COMBINATORICS OF PARKING FUNCTIONS

An Honors Thesis Presented

By

CHRISTO M KELLER

Approved as to style and content by:

**** Laura Colmenarejo Hernando 05/10/21 17:10 ****
Chair

**** Alejandro Henry Morales Borrero 05/11/21 12:52 ****
Committee Member

**** Sohrab M Shahshahani 05/11/21 12:57 ****
Honors Program Director

ABSTRACT

We study parking functions and their generalizations, particularly the k-Naples parking functions of Christensen et al (2020). We begin with a survey of the classical parking functions of Konheim and Weiss (1966) and their basic properties, before expanding into related areas of combinatorics. Then we discuss recent work about the k-Naples parking functions, focusing on our own results in Colmenarejo et al (2021). In the last sections we generalize a result of Stanley, describe a new algorithm for recursively generating the k-Naples parking functions, study the rearrangements of $k = n - 2$ Naples parking functions, and discuss our current work on the characteristic polynomials of parking matrices.

Epigraph

Mantua me genuit, Calabri rapuere, ***tenet nunc Parthenope****; cecini pascua rura duces.*

Explanation

This quote is taken from the epitaph on the tomb of Publius Vergilius Maro, the classical Roman author of the *Eclogues*, the *Georgics*, and most famously the *Aeneid.* Born in 70 BC in Mantua, in 19 BC Virgil died at Brindisi (then Calabri) and was buried in Naples (then Parthenope).

The epigraph translates to *Mantua bore me, Brindisi took me,* ***now Naples holds me****; I sang of pastures, countryside, and rulers.* This pun reflects the development of my interest in the k-Naples parking functions and although my work is not as polished as Virgil's poetry, the moral of his life is that there are always improvements to be made anyways.

According to legend, Virgil commanded on his deathbed that the *Aeneid* be burned, perhaps because he considered it incomplete. But even unfinished, the *Aeneid* is one of the great literary masterpieces of history. Remember this story if you see any typos. **;-)**

Acknowledgements

This work is partly from the 2020 UMass Math REU and the AIM UP program — see in particular [1] and [2]. Many thanks to my collaborators, family, and friends who made this thesis possible.

Contents

Introduction

Parking functions are combinatorial objects corresponding to the ways cars can park under a given rule. They are also more than this due to a series of different characterizations and bijections with other combinatorial objects. They are important insofar as they appear in other subjects such as discrete geometry [3], hyperplane arrangements [4], and diagonal harmonics [5]. They are counted by the number $(n+1)^{n-1}$, which is a special case of Cayley's formula for the number of labelled rooted trees. It is the number of labelled forests on n vertices.

Recently interest has developed in various abstractions of the parking functions [6]. People have begun to study them in their own right as in [7], and begun to draw connections to other topics in combinatorics as shown in [3] and [8]. The Naples parking functions are a generalization of the parking functions allowing a car to parallel park. These were extended to the k-Naples parking functions in [9], where a recursion was given for their number. In [1] we give a closed formula for this number, and generalize a statistic on the parking functions to the k-Naples. Finally, we show that a natural q-analog of our formula is exactly the distribution of our statistic.

Having entered the world of algebraic combinatorics, we prove a new statistical result for the parking functions in this thesis. In [10], Richard Stanley defines permutation-compatible functions and proves a formula for the distribution of their sizes. We extend this result to the parking functions, then turn to questions about the k-Naples generalization.

We present an algorithm recursively generating the decreasing k-Naples parking functions, which have a convenient lattice path representation, and slightly improve our understanding of rearrangements of k-Naples parking functions. While parking functions are freely permutable, things are more complicated for the k-Naples case for which only $k \leq 1$ and $k \geq n-1$ are fully characterized. We give a sufficient condition for $k = n-2$, and use this to define a group action on a subset of the $k = n-2$ Naples parking functions.

We explore what we call parking matrices which we conjecture a formula for the characteristic polynomial of. We are continuing this work with our collaborators, and hope to finish the proof of the conjecture and look for ways to apply it to the combinatorics of parking functions and their generalizations. In particular, we hope to count the parking functions in terms of the unique characteristic polynomials of parking matrices.

Throughout this work we pose mostly unexplored problems related to or extending our results. There is a lot that can be done, and it is all very accessible — hopefully someone will come fill in the details. There are also a series of interesting remarks throughout the exposition, even in the parts explaining classical ideas. We hope these can be useful someday.

The structure of this thesis is basically as appears in this introduction. In Chapter 1, we introduce the parking functions and prove their basic properties. Then we digress into combinatorial interpretations and various applications, particularly the results of [11]. In Chapter 2, we generalize to the k-Naples parking functions and explain the basic results of [9]. We then present our own work in [1] and recent results from [12]. Chapter 3 consists of the new results: the generalization of Stanley's lemma, the description of Zakiya's algorithm, facts about rearrangements of k-Naples parking functions, and the conjecture on parking matrices.

We hope this thesis can inspire some interest in the combinatorics of parking functions, and that the problems it poses are addressed someday. At the very least, we are glad to have recorded somewhere many results that have not been clearly written down. We have learned a lot from the experience of writing this thesis, and would be thrilled for someone to pick up the torch.

Chapter 1

Classical parking functions

1.1 Definitions and basic results

Parking functions were introduced in [13] as a nice solution for computer memory hashing problems. An extensive theory has been built up around them as combinatorial objects, including applications to various other domains of mathematics; and in recent years many generalizations have appeared [6]. We will begin with a summary of basic definitions and results about parking functions, then explain some of the more abstract variants and ideas for future work. Anachronistically, we start at the mathematically pleasant perspective that *parking functions are generalized permutations* and show that to be equivalent to the more popular car-parking analogy of Konheim and Weiss.

A permutation of $[n] = \{1,\ 2,\ \ldots\ n\}$ can be characterized as a sequence $s_1,\ s_2,\ \ldots\ s_n$ the increasing rearrangement $t_1,\ t_2,\ \ldots\ t_n$ of which satisfies the sentence $t_i = i$. Parking functions generalize on the equality sign here, being sequences of elements in $[n]$ the increasing rearrangements of which satisfy the modified sentence $t_i \leq i$. Because the condition on the sequence only relies on its increasing rearrangement, it is clear that permutations of parking functions remain parking functions. The original definition of parking functions is less formal. That it is equivalent to the above characterization can be seen by induction on the number of cars.

Definition 1.1. A *parking preference* p is a vector of natural numbers $p = (p_1, p_2, \ldots, p_n) \in [n]^n$.

Definition 1.2. Imagine a one-way street with n parking spots, and n cars entering the street one by one. Suppose each car has a preferred spot, which it attempts to park in. If the spot is empty, it parks; otherwise, it moves on to the next available spot. When the cars actually do park without driving off the road, the sequence of their parking preferences is a parking function.

Example. The parking functions of length $n = 3$ are (omitting parentheses and commas)

111	112	121	211	113	131	311	122
212	221	123	132	213	231	312	321

One fascinating takeaway from studying parking functions is how these two definitions seem to be suited to different questions. For instance, it is probably not so obvious from the get-go that the cars' order can be freely permuted in the parking-spot analogy. But there are problems like enumerating the parking functions, where the conceptual nature of the analogy comes in handy. There are exactly $(n+1)^{n-1}$ parking functions with n cars, establishing a series of bijections which explain the surprising relevance of parking functions to topics outside of pure combinatorics.

Theorem 1.3. [15] The number of parking functions with n cars $|PF_n| = (n+1)^{n-1}$.

Proof. (Pollack c. 1974) Add an $(n+1)^{\text{th}}$ parking spot, and connect it back to the first in a circle. Now a sequence of length n in $[n+1]$ is a parking function if and only if the cars' attempt to park leaves this new spot open – but no matter what, some spot will be left open. Now shifting the values in the sequence modulo $n+1$ shifts the spot left open modulo $n+1$, so parking functions are the equivalence classes of sequences of length n in $[n+1]$ by the relation "p is a shift of q by some number modulo $n+1$". There are $(n+1)^n$ such sequences, and each equivalence class contains $n+1$ sequences, so that the number of parking functions is $(n+1)^n/(n+1) = (n+1)^{n-1}$ as desired. $\square$

1.2 Combinatorial interpretations

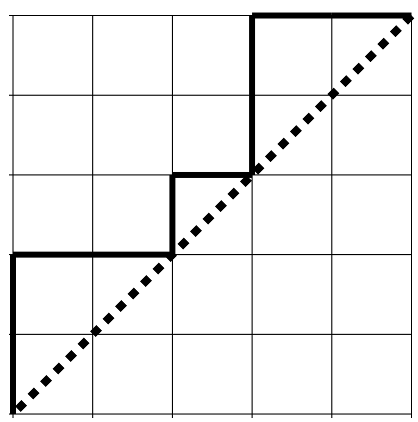
Cayley's formula says that the number of labelled trees on n vertices is n^{n-2}. There is a nice proof due to André Joyal [16] using the notion of *combinatorial species* (see [17] for an exposition). Schützenberger gave an explicit bijection between parking functions of length n and labelled trees on $n+1$ vertices based on the observation that the counting formulas have the same form [18]. As a consequence, we obtain a bijection to forests on n vertices by removing the root of the tree.

