^k r_k^c (n-k)! \tag{5.35}$$

where $\sum_{k=0}^{n} r_k^c x^k = \prod_{p\in\mathcal{P}_\alpha} F_p$, F_k is the k-th Fibonacci polynomial, and $\mathcal{P}_\alpha$ is the multiset corresponding to the size of the staircase-shaped sub-boards in the disjoint partition of B_α^c.

Proof. This follows directly from Corollary 5.5.8 together with Lemma 5.5.10 □

Now that we have computed the number of ménage permutations, Theorem 5.2.4 provides an efficient unranking algorithm for $\mathcal{M}$.

We will illustrate this with a specific example computing the number of ménage permutations with a given prefix.

Example 5.5.13. *We will continue with the running example illustrated in Figure 5.3 and expounded on in Example 5.5.11.*

We've already seen that the for $n = 12$, the prefix $\alpha = (3,6,1,8)$ partitions the derived complementary board into three nonempty sub-boards:

$$B_\alpha^c = \mathscr{E}_0 \sqcup \mathscr{O}_3 \sqcup \mathscr{E}_2 \sqcup \mathscr{O}_7^{\mathsf{T}}. \tag{5.36}$$

Lemma 5.5.7 tells us that the rook polynomial of B_α^c is

$$\begin{aligned} p_{B_\alpha^c}(x) &= F_3(x)F_2(x)F_7(x) && (5.37)\\ &= (1+3x+x^2)(1+2x)(1+7x+15x^2+10x^3+x^4) && (5.38)\\ &= 1+12x+57x^2+136x^3+170x^4+105x^5+27x^6+2x^7 && (5.39)\\ &= \sum_{k=0}^{7} r_k^c x^k. && (5.40) \end{aligned}$$

By Lemma 5.4.5, the number of ways to place eight rooks on B_α is is

$$\begin{aligned} N_0 &= \sum_{k=0}^{7} (-1)^k r_k^c (8-k)! && (5.41)\\ &= 1(8!) - 12(7!) + 57(6!) - 136(5!) + 170(4!) - 105(3!) + 27(2!) - 2(1!) && (5.42)\\ &= 8062. && (5.43) \end{aligned}$$

Therefore there are 8062 *ménage permutations in S_{12} that start with the prefix* $(3,6,1,8)$.

We can now repeatedly use the above counting technique in conjunction with Theorem 5.2.4 to unrank derangements.

Example 5.5.14. *There are $A000179(8) = 4738$ ménage permutations on* 8 *letters. Table 5.3 shows the steps of the algorithm that determines that the* 1000*th ménage permutation in lexicographic order is*

$$w_{1000} = 3\ 5\ 4\ 8\ 2\ 7\ 1\ 6. \tag{5.44}$$

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

Table 5.3: The recursive computation of the 1000th ménage permutation.

5.6 Generalizations and open questions

In this final section we explore several possible future directions for applying these ideas in new contexts. We can potentially apply these unranking techniques to position-restricted permutations, permutations that satisfy certain inequalities with respect to a permutation statistic, or words that avoid or match certain patterns.

5.6.1 Other restricted permutations

In a 2014 paper about finding linear recurrences for derangements, ménage permutations and other restricted permutations, Doron Zeilberger introduces a more general family of restricted permutations.

Definition 5.6.1 ([30])**.** *Let $S \subset \mathbb{Z}$ be a finite collection of integers. An S-**avoiding permutation** is a permutation $\pi \in S_n$ such that*

$$\pi(i) - i - s \not\equiv 0 \bmod n \quad \textit{for all} \quad i \in [n] \quad \textit{and} \quad s \in S. \tag{5.45}$$

Example 5.6.2. *In terms of S-avoiding permutations,*

- *ordinary permutations are $\emptyset$-avoiding permutations,*
- *derangements are $\{0\}$-avoiding permutations, and*
- *ménage permutations are $\{-1,0\}$-avoiding permutations.*

The results in the previous sections straightforwardly adapt to the cases of unranking $\{i\}$-avoiding and $\{i, i+1\}$ avoiding permutations.

Open Question 5.6.3. *For arbitrary finite subsets $S \subset \mathbb{Z}$, do there exist efficient unranking algorithms on S-avoiding permutations?*

The techniques used to unrank derangements and ménage permutations do not appear to generalize even to superficially similar domains. So, in the spirit of Richard Arratia's bounties, it is only fair to offer one of my own.

