PARKING FUNCTIONS ON TREES AND DIRECTED GRAPHS

A Dissertation

by

HAROLD WESTIN KING

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Catherine Yan
Committee Members,	Michael Anshelevich
	Eric Rowell
	Sergiy Butenko
Head of Department,	Emil Straube

August 2019

Major Subject: Mathematics

ABSTRACT

A parking function can be thought of as a sequence of n drivers, each with a preferred parking space, wanting to park along a one-way street with n parking spaces. Each driver checks her preferred parking space and, if it is occupied, parks in the first available space afterwards. One may consider how the enumeration of these sequences changes if the "parking lot" is made more complex, a question whose solution this dissertation lays the foundations for and answers in the case of certain families of graphs.

We begin by generalizing the underlying parking lot to a general digraph and give several equivalent characterizations. We then start by building on recent work in the case that the parking lot is a tree with edges directed towards a root. We generalize the notions of "prime" and "increasing" parking functions to give enumerative results concerning both. Additionally, we consider one of the numerous statistics on classical parking functions, the number of drivers who park in their desired spot, and show that it connects these tree parking functions that are prime and those that are both increasing and prime. Finally, we consider miscellaneous results on various families of trees and enumerate a generalization of Dyck paths in the process.

ACKNOWLEDGMENTS

I would like to thank my parents for their constant support throughout my two-and-a-half decades of schooling as well as my adviser, Catherine Yan, for her guidance and patience the last four years. Additionally, I want to thank Nick Satullo for his motivation not to give up.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Catherine Yan and Professors Michael Anshelevich and Eric Rowell of the Department of Mathematics and Professor Sergiy Butenko of the Department of Industrial & Systems Engineering.

The work in Chapters 3, 4, and 5 was conducted in joint work with Professor Catherine Yan.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by a fellowship from Texas A&M University.

NOMENCLATURE

$[n]$	$\{1, 2, \ldots, n\}$
$[n]_0$	$\{0, 1, 2, \ldots, n\}$
$V(D)$	The vertex set of digraph D
$E(D)$	The edge set of digraph D
$\mathfrak{S}_n$	The symmetric group on n elements
$\mathcal{P}_n$	The directed path of n vertices upon which classical parking functions are defined
T_v	The maximal subtree of T rooted at vertex v
$\mathcal{T}_n$	The set of sink trees with n vertices
$\widetilde{\mathcal{T}}_n$	The set of source trees with n vertices
$\mathcal{M}_n$	The set of mapping digraphs with n vertices
$\widetilde{\mathcal{M}}_n$	The set of inverse mapping digraphs with n vertices
$P(D, m)$	The number of parking functions on digaph D with m drivers
$\mathcal{PF}_n$	$\{(T, p) : T \in \mathcal{T}_n$ and p is a prime parking function on $T\}$
F_n	$\sum_{T \in \mathcal{T}_n} P(T, n)$
P_n	$\lvert\mathcal{PF}_n\rvert$

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

Konheim and Weiss first introduced parking functions in 1966 when they investigated the probability that a random hashing function would fill a hash table when linear probing was used to resolve collisions [13]. One-by-one n drivers, each with a preferred parking space attempt to park on a one-way street. If their preferred spot is unavailable, the drivers park in the first available space after their preferred one. If all drivers are able to park, the sequence of their preferences is called a parking function.

Parking functions have proven to be a fruitful field of study, in large part due to their interconnectedness with both trees and Catalan structures (see [19] for a survey on the subject). They have appeared in many places: chambers in Shi and braid arrangements, maximal chains in the lattice of noncrossing partitions, symmetric functions, and they even have their own polytope (see [8, 16, 17, 19] and the references within).

Parking functions can be characterized in multiple ways, which we discuss in Section 2, and have numerous generalizations as a result. Rational parking functions come from lattice paths [2], G-parking functions are related to the spanning trees on a digraph G [15], $\vec{x}$-parking functions generalize the growth restriction of the order statistics of a parking function [17], parking sequences consider cars of different sizes attempting to park along a street [9], and the digraph parking functions we generalize in this dissertation consider a parking lot more complex than a one-way street [14].

We take the following scenario as our motivation: one-by-one, n users attempt to connect to servers. If the initial server a user attempts to connect to is busy, the server passes the user to another server, according to an underlying digraph, until the user finds a free server. We are interested in how many ways the users can attempts to initially attempt to connect to servers such that all users can be simultaneously served. This dissertation presents an initial foray into the subject.

2. PRELIMINARIES

This section discusses the definitions and previous theorems appearing in the literature necessary for the following sections. For the rest of this dissertation, we assume any digraph has vertex set $[n]$, where n is a positive integer, and is connected, unless stated otherwise.

2.1 Graphs

We define the various types of graphs mentioned in the rest of the dissertation. Additionally, we state a theorem by Hall that gives a necessary and sufficient condition for the existence of a perfect matching on a bipartite graph.

A *graph* G is a pair (V, E) where V is the vertex set and E is the edge set. As mentioned, we take $V = [n]$ and $E \subseteq 2^V$ such that all elements of E have cardinality 2. We will denote $V = V(G)$ and $E = E(G)$ in several places and we let $|G| = |V| = n$. The *neighborhood of* u are the vertices $\{v : \{u, v\} \in E\}$. Figure 2.1 is a graph with $V = [4]$ and $E = \{\{3, 4\}, \{1, 3\}, \{2, 3\}\}$.

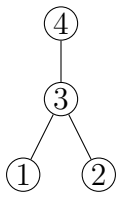

Figure 2.1: A graph.

A *directed* graph has $E \subseteq V \times V$ with no elements of the form (v, v). An edge (u, v) is understood to have an orientation with u being the origin vertex and v the terminal vertex. The edge is graphically represented by an arrow: $u \to v$. The *out-neighborhood* of u is $\{v : (u, v) \in E\}$. The cardinality of the out-neighborhood of u, denoted by $N(u)$, is called the *outdegree*.

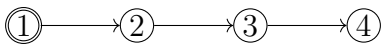

Figure 2.2: $\mathcal{P}_4$

2.1.1 Trees

Trees are connected graphs with n vertices and $n-1$ edges. A *rooted* tree is a tree with a distinguished node, called the root. Figure 2.3 is a tree with root 1. We denote the root by drawing the vertex as concentric circles. We consider two types of trees in this dissertation: sink trees and source trees. A *sink* tree is a rooted tree with edges directed towards the root. That is, if (u, v) is an edge, then v is on the unique path between u and the root. Similarly, a source tree is a rooted tree with edges directed away from the root. We call v the *parent* of u if the two are connected by an edge and v is closer to the root than u is. The edge orientation does not matter. In Figure 2.3, the vertex 2 is the parent of 7. In this situation, we also call u the *child* of v. Two vertices with a common parent are called *siblings*. We denote $\mathcal{P}_n$ to be the rooted tree with n vertices and edges $\{(i, i+1)\}_{i=1}^{n-1}$. Figure 2.2 is the graph $\mathcal{P}_4$.

Additionally, we consider two special types of rooted trees. A *caterpillar* is a tree on which there exists a path such that all vertices are distance at most 1 from some vertex on the path. A *star* is a caterpillar with n vertices and a vertex of degree $n-1$. For this paper and $a_i \geq 0$ integers, we let $\text{Cat}(a_1, a_2, \ldots, a_n)$ be the directed caterpillar with root 1, and edges $\{(i, i+1\}_{i=1}^{n-1} \bigcup_{j=1}^{n} \{(j, i + \sum_{\ell=1}^{j-1} a_\ell)\}_{i=1}^{a_j}$. Figure 2.3 shows $\text{Cat}(1, 3, 0, 2)$. We call the vertices $\{1, \ldots, n\}$ the *spine* vertices and all other vertices *leg* vertices. We note here that $\text{Cat}(0, \ldots, 0, 1)$ is the same as $\text{Cat}(0, \ldots, 0, 0, 0)$ (both are isomorphic to $\mathcal{P}_n$) as directed graphs, but we will consider them distinct for the purpose of the results in Chapter 6.

A *spider* is a rooted tree with at most one vertex of degree 3 or more. For $n_i \in \mathbb{N}$, we denote $S(n_1, n_2, \ldots, n_k)$ to be the tree formed by joining paths of n_i vertices to the root, $n = 1 + \sum_{i=1}^{k} n_i$. Let the path associated with n_1 have vertices between 1 and n_1, traveling away from the root, and in general the path associated with n_i having vertices $1 + \sum_{j=1}^{i-1} n_j$ to $\sum_{j=1}^{i} n_j$. For simplicity, we

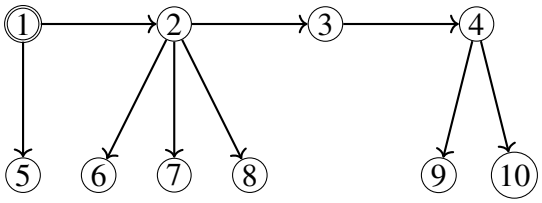

Figure 2.3: A an example of Cat$(1, 3, 0, 2)$.

call these paths *legs*. See Figure 2.4 for the graph of $S(2, 1, 1)$.

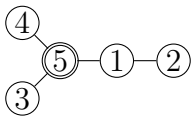

Figure 2.4: The spider $S(2, 1, 1)$.

2.1.2 Bipartite Graphs and Hall's Theorem

A *bipartite graph* B is an undirected graph whose vertex set may be partitioned into two disjoint sets X and Y such that for any $x_i, x_j \in X$ or $y_k, y_\ell \in Y$, $\{x_i, x_j\} \notin E(B)$ and $\{y_k, y_\ell\} \notin E(B)$. We say that $M \subseteq E$ is a *matching that saturates* X if, for all $x \in X$, there is a unique $y_x \in Y$ such that $\{x, y_x\} \in M$ and further there are no distinct $x_1 \neq x_2$ such that $\{x_1, y\} \in M$ and $\{x_2, y\} \in M$ for any $y \in Y$. The following gives a necessary and sufficient condition for the existence of such a matching M on B.

Theorem 1 (Hall's Matching Theorem)**.** *Let B be a bipartite graph with bipartition (X, Y). Then there exists a matching saturating X if and only if for all $A \subseteq X$ we have*

$$|A| \leq |\bigcup_{a \in A} N(a)|.$$

Figure 2.5 shows a bipartite graph on 6 vertices. The graph has a unique matching saturating

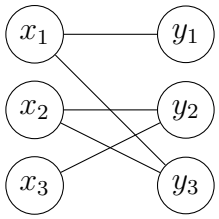

Figure 2.5: A bipartite graph.

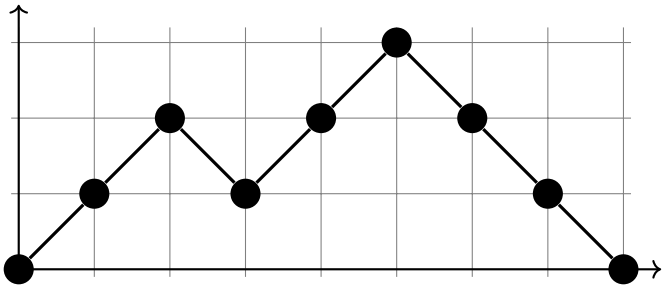

Figure 2.6: The Dyck path NNSNNSSS

$\{x_1, x_2, x_3\}$, $M = \{\{x_1, y_1\}, \{x_2, y_3\}, \{x_3, y_2\}\}$.

2.2 Lattice Paths

In this section, we discuss two lattice paths closely associated with parking functions: Dyck and Royal paths.

2.2.1 Dyck Paths

A *Dyck path of semilength* n is a lattice path in $\mathbb{Z}^2$ with steps N $= (1, 1)$ and S$= (1, -1)$ starting from $(0, 0)$ and ending at $(2n, 0)$ such that the path never dips below the x-axis. A Dyck path is called *prime* if it only touches the x-axis at $(0, 0)$ and $(2n, 0)$. We can also represent a Dyck path as a word of length $2n$ with letters $\{N, S\}$. Figure 2.6 gives an example of a prime Dyck path.

The number of Dyck paths of semilength n is well-known to be $C_n = \frac{1}{n+1}\binom{2n}{n}$. The numbers $\{C_n\}_{n\geq 0}$ are called the Catalan numbers and appear throughout combinatorics. If we count the Dyck paths of semilength n with k peaks, or appearances of "NS" in the word representation,

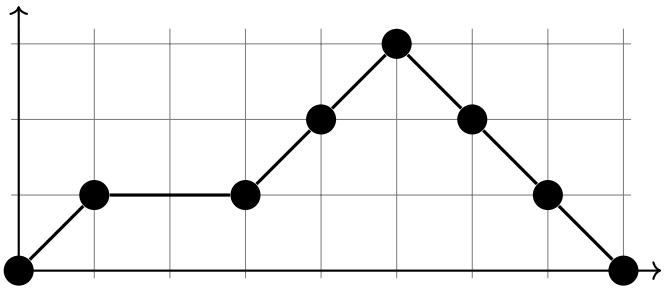

Figure 2.7: The Royal path NENNSSS

and sum over k, we get a refinement of the Catalan numbers by the Narayana numbers, $N_{n,k} = \frac{1}{n}\binom{n}{k}\binom{n}{k-1}$. So,

$$C_n = \sum_{k=1}^{n} N_{n,k}.$$

2.2.2 Royal Paths

A *Royal path of semilength* n is a slightly more general Dyck path; it is a lattice path with steps N$= (1,1)$, S$= (1,-1)$, and E$= (2,0)$ starting at $(0,0)$, ending at $(2n,0)$, and never going below the x-axis. Figure 2.7 gives an example.

Let S_n denote the number of Royal paths of semilength n. The numbers $\{S_n\}_{n\geq 0}$ are called the large Schröder numbers. After some consideration, we notice that we can turn a Royal path into a Dyck path by replacing the horizontal step (E) with a peak (NS) and there are precisely 2^k Royal paths that yield a given Dyck path in this manner, where k is the number of peaks in the Dyck path. We thus can say

$$S_n = \sum_{i=1}^{n} 2^k N_{n,k}. \tag{2.1}$$

We briefly note that both Dyck and Royal paths can be generalized further to Motzkin paths by replacing the E$= (2,0)$ steps in the Royal path with E$= (1,0)$. While also fascinating and useful, we will not discuss Motzkin paths for the remainder of this dissertation.

2.3 Parking Functions

Here we discuss both classical parking functions as well as recent work done by Lackner and Panholzer [14] for parking functions on sink trees.

2.3.1 Classical Parking Functions

Parking functions may be described as a sequence of drivers, each with a preferred parking space, searching for a place to park along a one-way street. Let $s \in [n]^n$ be the sequence of preferences and consider the path $\mathcal{P}_n$.

Definition 1 (Parking Process on $\mathcal{P}_n$)**.** The drivers attempt to park according to the following process:

1) Beginning with $i = 1$, driver i begins at vertex s_i.

2) If the current vertex is unoccupied, the driver parks there. If it is occupied and the current vertex is not n, the driver drives to the vertex in the out-neighborhood of the current one and repeats Step 2.

3) If she parks, the process continues with driver $i + 1$ attempting to park at vertex s_{i+1}. Otherwise, the process terminates.

If all n drivers parks, the sequence s is called a *parking function*. Figure 2.2 shows $\mathcal{P}_4$ for which 125 parking functions are defined. A few examples are: 1234, 1111, 3121, 2112, and 4311. Note that 1334 is *not* a parking function, because the final driver does not find a place to park.

While intuitive, this characterization does not lend well to mathematical manipulation. Luckily, we can see after some consideration that if all drivers can park then there is no i such that too many drivers prefer the vertices $\{i, i+1, \ldots, n\}$. Alternatively, if such a set exists, then the drivers cannot all park because they cannot park before their preferred vertex. Hence, we may alternatively define parking functions:

Definition 2. A *classical parking function* of length n is a sequence $s \in [n]^n$ such that for $i \in [n]$,

$$|\{j : s_j \geq i\}| \leq n - i + 1.$$

From this definition, it is immediately obvious that the order in which the drivers park does not affect their ability to do so. This is a useful property that will be preserved when we generalize to digraphs.

There are several bijections between parking functions of length n and trees with vertex set $[n+1]$ (see [19] for one involving a tree's Prüfer code), which means there are $(n+1)^{n-1}$ parking functions of length n.

One studied statistic on parking functions is the number of drivers who park in their preferred parking space.

Definition 3. Let s be a parking function. Driver i is *lucky* if she parks in spot s_i.

Gessel and Seo [10] noted that lucky drivers in parking functions were equidistributed with *proper* vertices, vertices whose label is smaller than all of their descendants, in rooted trees.

We will also consider special types of parking functions.

Definition 4 (Increasing and Prime Parking Functions)**.** A parking function $s \in [n]^n$ is *increasing* if $s_i \leq s_{i+1}$ for $1 \leq i \leq n-1$.

We say $p \in [n]^n$ is a *prime* parking function if, for all $2 \leq i \leq n$, we have

$$|\{j : s_j \geq i\}| < n - i + 1.$$

Notice the similarities in the definitions of regular and prime parking functions. Prime parking functions represent those for which too many drivers attempt to park in the vertices labeled $[k]$ for $k < n$. There are $(n-1)^{n-1}$ prime parking functions of length n and C_n-many increasing parking functions (see [18], problem 5.49).