Less concretely, in [19], Haglund and Loehr proved that the parking functions count the monomials of the Hilbert series of diagonal harmonics arising from questions in representation theory [5]. We will be mostly concerned with the bijection with what are called labelled Dyck paths.

Definition 1.4. A *Dyck path* (after Walter von Dyck) is a north-east path in the integer lattice $\mathbf{Z}^2$ starting from $(0,0)$ that does not cross the main diagonal $y = x$. (They can touch, however.)

Definition 1.5. The *length* of a Dyck path d is the number of lattice steps in d.

Example. The Dyck path $NNEENENNEE$ has length 10:



Proposition 1.6. The number of Dyck paths of length $2n$ is the n^{th} Catalan number C_n. These numbers count many combinatorial objects, hundreds of which are given in [20].

Proof. The Catalan numbers satisfy the recurrence $C_{n+1} = \sum_{k=0}^{n} C_k C_{n-k}$ with $C_0 = 1$. That the number of Dyck paths satisfies this relation can be seen by considering k as the number of up steps corresponding to the highest point where the number of up steps equals the number of east steps. $\square$

Proposition 1.7. The number of Dyck paths of length $2n$ has closed form $C_n = \frac{1}{n+1}\binom{2n}{n}$.

Proof. The Dyck paths of length $2n$ are clearly in bijection with strings of $2n$ matching parentheses called *Dyck words*, which we count instead. Chasing back the bijection shows that Dyck words satisfy the recurrence from the last paragraph. The rest of this proof uses methods and notation from [21]. Let a *balanced word* be any word consisting of an equal number of left and right brackets (not necessarily matching). A balanced word starts either with a left or right bracket. If with a left bracket, it can be decomposed into the form [c] b, where c is a Dyck word and b is a balanced word. If with a right bracket, it becomes] c^+ [b with c^+ a Dyck word with the brackets switched.

We have shown, then, that the number of Dyck words satisfies the recursive formula

$$B_{n+1} = \sum_{k=0}^{n} B_k C_{n-k} + \sum_{k=0}^{n} B_k C_{n-k} = 2 \sum_{k=0}^{n} B_k C_{n-k},$$

where B_n is the number of balanced words of length $2n$. A balanced string which is not Dyck starts with a Dyck word (possibly empty) then a right bracket, so we can say that

$$B_{n+1} - C_{n+1} = \sum_{k=0}^{n} \binom{2k+1}{k} C_{n-k} = \sum_{k=0}^{n} \frac{2k+1}{k+1} B_k C_{n-k},$$

since $B_n = \binom{2n}{n} = d_n C_n$ pops out of the binomial coefficient. Subtracting these equations,

$$C_{n+1} = 2 \sum_{k=0}^{n} d_k C_k C_{n-k} - \sum_{k=0}^{n} \frac{2k+1}{k+1} d_k C_k C_{n-k} = \sum_{k=0}^{n} \frac{d_k}{k+1} C_k C_{n-k},$$

so that d_k is $k+1$ by the recursion $C_k = \sum_{k=0}^{n} C_k C_{n-k}$ from before. Finally we see that $C_n = B_n/d_n = \frac{1}{k+1}\binom{2n}{n}$ is the number of Dyck words, equivalently the number of Dyck paths. $\square$

Now Dyck words/paths, being "Catalan objects", are interesting in their own respect. The surprising depth of Catalan combinatorics is expounded upon in many sources, for instance in [22]. But clearly there is no bijection from parking functions to Dyck paths since they are counted by different numbers. But we can modify the paths to put them in a clever bijection, unfortunately obscuring some of the Catalan features of the paths.

Label the up steps of a Dyck path with elements of $[n] =_{\text{def}} 1, 2, \ldots, n$, so that consecutive up steps preserve the canonical order of the integers. Such objects are called *labelled Dyck paths* of length $2n$, and exist in bijection with the parking functions with n cars. The columns of the up steps give the increasing rearrangement, and the labels give the permutation.

We can see the Catalan combinatorics begin to emerge, however, when considering the fact that rearrangements of parking functions freely permute the labels. So the number of equivalence classes of the parking functions under the rearrangement relation is C_n, and each equivalence class has a multinomial coefficient as its size. This problem is greatly complicated for the k-Naples case, which we will focus on later, with no simple lattice representation in general, nor closure under rearrangements. But this makes for interesting problems.

1.3 k-Dyck paths and m-ary trees

Recall that a combinatorial statistic on a set S is a function $\mathtt{stat} : S \to \mathbf{N}$. Two statistics are equi-distributed if they share the same distribution $D(\mathtt{stat}) = \sum_{s \in S} x^{\mathtt{stat}(s)}$. These generalize combinatorics insofar as the cardinality of S can be thought of as a specialization $x \to 1$ of $D(\mathtt{stat})$. Also, they algebraize problems, allowing new approaches to be used. Similarly, generalizations of the underlying objects can be useful as they specialize to the original objects. This can sometimes simplify proofs that rely on more abstract ideas.

It is worth wondering whether there are nice k-analogs of $\mathtt{pmaj}$ and $\mathtt{dinv}$ that are equi-distributed with $\mathtt{area}_k$ for all k. Similarly, if there are similar results for other generalizations of parking functions. Almost twenty such objects are described in [6], and more than ten of these have been explicitly studied. I will briefly introduce and discuss an example.

Definition 1.8. The *parking sequences* let cars take up more than just one spot. The car size vector (y_i) is the list of the number of parking spots car c_i takes up.

Theorem 1.9. The parking sequences with car size vector (y_i) are counted [7] by the number

$$(y_1 + n)(y_1 + y_2 + n - 1) \cdots (y_1 + y_2 + \cdots + y_{n-1} + 2),$$

which can be seen to be $|PF_n| = (n+1)^{n-1}$ for $(y_1, y_2, \ldots, y_n) = (1, 1, \ldots, 1)$ as expected.

Example. Take car size vector $y = (3, 1, 1)$. This has $n = 3$, so the number of y-parking sequences is

$$(3 + 3)(3 + 1 + 2) = 6^2 = 36.$$

Unfortunately it is not clear how to extend our results to this case, since our $\texttt{area}_k$ statistic has no obvious analog for (y_i) parking sequences. Perhaps a different generalization of Dyck paths may be involved, but there are many of these. I will introduce and discuss an example.

A k-Dyck path of down-length n is a first-quadrant lattice path from $(0,0)$ to $(nk+n,0)$ with up steps $(1,1)$ and down steps $(1,-k)$. So, for example, classical Dyck paths are 0-Dyck paths. We say the down-length of a Dyck path is its *semi-length.* We describe here results presented in [11].

An m-ary tree is a tree with at most m children under every node, labelled injectively with $[m]$. Call the set of m-ary trees on n vertices $\mathcal{T}_n^m$. Now the m-statistic $e : \mathcal{T}_n^m \to \mathbf{N}^m$ sending a tree to $(e_1, e_2, \ldots, e_m)$, with e_i the number of nodes labelled i, is equi-distributed with any of its permutations $(e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(m)})$. Let the labelled nodes be represented by ordered pairs (j,k) with $j \in [n]$ the vertex name and $j \in [m]$ the label. Now the bijection on $\mathcal{T}_n^m$ sending (e_i) to $e_{\pi(i)}$ for any $\pi \in S_m$ is given by sending (i, k_i) to $(i, \pi(k_i))$.

Proposition 1.10. For any $\pi \in S_n$, the distribution of the statistic $(e_{\pi(i)})$ is the same.

Proof. To prove this result, we give a bijection φ from $\mathcal{T}_n^m$ to itself, considering different statistics on each side, and showing that those statistics are equi-distributed.