$100 Problem. *Do there exist efficient unranking algorithms on* $\{-1,1\}$*-avoiding permutations?*

The main obstruction to using the techniques from Section 5.5 to resolve this question is that placing a rook and deleting a column does not necessarily cause the left and right sides of that column to be disjoint. As such, unranking $\{-1,1\}$-avoiding permutations appears to require a genuinely novel insight.

5.6.2 Permutation statistics

Another area for exploration, inspired by Chapter 3, is unranking permutations with a given permutation statistic.

Open Question 5.6.4. *Let* $\mathrm{inv}\colon S_n \to \mathbb{N}_{\geq 0}$ *be the map that counts inversions of a permutation. Since* inv *is a Mahonian statistic, the generating function for the number of permutations* $\pi \in S_n$ *such that* $\mathrm{inv}(\pi) = k$ *is given by the q analog of* $n!$*,* $n!_q$*.*

Does there exist an efficient unranking function on the set

$$\mathscr{I}_n^k = \{\pi \in S_n | \, \mathrm{inv}(\pi) = k\}, \tag{5.46}$$

and if so, how does one construct it?

We can, of course, substitute inv with any other permutation statistic of interest.

5.6.3 Pattern avoidance

In the field of combinatorics on words, there exists a notion of patterns and instances of a pattern. At this level of informality, this is probably best illustrated with an example (with undefined words in bold).

Example 5.6.5. *The word* 1100010110 *is an* ***instance*** *of the* ***pattern*** *ABA with* $A = 110$ *and* $B =$ 0010*. The word* 32123213212 *is said to* ***match*** *the pattern CC with* $C = 321$ *because it contains a substring of the form* 321321*.*

Open Question 5.6.6. *Given a pattern P, is it possible to unrank words of length n over an alphabet* $\mathscr{A}$ *that are not instances of the pattern P? That match the pattern P? That don't match the pattern P?*

5.6.4 Prefixes of Lyndon words

There are other collections of finite words that might be amenable to some of the above techniques. In particular, Kociumaka, Radoszewski, and Rytter [31] give polynomial time algorithms for unranking Lyndon words. We have some conjectures about prefixes of Lyndon words and open questions about other restricted words.

Definition 5.6.7. *A* ***Lyndon word*** *is a string over an alphabet of letters that is the unique minimum with respect to all of its rotations.*

Example 5.6.8. 00101 *is a Lyndon word because*

$$00101 = \min\{00101, 01010, 10100, 01001, 10010\} \tag{5.47}$$

is the unique minimum of all of its rotations.

011011 *is not a Lyndon word because while*

$$011011 = \min\{011011, 110110, 101101, 011011, 110110, 101101\}, \tag{5.48}$$

it is not the ***unique*** *minimum. (That is, rotating it three positions returns it to itself.)*

Definition 5.6.9 ([32])**.** *Suppose that* $\{a_i\}_{i=1}^{\infty}$ *and* $\{b_i\}_{i=1}^{\infty}$ *are integer sequences related by*

$$1 + \sum_{n=1}^{\infty} b_n x^n = \prod_{i=1}^{\infty} \frac{1}{(1-x^i)^{a_i}}. \tag{5.49}$$

Then $\{b_i\}_{i=1}^{\infty}$ *is said to be the* ***Euler transform*** *of* $\{a_i\}_{i=1}^{\infty}$*, denoted* $\mathcal{E}(\{a_i\}_{i=1}^{\infty}) = \{b_i\}_{i=1}^{\infty}$.

Definition 5.6.10. *Let* $\mathscr{L}_\alpha = \{\ell_n^\alpha\}_{n=1}^{\infty}$ *where* ℓ_n^α *is the number of Lyndon words with prefix* α *and length n over the alphabet* $\{0,1\}$.

Conjecture 5.6.11. *The Euler transform of the number of Lyndon words with prefix* α *and length n over the alphabet* $\{0,1\}$*,* $\mathcal{E}(\mathscr{L}_\alpha) = \{t_n^\alpha\}_{n=1}^{\infty}$*, follows a linear recurrence for all* $n \geq N_\alpha$.

This conjecture and the following conjectures are based on the data in Table 5.4.

We start with two specific conjectures about two families of prefixes.