Parking functions are intimately connected to Dyck paths. As their count suggests, increasing

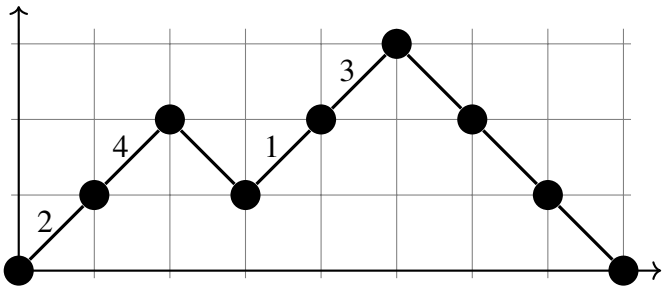

Figure 2.8: A labeled Dyck path

parking functions are in bijection with Dyck paths. One bijection is, given a Dyck path, the number of drivers preferring i is the number of N-steps immediately before the i^{th} S-step. For example, the Dyck path in Figure 2.6 corresponds to the parking function $s = 1122$. Classical parking functions are in bijection with Dyck paths whose runs of N-steps are labeled by subsets of $[n]$. Given a parking function s', the corresponding Dyck path has a run of $|\{i : s'_i = j\}|$ N-steps labeled with the set $\{i : s'_i = j\}$ immediately before the j^{th} S-step. For example, Figure 2.8 corresponds to the parking function $s' = 2121$. Finally, prime parking functions are those whose associated labeled Dyck path is prime.

The final parking function variant we must consider is one in which $m \leq n$ drivers attempt to park. This causes no problems with parking and only requires a slight change to the definition of a parking function:

Definition 5 (Parking Functions)**.** An (n, m)-*parking function* is a sequence $s \in [n]^m$ such that for $i \in [n]$,

$$|\{j : s_j \geq i\}| \leq n - i + 1.$$

If $m = n$, we will suppress the (n, n) in most cases.

2.3.2 Tree Parking Functions

Lackner and Panholzer were first to study parking functions on sink trees [14]. For sink trees, since the outdegree of each vertex is at most one, the process described in Definition 6 is still well-

defined. Thus, we say $s \in [n]^m$ is an (n, m)-*parking function on sink tree* T if the parking process works on T.

Let F_n be the number of parking functions on all sink trees with $|T| = n$ and define the formal power series $F(x) = \sum_{n\geq 1} F_n \frac{x^n}{n!}$. Then the authors of [14] showed that $F(x)$ satisfied

$$F(x) = T(2x) + \ln\left(1 - \frac{T(2x)}{2}\right), \tag{2.2}$$

where $T(x) = \sum_{n\geq 1} n^{n-1}\frac{x^n}{n!}$ is the tree function. Further, an explicit count of F_n was determined:

Theorem 2 ([14], Corollary 3.4)**.**

$$F_n = ((n-1)!)^2 \cdot \left(\sum_{i=0}^{n-1} \frac{(n-i)\cdot(2n)^i}{i!}\right).$$

2.4 Formal Power Series and a Useful Theorem

We introduce formal power series and state a formula useful for extracting coefficients.

2.4.1 Formal Power Series

For a function $f : \mathbb{N}_0 \to \mathbb{C}$ we associate the formal power series $F(x) = \sum_{n\geq 0} f(n)x^n$, called the *ordinary generating function* of f. Rather than using the functional notation, we may write $f(n) = f_n = [x^n]F(x)$, depending on which is convenient. If we use generating functions to count an object, sometimes adjustment in the generating function proves useful. We say $\hat{F}(x) = \sum_{n\geq 0} f_n \frac{x^n}{n!}$ is the *exponential generating function* of f. While factors other than $1/n!$ appear, they are significantly less common.

2.4.2 Lagrange Inversion

We state here a theorem useful for extracting coefficients in some situations.

Theorem 3 (Lagrange Inversion Formula)**.** *Let $F(x)$ and $G(x)$ be ordinary generating functions*

such that $F(x) = xG(F(x))$ *and* $G(0) \neq 0$. *Then for* $n \geq 1$,

$$[x^n]F(x) = \frac{1}{n}[x^{n-1}]G(x)^n.$$

3. PARKING FUNCTIONS ON DIGRAPHS[1]

This section is an expansion of Section 2 of [11].We begin by giving a general definition that is similar to Definition 1, but allows for parking on all directed graphs, rather than only those with maximum outdegree 1. Let drivers park according to the following process:

Definition 6 (Parking Process)**.** Pick n, m such that $0 \leq m \leq n$. Let $s \in [n]^m$ and D be a digraph with vertex set $[n]$.The m drivers attempt to park according to the following process:

1) Beginning with $i = 1$, driver i begins at vertex s_i.

2) If the current vertex is unoccupied, the driver parks there. If it is occupied, the driver chooses a vertex in the out-neighborhood of the current one and drives there.

3) The driver repeats step 2) until she either parks, and the next driver enters, or is unable to find an available parking space, and the process terminates.

In general, the maximaum outdegree of a vertex in D is larger than 1, so drivers will have to choose along which edges to travel in their search for a parking spot. We will consider the situation in which drivers could work cooperatively in order to allow everyone to park.

Definition 7. For a sequence $s \in [n]^m$ and digraph D, we say that s is a *parking function on* D if it is possible for all m drivers to park following the parking process. If s is a parking function on D, we call the pair (D, s) an (n, m)-*parking function*.

Figure 3.1 gives an example of a parking function. Drivers 2 and 5 are the only drivers who may make a choice and all drivers can park as long as at least one of those drivers uses the edge $(1, 4)$ during parking.

While Definition 7 is clearly a generalization of the classical case, it is rather intractable for applications. We seek a more workable definition and thus define a quasiorder $\preceq_D$ on the vertices

[1]Reprinted with permission from "Parking Functions on Oriented Trees" by W. King and C. H. Yan, 2018. *Séminaire Lotharingien de Combinatoire*, 80B, Copyright 2018 by W. King and C.H. Yan.

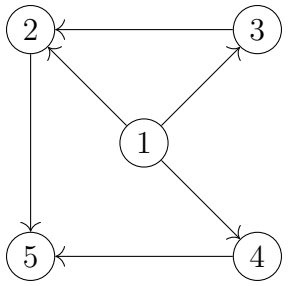

Figure 3.1: A digraph with parking function $s = (1, 1, 3, 2, 1)$.

of D by $v \preceq_D w$ if and only if there exists a directed path in D from v to w. We say by definition $v \preceq_D v$ and if $w \prec_D v$, then this means $w \preceq_D v$ but $w \neq v$. For each vertex v, we define the set of vertices *reachable from* v by

$$R_D(v) = \{w \in [n] : v \preceq_D w\}.$$

More generally, for $A \subseteq [n]$, we define the *reachable set* of A as

$$R_D(A) = \bigcup_{v \in A} R_D(v).$$

We can now give an alternative characterization of a parking function on D.

Theorem 4. *Let D be a digraph and $s \in [n]^m$. Let $\mathcal{C} = \{\mathcal{C}_1, \ldots \mathcal{C}_m\}$ be the set of cars, indexed such that $\mathcal{C}_i$ prefers spot s_i for each $i \in [n]$. Then s is a parking function on D if and only if for all $A \subseteq \mathcal{C}$ we have*

$$|A| \leq |\bigcup_{s_i : \mathcal{C}_i \in A} R_D(s_i)|.$$

Proof. If we let B be the bipartite graph with vertex set $\mathcal{C} \cup [n]$ where $\{\mathcal{C}_i, j\} \in E(B)$ if and only if $s_i \preceq_D j$, then by Hall's Theorem (Theorem 1), we are claiming that s is a parking function if and only if there exists a matching on B that saturates $\mathcal{C}$. If s is a parking function, then park the drivers in some manner and suppose $\mathcal{C}_i$ is parked on v_i for each i. Then, since we must have $s_i \preceq_D v_i$, the

edge $\{\mathcal{C}_i, v_i\}$ is in B. These edges define a matching.

The other direction is not as clear because the parking process requires drivers to park in the first empty spot they find. We use a matching $M = \{\{\mathcal{C}_i, v_i\}\}_{i=1}^m$ to determine a (not necessarily unique) way of parking on D.

The iterative process is the same for each $\mathcal{C}_i$, starting at $i = 1$. At step i, pick any x with $s_i \preceq_D x \preceq_D v_i$ such that there exists a path $s_i = y_0 \to y_1 \to \ldots \to y_k \to x$ such that the y_j's are spots occupied by cars in previous iterations for $0 \leq j \leq k < i$. If no such y_0 exists, then $x = s_i$. Park $\mathcal{C}_i$ in x and delete $\{\mathcal{C}_i, v_i\}$ from M. If $\{\mathcal{C}_j, x\} \in M$ for some $j > i$, then replace this edge with $\{\mathcal{C}_j, v_i\}$. Now repeat with $\mathcal{C}_{i+1}$.

In each step, the vertex x is the first unoccupied vertex along some walk between s_i and the vertex with which $\mathcal{C}_i$ is matched. At least one such x exists because, at the start of step i, $\mathcal{C}_i$ is matched with a vertex which is not occupied by any car. At the end of step i, we know the updated M is a matching saturating $\{\mathcal{C}_\ell\}_{\ell>i}$ because we know $s_j \preceq_D v_j = x$ from the edge $\{\mathcal{C}_j, v_j\}$ and $x \preceq_D v_i$ by our choice of x. Thus, $s_j \preceq_D v_i$, so the edge $\{\mathcal{C}_j, v_i\}$ is in B. □

Stated plainly, Theorem 4 says that matching all drivers with spots they could potentially park at (given their preferred spot) is both necessary and sufficient to conclude that (D, s) is a parking function. We can also characterize parking functions in terms of the number of drivers preferring vertices in the various reachable sets, $R_D(B)$, in the graph.

Corollary 1. *Let D be a digraph and $s \in [n]^m$. Then s is a parking function on D if and only if for all $B \subseteq [n]$ we have*

$$|\{\mathcal{C}_i : s_i \in R_D(B)\}| \leq |R_D(B)|.$$

Proof. Let s be a parking function on D and $B \subseteq [n]$. Let $A = \{\mathcal{C}_i : s_i \in R_D(B)\}$. Because s is a parking function, we know $|A| \leq |\bigcup_{s_i:\mathcal{C}_i\in A} R_D(s_i)|$, but also by the definition of $R_D(B)$, we have $\bigcup_{s_i:\mathcal{C}_i\in A} R_D(s_i) \subseteq R_D(B)$.

On the other hand, suppose for all $B \subseteq [n]$ we have $|\{\mathcal{C}_i : s_i \in R_D(B)\}| \leq |R_D(B)|$ and let

$A \subseteq \mathcal{C}$. Let $B = \bigcup_{\mathcal{C}_i \in A} \{s_i\}$. By definition $R_D(B) = \bigcup_{\mathcal{C}_i \in A} R_D(s_i)$ and so

$$|A| \leq |\{\mathcal{C}_i : s_i \in R_D(B)\}| \leq |R_D(B)| = |\bigcup_{\mathcal{C}_i \in A} R_D(s_i)|,$$

thus s is a parking function. □

From these set definitions, it is immediately clear that the order of the preference sequence s does not matter, a property shared with classical parking functions.

Corollary 2. *Let (D, s) be a parking function and $\sigma \in \mathfrak{S}_m$. Then $s_\sigma = (s_{\sigma(1)}, s_{\sigma(2)}, \ldots, s_{\sigma(m)})$ is also a parking function on D.*

Additionally, we can alternatively characterize the reachable sets in terms of the quasiorder to see how many exist.

Remark 1. *The number of distinct $R_D(B)$ is the same as the number of filters of the quasiorder $\preceq_D$, which is likely to be much less than 2^n.*

For example, there are 7 distinct $R_D(B)$ in Figure 3.1, which is indeed less than $2^5 = 32$.

We now give some definitions and notation that will be utilized throughout the subsequent sections. For a digraph D we define the number of parking functions on D to be given by

$$P(D, m) = |\{(D, s) : s \in [n]^m \text{ is a parking function on } D\}|.$$

Recall the two notions of classical parking functions given in Definition 4. In their spirits, we define a prime parking function and a parking distribution on a digraph D.

Definition 8 (Prime Parking Function)**.** A parking function (D, s) is called *prime* if, for every non-empty $A \subseteq [n]$ such that $R_D(A) \neq [n]$, we have

$$|\{C_i : s_i \in R_D(A)\}| < |R_D(A)|.$$

Definition 9 (Parking Distribution)**.** Let $f : [n] \to [m]_0$ such that $\sum_{i=1}^{n} f(i) = m$. Then we say (D, f) is a *parking distribution* if for all $A \subseteq [n]$, we have

$$\sum_{i \in R_D(A)} f(i) \leq |R_D(A)|.$$

For a parking distribution, $f(i)$ is understood to be the number of drivers preferring vertex i. We call them distributions, rather than increasing parking functions, for both precedential reasons [4] and to emphaize the distribution of driver preferences rather than the ordering of the preferences in a sequence.

4. PRIME PARKING FUNCTIONS ON SINK TREES[1][2]

This section is partially contained in [12] and is an expansion of Section 3 of [11]. In this section, we assume all trees are sink trees and all parking functions are (n, n)-parking functions. Parking functions on such trees were initially studied by Lackner and Panholzer [14] and serve as an appropriate launching pad for our investigation into digraph parking functions.

We let T_v be the maximal subtree rooted at v. For T, the maximal outdegree of a vertex is bounded above by 1, thus the order and location in which drivers park following the parking process of Definition 6 is well-defined. In the case of sink trees, it is convenient to characterize parking functions as those which have excess drivers preferring certain regions, rather than the "extra space" characterization seen in Corollary 1.

Proposition 1. *The pair (T, s) is a parking function if and only if for all $v \in [n]$, we have $|T_v| \leq |\{i : s_i \in T_v\}|$.*

Proof. If (T, s) is a parking function, then for $B_v = [n] \setminus V(T_v)$, by Corollary 1 we have

$$n - |\{i : s_i \in T_v\}| = |\{i : s_i \in R_T(B_v)\}| \leq |R_T(B_v)| = |B_v| = n - |T_v|.$$

On the other hand, for any $B \subseteq [n]$, the digraph induced by $[n] \setminus R_T(B)$ is a forest with roots $\{\rho_i\}_{i=1}^r$. Then

$$n - |\{i : s_i \in R_T(B)\}| = \sum_{j=1}^{r} |\{i : s_i \in T_{\rho_j}\}| \geq \sum_{j=1}^{r} |T_{\rho_j}| = n - |R_T(B)|.$$

□

[1]Reprinted with permission from "Prime Parking Functions on Rooted Trees" by W. King and C. H. Yan, 2019. *Journal of Combinatorial Theory, Series A*, 1-25, Copyright 2019 by Elsevier. This version is made available under the CC-BY-NC-ND 4.0 license http://creativecommons.org/licenses/by-nc-nd/4.0/

[2]Reprinted with permission from "Parking Functions on Oriented Trees" by W. King and C. H. Yan, 2018. *Séminaire Lotharingien de Combinatoire*, 80B, Copyright 2018 by W. King and C.H. Yan.

Thus, to check if (T, s) is a parking function, we need only check n inequalities, an improvement over those given in Theorem 4 and Corollary 1. Re-stating Definition 8 in these terms gives

Definition 10. A parking function (T, s) is *prime* if, for every non-root vertex $v \in [n]$, we have $|T_v| < |\{i : s_i \in T_v\}|$.

Figure 4.1 gives an example of a prime parking function.

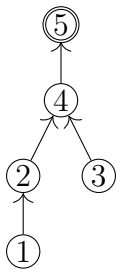

Figure 4.1: T with prime $p = (1, 3, 2, 3, 1)$.

We can understand $|T_v| < |\{i : s_i \in T_v\}|$ as an excess of drivers wanting to park in T_v. This means some driver *must* leave T_v during the parking process, so let us say

Definition 11. For a parking function (T, s), we say that an edge e is *used* by s if there exists some driver who, after failing to park at her preferred spot, crosses e during her search for an unoccupied spot.

Figure 4.2 shows a classical parking function in which the used edges are solid and the unused edges are dotted.

Figure 4.2: Solid edges are used by $s = (1, 3, 4, 4, 1)$

By considering the conditions under which an edge is used, we arrive at another characterization of prime parking functions.

Proposition 2. *Given a parking function (T, s), an edge $e = (u, v)$ is used by s if and only if $|T_u| < |\{i : s_i \in T_u\}|$. Thus, (T, s) is prime if and only if all edges are used by s.*

Proof. Suppose $e = (u, v)$ is used by s. Then at least one driver preferring T_u does not park in T_u. No cars preferring a vertex outside T_u can park inside, as T_u consists of all vertices w such that $w \preceq_T u$. The pair (T, s) is a parking function, so it follows that $|T_u| < |\{i : s_i \in T_u\}|$. On the other hand, if $|T_u| < |\{i : s_i \in T_u\}|$, then as s is a parking function on T, at least one driver preferring T_u must park outside. This driver must cross e in order to do so. □

We wish to arrive at a result similar to Theorem 2 for prime parking functions, so we define the set

$$\mathcal{PF}_n = \{(T, p) : |T| = n \text{ and } p \text{ is a prime parking function on } T\},$$

and we set $P_n = |\mathcal{PF}_n|$. We give two proofs to the following theorem:

Theorem 5. *The total number of prime tree parking functions P_n for $n \geq 1$ is given by*

$$P_n = (2n - 2)!$$

4.1 A Proof of Theorem 5 via Generating Functions

Recall Theorem 2. We define the formal power series

$$F(x) = \sum_{n \geq 1} F_n \frac{x^n}{(n!)^2} \qquad \text{and} \qquad P(x) = \sum_{n \geq 1} P_n \frac{x^n}{(n!)^2},$$

For a parking function (T, s), we consider a decomposition of T into a "core" component supporting a prime parking function, $(T_0, s^{(0)})$, and some collection of general parking functions, $(T_i, s^{(i)})$, attached to the core component. Let T_0 be the subtree of T containing the root and all vertices connected to the root via edges used by s. Let $s^{(0)}$ be the subsequence of s defined by

drivers preferring vertices in T_0. Notice that $s^{(0)}$ is a prime parking function by construction. The other subtrees T_i are the connected components remaining after deleting edges (u, v) unused by s, where $v \in V(T_0)$ and $u \notin V(T_0)$, and $s^{(i)}$ are the subsequences of s consisting of drivers preferring vertices in T_i.