Since the map φ only affects child-labellings, it preserves the tree structure. Now in each node's child-set C, the children are initially labelled injectively by a function $f : C \to [m]$, so the new labelling $\pi \circ f$ will also be injective since a composition of injective functions is injective. Hence the map is also well-defined, and it remains to show surjectivity. But for any permutation π there is a unique inverse π^{-1}, so $\pi^{-1} \circ f$ gives the unique inverses. So the map φ is a bijection. Now, we look at the statistics and how they are affected by the map φ. We notice that the bijection re-labels everything labelled i to $\pi(i)$, so the distributions are equi-distributed. $\square$

Burstein uses the result about m-ary trees to prove this result about k-Dyck paths: that the $(k+1)$-statistic $(\mathrm{pk}_0, \mathrm{pk}_1, \mathrm{pk}_2, \ldots, \mathrm{pk}_{k-1}, \mathrm{dd})$ with pk_i the number of peaks of a k-Dyck path at height $i \pmod k$ and dd the number of double descents is equi-distributed with its permutations.

Take the classical case $k = 1$. The 2-statistic is $(\mathrm{pk}, \mathrm{dd})$ with pk the number of peaks (equal to pk_0 for $k = 1$). In [11, Example 10] the bijection is explicitly given: $\eta(\phi) = \phi$ and $\eta(\pi u \pi' d) = \eta(\pi') u \eta(\pi) d$ with ϕ the empty path and π, π' any two paths (this definition runs recursively).

Now we see another interesting case where two statistics have the same distribution. This will be a theme, and we'll see an interesting theorem like this in section 2.3.

Proposition 1.11. The statistic $(\mathrm{pk}, \mathrm{dd})$ on 1-Dyck paths is equi-distributed with $(\mathrm{dd}, \mathrm{pk})$.

Proof. We proceed by induction on the down-size of the Dyck path. For the base case $n = 0$ there is only the empty path ϕ, with neither peaks nor descents, so $\eta(\phi) = \phi$ is enough, sending $(0, 0)$ to $(0, 0)$. Now for induction, suppose for some down-length $n > 0$, η is a bijection on PF_n reversing the statistic for all $0 \leq m < n$. Now any Dyck path $p \in PF_n$ can be written uniquely in the form $\pi u \pi' d$ with π, π' Dyck paths of semi-length less than n. Thus we have the equations:

$$\mathrm{pk}(p) = \begin{cases} \mathrm{pk}(\pi) + 1 & \text{if } \pi' = \phi, \\ \mathrm{pk}(\pi) + \mathrm{pk}(\pi') & \text{otherwise,} \end{cases}$$

$$\mathrm{dd}(p) = \begin{cases} \mathrm{dd}(\pi) & \text{if } \pi' = \phi, \\ \mathrm{dd}(\pi) + \mathrm{dd}(\pi') + 1 & \text{otherwise.} \end{cases}$$

Now to show the statistic is handled correctly, it is enough to consider for each parking function $p = \pi u \pi' d$, the cases where $\pi' = \phi$ and $\pi' \neq \phi$. When $\pi = \pi' = \phi$, $n = 1$ and there is only ud, again with neither peaks nor descents; and we can compute $\eta(ud) = \eta(\phi u \phi d) = \eta(\phi) u \eta(\phi) d = \phi u \phi d = ud$ is a bijection sending $(0, 0)$ to $(0, 0)$ again. When $\pi' = \phi \neq \pi$, $(\mathrm{pk}(p), \mathrm{dd}(p)) = (\mathrm{pk}(\pi) + 1, \mathrm{dd}(\pi))$. Now $\eta(p) \neq \phi$ has statistic $(\mathrm{pk}(\eta(\pi)), \mathrm{dd}(\eta(\pi)) + 1) = (\mathrm{dd}(\pi), \mathrm{pk}(\pi) + 1)$.

It remains to show η is a bijection for general semi-length. But as Burstein notes, η is an involution. Compute $\eta(\eta(p)) = \eta(\eta(\pi') u \eta(\pi) d) = \eta(\eta(\pi)) u \eta(\eta(\pi')) d = \pi u \pi' d = p$, taking ϕ as base case. This is enough because $\eta \circ \eta = \mathrm{id}$ means η has both inverses (itself), hence is bijective. □

Problem 1. Does Burstein's theory of k-Dyck paths and m-ary trees give us an interesting generalization of parking functions? That is, what happens if you label k-Dyck paths?

1.4 Other related combinatorics

There are other ideas related to the parking functions that we would like to present here. We do not present these results in depth as we want to focus on a particular generalization of parking functions called the k-Naples parking functions.

We will present the Pitman-Stanley polytope, the Shi hyperplane arrangement and the space of diagonal harmonics. Each of these objects is intimately connected to the parking functions, and we give a theorem relating each to them.

Problem 2. Can any of these objects be generalized to create an analogous theorem for a generalization of the parking functions? If so, can we extract information about the generalization from the data of the abstraction? Then, do these analogies yield interesting facts about classical parking functions?

Definition 1.12. The *Pitman-Stanley polytope* is the set

$$\Pi_n(x_1, x_2, \ldots, x_n) := \{y \in \mathbf{R}^n : \forall i \in [n], (y_i \geq 0 \wedge y_1 + \cdots + y_i \leq x_1 + x_2 + \cdots + x_i)\}.$$

Theorem 1.13. The volume of $\Pi_n(x_1, x_2, \ldots x_n)$ is [3] a sum over the parking functions

$$V_n\big(\Pi_n(x_1, x_2, \ldots, x_n)\big) = \frac{1}{n!} \sum_{(a_i) \in PF_n} x_{a_1} x_{a_2} \cdots x_{a_n}.$$

Example. Recall $PF_{2,0} = \{(1,1), (1,2), (2,1)\}$. We have that the volume of $\Pi_2(3,2)$ is

$$\frac{x_1x_1 + x_1x_2 + x_2x_1}{2} = \frac{21}{2}.$$

The polytope is the trapezoid with $h_1 = 5$, $h_2 = 2$ and $w = 3$ so indeed it has area

$$A = \frac{(h_1 + h_2)w}{2} = \frac{7 \cdot 3}{2} = \frac{21}{2}.$$

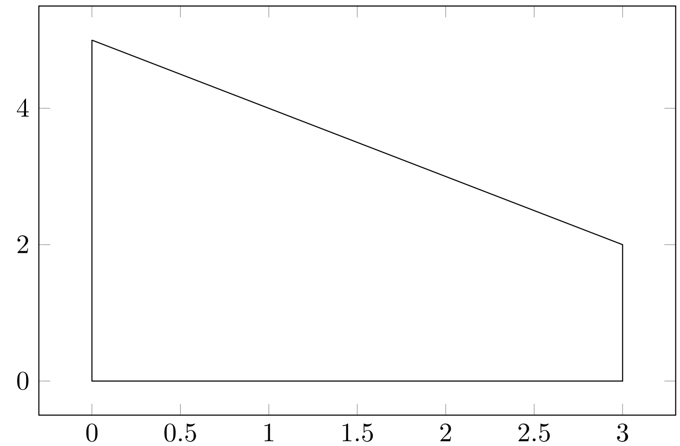


Figure 1.1: The Pitman-Stanley polytope $\Pi_n(3,2)$

Definition 1.14. The *Shi hyperplane arrangement*

$$\mathcal{S}_n = \{x_i - x_j = 0 \text{ for real } 1 \leq i < j \leq n\}.$$

Theorem 1.15. The Shi hyperplane arrangement $\mathcal{S}_n$ divides R^n into $(n+1)^{n-1}$ regions [15]. In fact there is a known bijections assigning parking functions to these regions in a methodical way. [8]

Definition 1.16. The *space of diagonal harmonics of order n*

$$DH_n = \left\{ f \in R_n : \sum_{i=1}^{n} \frac{\partial^h}{\partial x_i^h} \frac{\partial^k}{\partial y_i^k} f = 0 \text{ for } 1 \leq h + k \leq n \right\},$$

with $R_n = \mathbf{C}[x_1, x_2, \ldots x_n, y_1, y_2, \ldots y_n]$ the ring of polynomials in $2n$ commuting variables x_i and y_i, with $V_{h,k,n}$ decomposing R_n as a doubly graded vector space in some sense [5].