Conjecture 5.6.12. *For* $k \geq 1$*, let* $\mathscr{L}_\alpha$ *be the sequence of the number of Lyndon words of length n with prefix* $\alpha = (0,0,\ldots,0)$ *of length k over the alphabet* $\{0,1\}$*. Then the Euler transform* $\mathcal{E}(\mathscr{L}_\alpha) = \{t_n^\alpha\}_{n=1}^{\infty}$ *follows the linear recurrence* $t_{n+1}^\alpha = 2t_n^\alpha$ *for all* $n \geq k+2$.

Conjecture 5.6.13. *For* $k \geq 2$ *let* $\mathscr{L}_\alpha$ *be the sequence of the number of Lyndon words of length n with prefix* $\alpha = (1,0,0,\ldots,0)$ *of length k over the alphabet* $\{0,1\}$*. Then the Euler transform* $\mathcal{E}(\mathscr{L}_\alpha) = \{t_n^\alpha\}_{n=1}^{\infty}$ *follows the linear recurrence* $t_{n+k}^\alpha = \sum_{i=0}^{k-1} t_{n+i}^\alpha$ *for all* $n \geq 1$.

And more broadly, we have a conjecture in the case that the prefix is not the zero sequence.

Conjecture 5.6.14. *Let* $\mathscr{L}_\alpha$ *be the sequence of the number of Lyndon words of length n with prefix* α *over the alphabet* $\{0,1\}$ *such that* α *contains at least one* 1*. Then the Euler transform of the sequence,* $\mathcal{E}(\mathscr{L}_\alpha) = \{t_n^\alpha\}_{n=1}^{\infty}$*, follows the linear recurrence where all terms have coefficients of* 0 *or* 1.

Open Question 5.6.15. *If, as the evidence suggests,* $\mathcal{E}(\mathscr{L}_\alpha)$ *follows a linear recurrence, what is the length of the recurrence and what are the coefficients of the recurrence as a function of* α*?*

α	$\mathscr{L}_\alpha$ and $\mathcal{E}(\mathscr{L}_\alpha)$												Conjectured recurrence	
0	1	1	2	3	6	9	18	30	56	99	186	335	$a_{n+1} = 2a_n$	$n \geq 2$
	1	1	2	4	8	16	32	64	128	256	512	1024		
00	0	0	1	2	4	7	14	25	48	88	168	310	$a_{n+1} = 2a_n$	$n \geq 4$
	1	0	0	1	2	4	8	16	32	64	128	256		
01	0	1	1	1	2	2	4	5	8	11	18	25	$a_{n+2} = a_{n+1} + a_n$	$n \geq 1$
	1	0	1	1	2	3	5	8	13	21	34	55		
000	0	0	0	1	2	4	8	15	30	57	112	214	$a_{n+1} = 2a_n$	$n \geq 5$
	1	0	0	0	1	2	4	8	16	32	64	128		
001	0	0	1	1	2	3	6	10	18	31	56	96	$a_{n+3} = a_{n+2} + a_{n+1} + a_n$	$n \geq 1$
	1	0	0	1	1	2	4	7	13	24	44	81		
010	0	0	0	0	1	1	2	3	5	7	12	18	$a_{n+2} = a_{n+1} + a_n$	$n \geq 5$
	1	0	0	0	0	1	1	2	3	5	8	13		
011	0	0	1	1	1	1	2	2	3	4	6	7	$a_{n+3} = a_{n+2} + a_n$	$n \geq 1$
	1	0	0	1	1	1	2	3	4	6	9	13		
0000	0	0	0	0	1	2	4	8	16	31	62	121	$a_{n+1} = 2a_n$	$n \geq 6$
	1	0	0	0	0	1	2	4	8	16	32	64		
0001	0	0	0	1	1	2	4	7	14	26	50	93	$a_{n+4} = a_{n+3} + a_{n+2} + a_{n+1} + a_n$	$n \geq 1$
	1	0	0	0	1	1	2	4	8	15	29	56		
0010	0	0	0	0	1	1	3	5	9	16	30	53	$a_{n+3} = a_{n+2} + a_{n+1} + a_n$	$n \geq 6$
	1	0	0	0	0	1	1	3	5	9	17	31		
0011	0	0	0	1	1	2	3	5	9	15	26	43	$a_{n+4} = a_{n+3} + a_{n+2} + a_n$	$n \geq 1$
	1	0	0	0	1	1	2	3	6	10	18	31		
0101	0	0	0	0	1	1	2	3	5	7	12	18	$a_{n+2} = a_{n+1} + a_n$	$n \geq 5$
	1	0	0	0	0	1	1	2	3	5	8	13		
0110	0	0	0	0	0	0	1	1	1	2	3	4	$a_{n+3} = a_{n+2} + a_n$	$n \geq 6$
	1	0	0	0	0	0	0	1	1	1	2	3		
0111	0	0	0	1	1	1	1	1	2	2	3	3	$a_{n+4} = a_{n+3} + a_n$	$n \geq 1$
	1	0	0	0	1	1	1	1	2	3	4	5		