Figure 4.3 gives a general overview of this decomposition. Dashed edges are those unused by s but connected to the "core" component T_0. In Figure 4.2, the "core" component is the subtree induced by vertex set $\{4, 5\}$ while $s^{(0)} = (4, 4)$. The one other component is the subgraph induced by vertices $\{1, 2, 3\}$ (identical to $\mathcal{P}_3$) with $s^{(1)} = (1, 3, 1)$. Notice that $(1, 3, 1)$ is not a prime parking function on $\mathcal{P}_3$.

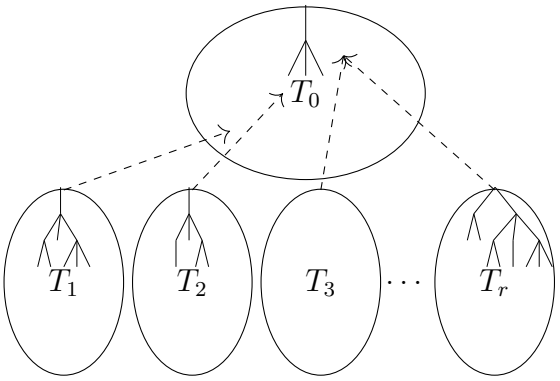

Figure 4.3: Decomposition into components.

In this way, we can construct any parking function (T, s) with $|T| = n$ by choosing a prime parking function $(T_0, s^{(0)})$ with $|T_0| = k_0$ and r-many other regular parking functions $\{(T_i, s^{(i)})\}_{i=1}^{r}$ with $k_i = |T_i| \geq 1$ and $\sum_{i=0}^{r} k_i = n$. From there, we can attach each T_i to T_0 in one of k_0-many places to form T. What remains is to choose the labels on T and choose which indices in s each $s^{(i)}$ is assigned. The $1/r!$ accounts for the order in which the T_i are chosen and attached. This means

$$F_n = \sum_{r\geq 0} \frac{1}{r!} \sum_{\sum_{i=0}^{r} k_i = n} P_{k_0} F_{k_1} \cdots F_{k_r} \binom{n}{k_0, k_1, \ldots, k_r}^2 (k_0)^r.$$

Since $F_1 = 1$, summing over $n \geq 1$, we get the relationship

$$F(x) = P\left(xe^{F(x)}\right). \tag{4.1}$$

Using (2.2),

$$P\left(xe^{F(x)}\right) = T(2x) + \ln\left(1 - \frac{T(2x)}{2}\right). \tag{4.2}$$

Setting $z = z(x) = xe^{F(x)}$, $y = y(x) = \frac{T(2x)}{2}$, and using the relation $T(x) = xe^{T(x)}$, we notice from Equation (2.2) that

$$z = y(1 - y).$$

Solving the quadratic equation gives

$$y = zC(z),$$

where $C(x) = \sum_{n\geq 0} C_n x^n$ is the ordinary generating function for the Catalan numbers with analytic expression

$$C(x) = \frac{1 - \sqrt{1 - 4x}}{2x}.$$

Since $z(0) = 0$ and $z_1 \neq 0$, the formal power series $z(x)$ has a compositional inverse. Rewriting Equation (4.2) and plugging in the inverse, we get

$$P(x) = 2xC(x) + \ln(1 - xC(x)).$$

The Catalan generating function $C(x)$ satisfies the recursion $C(x) = 1 + xC(x)^2$ and thus $C'(x)$

satisfies

$$C'(x) = \frac{C(x)^2}{1 - 2xC(x)}.$$

After some algebra, we can see

$$\frac{C(x)}{(xC(x))'} = \frac{1 - 2xC(x)}{1 - xC(x)}. \tag{4.3}$$

Then taking the derivative of $P(x)$ and using (4.3), we have

$$\begin{aligned} P'(x) &= 2(xC(x))' - \frac{(xC(x))'}{1 - xC(x)} \\ &= (xC(x))' \left(\frac{1 - 2xC(x)}{1 - xC(x)} \right) \\ &= C(x). \end{aligned}$$

Therefore,

$$P(x) = \sum_{n \geq 1} \frac{C_{n-1}}{n} x^n = \sum_{n \geq 1} (2n-2)! \frac{x^n}{(n!)^2}.$$

Hence, $P_n = (2n-2)!$ as claimed. Such a simple number demands a bijective proof, which we give in the next section.

4.2 A Bijective Proof of Theorem 5

In order to determine P_n, we define a bijection on prime parking functions $\psi : (T, p) \mapsto (\sigma, P)$ where $\sigma \in \mathfrak{S}_n$ and P is an ordered tree with n vertices whose non-root vertices are labeled by $[n-1]$. An ordered tree, also called a plane tree, is a rooted tree for which siblings are given a linear order.

To describe ψ, we first give a definition. Recall post-order labeling: given an ordered tree $\mathcal{T}$, travel around the left border of the tree starting from the root, labeling in increasing order as one

reaches the right side of a vertex. See the final picture in Figure 4.4 for a tree labeled by post-order.

Definition 12. For an ordered tree $\mathcal{T}$ with $|\mathcal{T}| = n$ and sequence $p \in [n]^n$, we say the pair $(\mathcal{T}, p)$ is a *standardized restricted prime parking function* if

1. $(\mathcal{T}, p)$ is a prime parking function when $\mathcal{T}$ is considered an unordered tree.
2. For any pair of sibling vertices, $\{u, v\} \subseteq V(\mathcal{T})$, the vertex v is ordered to the right of u if and only if the edge (v, w), where w is the parent vertex, is used by some driver before the edge (u, w) is used during the parking procedure.
3. $\mathcal{T}$ is labeled via post-order.

We denote the set of standardized restricted prime parking functions on n vertices by $\mathcal{SRPF}_n$. If $\mathcal{T}$ is an ordered tree, we say $(\mathcal{T}, p)$ is a prime parking function when the pair is a parking function if the ordering on $\mathcal{T}$ is forgotten.

The bijection ψ is constructed from two bijections, $\phi : (T, p) \mapsto (\sigma, (T_\sigma, p_\sigma))$ where $(T_\sigma, p_\sigma) \in \mathcal{SRPF}_n$, and $\alpha : (T_\sigma, p_\sigma) \mapsto P$, where P is the aforementioned ordered tree with non-root vertices labeled by $[n-1]$. The unlabeled ordered trees on n vertices are counted by the Catalan number C_{n-1}, so including the labellings there are $C_{n-1}(n-1)!$-many such P. Combining these two steps, we conclude that $P_n = n!C_{n-1}(n-1)! = (2n-2)!$.

4.2.1 The Bijection ϕ

We prove

Proposition 3. *For $n \geq 1$, we have*

$$P_n = n!|\mathcal{SRPF}_n|.$$

Proof. Let $(T, p) \in \mathcal{PF}_n$. We induce an ordering on T by using p. Since the parking function is prime, every edge must be crossed by some driver after failing to park at her preferred spot. Additionally, since there is only one path a driver may travel in search of a spot and because

drivers must park at the first available spot they encounter, the order in which edges are initially crossed is well-defined. Then, consider T as an ordered tree by ordering vertices to match property 2 in Definition 12.

For a $\sigma \in \mathfrak{S}_n$, let T_σ be the tree obtained by relabeling the vertices of T, $v \mapsto \sigma(v)$. Likewise, let $p_\sigma = (\sigma(p_1), \sigma(p_2), \ldots, \sigma(p_n))$. Then we define

$$\phi((T, p)) = (\sigma, (T_\sigma, p_\sigma)),$$

where σ is the unique permutation such that T_σ is labeled by post-order. This means $(T_\sigma, p_\sigma) \in \mathcal{SRP}_n$.

Since any two relabellings of an ordered tree are distinct, for $(\widetilde{T}, \widetilde{p}) \in \mathcal{SRP}_n$, the preimage $\phi^{-1}((\sigma, (\widetilde{T}, \widetilde{p}))$ is the prime parking function $(\widetilde{T}_{\sigma^{-1}}, \widetilde{p}_{\sigma^{-1}})$, where the ordering on the tree is forgotten. □

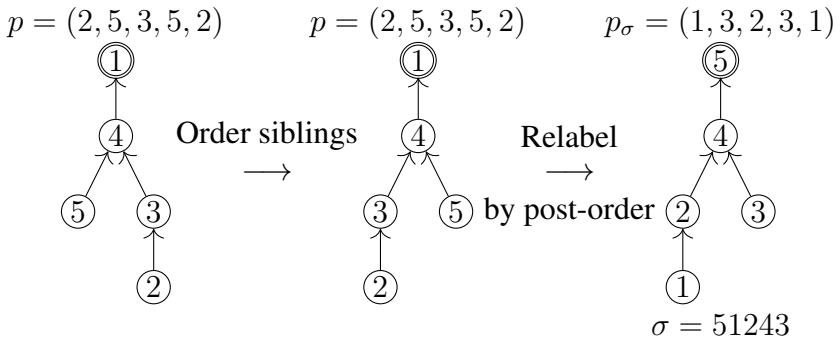

Figure 4.4: An example of ϕ.

Figure 4.4 shows an application of ϕ. In the left tree, the edge $(5, 4)$ is first used by the fourth driver, while the edge $(3, 4)$ is not used until the fifth driver. Thus, the vertex 5 is placed to the right of vertex 3, which gives the ordered tree in the middle of the figure. The right tree is obtained

from the middle one by relabeling via post-order, which gives the permutation σ. We now turn our attention to $|\mathcal{SRP}_n|$.

4.2.2 The Bijection α

We define $\alpha : \mathcal{SRP}_n \to \{\text{ordered trees with non-root vertices labeled by } [n-1]\}$. The main observation necessary for the construction of α is a decomposition of $(\mathcal{T}, p) \in \mathcal{SRP}_n$ into an ordered collection of components, each a relabeling of a standardized restricted prime parking function, based on the edges used by the first $n-1$ drivers. We use $\mathcal{T}$ to emphasize that $\mathcal{T}$ is an ordered tree. First, we give a proposition about where the final driver must park for prime parking functions, which also applies to parking functions in $\mathcal{SRP}_n$.

Proposition 4. *Let $(T, p) \in \mathcal{PF}_n$. Then the final driver parks at the root node.*

Proof. Let ω be the vertex the final driver parks at. If ω is not the root, it has a parent vertex v. Since drivers must park at the first empty vertex they arrive at, the edge (ω, v) can not be used by any driver prior to the final one, since ω is unoccupied. Since the final driver also does not use (ω, v), it remains unused. However, p is a prime parking function on T, so all edges must be used. Thus, there can be no edge (ω, v), and so ω is the root of T. □

We first describe the recursive construction of α, then prove it is a bijection. Begin with $(\mathcal{T}, p) \in \mathcal{SRP}_n$. We use the tree in Figure 4.5 as a running example.

Base Case. If $\mathcal{T}$ is a singleton, then $\alpha((\mathcal{T}, p))$ is an unlabeled singleton as $|\mathcal{SRP}_1| = 1$.

Step 1. Park all except the final driver, highlighting edges as they are used. Delete the non-highlighted edges and the root, marking the terminal vertex of any edge deleted and the vertex with label p_n.

Since p is a prime parking function on $\mathcal{T}$, the non-highlighted edges must lie on the path $\mathcal{P}$ between the vertex labeled p_n and the root. The highlighted edges define some collection of subtrees $\{\mathcal{T}_i\}_{i=1}^{r}$, linearly ordered by the order in which they are a part of $\mathcal{P}$. Since the root is always isolated, as the final driver is the only one to cross the edge connected to the root, we may ignore it. Let $p^{(i)}$ be the subsequence of $(p_1, \ldots, p_{n-1})$ consisting of all p_j such that $p_j \in V(\mathcal{T}_i)$.

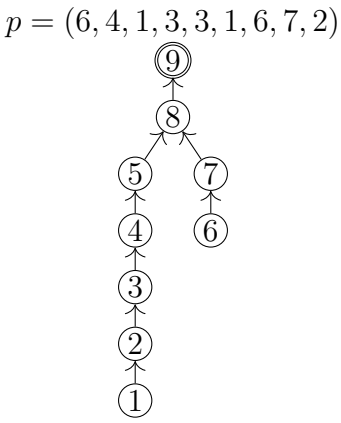

Figure 4.5: A $(\mathcal{T}, p) \in \mathcal{SRP}_n$.

By construction, $p^{(i)}$ is a prime parking function on $\mathcal{T}_i$ satisfying conditions 1 and 2 in Definition 12. See Figure 4.6 for this step applied to our running example. The non-highlighted edges are dotted and the marked vertices are shaded.

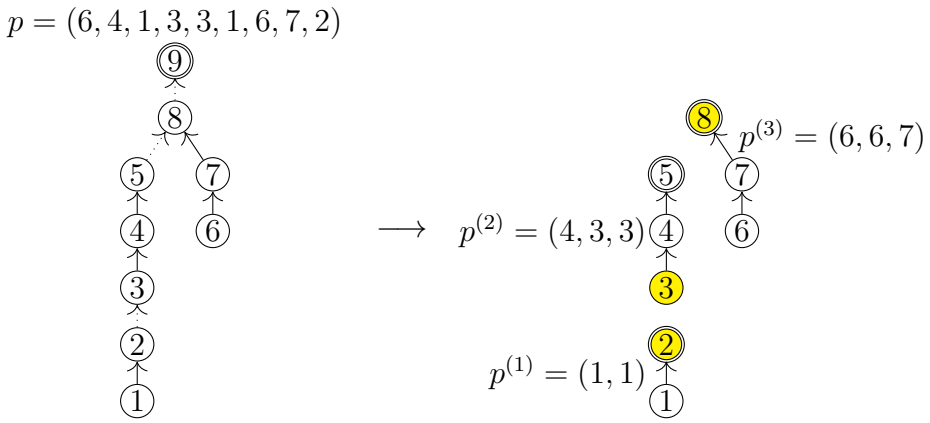

Figure 4.6: Step 1, decomposing.

Step 2. For each $(\mathcal{T}_i, p^{(i)})$, let $A_i = \{j \in [n-1] : p_j \in V(\mathcal{T}_i)\}$. For $1 \leq i \leq r$, if the marked vertex on $\mathcal{T}_i$ has the k^{th} smallest label among vertices in $\mathcal{T}_i$, mark the k^{th} smallest element in A_i.

Notice that the elements of A_i are precisely those j such that p_j appears in $p^{(i)}$. The marked

vertices and elements track how the $\mathcal{T}_i$ are connected to each other in $\mathcal{T}$. Figure 4.7 shows this step. The elements of A_i are represented by C_j instead of j to emphasize that j is an index from p, rather than a vertex of $\mathcal{T}$.

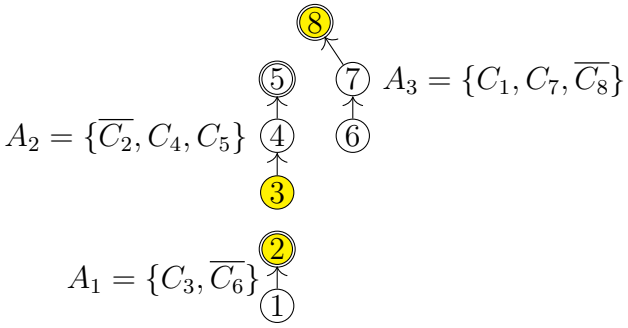

Figure 4.7: Step 2, tracking p and the shape of $\mathcal{T}$.

Step 3. Relabel each $(\mathcal{T}_i, p^{(i)})$ so that its labels are in post-order, meaning $(\overline{\mathcal{T}}_i, \overline{p}^{(i)}) \in \mathcal{SRP}_n$. Apply α to each $(\overline{\mathcal{T}}_i, \overline{p}^{(i)})$.

In Figure 4.8, we show components after they have been relabeled and apply α to each.

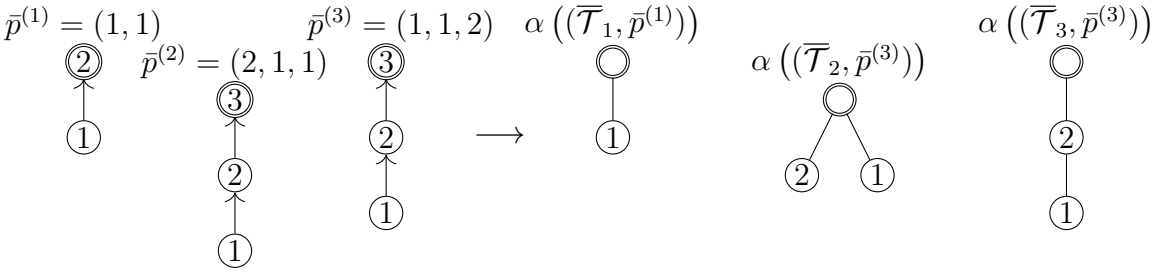

Figure 4.8: Step 3, applying α on the components.