Definition 1.17. The *Hilbert series of diagonal harmonics* is

$$H_n(q,t) = \sum_{h \geq 0} \sum_{k \geq 0} \dim(DH_n \cap V_{h,k,n}) q^h t^k.$$

Theorem 1.18. We have [5] that $q^{n(n-1)/2} H_n(1/q, q) = [n+1]_q^{n-1} = |PF_{n,0}|_q$.

Corollary 1.19. Specializing $q \to 1$, we see that $\dim DH_n = (n+1)^{n-1} = |PF_{n,0}|$

Chapter 2

k-Naples parking functions

2.1 Definition and basic results

The k-Naples parking functions can be thought of as cars whose parking preferences park under the parking rule modification allowing the cars to check the first k spots behind their spot. They were introduced for $k = 0$ by [23] and generalized in [9]. I studied this subject as part of the 2020 UMass Math REU and AIM UP, which culminated in a literature review and the publication of [1].

Definition 2.1. Imagine a one-way street with n parking spots, and n cars entering the street one by one. Suppose that each car has a preferred spot, which it attempts to park in. If the spot is empty, it parks; otherwise, it checks first the spot behind it, then the spot behind that, and so on up to k times. If all those spots are taken, it moves forward to the next available spot. When the cars park without driving off the road, the sequence of their preferences is a *k-Naples parking function.*

Example. Consider the parking preference 223. This is not a (0-Naples) parking function since no car can ever even check the first spot. But it is 1-Naples because the second car will come check the second spot, and seeing it is full, parallel park back into the first spot.

The decreasing k-Naples parking functions in [9] are related to the representation of parking functions as labelled Dyck paths. Recall that a Dyck path is defined by restricting a north-east lattice path from the origin in $\mathbf{N}^2$ from crossing the line $y = x$. In particular, this forces the first step to be north. A decreasing k-Naples parking function of length n loosens this restriction by moving the line over to $y = x - k$ while requiring the first step still be north.

Why the first step north requirement? It is necessary for the Dyck path abstraction to carry over to an abstraction of the car-parking analogy. In the definition of the parking function, if their spot is taken, the cars check zero spots behind their preferred spot before moving on. The word "zero" in that sentence corresponds to the fact that the parking functions are zero-Naples. In fact, we've said nothing about "decreasing"! The parking preference vectors underlying decreasing k-Naples parking functions above can be shown to be k-Naples parking functions, and the decreasing rearrangements of the k-Naples parking functions can be shown to be decreasing k-Naples parking functions. The Naples parking functions were introduced in [23].

Definition 2.2. Let $s = (s_i)$ be a vector in $\mathbf{N}^\ell$. Then an s-Dyck path is a north-east lattice path in the ℓ by $\sum_{i=1}^{\ell}(s_i - 1) + 1 = \sum_{i=1}^{\ell} s_i - \ell + 1$ grid lying above (but not on) the path from $\big(j,\ \sum_{i=1}^{j-1}(s_i - 1) + 1\big)$ to $\big(j,\ \sum_{i=1}^{j}(s_i - 1) + 1\big)$ at each row $1 \leq j \leq n$. This boundary path is called the s-Dyck ribbon, and s is called a Catalan signature.

Proposition 2.3. The increasing k-Naples parking functions with n cars are in bijection with the s-Dyck paths from [24] with k-Naples signature

$$s_k = (k+1, \underbrace{2, 2 \ldots, 2}_{n-k \text{ times}}, \underbrace{1, 1 \ldots, 1}_{k \text{ times}}).$$

Example. The decreasing 3-Naples parking functions of length 5 are in bijection with the s-Dyck paths with signature $(4, 2, 2, 1, 1, 1)$.

Note that a classical Dyck path is the s-Dyck path with signature $s = s_1 = (1, 2, \ldots, 2)$ and the s-Dyck ribbon is the sequence of north and east steps following the diagonal, and a k-lattice path is an s_k-Dyck path with signature s_k as defined before. The following algorithm describes how to generate the s-Dyck ribbon given the signature.

Algorithm 1: The s-Dyck ribbon

Result: Generate the s-Dyck ribbon
given an n-Dyck signature $s = (s_i)$
for i from 1 to n **do**
  put $s_i - 1$ east steps;
  put a single north step;
end

Definition 2.4. The T-map $T : PP_n \to PP_n$ is given by the formula $T((a_i)) = (\tau(a_i))$ with

$$\tau(a_i) = \begin{cases} a_i & \text{if } i = 1\text{, or if } a_i = 1\text{, or if } a_i \neq 1 \text{ and } a_i \neq \tau(a_j) \text{ for all } j < i, \\ a_i - 1 & \text{if } i \neq 1\text{, and } a_i \neq 1\text{, and } a_i = \tau(a_j) \text{ for some } j < i. \end{cases}$$

Theorem 2.5. A parking preference α is 1-Naples if and only if $T(\alpha)$ is 0-Naples. [9]

Example. Take $\alpha = (3, 3, 3, 2) \in PP_4$. Then $T(\alpha) = (3, 2, 2, 1) \in PF_{4,0}$, so $\alpha \in PF_{4,1}$. Conversely, let $\beta = (3, 3, 3, 3) \in PP_4$. Then $T(\beta) = (3, 2, 2, 2) \notin PF_{4,0}$, so $\beta \notin PF_{4,1}$.

For the k-Naples parking functions, with $k \geq 1$, this was extended to a more general form characterizing k-Naples parking functions as preferences mapping to a 0-Naples in [2].

Problem 3. Is there a similar characterization for other generalizations of parking functions?

Theorem 2.6.

$$|PF_{n,k}| = \sum_{i=0}^{n} \binom{n}{i} \min((i+1)+k, n+1)|PF_{i,k}|(n-i+1)^{n-i-1}. \quad [9]$$

Proof. Each term counts the number of ways that $n+1$ cars can park such that c_n parks in the spot $i+1$. There are $\binom{n}{i}$ choices for the cars that park to the left of spot $i+1$, then $PF_{i,k}$ ways for them to park. The $(n-i+1)^{n-i+1}$ term then counts the number of ways for $n-i+1$ cars to park such that no car would ever try to check before the first parking spot if that was possible. Christensen et al call this the set of *contained parking functions* $B_{n,k}$ and prove [9, Lemma 3.4] the counting formula by a generalization of Pollack's circluar parking lot trick [15] mentioned before. □

Problem 4. Can other generalizations of parking functions also be counted recursively?

2.2 Counting the k-Naples functions

Together with Laura Colmenarejo, Pamela E. Harris, Zakiya Jones, Andrés Ramos Rodríguez, Eunice Sukarto, and Andrés R. Vindas-Meléndez in [1], I continued the study of the k-Naples parking functions. Extending a result from [25], we give a formula for the number of k-Naples parking functions causing a particular parking outcome. This allows us to give a closed form for the number of k-Naples parking functions in terms of the permutations corresponding to how the cars park.

Definition 2.7. Given a k-Naples parking function, consider the function $\varphi_k : PF_{n,k} \to S_n$ given by mapping a k-Naples parking function $\alpha \in PF_{n,k}$ to the permutation denoting the position in which the cars park under α using the k-Naples parking rule.

Remark 1. The statistic φ_0 is statistic `St001346` in [26], for which we give a formula in [1].

Example. Consider $\alpha = (4, 2, 2, 4, 1)$. We know that α is a parking function with $\phi_0(\alpha) = 52314$. In fact, α is also a 1-Naples parking function. Under the 1-Naples parking rule, the cars park as

spot 1	spot 2	spot 3	spot 4	spot 5
c_3	c_2	c_4	c_1	c_5

Therefore, $\varphi_1(\alpha) = 32415$. Notably, this example shows $\varphi_1(\alpha) \neq \varphi_0(\alpha)$ in general. To formally express this idea, we'll need to define an auxiliary function:

Definition 2.8. For $k \geq 0$, $0 < i < n+1$ and $\sigma \in S_n$,

$$\ell_k\left(i;\sigma\right) := \begin{cases} \ell\left(i;\sigma\right) - 1 + \min(k+1, \ell\left(n+1-i;\mathrm{rev}(\sigma)\right)) & \text{if } \ell\left(i;\sigma\right) = i \\ \max(\ell\left(i;\sigma\right) - 1 - k, 0) + \min(k+1, \ell\left(n+1-i;\mathrm{rev}(\sigma)\right)) & \text{if } \ell\left(i;\sigma\right) < i. \end{cases}$$

The idea here is that ℓ counts the number of spots car c_i can possibly have as its preference given that it parks ultimately in spot σ_i. This allows us to derive a closed form expression for the size of the fibers of φ_k. Summing these fibers gives us as a corollary the number of k-Naples parking functions of length n in a non-recursive manner.