Table 5.4: A table of conjectures about the number of Lyndon words of length n and prefix α. Each row contains a prefix α, a sequence counting Lyndon words with α ($\mathscr{L}_\alpha$), the Euler transformation of that sequence ($\mathcal{E}(\mathscr{L}_\alpha)$), a conjectured recurrence for the Euler-transformed sequence, and the valid range for the conjectured recurrence.

References

1. Winkler, P. *Mathematical Puzzles: A Connoisseur's Collection* (AK Peters, Natick, Mass, 2004).
2. Gardner, M. Mathematical Games. *Scientific American* **240,** 16–27 (1979).
3. Gardner, M. Mathematical Games. *Scientific American* **240,** 21–31 (1979).
4. Yehuda, R. B., Etzion, T. & Moran, S. Rotating-Table Games and Derivatives of Words. *Theor. Comput. Sci.* **108,** 311–329 (1993).
5. Ehrenborg, R. & Skinner, C. M. The Blind Bartender's Problem. *Journal of Combinatorial Theory, Series A* **70,** 249–266. doi:https://doi.org/10.1016/0097-3165(95)90092-6 (1995).
6. Roeder, O. The Riddler. *FiveThirtyEight* (2019).
7. Sidana, T. *Constacyclic codes over finite commutative chain rings* PhD thesis (Indraprastha Institute of Information Technology, Delhi, 2020).
8. Rabinovich, Y. A generalization of the Blind Rotating Table game. *Information Processing Letters* **176,** 106233. doi:https://doi.org/10.1016/j.ipl.2021.106233 (2022).
9. Rotman, J. J. *An Introduction to the Theory of Groups* (Springer, New York, 1999).
10. OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences* 2021.
11. Mazurov, V. D. On Generation of Sporadic Simple Groups by Three Involutions Two of Which Commute. *Siberian Mathematical Journal* **44,** 160–164 (2003).
12. Nuzhin, Y. N. Generating triples of involutions of alternating groups. *Mathematical Notes* **51,** 389–392. doi:10.1007/BF01250552 (1992).
13. Nuzhin, Y. N. Generating triples of involutions of Chevalley groups over a finite field of characteristic 2. *Algebra i Logika* **29,** 192–206, 261. doi:10.1007/BF02001358 (1990).
14. Nuzhin, Y. N. Generating triples of involutions of Lie-type groups over a finite field of odd characteristic. I. *Algebra i Logika* **36,** 77–96, 118. doi:10.1007/BF02671953 (1997).
15. Nuzhin, Y. N. Generating triples of involutions of Lie-type groups over a finite field of odd characteristic. II. *Algebra i Logika* **36,** 422–440, 479 (1997).
16. Willse, T. *The unique loop (quasigroup with unit) L of order 5 satisfying $x^2 = 1$ for all $x \in L$* Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/1216907 (version: 2018-12-27).
17. Winkler, P. *Mathematical Puzzles* (CRC Press, 2021).
18. Conger, M. A refinement of the Eulerian numbers, and the joint distribution of $\pi(1)$ and $\mathrm{Des}(\pi)$ in S_n. *Ars Combinatoria* **95** (2010).
19. Goncharov, V. Du domaine de l'analyse combinatoire. *Izv. Akad. Nauk SSSR Ser. Mat.* **8,** 3–48 (1 1944).
20. Arratia, R. & Tavare, S. The Cycle Structure of Random Permutations. *The Annals of Probability* **20,** 1567–1591 (1992).