Step 4. For each ordered tree $\alpha((\overline{\mathcal{T}}_i, \overline{p}^{(i)}))$, label the root with the marked element of A_i and

relabel the rest of the vertices with the unmarked elements of A_i, preserving relative ordering. Denote these trees by $\{P_i\}_{i=1}^{r}$. Attach their roots to an unlabeled vertex and arrange the subtrees so that the subtree P_i is to the left of P_j if $i < j$. This is $\alpha((\mathcal{T}, p))$.

Figure 4.9 shows the relabeling and Figure 4.10 shows the final result of the running example. We constructed α to prove the following lemma.

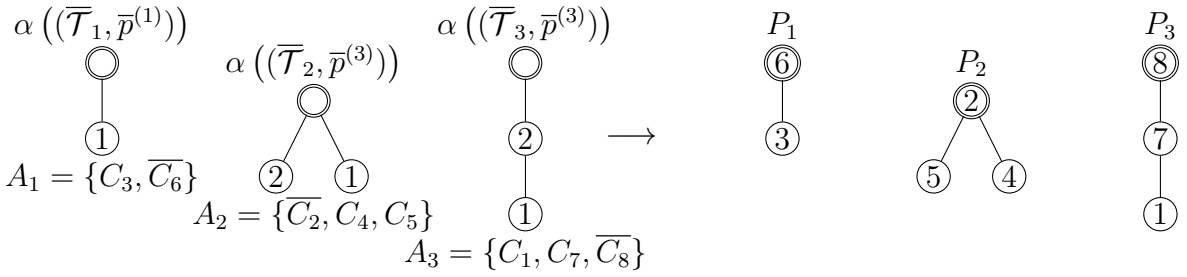

Figure 4.9: Step 4, relabeling the trees from Figure 4.8 with the sets from Figure 4.7.

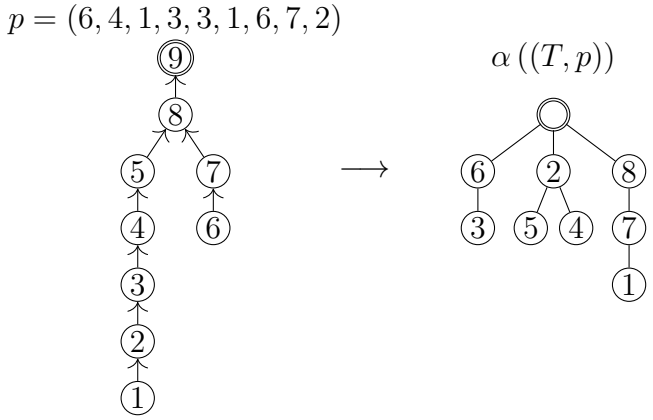

Figure 4.10: The result of α.

Lemma 1. *For* $n \geq 1$

$$|\mathcal{SPF}_n| = C_{n-1}(n-1)!$$

Proof. There are $C_{n-1}(n-1)!$ ordered trees with n vertices with non-root vertices labeled by $[n-1]$. Therefore, it is sufficient to show that α is a bijection, which we do inductively. In the case that $|T| = 1$, the only pair is $(T, 1)$ and $\alpha((T, 1))$ is an unlabeled singleton.

Now suppose α is a bijection for all parking functions in $\mathcal{SRP}_k$ with $k < n$ and let $(\mathcal{T}, p) \in \mathcal{SRP}_n$. Run the parking procedure on T for all but the final driver. Since p is a prime parking function, all edges except for some on the path $\mathcal{P}$ from p_n to the root have been used. Deleting these unused edges creates a collection of trees which are naturally ordered by the order their vertices appear on $\mathcal{P}$. Since we know the root component is a singleton by Proposition 4, we may ignore it and label the other components $\{\mathcal{T}_i\}_{i=1}^r$ for some r.

We claim that $\mathcal{T}_i$ has smallest vertex label $1 + \sum_{j=1}^{i-1} |\mathcal{T}_j|$ and largest label $|\mathcal{T}_i| + \sum_{j=1}^{i-1} |\mathcal{T}_j|$. Further, if $\sum_{j=1}^{i-1} |\mathcal{T}_j|$ is subtracted from each vertex, the resulting tree will be labeled by post-order. This means if we consider the subsequence of p, denoted $p^{(i)}$, consisting of drivers preferring $\mathcal{T}_i$, the pair $(\mathcal{T}_i, p^{(i)})$ is a relabeling of a parking function in $\mathcal{SRP}_{|\mathcal{T}_i|}$.

Proof of claim. Because the edge leaving the root of every $\mathcal{T}_i$ is not used until the very last driver and $(\mathcal{T}, p) \in \mathcal{SRP}_n$, the roots of the $\mathcal{T}_i$'s, denoted $\{\rho_i\}_{i=1}^r$, must lie on the left border of the tree.

By post-order labeling, a vertex is not labeled before all vertices below and all of its left siblings are given a label. Since all vertices below ρ_1 belong to $\mathcal{T}_1$, all other vertices of $\mathcal{T}_1$ are labeled before ρ_1. No vertex in $\mathcal{T}_2$ is labeled before ρ_1 since ρ_1 is on the left border of the tree, to the left of any of its siblings. Hence, $\mathcal{T}_1$ is labeled first and is in post-order.

In general, as ρ_{i-1} is on the left border of $\mathcal{T}$, the vertices below ρ_{i-1} are labeled before any vertex in $\mathcal{T}_i$. Thus, $\mathcal{T}_i$ has vertices labeled $1 + \sum_{j=1}^{i-1} |\mathcal{T}_j|$ to $|\mathcal{T}_i| + \sum_{j=1}^{i-1} |\mathcal{T}_j|$. Subtracting $\sum_{j=1}^{i-1} |\mathcal{T}_j|$ from each vertex is the same as labeling, via post-order, the maximal subtree with root ρ_i and vertices below ρ_{i-1} (inclusive) deleted, as this deletion removes a branch on the left side of the tree that is labeled before any other vertex. □

Let $A_1, \ldots, A_r$ be a partition of $[n-1]$ such that $j \in A_\ell$ if and only if p_j is a vertex in $\mathcal{T}_\ell$. A_i is the set of indices of the drivers preferring the component $\mathcal{T}_i$, meaning $|A_i| = |\mathcal{T}_i|$. For each deleted edge (u, v), except for when v is the root, if v is the k^{th} smallest vertex in its component, $\mathcal{T}_j$, mark

the k^{th} smallest element in A_j. For A_1, let $v = p_n$. These marked elements track both the final driver's preference and how to reconstruct the tree from its components. The collection $\{A_i\}_{i=1}^r$ partitions $[n-1]$ and will become the set of labels on the resulting ordered tree.

Relabel $\{(\mathcal{T}_i, p^{(i)})\}_{i=1}^r$ by post-order (notice this is the same as subtracting $\sum_{j=1}^{i-1} |\mathcal{T}_j|$ from $v \in V(\mathcal{T}_i)$), denoting them $\{(\overline{\mathcal{T}}_i, \overline{p}^{(i)}\}_{i=1}^r$. Use the inductive hypothesis to obtain the trees $\{\alpha\left((\overline{\mathcal{T}}_i, \overline{p}^{(i)})\right)\}_{i=1}^r$. Label the root of $\alpha\left((\overline{\mathcal{T}}_i, \overline{p}^{(i)})\right)$ with the marked vertex of A_i, then relabel the remaining vertices with the unmarked elements of A_i, preserving relative order. This is possible because $|A_i| = |\mathcal{T}_i| = |\alpha\left((\overline{\mathcal{T}}_i, \overline{p}^{(i)})\right)|$. Attach the roots of these trees to an unmarked vertex and order them so that the tree using the labels in A_i is i^{th}-from-the-left. This tree is $\alpha\left((\mathcal{T}, p)\right)$.

To reverse, let P be an ordered tree with non-root vertices labeled by $[n-1]$. Deleting the root yields several components, denoted left-to-right as P_i for $1 \leq i \leq r$. The set A_i is the set of labels of P_i where the root of P_i is the marked element. Let k_i denote the relative size of the marked element in A_i. For each, delete the root's label and relabel using $[|P_i| - 1]$, preserving relative order, and apply α^{-1} to get the collection $\{(\overline{\mathcal{T}}_i, \overline{p}^{(i)})\}_{i=1}^r$. To each $v \in \overline{\mathcal{T}}_i$ and letter of $\overline{p}^{(i)}$, add $\sum_{j=1}^{i-1} |\overline{\mathcal{T}}_j|$ to recover the pairs $(\mathcal{T}_i, p^{(i)})$.

Then for $1 \leq i \leq r-1$, attach the root of $\mathcal{T}_i$ to the vertex labeled $k_i + \sum_{j=1}^{i-1} |\mathcal{T}_j|$, placing it to the left of any siblings. These vertices are those marked in Step 1 of the description of α. Attach the root of $\mathcal{T}_r$ to a singleton with label n, the root of $\mathcal{T}$. Let the sequence $\{i_j\}$ be the increasing sequence of the elements of A_i. Then let $p_{i_j} = p_j^{(i)}$ to recover p. □

Finally, we combine results to prove Theorem 5.

Proof of Theorem 5. Let (T, p) be a prime parking function, if $\phi((T, p)) = (\sigma, T_\sigma, p_\sigma)$, we define

$$\psi(T, p) = (\sigma, \alpha((T_\sigma, p_\sigma)).$$

The function ψ is a bijection by Proposition 3 and Lemma 1, so we conclude that

$$P_n = n!|\mathcal{SRP}_n| = n! \cdot (n-1)! \cdot C_{n-1} = (2n-2)!.$$

□

4.3 Preimages of Paths Under α

We investigate special families of trees and parking functions of interest.

4.3.1 Preimage of Paths

We study what kinds of parking functions (T, p) appear under α^{-1} when we restrict the domain to trees which are paths. Such paths will have $n+1$ nodes with the non-root nodes labeled by $[n]$.

Definition 13. We denote the paths in the domain of α^{-1} by P_σ, where $\sigma \in \mathfrak{S}_n$ is the permutation obtained by reading labels traveling away from the root.

For a tree of size $n+1$, there are $n!$ such paths. Recall that $\mathcal{P}_n$ is the path of n vertices upon which classical parking functions are defined.

Proposition 5. *Let $(\mathcal{P}_{n+1}, s)$ be a parking function satisfying $s_1 = 1$ and $s_i \leq i-1$ for $i \geq 2$. Then $\alpha(\mathcal{P}_{n+1}, s)$ is one of the $n!$ paths with non-root vertices labeled by $[n]$.*

Proof. That s is prime is easily checked. Since $s_1 = 1$, we may delete s_1 and consider $s' = (s_2, s_3, \ldots, s_{n+1})$. Since $s'_i = s_{i+1} \leq i$, we have $|\{i : s'_i \leq k\}| \geq k$ for any $k \in [n]$, so s' is a parking function on $\mathcal{P}_n$. Further, s'_i may be one of i-many choices, so there are $n!$-many s', and thus s.

Park the drivers in order. The first driver takes spot 1 and since $s_2 = 1$, the second driver takes spot 2, crossing the edge from 1 to 2. Next, since $s_2 \leq 2$, and spots 1 and 2 are taken, the third driver takes spot 3, crossing the edge from 2 to 3. Continuing this, the i^{th} driver always parks at, but never prefers, spot i. When the final driver parks, all spots except for $n+1$ are filled and every edge has been used except the one between n and $n+1$. Thus, by the construction of α, the root of $\alpha(s, \mathcal{P}_{n+1})$ has one child. The inductive step in the proof of Lemma 1 tells us that the shape of the tree obtained by deleting the root from $\alpha(s, \mathcal{P}_{n+1})$ is the same as that of $\alpha\left((1, s_1, \ldots, s_{n-1}), \mathcal{P}_n\right)$. But $(1, s_1, \ldots, s_{n-1})$ is a prime parking function on $\mathcal{P}_n$ with the same growth property as s. Therefore, its root also has one child. Iterating this argument, we see that the

image of such parking functions under α consists of paths with non-root vertices labeled by $[n]$. Since α is a bijection and there are $n!$ choices for s, all $n!$-such paths must appear. $\square$

It will prove useful to have a characterization of the s_i in terms of the labels of $\alpha(\mathcal{P}_{n+1}, s)$.

Lemma 2. *Let $(\mathcal{P}_{n+1}, s)$ be a prime parking function such that $s_1 = 1$ and $s_i \leq i-1$ for $2 \leq i \leq n+1$ and let $P_\sigma = \alpha(\mathcal{P}_{n+1}, s)$ for $\sigma \in \mathfrak{S}_n$. Then for $2 \leq i \leq n+1$, we have*

$$s_i = |\{j > n+2-i : \sigma_j < \sigma_{n+2-i}\}| + 1.$$

Proof. By the construction of α in Lemma 1, we know $\sigma_1 = s_{n+1}$ and so σ_2 is the element in $[n] \setminus \{\sigma_1\}$ larger than exactly $s_n - 1$ others (the s_n^{th} smallest element). In general, σ_i is the s_{n+2-i}^{th} smallest element in $[n] \setminus \{\sigma_1, \sigma_2, \ldots, \sigma_{i-1}\}$. Thus, for $i \geq 2$, s_i is given by the relative size of σ_{n+2-i} in the set $[n] \setminus \{\sigma_1, \sigma_2, \ldots, \sigma_{n+1-i}\}$. We may write this as $s_i = |\{j > n+2-i : \sigma_j < \sigma_{n+2-i}\}| + 1$. $\square$

Example 1. Consider the classical prime parking function $p = (1, 1, 2, 3, 3, 1)$ and tree $\mathcal{P}_6$. That $s_i \leq i-1$ for $i \geq 2$ means the root of $\alpha((\mathcal{P}_6, p))$ has only one child. Further, as the final driver has preference 1, the tree $\alpha((\mathcal{P}_6, p))$ is constructed by labeling the root of $\alpha((\mathcal{P}_5, (1, 1, 2, 3, 3))$ with 1 and re-labeling the other nodes to preserve relative order.

To determine σ_2, we note that the root child of $\alpha((\mathcal{P}_5, (1, 1, 2, 3, 3)))$ is 3 because the final driver prefers 3. But for $\alpha((\mathcal{P}_6, p))$, we must adjust to account for the 1 used and in order to preserve relative order. Therefore, σ_2 is the 3rd smallest in $[6] \setminus \{1\}$, which is 4.

Continuing this trend, σ_3 is the label of the root child of $\alpha((\mathcal{P}_4, (1, 1, 2, 3)))$ adjusted to account for the labels 1 and 4 already having been used. The final driver prefers 3, so σ_3 is the third smallest in the set $[6] \setminus \{1, 4\}$, which is 5. Determining σ_i for $i \geq 4$ proceeds in a similar fashion.

Figure 4.11 presents $\alpha((\mathcal{P}_6, p))$ and $\alpha((\mathcal{P}_4, (1, 1, 2, 3)))$. Note that the leaf of $\alpha((\mathcal{P}_6, p))$ and its two generations of parents have the same relative order as the labeled vertices in $\alpha((\mathcal{P}_4, (1, 1, 2, 3)))$.

Of particular interest are the increasing parking functions that obey this growth restriction.

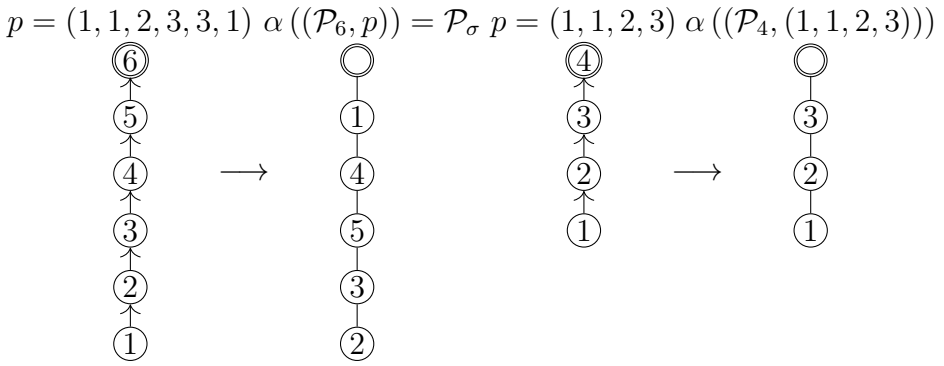

Figure 4.11: Applying α to several classical parking functions.

We now examine which labellings σ appear for $\alpha((\mathcal{P}_{n+1}, s))$ with s an increasing prime parking function.