Theorem 2.9. Let $\sigma \in S_n$ be a permutation. Then

$$|\varphi_k^{-1}(\sigma)| = \prod_{i=1}^{n} \ell_k\,(i;\sigma)\,.$$

Corollary 2.10. For all $n \geq 1$ and $0 \leq k \leq n-1$,

$$|PF_{n,k}| = \sum_{\sigma \in S_n} \left(\prod_{i=1}^{n} \ell_k\,(i;\sigma) \right).$$

Example. Take $\sigma = 2134 \in S_n$ and $k = 1$. This has $\ell_1(1;\sigma) = 1$, $\ell_1(2;\sigma) = 2$, $\ell_3(1;\sigma) = 3$, and $\ell_1(4;\sigma) = 4$. So by the corollary we have $|\varphi_1^{-1}(\sigma)| = \prod_{i=1}^{4} \ell_1(i;\sigma) = 2 \cdot 1 \cdot 3 \cdot 4 = 24$. These only differ from the ℓ_0 terms in that $\ell_0(2;\sigma) = 1$. And indeed $|\varphi_1^{-1}(\sigma)| = 2|\varphi_0^{-1}(\sigma)|$ (compare at [26]).

2.3 The k-Naples area statistic

Recall that a (combinatorial) *statistic* on a (combinatorial) set S is a function $\texttt{stat} : S \to \mathbf{N}$. The distribution $D(\texttt{stat})$ of a statistic $\texttt{stat} : S \to \mathbf{N}$ is the formal power series $D(\texttt{stat}) = \sum_{s \in S} q^{\texttt{stat}(s)}$. The area of a Dyck path is the number of full boxes between the path and the line $y = x$, i.e.

$$\texttt{area}(a_1, a_2, \ldots, a_n) = \sum_{i=1}^{n} (n - i - a_i + 1) = \frac{n^2 + n}{2} - \sum_{i=1}^{n} a_i.$$

A q-analog of a formula F is a polynomial P in q with $\lim\limits_{q \to 1} P = F$. We show that $\sum_\sigma \prod_{i=1}^{n} \ell_0\,(i;\sigma)_q = \sum_{p \in PF_n} q^{\texttt{area}(p)} = D(\texttt{area})$. Using results from [5], we get that this is $D(\texttt{pmaj}) = D(\texttt{dinv})$, connecting our q-analog to two equi-distributed statistics, and a related corollary for any statistic $\texttt{stat} : PF_n \to \mathbf{N}$. Where $\texttt{right}_k(i, \sigma)$ is the number of spots to the right of and including i which c_i can have as a preference and still park in spot σ_i, we define the k-Naples area statistic as follows.

Definition 2.11. Let $\sigma \in S_n$ be a permutation. Then

$$\texttt{area}_k(\sigma) = \sum_{i=1}^{n} [n - i + \texttt{right}_k(i, \phi_k(\sigma)) - \sigma_i]$$

Theorem 2.12 (Main result)**.** For all $n \geq 1$ and $0 \leq k \leq n-1$,

$$\sum_{\sigma \in S_n} \prod_{i=1}^{n} \ell_k \left(i;\sigma\right)_q = D(\texttt{area}_k)$$

. This is proved bijectively, splitting the sum up into the fibers of φ_k, defining the statistic $\texttt{stat}((u_i)) = \sum u_i$ and recognizing that

$$\prod_{i=1}^{n} \ell_k(i;\sigma)_q = \sum_{(u_i)\in\mathcal{I}} q^{\texttt{stat}((u_i))}$$

in the sense that it is a generating function for objects $(u_i) \in \mathcal{I}$ with $u_i < \ell_k(i;\sigma)$. Then we take a bijection $f : \mathcal{I} \to \varphi_k^{-1}(\sigma)$ with $\texttt{stat} = \texttt{area}_k \circ f$. In particular we choose the map

$$f((u_i)) = (\pi_1 + \texttt{right}_k(1;\sigma) - u_1 - 1, \pi_2 + \texttt{right}_k(2;\sigma) - u_2 - 1. \ldots, \pi_n + \texttt{right}_k(n;\sigma) - u_n - 1)$$

where σ is as above and π is defined by the formula $\sigma_{\pi_i} = i$. See in [1] for the details.

2.4 An upper bound for $PF_{n,k}$

In a recent preprint [12], Roger Tian presents an upper bound for $|PF_{n,k}|$ by exploiting connections with other generalizations of parking functions. Let $PF_{m,n,k}$ be the k-Naples parking functions with m cars parking on n spots and $LPF(m,n,k)$ the set of parking functions with m cars parking on n spots, the first k of which are obstructed. Then we have the following results.

Theorem 2.13.

$$|PF_{m,n,k}| \leq |LPF(m, n+k, k)|$$

Corollary 2.14.

$$|PF_{n,k}| \leq (k+1)(k+n+1)^{n-1}$$

You can see that the formula specializes correctly to $|PF_{n,0}| \leq (n+1)^{n-1}$. In fact $PF(m,n,k)$ injects into $LPF(m, n+k, k)$ and along the way we pick up a bijection between the contained parking functions $B(m,n,k)$ and $PF(m,n)$, extending a conjecture from [9]. This begins to demonstrate that different generalization of parking functions as in [6] are more closely related than they appear,

Chapter 3

New directions

3.1 Statistics compatible with a parking function

Alejandro H. Morales suggested generalizing the notion of permutation-compatible functions to parking functions. In [10] Stanley introduced this notion for permutations.

Definition 3.1. For $w \in S_n$, call $f : [n] \to \mathbf{N}$ as a w-compatible function if

a. $f(w_1) \geq f(w_2) \geq \cdots \geq f(w_n)$,

b. $f(w_i) > f(w_{i+1})$ when $w_i > w_{i+1}$.

Example. Take $w = 213$. Then $f : [3] \to \mathbf{N}$ is w-compatible if $f(2) > f(1) \geq f(3)$. So for instance we could define $g : [3] \to \mathbf{N}$ by $g(1) = 1$, $g(2) = 2$ and $g(3) = 1$. So g is w-compatible.

Proposition 3.2. Each statistic $f : [n] \to \mathbf{N}$ is w-compatible for a unique $w \in S_n$. [10]

For $f : [n] \to \mathbf{N}$, write $|f| = \sum_{i=1}^{n} f(i)$. Then let $\mathcal{A}(w)$ be the set of w-compatible functions.

Proposition 3.3 ([10], Lemma 1.4.12b)**.** We have for each w in S_n that

$$\sum_{f \in \mathcal{A}(w)} q^{|f|} = \frac{q^{\mathtt{maj}(w)}}{(1-q)\,(1-q^2)\cdots(1-q^n)}.$$

I have extended the idea to parking functions and proved the analogous result in this case.

Definition 3.4. Let $p = (p_1, p_2, \dots, p_n)$ be a parking function of length n. We say that a function $f : [n] \to [n]$ is p-compatible provided the following conditions hold of it:

a. $f(p_1) \le f(p_2) \le \cdots \le f(p_n)$,

b. $f(p_i) < f(p_{i+1})$ when $p_i \le p_{i+1}$.

These roughly correspond to filling in the labelled Dyck path. The choice of direction of the inequality doesn't really matter up to the S_n isomorphism $i \mapsto n - i + 1$, but we choose to use increases rather than decreases since this is usually how labelled Dyck paths are drawn in the literature.

Problem 5. Is there is a formula for the number of parking functions a statistic is compatible with?

Let $p = (p_1, p_2, \dots, p_n)$. Define $|f| = \sum_{i \in [n]} f(i)$ and $\texttt{comaj}(p) = \left(\sum_{i \le |p|} i\right) - \texttt{maj}(p)$. Now let $\mathcal{A}(p)$ be the set of p-compatible functions. The following result, which is the main result of this section, extends Stanley's lemma to parking functions.