21. Assaf, S. H. Cyclic Derangements. *The Electronic Journal of Combinatorics* **17** (2010).
22. Rubey, M., Stump, C., *et al. FindStat - The combinatorial statistics database* Accessed: June 16, 2022.
23. Foata, D. *Distributions Euleriennes et Mahoniennes sur le Groupe des Permutations* in *Higher Combinatorics* (ed Aigner, M.) (Springer Netherlands, Dordrecht, 1977), 27–49.
24. Kaplansky, I. & Riordan, J. The problem of the rooks and its applications. *Duke Mathematical Journal* **13,** 259–268. doi:10.1215/S0012-7094-46-01324-5 (1946).
25. Shevelev, V. S. Some problems of the theory of enumerating the permutations with restricted positions. *Journal of Soviet Mathematics* **61,** 2272–2317. doi:10.1007/BF01104103 (1992).
26. Valiant, L. The complexity of computing the permanent. *Theoretical Computer Science* **8,** 189–201. doi:https://doi.org/10.1016/0304-3975(79)90044-6 (1979).
27. Riordan, J. *An Introduction to Combinatorial Analysis* (Princeton University Press, USA, 1980).
28. Mikawa, K. & Tanaka, K. Lexicographic ranking and unranking of derangements in cycle notation. *Discret. Appl. Math.* **166,** 164–169 (2014).
29. Stanley, R. P. *Enumerative Combinatorics: Volume 1* 2nd (Cambridge University Press, USA, 2011).
30. Zeilberger, D. Automatic Enumeration of Generalized Ménage Numbers. *Séminaire Lotharingien de Combinatoire* **71** (2014).
31. Kociumaka, T., Radoszewski, J. & Rytter, W. *Computing k-th Lyndon Word and Decoding Lexicographically Minimal de Bruijn Sequence* in *Combinatorial Pattern Matching* (Springer International Publishing, 2014), 202–211.
32. Sloane, N. J. A. & Plouffe, S. *Encyclopedia of Integer Sequences* (Academic Press, San Diego, 1995).

Appendices

A An algorithm for unranking ménage permutations

In this section, we will provide Haskell implementations of the functions in Chapter 5 that were used to compute the permutations requested by Richard Arratia's bounties. Each code sample will reference its precedent in Chapter 5. We start by importing some relevant libraries and establishing some naming conventions.

```
import Data.List (sort, nub, intersect)

type Prefix = [Int]
type Coefficients = [Integer]
type PrefixCount = Prefix -> Integer
```

A.1 Unranking from prefix counts

We begin with the function `unrank`, which implements $\text{unrank}_{\mathscr{C}}(i)$, as shown in Theorem 5.2.4. The internal function `recurse` corresponds to the recursive function $f_i^{\mathscr{C}}$.

```
unrank :: Int ->                 -- Alphabet of n letters
          Int ->                 -- Words of length k
          (Prefix -> Integer) -> -- #prefix function
          Integer ->             -- Unrank at i
          Prefix                 -- Word at rank i
unrank n k prefixCounter i = recurse (0, 0) 1 [] where
  recurse :: (Integer, Integer) -> -- index range for prefix (a, b]
             Int ->                -- candidate for current letter
```

```
            Prefix ->             -- established prefix
            Prefix                -- word at index
  recurse (a, b) c prefix
    | c > n              = error "Out of range!"
    | length prefix == k = prefix
    | a < i && i <= b    = recurse (a, b') 1 (prefix  ++ [c])
    | otherwise          = recurse (b, b'') (c + 1) prefix where
    b'      = a + prefixCounter (prefix ++ [c, 1])
    b''     = b + prefixCounter (prefix  ++ [c + 1])
```

A.2 Counting derangements with prefixes

This part of the code contains three functions. Each of them takes a parameter n, which indicates the number of letters in the derangement. The first function, `invalidDerangementPrefix`, also takes a prefix α and checks to see if it includes duplicate values or has letters in forbidden positions. The function `derangementPrefixCount` is equivalent to the function $\#\text{prefix}_{\mathscr{D}}$, as described in Corollary 5.4.6. The function `unrankDerangement` is equivalent to the function $\text{unrank}_{\mathscr{D}}$ as described in Theorem 5.2.4.

```
invalidDerangementPrefix :: Int -> Prefix -> Bool
invalidDerangementPrefix _ prefix = containsDupes || invalidPosition where
  containsDupes = prefix /= (nub prefix)
  invalidPosition = any (id) $ zipWith (==) [1..] prefix

derangementPrefixCount :: Int -> Prefix -> Integer
derangementPrefixCount n prefix
  | invalidDerangementPrefix n prefix = 0
  | otherwise = sum $ map (\j -> term j) [0..squareCount] where
    k = length prefix
    squareCount = fromIntegral $ n - k - length (intersect [k+1..n] prefix)
    n' = fromIntegral n
    k' = fromIntegral k
    term j = (-1)^j * binomial squareCount j * factorial (n' - k' - j)

unrankDerangement :: Int -> Integer -> Prefix
unrankDerangement n = unrank n n (derangementPrefixCount n)
```