4.3.2 Image of Classical Increasing Parking Functions

Borie [3] gives a bijection between $\mathfrak{S}_n(132)$, the permutations of length n which avoid a 132 pattern, and classical increasing parking functions of length n. Let $\sigma \in \mathfrak{S}_n(132)$. Define for $m \in [n]$:

$$f_\sigma(m) = |\{i : |\{j : j < i, \sigma_j > \sigma_i\}| \geq m\}|.$$

We briefly note that our $f_\sigma(m)$ is equivalent to the function $mmp(0, m, 0, 0)(\sigma)$ in [3], which we have shortened for readability. Set

$$\phi(\sigma) = (f_\sigma(n) + 1, f_\sigma(n-1) + 1, \ldots, f_\sigma(1) + 1).$$

Then $\phi(\sigma)$ is a classical increasing parking function and we have the following theorem due to Borie:

Theorem 6 (Theorem 3.3, [3])**.** *ϕ is a bijection between $\mathfrak{S}_n(132)$ and increasing parking functions of length n.*

We will show

Theorem 7. *For $\sigma \in \mathfrak{S}_n(132)$, $\phi(\sigma)$ is the parking function obtained after deleting the leading 1 from $\alpha^{-1}(P_\sigma)$.*

Proof. Fix $\sigma \in \mathfrak{S}_n(132)$ and let $(\mathcal{P}_{n+1}, p') = \alpha^{-1}(P_\sigma)$. By Proposition 5, we know we may write $p' = (1, p_1, p_2, \ldots, p_n)$, where $p = (p_1, p_2, \ldots, p_n)$ is a classical increasing parking function. For brevity, define for $m \in [n]$:

$$A_m = \{j : j > m \text{ and } \sigma_j < \sigma_m\},$$

and

$$\begin{aligned} B_m &= \{i : |\{j : j < i, \sigma_j > \sigma_i\}| \geq m\} \\ &= \{i > m : \sigma_i \text{ smaller than at least } m \text{ of } \{\sigma_1, \ldots, \sigma_{i-1}\}\}. \end{aligned}$$

Since $|B_m| = f_\sigma(m)$, we have $\phi(\sigma) = (|B_n| + 1, |B_{n-1}| + 1, \ldots, |B_1| + 1)$. From Lemma 2 we have $p_i = p'_{i+1} = |A_{n+1-i}| + 1$, so it is sufficient to show that $|A_m| = |B_m|$ for $m \in [n]$. We show the sets are the same.

Fix $m \in [n]$ and let $k \in A_m$. By definition, $m < k$ and $\sigma_k < \sigma_m$. For $i < m$, if $\sigma_i < \sigma_k$, then σ has the 132 pattern $\sigma_i \sigma_m \sigma_k$, which is not possible. Hence, $\sigma_k < \sigma_i$ for $1 \leq i \leq m$, so $k \in B_m$.

On the other hand, let $j \in B_m$. If $\sigma_j < \sigma_m$, then $j \in A_m$, so suppose $\sigma_j > \sigma_m$. Since $\sigma \in \mathfrak{S}_n(132)$, $\sigma_i < \sigma_j$ for $m + 1 \leq i \leq j - 1$. Thus, for indices smaller than j, only elements from $\{\sigma_1, \sigma_2, \ldots, \sigma_{m-1}\}$ may be larger than σ_j. However, $|\{\sigma_1, \sigma_2, \ldots, \sigma_{m-1}\}| = m - 1 < m$, contradicting that $j \in B_m$. Therefore $\sigma_j < \sigma_m$, so $j \in A$. □

4.4 Lucky Drivers

Recall Definition 3. For a parking function (T, s), let $L((T, s))$ be the number of lucky drivers. Define the enumerator

$$\text{PTL}_n(t) = \sum_{(T,p)\in\mathcal{PF}_n} t^{L((T,p))}.$$

We prove

Theorem 8. *When $n \geq 2$, the number of prime parking functions, (T,p), with $|T| = n$ and k lucky drivers is given by*

$$n!(n-1)!N_{n-1,k},$$

where $N_{n-1,k} = \frac{1}{n-1}\binom{n-1}{k}\binom{n-1}{k-1}$, a Narayana number.

Proof. For $n \geq 1$, let $(T,p) \in \mathcal{PF}_n$. We claim $L((T,p))$ is the number of leaves in the tree $\alpha((T_\sigma, p_\sigma))$, the second component of $\psi(T,p)$. When $n = 1$, there is precisely one prime parking function and one lucky driver and the only tree under α is a singleton, which has one leaf.

For $n \geq 2$, suppose the claim is true for $m < n$. Recall from Proposition 4 that the final driver must park at the root, but can not prefer the root, and thus is not lucky. Therefore, $L((T,p)) = \sum_{i=1}^{r} L((\overline{\mathcal{T}}_i, \overline{p}^{(i)}))$ where $\{(\overline{\mathcal{T}}_i, \overline{p}^{(i)})\}_{i=1}^{r}$ are the standardized parking functions from Step 3 in the description of α. Since $|\overline{\mathcal{T}}_i| < n$, the inductive hypothesis tells us that $L((\overline{\mathcal{T}}_i, \overline{p}^{(i)}))$ is the number of leaves in $\alpha((\overline{\mathcal{T}}_i, \overline{p}^{(i)}))$. The shape of $\alpha((T_\sigma, p_\sigma))$ is obtained by attaching the roots of the trees $\{\alpha((\overline{\mathcal{T}}_i, \overline{p}^{(i)}))\}_{i=1}^{r}$ to a new root, so $L((T,p))$ is indeed the number of leaves in $\alpha((T_\sigma, p_\sigma))$.

By Lemma 1, there are $(n-1)!$-many choices of parking functions (T', p') such that $\alpha((T', p'))$ has the same shape as $\alpha((T_\sigma, p_\sigma))$. Proposition 3 tells us there are $n!$-many choices of parking functions whose representative in $\mathcal{SRP}_n$ is (T_σ, p_σ). Finally, the Narayana number $N_{n-1,k}$ counts the number of ordered plane trees with n vertices and k leaves. □

Summing over k, we can determine the enumerator $\text{PTL}_n(t)$.

Corollary 3. *$PTL_1(t)$=t and for $n \geq 2$, we have*

$$\begin{aligned} PTL_n(t) &= n!(n-1)! \sum_{k=1}^{n-1} N_{n-1,k} t^k \\ &= n!(n-2)! \sum_{k=1}^{n-1} \binom{n-1}{k} \binom{n-1}{k-1} t^k. \end{aligned}$$

Using the parking function and corresponding plane tree in Figure 4.10, we note that $\alpha((T,p))$ has 4 leaves while the first four drivers are the four lucky drivers.

4.5 Parking Distributions

Following the decomposition in Section 4.1, we let $\widetilde{F}_n$ be the total number of parking distributions on trees with n vertices, and $\widetilde{P}_n$ be the corresponding number of prime parking distributions. Set

$$\widetilde{F}(x) = \sum_{n \geq 1} \widetilde{F}_n \frac{x^n}{n!} \qquad \text{and} \qquad \widetilde{P}(x) = \sum_{n \geq 1} \widetilde{P}_n \frac{x^n}{n!}.$$

Notice that these are exponential generating functions, unlike $F(x)$ and $P(x)$. Decomposing a parking distribution (T,s) into the "core" prime component and collection of r other components, as in Section 4.1, we get

$$\widetilde{F}_n = \sum_{r \geq 0} \frac{1}{r!} \sum_{\substack{r \\ \sum_{i=0} k_i = n}} \widetilde{P}_{k_0} \widetilde{F}_{k_1} \cdots \widetilde{F}_{k_r} \binom{n}{k_0, k_1, \ldots, k_r} (k_0)^r.$$

Summing over n, we find

$$\widetilde{F}(x) = \widetilde{P}\left(x e^{\widetilde{F}(x)}\right), \tag{4.4}$$

which is the same relationship for parking functions in Equation (4.1). We then turn our attention to prime parking distributions and prove

Theorem 9. *The total number of prime parking distributions on trees with $n \geq 1$ vertices is given*

by

$$\widetilde{P}_n = (n-1)!S_{n-1},$$

where $\{S_i\}_{i\geq 0}$ *are the large Schröder numbers.*

Proof. We let P_n^* be the total number of tuples (T, p, v), called *marked* prime parking distributions, where (T, p) is a prime parking distribution and v is a leaf of T. Define the exponential generating function

$$P^*(x) = \sum_{n\geq 1} P_n^* \frac{x^n}{n!}.$$

We count P_n^* in two ways in order to determine the coefficients of $\widetilde{P}(x)$. We may construct a marked prime parking distribution from a prime parking distribution (T, p) with $|T| = n-1$ by "growing" the marked leaf from some vertex in T. We choose one vertex w in T which has $j \geq 1$ drivers preferring it, select a label for the marked leaf v (out of n choices), add 1 to the label of any vertex with label greater than or equal to the label chosen for v, attach v as a child of w, add a driver preferring v, and change i drivers, for some $1 \leq i \leq j$, preferring w to prefer v instead. The number of choices we can make for v's parents and number of drivers whose preferences we change is $\sum_{w\in V(T)} |\{i : p_i = w\}| = n-1$.

On the other hand, any marked prime parking distribution can be changed to a regular prime parking distribution by deleting the marked vertex v, deleting one driver preferring it, and changing the preference of other drivers preferring v to instead prefer v's parent. Since the marked parking distribution is prime, at least two drivers prefer the marked leaf. Figure 4.12 has two examples where the shaded vertex is added and marked letters of p are drivers whose preference was changed.

This means for $n \geq 2$,

$$P_n^* = n(n-1)\widetilde{P}_{n-1},$$

so noting $P_0^* = 0$ and $P_1^* = 1$ and summing over n, we get

$$P^*(x) = x + x^2\widetilde{P}'(x). \tag{4.5}$$

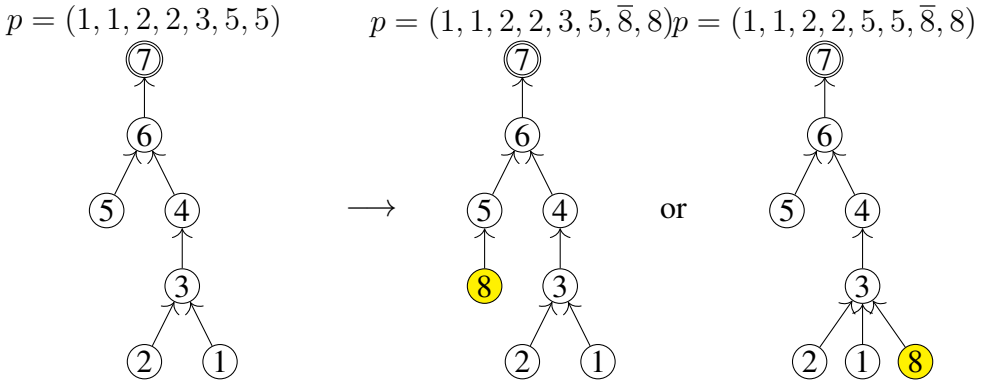

Figure 4.12: Two possibilities when adding the marked leaf.

For the second equation, we decompose a marked prime parking distribution (T, p, v) as in Section 4.2.2: park all drivers except for one preferring v. For a general picture, see Figure 4.13. Dashed edges denote those unused before the final driver. As before, the edges which have not yet been used define $r + 1$, for some $r \geq 0$, components of size k_i with prime parking functions, one of which is marked. Accounting for the edges connecting the components, the label of the root, and the labels on the components, we have for $n \geq 2$,

$$P_n^* = n \sum_{r \geq 0} \sum_{\sum k_i = n-1} \binom{n-1}{k_0, k_1, \ldots, k_r} P_{k_0}^* \widetilde{P}_{k_1} \cdots \widetilde{P}_{k_r} k_1 k_2 \cdots k_r,$$

so multiplying by $\frac{x^n}{n!}$ and summing over $n \geq 2$, we get

$$\sum_{n\geq 2} P_n^* \frac{x^n}{n!} = x \cdot \sum_{n\geq 2} \sum_{r\geq 0} \sum_{\sum k_i = n-1} \frac{P_{k_0}^*}{k_0!} \cdot \frac{\widetilde{P}_{k_1}}{(k_1-1)!} \cdots \frac{\widetilde{P}_{k_r}}{(k_r-1)!} x^{n-1}$$
$$= x \cdot \sum_{n\geq 2} \sum_{k_0=1}^{n-1} \frac{P_{k_0}^*}{k_0!} \cdot [x^{n-1-k_0}] \frac{1}{1 - x\widetilde{P}'(x)} x^{n-1}.$$

After changing indices and noting $P_0^* = 0$ and $P_1^* = 1$, we see

$$P^*(x) = x + \frac{xP^*(x)}{1 - x\widetilde{P}'(x)}. \tag{4.6}$$

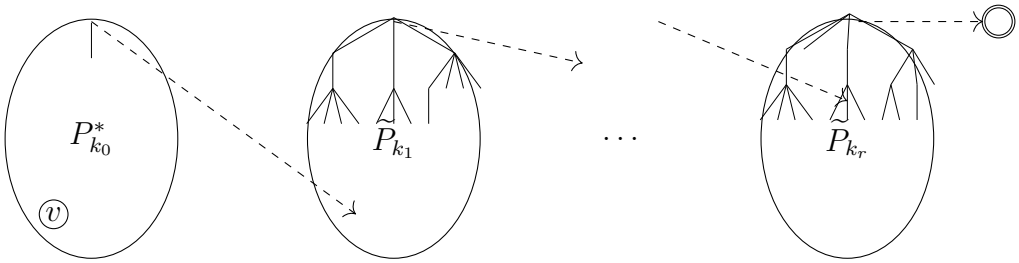

Figure 4.13: Decomposition based on final driver's movement.

Combining Equations (4.5) and (4.6), we see

$$x\left(\widetilde{P}'(x)\right)^2 + (x-1)\widetilde{P}'(x) + 1 = 0,$$

and so as $\widetilde{P}'(0) = 1$,

$$\widetilde{P}'(x) = \frac{1 - x - \sqrt{x^2 - 6x + 1}}{2x},$$

which is the ordinary generating function for the large Schröder numbers. Hence,

$$\widetilde{P}'(x) = \sum_{n\geq 0} n! S_n \frac{x^n}{n!},$$

meaning

$$\widetilde{P}(x) = \sum_{n\geq 1} (n-1)! S_{n-1} \frac{x^n}{n!}.$$

□

As mentioned in Section 2.2.2, the large Schröder numbers count royal paths and the large Schröder number S_{n-1} has the formula

$$S_{n-1} = \sum_{k=0}^{n-1} N_{n-1,k} 2^k.$$

We have seen something like this before! From Corollary 3, it immediately follows that

Corollary 4. *For $n \geq 1$, we have*

$$PTL_n(2) = n! \widetilde{P}_n.$$

In addition to the relationship in Equation (4.4), we can let F_n^* denote the total number of parking distributions on trees with n vertices with one leaf marked and let

$$F^*(x) = \sum_{n\geq 1} F_n^* \frac{x^n}{n!}.$$

Constructing a parking distribution counted by F_n^* by "growing" it from a parking distribution counted by $\widetilde{F}_{n-1}$ gives n choices for the leaf label, $n-1$ choices of nodes to attach the marked leaf to without reassigning drivers, and $n-1$ choices for attaching the marked leaf to a node and reassigning at least one driver from the parent node. Thus, $F_n^* = 2n(n-1)\widetilde{F}_{n-1}$, so

$$F^*(x) = x + 2x^2 \widetilde{F}'(x). \tag{4.7}$$

On the other hand, if a parking distribution on a marked tree with n vertices is decomposed by considering the final unfilled node when the drivers on the marked leaf park last, the final node is of one of three types: the root, the marked leaf, or neither. Therefore, we have for $n \geq 2$

$$\begin{aligned} F_n^* = &\sum_{r\geq 0} \frac{n}{r!} \sum_{\sum_{i=0}^{r} k_i = n-1} \binom{n-1}{k_0, \ldots, k_r} F_{k_0}^* \widetilde{F}_{k_1} \cdots \widetilde{F}_{k_r} \\ &+ n(n-1)F_{n-1} \\ &+ \delta_{n\geq 3} \sum_{r\geq 0} \frac{n}{r!} \sum_{k+\sum_{i=0}^{r} k_i = n-1} \binom{n-1}{k_0, \ldots, k_r, k} F_{k_0}^* \widetilde{F}_{k_1} \cdots \widetilde{F}_{k_r} \widetilde{F}_k k, \end{aligned}$$

where $\delta_{n\geq 3}$ is 1 if $n \geq 3$ and 0 otherwise. Summing over n we get

$$F^* = x + xF^* e^F + x^2 F' + x^2 F^* F' e^F. \tag{4.8}$$

Combining Equations (4.7) and (4.8) proves

Theorem 10. *The exponential generating function of parking distributions on trees, $\widetilde{F}(x)$, satisfies the differential equation*

$$\widetilde{F}' = e^{\widetilde{F}}(1 + x\widetilde{F}')(1 + 2x\widetilde{F}'),$$

with initial condition $\widetilde{F}(0) = 0$

We note that the generating function for general tree parking functions, $F(x)$, satisfies the similar equation

$$F' = e^F (1 + xF')^2.$$

For details, see Equation (7) in [14].

5. SOURCE TREES AND INVERSE MAPPING DIGRAPHS[1]

This section is an expansion of Section 4 of [11]. We now combine the subject matters of Sections 3 and 4 to investigate parking functions on source trees.

If T is a sink tree, let $\widetilde{T}$ be the source tree obtained by reversing the orientation of all the edges. Similarly, if M_f is the digraph obtained from $f : [n] \to [n]$ by letting $V(M_f) = [n]$ with edge set $E = \{(i, f(i))\}_{i=1}^{n}$, we let $\widetilde{M_f}$ be the digraph obtained from M_f by reversing edge orientations. That is, $E(\widetilde{M_f}) = \{(f(i), i)\}_{i=1}^{n}$. We will call these *mapping* and *inverse mapping digraphs*, respectively. Define the sets

$$\mathcal{T}_n = \{T : |V(T)| = n \text{ and } T \text{ is a sink tree}\},$$

$$\mathcal{M}_n = \{M : M \text{ is the mapping digraph of some } f : [n] \to [n]\}.$$

We similarly define $\widetilde{\mathcal{T}_n}$ and $\widetilde{\mathcal{M}_n}$ for the source trees and inverse mapping digraphs.

Recall that $P(D, m) = |\{(D, s) : s \in [n]^m \text{ is a parking function on digraph } D\}|$, the number of (n, m)-parking functions on D. We begin by finding upper and lower bounds for $P(\widetilde{T}, m)$.