Lemma 3.5.

$$\sum_{f \in \mathcal{A}(p)} q^{|f|} = \frac{q^{\texttt{comaj}(p)}}{(1-q)(1-q^2)\cdots(1-q^n)}.$$

Proof. Following Stanley's proof, we can form the inequality

$$1 \le f(p_1) \le f(p_2) \le \cdots \le f(p_n)$$

with strict inequality between $f(p_i)$ and $f(p_{i+1})$ if $p_i \le p_{i+1}$. Define $g(p_i)$ by reducing $f(p_i)$ by the number of ascents and ties to the left of and including position $i+1$, so that

$$0 \le g(p_1) \le g(p_2) \le \cdots \le g(p_n).$$

We can use this g function to compute the distribution of $|f|$ over $\mathcal{A}(p)$:

$$\begin{aligned}
\sum_{f \in \mathcal{A}(p)} q^{|f|} &= q^{\texttt{comaj}(p)} \sum_{0 \le g(p_1) \le g(p_2) \le \cdots \le g(p_n)} q^{g(1)+g(2)+\cdots+g(n)} = q^{\texttt{comaj}(p)} \sum_{m>0} \sum_{\lambda \vdash_n m} q^{|\lambda|} \\
&= q^{\texttt{comaj}(p)} \sum_{k_1 \ge 0} \sum_{k_2 \ge 0} \cdots \sum_{k_n \ge 0} q^{k_1 + 2k_2 + \cdots + nk_n} = q^{\texttt{comaj}(p)} \Big(\sum_{k_1 \ge 0} q^{k_1}\Big)\Big(\sum_{k_2 \ge 0} q^{2k_2}\Big) \cdots \Big(\sum_{k_n \ge 0} q^{nk_n}\Big) \\
&= q^{\texttt{comaj}(p)} / [(1-q)(1-q^2)\cdots(1-q^n)].
\end{aligned}$$

The second line recognizes $\sum_{f\in\mathcal{A}(p)} q^{|f|}$ as the sum of partitions with at most n parts and the third uses the bijection from partitions with n parts to partitions with each part at most n (this corresponds to a rotation of the Ferrers diagram of the partition). $\square$

Reversing the inequalities in the definition gives $\sum_{i\in[n]} i\chi_{\text{Tie}\vee\text{Des}}(p_i)$ as the statistic. This more clearly specializes to Stanley, but less clearly matches the usual way of writing labelled Dyck paths. Here we follow [5] in using χ_φ as the function mapping an object x to 1 if $\varphi[x]$ and 0 otherwise. This can be thought of as the Kronecker delta $\delta_{\varphi[x],\text{True}}$. Also $\text{Tie}(p_i)$ checks if i is a tie, that is if $p_i = p_{i-1}$ and $\text{Des}(p_i)$ checks if i is a descent, that is if $p_i < p_{i-1}$.

Problem 6. Can this result can be extended to generalizations of parking functions?

3.2 Recursively generating the k-Naples parking functions

Writing our joint work [1], Zakiya Jones conjectured an algorithm for recursively generating the decreasing k-Naples parking functions. We call this *Zakiya's algorithm*:

Algorithm 2: Zakiya's algorithm

Result: Generate $PF^d_{n,k}$ given $PF^d_{n-1,k}$, the decreasing k-Naples PFs of length $n-1$

let *set1* **be** $PF^d_{n,0}$, the decreasing 0-Naples PFs of length n

let *set2* **be** decreasing LPs beginning with n then n

let *proto-set3* **be** $PF^d_{n-1,k} \setminus PF_{n-1,0}$, the *proper* k-Naples PFs of length $n-1$

for *func* in proto-set3 **do**
- let $m = func_1$ be the first entry in *func*
- **for** i in $[n] \setminus [m-1]$ **do**
  - add the preference $(i, func_1, func2, \dots, func_n)$ to *set3*
- **end**

end

return the union of *set1*, *set2*, and *set3*

Christensen et al. gives a bijection from $PF^d_{n,k}$ to the k-lattice paths (See Theorem 1.3 in [1].) Then, Zakiya's algorithm follows from noting that

$$PF^d_{n,k} = \prod_{i=0}^{n-1} [\min(n+k-i, n)] = \prod_{i=0}^{n-1} \{1, 2, \ldots \min(n+k-i, n)\}$$

where multiplication is the "decreasing" Cartesian product that only appends weakly smaller numbers. For example, $(3, 2) \times [3] = \{(3, 2, 2), (3, 2, 1)\}$. However this can be improved by noticing that $PF^d_{n,k}$ is the union of such tuples beginning n, n and $PF_{n-1,k}$ with initial elements inserted in the possible spots as in figure 3.1 to make sure the corresponding lattice path is a decreasing k-lattice path. In fact, this set splits into $PF^d_{n,0}$ and $PF^d_{n-1,k} \setminus PF_{n-1,0}$ modified as such.

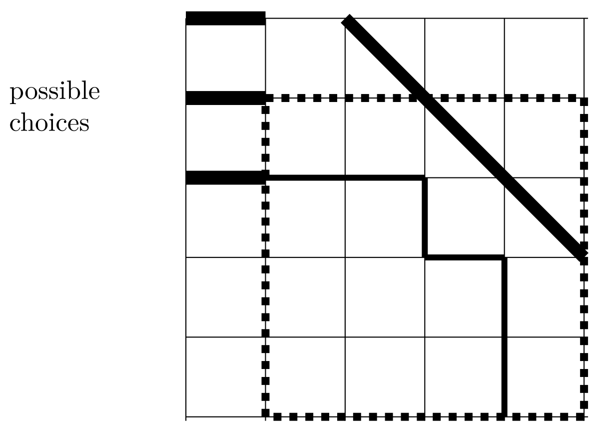


Figure 3.1: Splitting $PF_{n,k}$ in two

Problem 7. Could analysis of Zakiya's algorithm yield a new recursion for $PF_{n,k}$?

Problem 8. Is there a generalization of Zakiya's algorithm for a Dyck path of arbitrary signature?

We have run several experiments comparing Zakiya's algorithm with the recursion from [9] and the naive method of checking all n^n parking preferences individually. To my surprise, Zakiya's algorithm runs significantly slower in its various iterations.

Problem 9. What are the time complexities of the naive method and Zakiya's algorithm?

3.3 Characterizing rearrangements of k-Naples parking functions

We have explored when rearrangements of a 1-Naples parking function are 1-Naples.

Definition 3.6. A T-decrease in α is an index $i \leq n$ with $\tau(\alpha_i) = \alpha_i - 1$.

Proposition 3.7. If the increasing arrangement $\mathrm{Inc}(\alpha)$ is 1-Naples, then so is α.

Proof. Because $\mathrm{Inc}(\alpha)$ is Naples, its T-image $T(\mathrm{Inc}(\alpha))$ is a parking function [9]. It suffices to show that $\mathrm{Inc}(\alpha)$ has weakly fewer decreases than α since weakly decreasing a parking function yields a parking function. The definition of T ensures that no component falls below 1.

In particular, each tie in $\mathrm{Inc}(\alpha)$ corresponds to a T-decrease in α, but not every T-decrease in α corresponds to a decrease in $\mathrm{Inc}(\alpha)$. For each T-decrease in $\mathrm{Inc}(\alpha)$, say at position i, there are indices $j < i$ with $\mathrm{Inc}(\alpha)_i = \tau(\mathrm{Inc}(\alpha)_j) \leq \mathrm{Inc}(\alpha)_j$. Now because $\mathrm{Inc}(\alpha)$ is in increasing order with $j < i$, $\mathrm{Inc}(\alpha)_i \geq \mathrm{Inc}(\alpha)_j$, so $\mathrm{Inc}(\alpha)_i = \mathrm{Inc}(\alpha)_j$. But there may be T-decreases in α with $\alpha_i \neq \alpha_j$. $\square$

Definition 3.8. A bump for α is an index $i \in [n]$ with $\alpha_i = n - i + 2$ and either $i = n$ or $\alpha_{i+1} < n - i + 2$.

Proposition 3.9. Given a decreasing 1-Naples parking function, all its rearrangements are 1-Naples parking functions if and only if it has no adjacent bumps.