A.3 Decomposition of derived complementary boards

These next two functions decompose a derived complementary board into staircase-shaped blocks. The first function, `getColGroups`, corresponds to the partition into contiguous parts referred to in Lemma 5.5.10. The second function, `blockSizesFromPrefix` goes on to implement the sum in this lemma.

```
getColGroups :: Int -> Prefix -> [[Int]]
getColGroups n prefix = filter (not . null) $ colGroups where
  cols = 0 : (sort prefix) ++ [n+1]
  colGroups = zipWith (\a b ->  [a+1..b-1]) cols (tail cols)

blockSizesFromPrefix :: Int -> Prefix -> [Int]
blockSizesFromPrefix n prefix = map (sum . map colCells) colGroups where
  colGroups = getColGroups n prefix
  k = length prefix
  colCells c
    | c < k      = 0
    | c == k     = 1
    | c == n     = 1
    | otherwise  = 2
```

A.4 Complementary polynomials

Since the functions above compute the size of the staircase-shaped sub-boards, they can hand these values to `fibonacciPoly`, which implements Fibonacci polynomials as described in Definition 5.5.6. These match the rook polynomials of the disjoint staircase-shaped sub-boards, so in `complementaryRookPolynomial`, the appropriate polynomials are multiplied together and the coefficients are extracted to implement Lemma 5.4.5.

```
fibonacciPoly :: Int -> Coefficients
fibonacciPoly = (!!) fibonacciPolynomials where
  fibonacciPolynomials = [1] : [1] : recurse [1] [1] where
    recurse f g = h : recurse g h where
      h = ([0,1] .*. f) .+. g
```

```
complementaryRookPolynomial :: Int -> Prefix -> Coefficients
complementaryRookPolynomial n prefix = foldr (.*.) [1] blockPolys where
  blockPolys = map (\i -> fibonacciPoly (i + 1)) blockSizes where
    blockSizes = blockSizesFromPrefix n prefix
```

A.5 Putting it together

This part of the code contains three functions. Each takes a parameter n, which indicates the number of letters in the ménage permutation. The first function, `invalidMenagePrefix`, also takes a prefix α and checks to see if it includes duplicate values or has letters in forbidden positions. The function `menagePrefixCount` is equivalent to the function $\#\text{prefix}_{\mathcal{M}}$, as described in Corollary 5.4.6. The function `unrankDerangement` is equivalent to the function $\text{unrank}_{\mathcal{M}}$ as described in Theorem 5.2.4.

```
invalidMenagePrefix :: Int -> Prefix -> Bool
invalidMenagePrefix n prefix = containsDuplicates || invalidPosition where
  containsDuplicates = prefix /= (nub prefix)
  invalidPosition = any inRestrictedPosition $ zip [0..] prefix where
    inRestrictedPosition (i, x) = (x `mod` n == i) || (x == i + 1)

menagePrefixCount :: Int -> Prefix -> Integer
menagePrefixCount n prefix
  | invalidMenagePrefix n prefix = 0
  | otherwise      = recurse 0 crp where
  n' = fromIntegral (n - length prefix)
  crp = complementaryRookPolynomial n prefix
  recurse k (c:cs) = (-1)^k * c * factorial (n'-k) + recurse (k+1) cs
  recurse _ [] = 0

unrankMenage :: Int -> Integer -> Prefix
unrankMenage n = unrank n n (menagePrefixCount n)
```