5.1 Extremal Values for Source Trees

Proposition 6. *Let $\widetilde{T}$ be a source tree, u a non-root vertex, v the parent of u, and w such that $v \preceq_{\widetilde{T}} w$ and $w \notin \widetilde{T}_u$. Let $\widetilde{T}'$ be the tree obtained by removing the edge (v, u) and adding the edge (w, u). Then $P(\widetilde{T}, m) \leq P(\widetilde{T}', m)$.*

Proof. To clarify which tree we are considering, we denote by x' the vertex in $\widetilde{T}'$ with label x. Let $(\widetilde{T}, s)$ be an (n, m)-parking function. By Corollary 1, we must check $|\{i : s_i \in \widetilde{T}'_{x'}\}| \leq |\widetilde{T}'_{x'}|$ for all $x \in V(\widetilde{T}')$. By the construction of $\widetilde{T}'$, $|\{i : s_i \in \widetilde{T}'_{y'}\}| = |\{i : s_i \in \widetilde{T}_y\}|$ and $|\widetilde{T}'_{y'}| = |\widetilde{T}_y|$ for all y' not satisfying $v' \prec_{\widetilde{T}'} y' \preceq_{\widetilde{T}'} w'$.

Therefore, let y' be a vertex satisfying $v' \prec_{\widetilde{T}'} y' \preceq_{\widetilde{T}'} w'$. We thus have:

[1]Reprinted with permission from "Parking Functions on Oriented Trees" by W. King and C. H. Yan, 2018. *Séminaire Lotharingien de Combinatoire*, 80B, Copyright 2018 by W. King and C.H. Yan.

$$\begin{aligned}|\{i : s_i \in \widetilde{T}'_{y'}\}| &= |\{i : s_i \in \widetilde{T}_y\}| + |\{i : s_i \in \widetilde{T}_u\}| \\ &\leq |\widetilde{T}_y| + |\widetilde{T}_u| \\ &= |\widetilde{T}'_{y'}|.\end{aligned}$$

□

As a result, we obtain an upper and lower bound on $P(\widetilde{T}, m)$.

Corollary 5. *The number of (n, m)-parking functions is maximized when $\widetilde{T}$ is a directed path and minimized when $\widetilde{T}$ is a star, meaning*

$$\sum_{i=0}^{m} \binom{n-1}{m-i} \binom{m}{i} (m-i)! \leq P(\widetilde{T}, m) \leq (n-m+1)(n+1)^{m-1}.$$

Proof. For a path, the parking functions, up to vertex labeling, are classical $(n, m)-$parking functions, and thus number $(n-m+1)(n+1)^{m-1}$.

On a star, for $0 \leq i \leq m$, when i drivers prefer the root, there are $\binom{n-1}{m-i}$ ways to choose the preferred non-root vertices, $\binom{m}{i}$ ways to place the drivers preferring the root in s, and $(m-i)!$ ways to order the drivers preferring non-roots in s. □

5.2 Tree and Inverse Mapping Parking Functions

Define the following for $n \geq 1$:

$$F_{n,m} = \sum_{T \in \mathcal{T}_n} P(T, m), \text{ and } M_{n,m} = \sum_{M \in \mathcal{M}_n} P(M, m).$$

Similarly, define $\widetilde{F}_{n,m}$ and $\widetilde{M}_{n,m}$ for source trees and inverse mappings.

Lackner and Panholzer [14] proved $n \cdot F_{n,m} = M_{n,m}$. In fact, this relationship still holds when the edge orientations are reversed. We first prove this claim when $n = m$, then we will show the more general case. While the general idea of the proofs are similar to their counterparts in [14],

there are some technical difficulties on source trees that are not present on sink trees. For source trees, the drivers no longer necessarily have a unique path along which to search for a parking spot, so we must instead identify edges that are not necessary for *some* successful parking. This is simple enough on trees using the characterization of Corollary 1, but it is not immediately clear for inverse mapping digraphs. So, we first prove that at least one cycle edge on each component of an inverse mapping digraph is not needed for parking.

Lemma 3. *Let $(\widetilde{M}_f, s)$ be a parking function. Then there exists at least one edge in each cycle of $\widetilde{M}_f$ that can be deleted such that all drivers can still park.*

Proof. It is sufficient to prove the claim for an inverse mapping digraph with only one component, and thus one cycle. Suppose such an edge does not exist. Label the vertices of the cycle $\{u_i\}_{i=1}^r$ such that u_1 is the minimal among the cycle vertices and the edge $e_i := (u_{i-1}, u_i)$ is an edge in the graph. Denote by $T_{i,f}$ the tree rooted at u_i obtained by deleting edge e_i. By assumption, the sequence s is no longer a parking function on $T_{i,f}$ for any i. Begin by deleting e_1. Thus, there is some $A \subseteq [n]$ such that $|R_{T_{1,f}}(A)| < |\{i : s_i \in R_{T_{1,f}}(A)\}|$. Corollary 1 tells us that we may consider $A = R_{T_{1,f}}(i)$ for some $i \in [n]$. Since the viewable set of a non-cycle vertex is not affected by the deletion of a cycle edge, we know i must be a cycle vertex. Therefore, let k_1 be the index of the cycle vertex with maximal distance following the cycle orientation from u_1 such that $|R_{T_{1,f}}(u_{k_1})| < |\{i : s_i \in R_{T_{1,f}}(u_{k_1})\}|$.

Next, we consider the tree $T_{k_1,f}$ and choose index k_2 in a similar fashion, of maximal distance from u_{k_1}. We claim $k_2 < k_1$. Suppose not. We cannot have $k_1 = k_2$ as then $|R_{T_{k_1,f}}(u_{k_2})| = n \geq |\{i : s_i \in R_{T_{k_1,f}}(u_{k_2})\}|$. Therefore, let $k_2 > k_1$. This implies that $\text{dist}(u_1, u_{k_1}) < \text{dist}(u_1, u_{k_2})$ along the cyle, so by our choice of k_1, we know $|R_{T_{1,f}}(u_{k_2})| \geq |\{j : s_j \in V_{T_{1,f}}(u_{k_2})|$. Hence we

have

$$\begin{aligned}|R_{T_{k_1,f}}(u_{k_2})| &= |R_{T_{1,f}}(u_{k_2})| + |R_{T_{k_1,f}}(u_1)| \\ &\geq |\{j : s_j \in R_{T_{1,f}}(u_{k_2})| + |\{j : s_j \in R_{T_{k_1,f}}(u_1)\}| \\ &= |\{j : s_j \in R_{T_{k_1,f}}(u_{k_2})\}|,\end{aligned}$$

contradicting how we chose k_2.

Further, we show $k_2 \neq 1$. Supposing they are the same, we have

$$\begin{aligned}n &= |R_{T_{k_1,f}}(u_{k_2})| + |R_{T_{1,f}}(u_{k_1})| \\ &< |\{j : s_j \in R_{T_{k_1,f}}(u_{k_2})| + |\{j : s_j \in R_{T_{1,f}}(u_1)\}| \\ &= n,\end{aligned}$$

which is clearly false.

We use this base case to inductively prove that $k_{i+1} < k_i$, so assume this holds for all $i < m$. Because $R_{T_{k_m,f}}(u_{k_m}) = [n]$, we know $k_{m+1} \neq k_m$, so suppose $k_{m+1} > k_m$ and suppose further that there is some $0 \leq \alpha \leq m-1$ such that $k_{\alpha+1} < k_{m+1} < k_\alpha$, where we take $k_0 = 1$. Figure 5.1 shows a sketch of the tree $T_{k_m,f}$, where the vertex u_i is labeled i for readability. The dashed red vertices and edges are those associated with $R_{T_{k_\alpha,f}}(u_{k_{m+1}})$, while the dotted blue ones are associated with $R_{T_{k_m,f}}(u_{k_\alpha})$. Together, the vertices form $R_{T_{k_m,f}}(u_{k_{m+1}})$.

In this case we have $\text{dist}(u_{k_\alpha}, u_{k_{m+1}}) > \text{dist}(u_{k_\alpha}, u_{k_{\alpha+1}})$ and $\text{dist}(u_{k_m}, u_{k_\alpha}) > \text{dist}(u_{k_m}, u_{k_{m+1}})$ and so we can use our inductive hypothesis to write

$$\begin{aligned}|R_{T_{k_m,f}}(u_{k_{m+1}})| &= |R_{T_{k_\alpha,f}}(u_{k_{m+1}})| + |R_{T_{k_m,f}}(u_{k_\alpha})| \\ &\geq |\{j : s_j \in R_{T_{k_\alpha,f}}(u_{k_{m+1}})| + |\{j : s_j \in R_{T_{k_m,f}}(u_{k_\alpha})\}| \\ &= |\{j : s_j \in R_{T_{k_m,f}}(u_{k_{m+1}})\}|,\end{aligned}$$

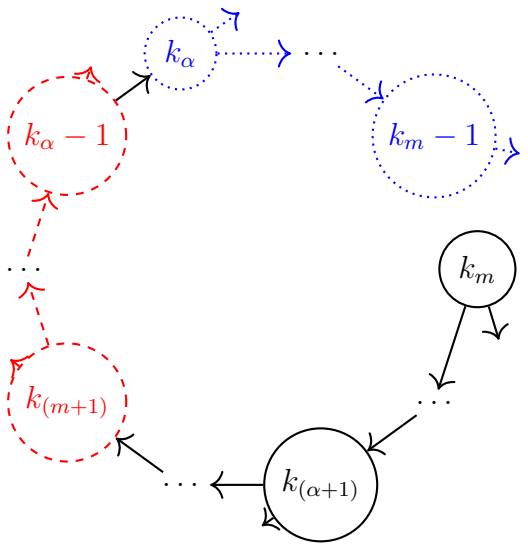

Figure 5.1: A view of the "cycle" vertices of $T_{k_m,f}$ if $k_{m+1} > k_m$.

which is also a contradiction. Our final case is if $k_{m+1} = k_\ell$ for some $0 \leq \ell < m$. Similar to the k_2 case, we have

$$\begin{aligned} n &= \sum_{i=\ell}^{m} |R_{T_{k_i,f}}(u_{k_{i+1}})| \\ &< \sum_{i=\ell}^{m} |\{j : s_j \in R_{T_{k_i,f}}(u_{k_{i+1}})\}| \\ &= n, \end{aligned}$$

which is still false.

Hence, we must have a strictly decreasing sequence $k_1 > k_2 > k_3 > \ldots$ which cannot indefinitely continue. Therefore, there must be some k_m such that $(T_{k_m,f}, s)$ is a parking function. □

Figure 5.2 gives an example of this process which terminates at k_2. For simplicity u_i has label i. If e_1 is deleted, the two drivers preferring 4 are not able to both park, and as $\text{dist}(1, 4)$ is maximal among cycle vertices, we must have $k_1 = 4$. Next, delete edge e_4 instead. Only two drivers prefer

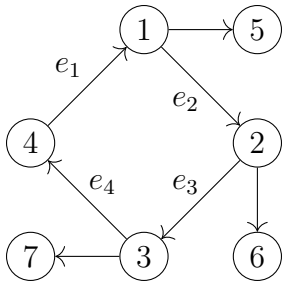

Figure 5.2: A mapping digraph with parking function $s = (2, 2, 2, 3, 3, 4, 4)$

$\{3, 7\}$, so this is no issue. However, 5 drivers prefer $\{2, 3, 6, 7\}$, which means s is not a parking function when e_4 is deleted and that $k_2 = 2$. After some inspection, we see that everyone is able to park even after e_2 is deleted, thus we have found *one* unnecessary cycle edge. In this case, the unnecessary edge is unique. However, it need not be.

Professor Chun-Hung Liu suggested the following, simpler proof of Lemma 3.

Second Proof of Lemma 3. We induct on the number of vertices in a cycle. Let $(\widetilde{M_f}, s)$ be an (n, n)-parking function. Without loss of generality, we may assume there is only one component of $\widetilde{M_f}$ and thus a unique cycle in the graph. If only one vertex is in the cycle, then there is an edge of the form (u, u) which is useless for parking and may be deleted.

Now suppose the cycle has length $r > 1$. Furthermore, suppose without loss of generality that the vertices of the cycle are labeled by $[r]$, $f(r) = 1$, and $f(i) = i + 1$ for $i \in [r - 1]$. Let M be the graph obtained by deleting all cycle edges. Define for $1 \leq i \leq r$, $V_i := |R_M(i)|$ and $\alpha_i := |\{j : s_j \in R_M(i)\}|$. These are the numbers of vertices in and drivers preferring the subtree induced by the vertex i along with all $i \preceq_{\widetilde{M_f}} v$ for v non-cycle vertices in $\widetilde{M_f}$.

If $\alpha_i < V_i$ for all $i \in [r]$, then there are strictly fewer than n drivers attempting to park, contradicting the assumption that $(\widetilde{M_f}, s)$ is an (n, n)-parking function. Hence, we know $\alpha_i \geq V_i$ for some i. Since s is a parking function on $\widetilde{M_f}$, at least one driver prefers i (otherwise, too many drivers prefer non-cycle vertices). We construct an $(n - 1, n - 1)$-parking function by contracting the edge $(i, i - 1)$ to identify the vertices $i - 1$ and i as a single vertex (with label $i - 1$) to form

the digraph M', deleting the first instance of i from s, and changing all others to $i-1$ to form the sequence s'. For non-cycle vertex v, $|R_{\widetilde{M}_f}(v)| = |R_{M'}(v)|$, as are the number of drivers preferring each set. If v is instead a cycle vertex, then $n = |R_{\widetilde{M}_f}(v)| = |R_{M'}(v)| + 1$, while n drivers prefer $R_{\widetilde{M}_f}(v)$ and $n-1$ drivers prefer $R_{M'}(v)$. Thus, (M', s') is a parking function with $r-1$ vertices in the cycle. By the inductive hypothesis, there exists an edge e that can be deleted from M'. We claim this same edge in $\widetilde{M}_f$ is not necessary for parking via s.

Let T be the digraph obtained by deleting e from $\widetilde{M}_f$ and T' be obtained by deleting e from M'. Since (T', s') is a parking function, we know for any $v \in T'$, we have $|\{j : s'_j \in R_{T'}(v)\}| \leq |R_{T'}(v)|$. We now check the vertices of T to determine if (T, s) is a parking function.

Case 1: $i-1 \prec_T v$. We have

$$|\{j : s_j \in R_T(v)\}| = |\{j : s_j \in R_{T'}(v)| \leq |R_{T'}(v)| = |R_T(v)|.$$

Case 2: $v \prec_T i$. Then,

$$|\{j : s_j \in R_T(v)\}| = |\{j : s_j \in R_{T'}(v)| + 1 \leq |R_{T'}(v)| + 1 = |R_T(v)|.$$

Case 3: $v = i$ gives

$$|\{j : s_j \in R_T(i)\}| = |\{j : s_j \in R_{T'}(i-1)| + 1 \leq |R_{T'}(i-1)| + 1 = |R_T(i)|.$$

Case 4: $v = i-1$. Using the fact that $-\alpha_i \leq -V_i$, we know

$$\begin{aligned} |\{j : s_j \in R_T(i-1)\}| &= |\{j : s_j \in R_{T'}(i-1)| + 1 - \alpha_i \\ &\leq |R_{T'}(i-1)| + 1 - V_i \\ &= (|R_T(i-1)| + V_i - 1) + 1 - V_i \\ &= |R_T(i-1)|. \end{aligned}$$

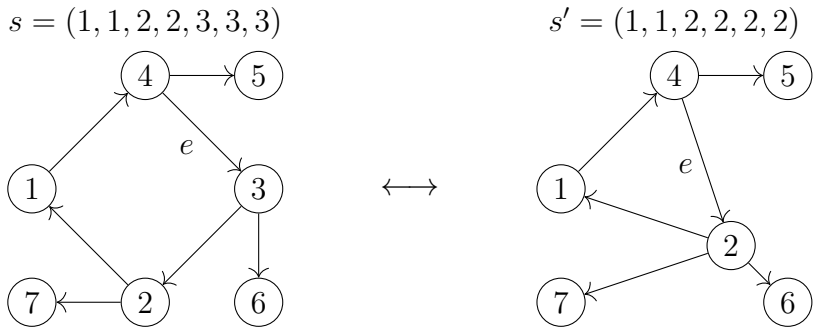

Figure 5.3: Identifying a deletable edge e by contracting $(3, 2)$.

Case 5: all other v. Since v is not a cycle vertex in $\widetilde{M}_f$ and s is a parking function on $\widetilde{M}_f$, the deleting of e does not affect the reachable set of v. Thus,

$$|\{j : s_j \in R_T(v)\}| = |\{j : s_j \in R_{\widetilde{M}_f}(v)\}| \leq |R_{\widetilde{M}_f}(v)| = |R_T(v)|.$$

So (T, s) is indeed a parking function and we know the edge e is not necessary for parking on $\widetilde{M}_f$. □

Figure 5.3 gives an example of the contraction to M' with $i = 3$. We identify a deletable e on M', which gives a deletable e on $\widetilde{M}_f$. We now use Lemma 3 to prove

Theorem 11. *For $n \geq 1$, we have the relationship*

$$n \cdot \widetilde{F}_{n,n} = \widetilde{M}_{n,n}.$$

Proof. Let $(\widetilde{T}, s)$ be a parking function for source tree $\widetilde{T}$, and pick $v \in V(\widetilde{T})$. We define a bijection ψ such that $\psi\left((\widetilde{T}, s, v)\right) = (\widetilde{M}_f, s)$ for some appropriate inverse mapping digraph $\widetilde{M}_f$, constructed by identifying edges in $\widetilde{T}$ that can be manipulated without affecting the ability of the cars to park. The sequence s will not change.