Proof. Since our parking function α has no adjacent bumps, we have that

$$a_j \begin{cases} \leq n - j + 1 & \text{if } j \text{ is not a bump,} \\ = n - j + 1 & \text{if } j + 1 \text{ is a bump,} \\ = n - j + 2 & \text{if } j \text{ is a bump.} \end{cases}$$

Since α has no adjacent bumps, if i is a bump, then $i - 1$ is not a bump, so $a_{i-1} = a_i = n - i + 2$. Now for each $\pi \in S_n$, it suffices to show that $T(\pi(\alpha)) \in PF_n$. Since

$$\tau(b_j) \leq b_j = a_{\pi(j)} \begin{cases} \leq n - \pi(j) + 1 & \text{if } \pi(j) \text{ is not a bump or if } \pi(j) + 1 \text{ is a bump,} \\ \leq n - \pi(j) + 2 & \text{if } \pi(j) \text{ is a bump,} \end{cases}$$

this shows that for each bump i, either $\tau(b_{\pi^{-1}(i-1)})$ or $\tau(b_{\pi^{-1}(i)}) \leq n-i+1$, where the same argument as above works since $\tau(b_{\pi^{-1}(i-1)}),\ \tau(b_{\pi^{-1}(i)}) \leq n-i+2$. Thus, $\pi(\alpha) \in PF_{n,1}$.

But if α has adjacent bumps, let $1 < i < n$ be the largest integer such that i and $i+1$ are bumps, so $a_i = n-i+2$, $a_{i+1} = n-i+1$. Consider the increasing rearrangement $\alpha^i = (a_n, \ldots, a_1)$ of α. We claim $\alpha^i \notin PF_{n,1}$. Suppose that under the parking preference α^i, the cars $c_1, \ldots, c_{n-i-1}$ do not all park within the spots $1, \ldots, n-i-1$. So, there exists a spot $1 \leq m \leq n-i-1$ which remains unoccupied after $c_1, \ldots, c_{n-i-1}$ have parked. Since for all $j \geq n-i$ the car c_j has parking preference greater than or equal to $a_{i+1} = n-i+1$, so c_j can only check spots $n-i, n-i+1, \ldots, n$, so cannot park at spot m. So m remains empty, hence there exists a car which is unable to park.

By contradiction, cars $c_1, \ldots, c_{n-i-1}$ must park in spots $1, \ldots, n-i-1$. Since c_{n-i} has parking preference $a_{i+1} = n-i+1$, it checks spot $n-i+1$ is empty and parks there. Since for all $j \leq n-i+1$ c_j has parking preference greater than or equal to $a_i = n-i+2$, c_j only checks spots $n-i+1, \ldots, n$, so cannot park at spot $n-i$. So $n-i$ remains empty, thus there is a car which is unable to park. $\square$

Corollary 3.10. Rearrangements of a decreasing parking preference with exactly one bump are 1-Naples.

Problem 10. Is there a corresponding notion of k-bump for arbitrary k?

Further, in their preprint [2], João Carvalho, Pamela Harris, Gordon Kirby, Nico Tripeny, and Andrés Vindas-Meléndez generalize the T map to $k > 1$ (see appendix 3.4). Thereby they derive a characterization of the subset of $PF_{n,k}$ closed under rearrangements.

Theorem 3.11. Given a parking preference, all of its rearrangements will be $k-$Naples parking functions if and only if its ascending rearrangement is a k-Naples parking function.

The question of determining how rearrangements work for k-Naples parking functions with ascending rearrangement not a k-Naples is more complicated, as it can be difficult to keep track of all the cars moving around. So we focus on the simpler problem when just two cars swap. In particular, this reduces the problem since any permutation can be factored into a composition of transpositions.

Now we focus talk about "decreasing" swaps on $PF_{n,k}$. These are relatively nice rearrangements since iterating them will *eventually* return to a k-Naples parking function: the decreasing rearrangement of any k-Naples is k-Naples. Does this mean every decreasing swap preserves k-Naplesness?

Unfortunately, the answer to this question is a resounding *no* in general. Take for example, $[1, 4, 5, 5, 3] \in PF_{5,2}$. Swapping the first and last terms decreases the function, but $[3, 4, 5, 5, 1] \notin PF_{5,2}$. But we can rescue our hypothesis for the case $k = n - 2$, recovering the following result.

Proposition 3.12. Let $p = p_1 p_2 \dots p_n \in PF_{n,n-2}$ and suppose $p_r < p_s$ for some indices $r < s$. Then the decreasing swap p' formed by swapping p_r and p_s remains in $PF_{n,n-2}$.

Proof. Suppose to the contrary that $p' \notin PF_{n,n-2}$. Then p' does not park using the $k = n-2$ Naples parking algorithm. We know $c_1, c_2, \dots, c_{n-1}$ all park with p' their preferences since they can all check the last $n - 1$ spots. This means that c_n fails to park, which means $p'_n = n$ and no car parks in spot 1. But some car parks in spot 1 with p the preferences, call it c_j. Either $p_j = 1$ or $p_j > 1$ and c_j backs up. In the first case, spot 1 is taken when the corresponding car comes in under the p' preference and we're done. Otherwise, $1 < p_j < n$ and there is a set of cars which c_j depends on to park (car c_j itself, cars c_j backs up over, cars they back up over, etc). If all these cars have preferences less than n, they can be rearranged any way and still fill in spot 1 because they can check all the parking spots. Otherwise, the cars with preference n have to not fail to park. *This is where the decreasing requirement for the rearrangement comes in!* Cars with preference n can only be moved forward by decreasing swaps, which guarantees they still park. □

What changes for $k = n - 3$? Now cars with preference $n - 1$ are also an issue, *but they can be moved forward by decreasing rearrangements!* For instance, $3324 \in PF_{4,1}$ with $j = 3$. The cars c_3 depends on to park are c_1, c_2, and itself. We can swap decreasingly, taking $r = 1$ and $s = 4$, and move a car with preference $n - 1 = 3$ forward. Now the function fails.

So there is something nice about $k = n - 2$ just like $k = 1$. This is at least some progress! In fact, the requirement that the swap be decreasing can be refined to simply not moving n forward in this case. So if $n \notin p \in PF_{n,n-2}$, then for all permutations $\pi \in S_n$ we have that the action $\pi(p) \in PF_{n,n-2}$. So S_n acts on the subset of $k = n - 2$ Naples parking functions not containing n.

Problem 11. Can anything be said about the orbits of this group action? Their sizes for instance?

Corollary 3.13. The number of k values for which a rearrangement fails at all seems to be less than or equal to the number of indices i with values p_i larger than i.

3.4 Parking matrices and their characteristic polynomials

Together with Andrés Ramos Rodríguez, Pamela Harris, and Laura Colmenarejo, we are currently exploring another idea. There is a notion of permutation matrix, since permutation of basis vectors is a linear operation. But parking functions extend permutations, so we can generalize.

Definition 3.14. Let f be a parking function of length n. Then the parking matrix M_f associated to f is the $n \times n$ matrix with 1 in entries M_{i,f_i} and 0 otherwise.

The idea of our conjecture is to look for another way of thinking about parking functions to see if it can be extended to other generalizations and help us understand those better. In particular, we consider the characteristic polynomials of the parking matrices.

Definition 3.15. The characteristic polynomial $P_M(x)$ of a matrix M is $\det(xI - M)$.

Conjecture 1. We have that $P_{M_f}(x) = x^t P_{M_\sigma}(x)$ where σ is a permutation.

We have been able to conjecture more specifically about the nature of t and σ:

Conjecture 2. In the above conjecture, $t = n - m$ and σ is the permutation corresponding to the largest bijection in f, viewing f as a function $[n] \to [n]$.

We are currently attempting to prove this by elementary linear algebraic methods. The goal is to manipulate an arbitrary matrix to nicely separate the permutation submatrix.