A.6 Miscellanea

These are helper functions that need to be included in order for the code to compile. The first two functions add and multiply list representations of polynomials, and the third and fourth function implement the factorial function and the binomial coefficient.

```
-- The polynomial a + bx + cx^2 ... is represented as
-- [a, b, c, ...]
-- These are helper functions for adding and multiplying polynomials
(.+.) :: Coefficients -> Coefficients -> Coefficients
(.+.) p1 [] = p1
(.+.) [] p2 = p2
(.+.) (a:p1) (b:p2) = (a + b) : (p1 .+. p2)

(.*.) :: Coefficients -> Coefficients -> Coefficients
(.*.) p1 [] = []
(.*.) [] p2 = []
(.*.) p1 p2 = foldr1 (.+.) termwiseProduct where
  termwiseProduct = map f $ zip [0..] p1 where
    f (i, x) = replicate i 0 ++ map (*x) p2

factorial :: Integer -> Integer
factorial n = factorial' n 1 where
  factorial' 0 accum = accum
  factorial' n accum = factorial' (n - 1) (accum * n)

binomial :: Integer -> Integer -> Integer
binomial _ 0 = 1
binomial 0 _ = 0
binomial n k
  | n < k'    = 0
  | otherwise = product [k' + 1..n] `div` factorial (n - k') where
    k' = max k (n - k)
```

B Unranking combinatorial objects

In this appendix, we provide some brief examples of prefix-counting functions, that together with Theorem 5.2.4, give a method for unranking combinatorial objects.

B.1 Compositions

Definition B.1. *A **composition** of $n \in \mathbb{N}_{>0}$ is a sequence of strictly positive integers summing to n. We denote the set of compositions of n by $\mathcal{C}_n$.*

Lemma B.2 ([29])**.** *The number of compositions of n is $\#\mathcal{C}_n = 2^{n-1}$.*

Proposition B.3. *If α is a sequence of positive integers such that $\mathrm{sum}(\alpha) \leq n$, then the number of compositions of n that start with the prefix α is*

$$\#\mathrm{prefix}_{\mathcal{C}_n}(\alpha) = \begin{cases} 1 & \alpha \in \mathcal{C}_n \\ 2^{n-\mathrm{sum}(\alpha)-1} & \textit{otherwise}. \end{cases} \tag{B.1}$$

Definition B.4. *For $k > 0$, a **k-composition** of $n \in \mathbb{N}_{>0}$ is a sequence of length k consisting of strictly positive integers that sum to n. We denote the set of k-compositions of n by $\mathcal{C}_n^k$.*

Lemma B.5 ([29])**.** *The number of k-compositions of n is $\#\mathcal{C}_n^k = \binom{n-1}{k-1}$.*

Proposition B.6. *If $\alpha \in \mathcal{C}_n^k$ or if α is a sequence of positive integers of length $\ell < k$ such that $\mathrm{sum}(\alpha) < n$, the number of k-compositions of n that start with the prefix α is*

$$\#\mathrm{prefix}_{\mathcal{C}_n}(\alpha) = \begin{cases} 1 & \alpha \in \mathcal{C}_n^k \\ \binom{n-\mathrm{sum}(\alpha)-1}{k-\ell-1} & \textit{otherwise}. \end{cases} \tag{B.2}$$

B.2 Partitions

Definition B.7. *A **partition** of $n \in \mathbb{N}_{>0}$ is a weakly decreasing sequence of strictly positive integers summing to n. We denote the set of partitions of n by $\mathcal{P}_n$.*

The set of partitions of n with largest part at most k is denoted $\mathcal{P}_n^{\leq k}$.

Lemma B.8 ([29])**.** *The ordinary generating function for the number of partitions of n is*

$$\sum_{n=0}^{\infty} \#\mathcal{P}_n q^n = \prod_{j=1}^{\infty} \frac{1}{1-q^j}, \tag{B.3}$$

and the ordinary generating function for the number of partitions of n with largest part as most k is

$$\sum_{n=0}^{\infty} \#\mathcal{P}_n^{\leq j} q^n = \prod_{j=1}^{k} \frac{1}{1-q^j}. \tag{B.4}$$

Proposition B.9. *If either $\alpha \in \mathcal{P}_n$ or α is a decreasing sequence of positive integers such that $\mathrm{sum}(\alpha) < n$ or then the number of partitions of n that start with the prefix α is*

$$\#\mathrm{prefix}_{\mathcal{P}_n}(\alpha) = \begin{cases} 1 & \alpha \in \mathcal{C}_n^k \\ \#\mathcal{P}_{n-\mathrm{sum}(\alpha)}^{\leq \min(\alpha)} & \textit{otherwise}. \end{cases} \tag{B.5}$$