Let (u, w) be an edge in $\widetilde{T}$. If $|\{i : s_i \in \widetilde{T}_w\}| = |\widetilde{T}_w|$, then for any successful parking, no car may cross (u, w) as otherwise too many cars would attempt to park in the subtree $\widetilde{T}_w$. Additionally,

at least one driver must prefer w, as one driver must park in w and no driver may use the edge (u, w). These two observations will allow us to select edges to manipulate in $\widetilde{T}$.

Consider the path root$(\widetilde{T}) = v_1 \to v_2 \to \ldots \to v_k = v$ for some $k \geq 1$. We first identify the edges that are freely manipulatable, then we use the order of s to choose a subset of those. For $1 \leq i \leq k$, let $v_i \in A$ if and only if $|\{j : s_j \in \widetilde{T}_{v_i}\}| = |\widetilde{T}_{v_i}|$. Since $\widetilde{T}_{v_1} = \widetilde{T}$, A is nonempty. By the second observation above, all $v_i \in A$ appear as preferences in s. The edge (v_{i-1}, v_i) is not used by any driver, so we may manipulate it (even delete it) without affecting parking. Next, for the $v_i \in A$, we define the rank $d(v_i)$ to be the index of the first appearance of v_i in s. We now let $B \subseteq A$ be given by the elements $\{v_i : \forall j < i, d(v_j) < d(v_i)\}$. That is, the elements of B are those in A that appear in s after their ancestors from A. Note that $B \neq \emptyset$ as root$(\widetilde{T}) = v_1 \in B$.

Consider the unique sequence $\{v_{i_j}\}_{j=1}^{|B|}$ such that $v_{i_j} \in B$ and $i_j < i_{j+1}$. For $j > 1$, remove the edge (v_{i_j-1}, v_{i_j}) and add the edge $(v_{i_j-1}, v_{i_{(j-1)}})$. Finally, add the edge $(v, v_{i_{|B|}})$ (if v is the root, this will be a loop as then $v_{i_1} = v$). The resulting graph is an inverse mapping digraph, $\widetilde{M_f}$, where $f(i)$ is the unique j such that (j, i) is an edge. In particular, the new edges are $\{(f(v_{i_j}), v_{i_j})\}_{j=1}^{|B|-1} \cup \{(f(v_{i_{|B|}}), v_{i_{|B|}})\}$.

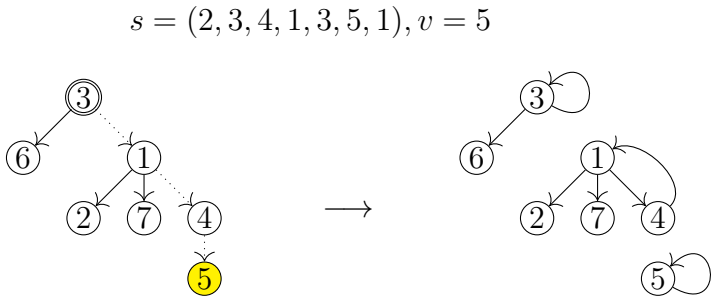

Figure 5.4: Turning a source tree into an inverse mapping digraph.

Figure 5.4 gives an example of ψ. In it, $A = \{1, 3, 4, 5\}$ and $B = \{1, 3, 5\}$, with $v_{i_1} = 3$, $v_{i_2} = 1$, and $v_{i_3} = 5$.

Remark 2. *In each component, the vertex appearing in B has the highest rank among all other vertices appearing in the cycle of that component. Further, if an edge was necessary for parking on $\widetilde{T}$, it is still necessary for parking on $\widetilde{M}_f$.*

For the inverse, we know by Lemma 3 that at least one edge in each cycle of the graph is not necessary for parking. We let the set $\hat{A}$ be the set of vertices in the cycles of $\widetilde{M}_f$ that are the terminal vertices of an edge that is not necessary for parking. Define $\hat{B} \subseteq \hat{A}$ as the set of vertices which have the highest rank in each cycle. By Remark 2, if $(\widetilde{M}_f, s) = \psi\left((\widetilde{T}, s, v)\right)$, we know $B = \hat{B}$. Label these elements of $\hat{B}$ by $\{b_i\}_{i=1}^{|\hat{B}|}$ such that $d(b_1) < d(b_2) < \ldots < d(b_{|\hat{B}|})$. For $1 \leq i \leq |\hat{B}| - 1$, remove the edge $(f(b_i), b_i)$ and add the edge $(f(b_i), b_{i+1})$. Finally, delete the edge $\left(f(b_{|\hat{B}|}), b_{|\hat{B}|}\right)$ and mark $f(b_{|\hat{B}|})$. The resulting tree is $\widetilde{T}$, so $\psi^{-1}\left((\widetilde{M}_f, s)\right) = (\widetilde{T}, s, f(b_{|\hat{B}|}))$. $\square$

As promised, we can extend this result to (n, m)-parking functions.

Theorem 12. *Let $n \in \mathbb{N}$ and $0 \leq m \leq n$. Then*

$$n \cdot \widetilde{F}_{n,m} = \widetilde{M}_{n,m}.$$

Proof. Let $s \in [n]^m$ be a parking function on $\widetilde{T} \in \widetilde{\mathcal{T}}_n$, and let $v \in V(\widetilde{T})$. Our goal is to extend s to $s' \in [n]^n$ in a reversible manner, then apply ψ. We must do so in a way that is not affected by the change in edges caused by ψ, which suggests we avoid using edges along the path root$(\widetilde{T}) \to v$.

To this end, drivers choose to park as follows. We recursively define $\{A_i\}_{i=1}^m$ so that $A_1 = \{s_1\}$ and in general A_i are the spots that driver i could park at so that the remaining drivers may successfully park and given the first $i-1$ drivers are parked. Let $B_i \subseteq A_i$ be the vertices of A_i that are reachable from s_i by utilizing a minimal number of edges in the path root$(\widetilde{T}) \to v$. From there, driver i parks at the vertex with label $\min(B_i)$. Once driver i is parked, we construct A_{i+1} and continue until all drivers have parked.

Let $\{x_i\}_{i=1}^{n-m}$ be the unoccupied spaces after the drivers have parked in this manner, ordered in

an increasing manner. We then define s' as follows:

$$s'_i = \begin{cases} s_i & \text{if } i \leq m \\ x_j & \text{if } i = m + j \end{cases}$$

Then, we apply ψ to $(\widetilde{T}, s', v)$. Since s' does not change under ψ and is a parking function, s is also a parking function on the resulting mapping digraph $\widetilde{M_f}$. In order to reverse, we must be able to extend s to s' on $\widetilde{M_f}$. Because the edges on the path between root$(\widetilde{T})$ and v become the cycle edges, drivers park as defined in the first paragraph, but instead of utilizing a minimal number of path edges, they use a minimal number of cycle edges. □

5.3 Source Trees vs Sink Trees

We consider the relationship, if any, between $P(T, m)$ and $P(\widetilde{T}, m)$. If $m = 0, 1$, then the two are equal as there is only one parking function with 0 cars and the first car parks regardless of the underlying graph. We begin with the third-easiest case and let $m = n$ and we prove

Theorem 13. *Let $T \in \mathcal{T}_n$. Then*

$$P(T, n) \leq P(\widetilde{T}, n)$$

with equality if and only if T is a path.

Proof. Let (T, s) be a parking function on sink tree T. We give a process to determine an involution $\tau \in \mathfrak{S}_n$ such that $\tau(s) = (\tau(s_1), \tau(s_2), \ldots, \tau(s_n))$ is a parking function on the source tree $\widetilde{T}$. Park cars on T following the parking procedure, highlighting an edge if it is used by a driver after failing to park at her preferred spot. Since T is a sink tree, each vertex has outdegree at most 1, so parking is deterministic. We define τ by individually considering the components connected by highlighted edges. So without loss of genreality, we may assume that every edge in T is highlighted.

We define a collection of length ≥ 2 "paths" in T, one for each leaf, whose vertices form a partition of the vertices of T. The purpose of these "paths" is to identify a section of the tree where

we can "flip" the edge orientations and driver preferences and still guarantee each driver a spot to park.

Let $\{v_i\}$, for $1 \leq i \leq k$ be the leaves of T indexed such that the labels $v_i < v_{i+1}$. We recursively define the sets P_i: the set P_i is the smallest set of vertices of the path between v_i and the root such that no vertices from $P_j, j < i$ are in P_i, there are $|P_i|$ drivers preferring P_i, and all vertices of P_i are connected to v_i through vertices in $\{P_j\}_{j=1}^{i}$. For example, in Figure 5.5, we have sets $P_1 = \{1, 2, 3, 5\}$ and $P_2 = \{4, 6\}$. Two drivers prefer the leaf $\{1\}$, three drivers prefer the vertices $\{1, 2\}$, four drivers prefer $\{1, 2, 3\}$ and $\{1, 2, 3, 5\}$, so the latter is P_1. When determining P_2, we skip over vertices in previously-chosen paths, in this case the vertex labeled 5.

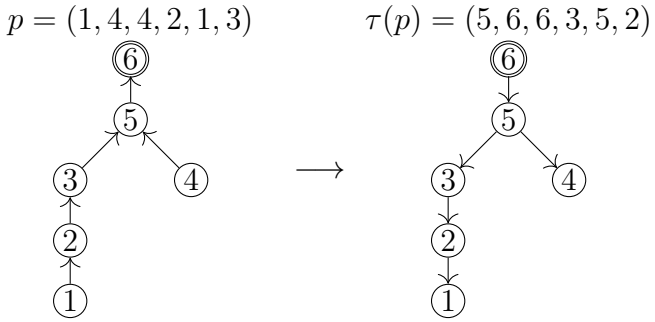

Figure 5.5: Constructing a parking function on $\widetilde{T}$ from one on T. $\tau = (15)(23)(46)$

The collection $\{P_i\}_{i=1}^{k}$ must partition the vertices of T. Suppose it does not and let $v \notin P_i$ for any i be such that v is the only vertex in T_v, the subtree rooted at v, with this property. The vertex v is not a leaf of T, as all leaves are in the sets $\{P_i\}_{i=1}^{k}$ by construction, and thus is the terminus of at least one edge, (u, v). Since none of the "paths" corresponding to leaves of T_v contain v and because all cars can park, all cars preferring spots $w \preceq_T u$ can park without occupying v. This means the edge (u, v) is not used by any driver after failing to park in her preferred spot, which contradicts our assumption that this was true of all edges. Therefore, such a v can not exist.

For each i, let $n_i = |P_i|$ and label the elements of P_i by $w_{i,j}$ such that $w_{i,1} = v_i \preceq_T w_{i,2} \preceq_T \ldots \preceq_T w_{i,n_i}$. Finally, we define $\tau(w_{i,j}) = w_{i,n_i+1-j}$. This reverses the driver preference along the

"path" so that when the edge orientation is flipped for $\widetilde{T}$, the drivers may park as they did on T.

We can recover the P_i from $(\widetilde{T}, \tau(s))$ using the exact same method. Thus, we can invert the process. In Figure 5.5, zero drivers prefer $\{1\}$, one driver prefers $\{1, 2\}$, two drivers prefer $\{1, 2, 3\}$, and four drivers $\{1, 2, 3, 5\}$, so this is P_1. For P_2, no drivers prefer $\{4\}$, we skip over 5 as it is already in P_1, and two drivers prefer $\{4, 6\}$.

If T is not a path, then this process is not surjective because the parking function in which all n drivers prefer the root of $\widetilde{T}$ is not obtainable in this manner as at least one driver prefers each leaf vertex. □

Summing over all $T \in \mathcal{T}_n$ gives us

Corollary 6. *For $n \geq 1$,*

$$F_{n,n} \leq \widetilde{F}_{n,n},$$

with equality only when $n \in \{1, 2\}$.

When we replace n by $m < n$ in Theorem 13, the result does not necessarily hold. For the tree in Figure 5.6 with root vertex 4, $P(T, 2) = 15$, as any sequence except $(4, 4)$ parks. However, $P(\widetilde{T}, 2) = 14$ as neither $(1, 1)$ nor $(2, 2)$ are parking functions.

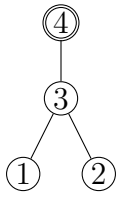

Figure 5.6: A tree for which $P(T, 2) > P(\widetilde{T}, 2)$.

6. MISCELLANEOUS STRUCTURES

In this final section of results, we consider parking distributions on source caterpillars and both parking functions and parking distributions on source and sink spiders. All parking functions here are (n, n)-parking functions.

6.1 Caterpillars

We recall from Section 2.1.1 the definition of a caterpillar. Butler, Graham, and Yan [4] give a bijection between parking distributions on sink caterpillars and lattice paths in $\mathbb{Z}^2$ with steps $\{(0, 1), (1, 0)\}$, avoiding a certain region. In this subsection, we give a bijection between source caterpillars and more complex lattice paths in $\mathbb{Z}^2$ that generalize Dyck paths.

Theorem 14. *The parking distributions on Cat$(a_1, a_2, \ldots, a_n)$ are in bijection with the lattice paths in $\mathbb{Z}^2$ weakly above the x-axis, starting at $(0, 0)$, ending at $(\sum_{i=1}^{n}(a_i + 2), 0)$, with down steps $(1, -1)$, and where the j^{th} up-step has the form $(1 + a_{n-j+1} - i_j, 1 + i_j)$ and is colored one of $\binom{a_{n-j+1}}{i_j}$ colors for some $0 \leq i_j \leq a_{n-j+1}$.*

Proof. Consider a caterpillar $C = \text{Cat}(a_1, a_2, \ldots, a_n)$, set $N := n + \sum_{i=1}^{n} a_i$, and let $f : [N] \to [N]$ be a parking distribution on C. For $1 \leq i \leq n$, let $\{v_{i,j}\}_{j=1}^{a_i}$ be the non-spine children of v_i. We construct a lattice path with the above properties from C and f.

For the j^{th} up-step of the path, $i_j = a_{n+1-j} - \sum_{k=1}^{a_{n+1-j}} f(v_{n+1-j,k})$, the number of leaf children of v_{n+1-j} which no drivers prefer. The color of the up-step corresponds to one of the $\binom{a_{n+1-j}}{i}$ ways to select these spots. The number of downs-steps following the j^{th} up-step is given by $f(v_{n+1-j})$. We must show this path is weakly above the x-axis and ends at $(\sum_{i=1}^{n}(a_i + 2), 0)$.

First, we consider the ending x-coordinate of the path. Knowing that $\sum_{v \in D} f(v) = N$ and summing up the x-components of the up and down-steps,

$$\begin{aligned}\sum_{j=1}^{n}(1+(a_{n-j+1}-i_j))+\sum_{j=1}^{n}f(v_j) &= n+\sum_{j=1}^{n}\sum_{k=1}^{a_{n+1-j}}f(v_{n+1-j,k})+\sum_{j=1}^{n}f(v_j)\\ &= n+N\\ &= 2n+\sum_{j=1}^{n}a_j\\ &= \sum_{j=1}^{n}(a_j+2).\end{aligned}$$

We now show that the path ends at height $y=0$. The final height is given by

$$\sum_{k=1}^{n}\left(1+a_{n+1-k}-\sum_{\ell=1}^{a_{n+1-k}}f(v_{n+1-k,\ell})\right)-\sum_{k=1}^{n}f(v_{n+1-k})=N-N=0.$$

Next, we show that the path is weakly above the x-axis by assuming there is some down-step that brings the path below the x-axis, occurring in the sequence of down steps appearing after the j^{th} up-step in the path. The height of the path immediately before the $(j+1)^{\text{st}}$ up-step is given by

$$\sum_{k=1}^{j}\left(1+a_{n+1-k}-\sum_{\ell=1}^{a_{n+1-k}}f(v_{n+1-k,\ell})\right)-\sum_{k=1}^{j}f(v_{n+1-k}),$$

so

$$\sum_{k=1}^{j}(1+a_{n+1-k})<\sum_{\ell=1}^{a_{n+1-k}}f(v_{n+1-k,\ell})+\sum_{k=1}^{j}f(v_{n+1-k}). \tag{6.1}$$

However, the left hand side of Equation (6.1) is $|R_C(v_{n+1-j})|$ and the right hand side is the number of drivers preferring that subtree. This contradicts the assumption that f is a parking distribution, thus there can be no steps below the x-axis.

On the other hand, given such a path with n up-steps, we may recover both f and the structure of C. The value of a_{n+1-j} is obtainable from the dimensions of the j^{th} up-step, say (x,y). Then $a_{n+1-j}=x+y-2$. Thus, we know $C=\text{Cat}(a_1,a_2,\ldots,a_n)$. The number of down-steps after the j^{th} up-step tells us $f(v_{n+1-j})$, and the color of the j^{th} up-step tells us which of $\{f(v_{n+1-j,k})\}_{k=1}^{a_{n+1-j}}$

are 0 and which are 1. We must now show the resulting f is a parking distribution on C.