Example. Take $f = (2, 1, 1) \in PF_{3,0}$. We have the permutation $\sigma = (2, 1)$ associated to f so that $P_{M_f}(x) = xP_{M_\sigma(x)} = x(x^2 - 1) = x^3 - x$ by the conjecture since

$$P_{M_\sigma}(x) = \det(xI - M_\sigma) = \det \left(\begin{smallmatrix} x & -1 \\ -1 & x \end{smallmatrix}\right) = x^2 - 1.$$

Now we can check that this is correct by directly computing the characteristic polynomial

$$P_{M_f}(x) = \det(xI - M_f) = \det\begin{pmatrix} x & -1 & -1 \\ -1 & x & 0 \\ 0 & 0 & x \end{pmatrix} = x\det\begin{pmatrix} x & -1 \\ -1 & x \end{pmatrix} = xP_{M_\sigma(x)} = x(x^2-1).$$

Example. Take $(3,3,5,1,1) \in PF_{5,0}$. We have the permutation $\sigma = (2,3,1)$ associated to f so that $P_{M_f}(x) = x^2 P_{M_\sigma(x)} = x^2(x^3-1)$ by the conjecture since

$$P_{M_\sigma}(x) = \det(xI - M_\sigma) = \det\begin{pmatrix} x & 0 & -1 \\ -1 & x & 0 \\ 0 & -1 & x \end{pmatrix} = x^3 - 1.$$

Now we can check that this is correct by directly computing the characteristic polynomial

$$P_{M_f}(x) = \det(xI - M_f) = \det\begin{pmatrix} x & 0 & 0 & -1 & -1 \\ 0 & x & 0 & 0 & 0 \\ -1 & -1 & x & 0 & 0 \\ 0 & 0 & 0 & x & 0 \\ 0 & 0 & -1 & 0 & x \end{pmatrix} \rightsquigarrow_{\text{cols}} \det\begin{pmatrix} x & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & x \\ -1 & x & 0 & 0 & -1 \\ 0 & 0 & 0 & x & 0 \\ 0 & -1 & x & 0 & 0 \end{pmatrix}$$

$$\rightsquigarrow_{\text{rows}} \det\begin{pmatrix} x & 0 & -1 & -1 & 0 \\ -1 & x & 0 & 0 & -1 \\ 0 & -1 & x & 0 & 0 \\ 0 & 0 & 0 & x & 0 \\ 0 & 0 & 0 & 0 & x \end{pmatrix} = x^2 P_{M_\sigma(x)} = x^2(x^3-1).$$

Remark 2. You can see in both of these cases that we ended up with a permutation matrix in the upper left and a triangular component on the right. We intend to prove the conjecture by showing that any parking matrix can be factored that way. The next proposition will be useful.

Proposition 3.16. Let $\sigma \in S_n$ be a permutation with cycle structure $(C_1, C_2, \ldots, C_n)$. Then the characteristic polynomial of M_σ is given by the formula [27]

$$P_{M_\sigma}(x) = (-1)^n \prod_{k=1}^{n} (\lambda^k - 1)^{C_k}$$

Thus the number of characteristic polynomials of permutation matrices is $p(n)$, the partition number of n. Our conjecture marks a first step towards extending this idea to the parking functions.

This idea of algebraizing parking functions is not new. It seems difficult to find a natural group structure on the parking functions, but in [28] the authors endow them with a Hopf algebra structure.

Problem 12. Do any of these results extend nicely to generalizations of parking functions?

Appendix: Code

Experiments run in Sage [29], building on code by Andrés Ramos Rodríguez [30]. The source:

```
import sage.combinat.parking_functions as pf
from andresramos5.knpf import is_knaples


def is_a_ramos(x, k=0, n=0) -> bool:
    # Check whether a list is a `k`-Naples parking function.
    # If a size `n` is specified, checks if a parking function of size `n`.
    # Uses the naive algorithm, as implemented by Andres Ramos Rodriguez.

    return is_knaples(x, k) if n==0 or len(x)==n else return False



def is_a_carvalho(x, k=0, n=0) -> bool:
    # Check whether a list is a `k`-Naples parking function.
    # If a size `n` is specified, checks if a parking function of size `n`.
    # Uses the componentwise tau map algorithm of Carvalho et al (forthcoming).

    return pf.is_a(list(map(lambda t:tau(x,t,len(x),k), \
        range(1,len(x)+1)))) if n==0 or len(x)==n else return False
```

```python
def is_a(x, k=0, carvalho=False, n=0) -> bool:
    # Check whether a list is a `k`-Naples parking function.  By default `k` is zero.
    # If a size `n` is specified, checks if a list is a parking function of size `n`.
    # Uses Ramos by default, or Carvalho if carvalho=True. By default carvalho=False.

    if carvalho:
        return is_a_carvalho(x, k, n)
    else:
        return is_a_ramos(x, k, n)


def tau(a, i, n, k) -> sage.rings.integer.Integer:
    # Computes the generalized tau function of Carvalho et al. (forthcoming).
    # A tuple is k-Naples iff the componentwise action of tau is a parking function.
    # The idea is to return the leftmost spot each car checks under the parking rule.

    if (i==1 or a[i-1]==1 or (a[i-1]!=1 and \
            all(a[i-1]!=tau(a, j, n, k) for j in range(1,i)))):
        return a[i-1]
    elif (a[i-1]>k and all(any(a[i-1]-p==tau(a, j, n, k) \
            for j in range(1,i)) for p in range(k))):
        return a[i-1]-k
    else:
        for m in reversed(range(k)):
            if (a[i-1]>m and all(any(a[i-1]-p==tau(a, j, n, k) \
                    for j in range(1,i)) for p in range(m))):
                return a[i-1]-m
```

Bibliography

[1] Laura Colmenarejo et al. Counting k-naples parking functions through permutations and the k-naples area statistic. *Enumerative Combinatorics and Applications*, 1:2, 2021.

[2] Joao Carvalho et al. Enumerating k-naples parking functions through catalan objects. *Preprint*.

[3] Jim Pitman and Richard P. Stanley. A polytope related to empirical distributions, plane trees, parking functions, and the associahedron. *Discrete Comput. Geom.*, 27:603–634, 2002.

[4] Jian-Yi Shi. *The Kazhdan-Lusztig cells in certain affine Weyl groups*. Springer-Verlag, 1986.

[5] Nicholas Loehr. Combinatorics of q, t-parking functions. *Adv. Appl. Math.*, 34:408–425, 2004.

[6] Joshua Carlson et al. Parking functions: Choose your own adventure. *arXiv preprint*, 2020.

[7] Richard Ehrenborg and Alex Happ. Parking cars of different sizes. *Am. Math. Mon. 123*, 2016.

[8] Ivan Pak and Richard P. Stanley. Hyperplane arrangements, interval orders, and trees. *Proceedings of the National Academy of Science*, 93:6:2620–2625, 1996.

[9] Alex Christensen et al. A generalization of parking functions allowing backward movement. *Electronic Journal of Combinatorics*, 27(1):33, 2020.

[10] Richard P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, 2011.

[11] Alexander Burstein. Distribution of peak heights modulo k and double descents on k-Dyck paths. *arXiv preprint*, 2020.

[12] Roger Tian. Connecting k-naples parking functions and obstructed parking functions via involutions. *arXiv preprint*, 2021.

[13] Alan G. Konheim and Benjamin Weiss. An occupancy discipline and applications. *Society for Industrial and Applied Mathematics Journal on Applied Mathematics*, 14, 1966.

[14] Dan Saracino. *Abstract algebra: A first course.* Waveland Press, 2008.

[15] Richard P. Stanley. Parking functions. `math.mit.edu/~rstan/transparencies/parking.pdf`.

[16] André Joyal. Une théorie combinatoire des séries formelles. *Adv. Math.*, 42, 1981.

[17] Tom Leinster. A visual telling of joyal's proof of cayley's formula. `https://golem.ph.utexas.edu/category/2019/12/a_visual_telling_of_joyals_pro.html`, 2019.

[18] Marcel-Paul Schützenberger. On an enumeration problem. *J. Comb. Theory*, 4:219–221, 1968.

[19] Jim Haglund and Nicholas Loehr. A conjectured combinatorial formula for the hilbert series for diagonal harmonics. *Discrete Mathematics*, 298(1):189–204, 2005.

[20] Richard P. Stanley. *Catalan numbers.* Cambridge University Press, New York, 2015.

[21] Wikipedia contributors. Catalan number. `en.wikipedia.org/wiki/Catalan_number`, 2020.

[22] Jim Haglund. *The q,t-Catalan Numbers and the Space of Diagonal Harmonics*, volume 41 of *University Lecture Series.* American Mathematical Society, 2008.

[23] Alyson Baumgardner. The naples parking function. *Florida Gulf Coast University*, 2019.

[24] Cesar Ceballos and Rafael D'León. Signature catalan combinatorics. *J. of Comb.*, 10, 2019.

[25] Richard P. Stanley. *Enumerative Combinatorics*, volume 2. Cambridge University Press, 1999.

[26] Martin Rubey et al. Findstat, the combinatorial statistics database. `http://www.findstat.org`.

[27] Nathaniel Blair-Stahn. Random permutation matrices. *University of Arizona Math*, 2000.

[28] Jean-Christophe Novelli and Jean-Yves Thibon. A hopf algebra of parking functions. *Proceedings of Formal Power Series and Algebraic Combinatorics*, Vancouver, 2004.

[29] William A. Stein et al. *Sage Mathematics Software.* The Sage Development Team, 2020.

[30] Andrés Ramos Rodríguez. andresramos5/naples-parking-function. *Github repository*, 2019.