Since C is a source tree, it is sufficient to only check the number of drivers preferring the maximal subtrees rooted at each vertex. If v is a leaf, then by the above correspondance, we know $f(v) \leq 1$. Now consider spine vertex v_j. The height of the path immediately before the $(n-j)^{\text{th}}$ up-step is

$$\sum_{k=1}^{n-j+1} \left(1 + a_{n+1-k} - \sum_{\ell=1}^{a_{n+1-k}} f(v_{n+1-k,\ell})\right) - \sum_{k=1}^{n-j+1} f(v_{n+1-k}).$$

Changing indices and noting that the quantity is non-negative, we know

$$\sum_{k=j}^{n} \left(f(v_k) + \sum_{\ell=1}^{a_k} f(v_{k,\ell})\right) \leq \sum_{k=j}^{n} (a_k + 1).$$

The left hand side is the number of drivers preferring a spot in $R_C(v_j)$ and the right hand side is $|R_C(v_j)|$. This holds for all spine vertices v_j, so indeed the constructed f is a parking distribution on C. □

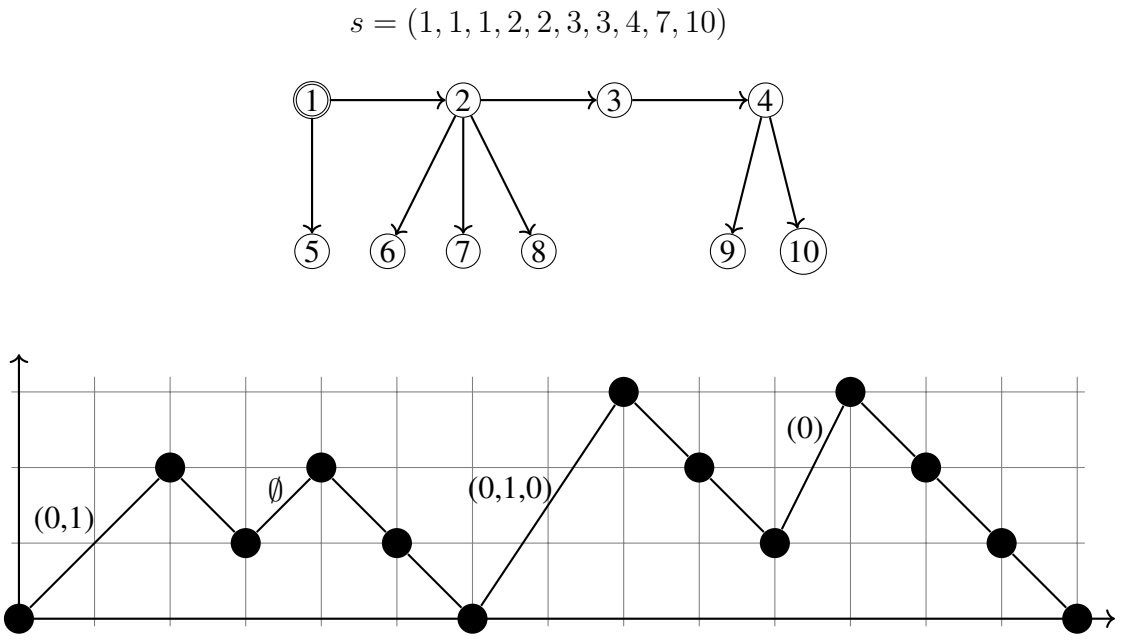

Figure 6.1: A parking distribution (represented as a weakly increasing sequence) on Cat$(1, 3, 0, 2)$ and corresponding lattice path.

Figure 6.1 gives an example of a parking distribution on $\mathrm{Cat}(1,3,0,2)$ and its corresponding lattice path. We choose to represent the color of the up-step $(\alpha,\beta) \to (\alpha+a_{n+1-j}+1-i_j, \beta+i_j+1)$ by an a_{n+1-j}-tuple with entries from $\{0,1\}$ and exactly i_j-many 0s. The tuple represents, from smallest to largest label, the leaves whose labels appear in s (the 1s) and those which do not (the 0s).

The exact number of such paths are difficult to count, but if we impose some regularity on the caterpillars, the problem becomes tractable. Let $\mathrm{Cat}_{n,k} = \mathrm{Cat}(k,k,\ldots,k)$, the caterpillar with n spine vertices and k leg vertices attached to each spine vertex. Then simplifying the statement of Theorem 14, we get:

Corollary 7. *The parking distributions on $\mathrm{Cat}_{n,k}$ are in bijection with the lattice paths in $\mathbb{Z}^2$ weakly above the x-axis, starting at $(0,0)$, ending at $(n(k+2),0)$, with down steps $(1,-1)$, and up steps of the form $(k+1-i,1+i)$, colored one of $\binom{k}{i}$ colors, for some $0 \le i \le k$.*

Let $a_{n,k}$ be the number of such paths and let $A_k(x) = 1 + \sum_{n\ge 1} a_{n,k}x^n$ be the ordinary generating function.

Theorem 15. *The generating function $A_k(x)$ satisfies the differential equation*

$$A_k(x) = 1 + x\left(A_k(x)\right)^2\left(1+A_k(x)\right)^k,$$

and so for $n \ge 1$,

$$a_{n,k} = \frac{1}{n}\sum_{j=0}^{n-1}\binom{2n}{j}\binom{kn}{n-1-j}2^{n(k-1)+j+1}.$$

Proof. We decompose a path counted by $a_{n,k}$ for $n \ge 1$ to get a recurrence relation. Given such a path, consider the first step on the path. It has the form $(0,0) \to (k+1-i, i+1)$ and one of $\binom{k}{i}$ colors for some $1 \le i \le k$. Then, for each $0 \le j \le i$, consider the first down step taking the path from height $j+1$ to height j. For appropriate α_j, these steps are $(\alpha_j, j+1) \to (\alpha_j+1, j)$. Set $\alpha_{i+1} = k-i$. For any j, the segment of the path between $x = \alpha_{j+1}+1$ and $x = \alpha_j$ must be a path counted by $a_{n_j,k}$ for $n_j = \dfrac{\alpha_{j-1}-\alpha_j-1}{k+2}$ when $j \ge 1$, and $n_0 = \dfrac{n(k+2)-\alpha_0-1}{k+2}$.

The sequence $(n_0, n_2, \ldots, n_{i+1})$ satisfies $\sum_{j=0}^{i+1} n_j = (n-1)$, so we may write

$$a_{n,k} = \sum_{i=0}^{k} \binom{k}{i} \sum_{\sum n_j = (n-1)} \prod_{j=0}^{i+1} a_{n_j,k}.$$

Figure 6.2 gives an example of this decomposition. Multiplying by x^n, summing over n, and adding $a_{0,k} = 1$ gives

$$\begin{aligned} A_k(x) &= 1 + x \sum_{i=0}^{k} \binom{k}{i} (A_k(x))^{i+2} \\ &= 1 + x (A_k(x))^2 (1 + A_k(x))^k . \end{aligned}$$

We now extract coefficients. Let $G_k(x) = A_k(x) - 1$, so

$$G_k(x) = x (G_k(x) + 1)^2 (2 + G_k(x))^k ,$$

letting $\phi(\lambda) = (1+\lambda)^2(2+\lambda)^k$, we see by Lagrange inversion for $n \geq 1$ that

$$\begin{aligned} a_{n,k} = [x^n]G_k(x) &= \frac{1}{n}[\lambda^{n-1}](1+\lambda)^{2n}(2+\lambda)^{kn} \\ &= \frac{1}{n} \sum_{j=0}^{n-1} \binom{2n}{j} \binom{kn}{n-1-j} 2^{n(k-1)+j+1}. \end{aligned}$$

□

We note that when $k = 0$, the caterpillar is a path, so any parking distribution is an increasing classical parking function. We should have $a_{n,0} = C_n$. Indeed, when $k = 0$, the only nonzero term in the sum is for $j = n-1$, so $a_{n,0} = \frac{1}{n}\binom{2n}{n-1} = C_n$, as expected. This result is a generalization of the results in [6], which presents several solutions for the $k = 1$ case.

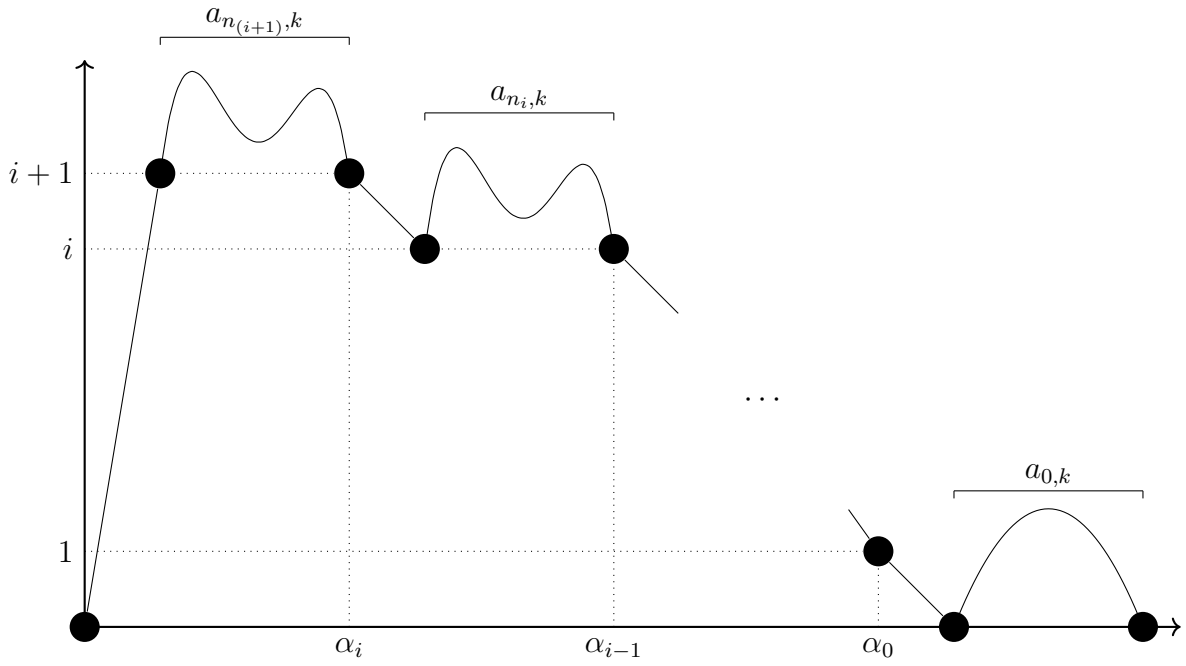

Figure 6.2: The decomposition of $a_{n,k}$.

6.2 Spiders

Recall our definition of and notations concerning spiders from Section 2.1.1. We consider the number of parking distributions and functions on an abritrary spider considered as either a sink and a source tree.

Theorem 16. *Let $S = S(n_1, n_2, \ldots, n_k)$ for $k \geq 1$. Let PF_S, PD_S, PF^S, and PD^S be the number of parking functions and distributions on sink and source tree S, respectively. Then:*

$$PD^S = \prod_{i=1}^{k} C_{(n_i+1)}, \tag{6.2}$$

$$PD_S = \prod_{i=1}^{k} C_{n_i} + \sum_{j=1}^{k} \left(\left(C_{(n_j+1)} - C_{n_j}\right) \cdot \prod_{i \in [k]\setminus\{j\}} C_{n_i} \right), \tag{6.3}$$

$$PF^S = \sum_{j=1}^{n} \left(\sum_{\substack{\sum_i \widetilde{n}_i = n-j \\ 0 \le \widetilde{n}_i \le n_i}} \binom{n}{j, \widetilde{n}_1, \widetilde{n}_2, \ldots, \widetilde{n}_k} \prod_{i=1}^{k} (n_i + 1 - \widetilde{n}_i)(n_i+1)^{\widetilde{n}_i - 1} \right), \tag{6.4}$$

$$PF_S = \binom{n}{1, n_1, n_2, \ldots, n_k} \prod_{i=1}^{k} (n_i+1)^{(n_i-1)} + \sum_{j=1}^{k} \left(\binom{n}{n_1, \ldots, n_j+1, \ldots, n_k} ((n_j+2)^{n_j} - (n_j+1)^{n_j}) \prod_{i \in [k] \setminus \{j\}} (n_i+1)^{(n_i-1)} \right). \tag{6.5}$$

Proof. For Equation (6.2), consider parking distributions for k paths of length n_i+1: $\{(P_i, f_i)\}_{i=1}^k$. Identify the roots, denoted ρ, with each other to form the spider. Define the parking distribution f as follows:

$$f(v) = \begin{cases} f_i(v) & \text{if } v \neq \rho \text{ and } v \in V(P_i) \\ 1 - k + \sum_{i=1}^k f_i(\rho) & \text{if } v = \rho \end{cases}$$

The functions f_i are recoverable by letting $f_i(\rho) = n_i - \sum_{v \in V(P_i) \setminus \{\rho\}} f(v)$ and $f_i(v) = f(v)$ for other $v \in V(P_i)$.

For Equation (6.3), if a driver prefers the root, then there are $\prod_{i=1}^k C_{n_i}$ choices for the drivers preferring the legs. On the other hand, suppose no driver prefers the root. Then one of the legs of S has an additional driver. Suppose it is the one associated with n_j. That leg has a parking distribution of length $n_j + 1$ but the final spot can not be preferred, so we must subtract these C_{n_j} options. For the other legs, choose a parking distribution of length n_i. Finally, sum over $1 \le j \le k$.

For Equation (6.4), we choose $1 \le j \le n$ drivers to prefer the root. We then must select $0 \le \widetilde{n}_i \le n_i$ drivers to prefer each leg such that $\sum_{i=1}^k \widetilde{n}_i = n - j$. Therefore for each leg, we select an $(n_i, \widetilde{n}_i)$-parking function on a path of length n_i. There are $(n_i+1-\widetilde{n}_i)(n_i+1)^{\widetilde{n}_i - 1}$-many choices for such parking functions. Then, we must arrange the sequences in one of $\binom{n}{j, \widetilde{n}_1, \ldots, \widetilde{n}_k}$-many ways.

Finally, for Equation (6.5), if a driver prefers the root, then we select parking functions for each leg and arrange the sequences in $\binom{n}{1, n_1, \ldots, n_k}$-many ways. On the other hand, suppose no driver

prefers the root. One of the legs must have an extra driver attempting to park. For this leg, we choose a parking function of length $n_j + 1$ and subtract out the cases where some driver prefers the root. If a driver prefers the root, there are $(n_j + 1)$-many places for her in the line of cars and the remaining cars must constitute a parking function of length n_j, giving $(n_j + 1)^{n_j}$-many choices. For the other legs, we select a parking function of length n_i.

□

It is good to check that these results match known ones. In the case of $S(n-1)$, we should have $\text{PD}^S = \text{PD}_S = C_n$ and $\text{PF}^S = \text{PF}_S = (n+1)^{(n-1)}$. These are all immediate except perhaps PF^S, which follows from the Binomial Theorem. Figure 6.3 shows an example of $S(2,1,1)$. Using the formulae above, we have $\text{PD}^S = 20$, $\text{PD}_S = 9$, $\text{PF}^S = 631$, and $\text{PF}_S = 500$. The parking distributions are simple enough to check by hand and the parking functions can be checked by counting the permutations of the parking distributions.

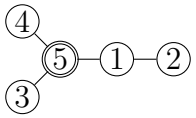

Figure 6.3: A labeling of $S(2,1,1)$.

The formula for PD^S is the simplest of the four, so let us consider summing over all choices of S such that S has n vertices and then sum over n. Define $\text{PD}^s(x) := x + \sum_{n\geq 2} \sum_{k=1}^{n-1} \sum_{\substack{\sum_{i=1}^{k} n_i = n-1, \\ 1 \leq n_i}} \text{PD}^S x^n$.

Corollary 8. *We have*

$$PD^s(x) = \frac{x}{2 - (C(x))^2},$$

where $C(x) = \dfrac{1 - \sqrt{1-4x}}{2x}$*, the Catalan generating function.*

Proof. Since $C(x) = 1 + x(C(x))^2$, we know $\dfrac{C(x) - 1 - x}{x} = (C(x))^2 - 1$. Substituting, we have

$$\begin{aligned}
\text{PD}^s(x) &= x + \sum_{n\geq 2}\sum_{k=1}^{n-1} \sum_{\substack{\sum_{i=1}^{k} n_i = n-1, \\ 1\leq n_i}} \prod_{j=1}^{k} C_{n_j+1} x^n \\
&= x + \sum_{n\geq 2}\sum_{k=1}^{n-1} x[x^{n-1}] \left(\frac{C(x) - 1 - x}{x}\right)^k \\
&= \sum_{n\geq 1}\sum_{k=0}^{n-1} x[x^{n-1}] \left(\frac{C(x) - 1 - x}{x}\right)^k \\
&= x \sum_{k\geq 0} \left((C(x))^2 - 1\right)^k \\
&= \frac{x}{1 - ((C(x))^2 - 1)}.
\end{aligned}$$

□

$\text{PD}^S(x)$ appears in queueing theory as the mixing moments for the waiting time in a M/G/1 waiting queue [1].

7. CONCLUSION AND FINAL REMARKS

This dissertation gives a new generalization of parking functions to directed graphs that opens many new avenues for research, particularly through Theorem 4. In this section, we expand on several of these possibilities.

In Section 4, we discussed the "lucky drivers" statistic, the number of drivers who park in their preferred spot. Several other statistics we may consider are the total distance driven by drivers, the number of distinct driver preferences and the number of spots the most unlucky driver had to check before parking. For digraphs with maximal outdegree larger than 1, one would likely have to define these statistics as the maximal or minimal value among all valid parkings. Another statistic that may yield interesting results is the number of complete squares between the x-axis and the path in a classical parking function's Dyck path representation. In the case of prime parking functions on sink trees, through Theorem 9 and Corollary 4, it seems likely we can associate a prime parking function with a Royal path and two permutations. One may then ask what the area under the Royal path counts or if it is equidistributed with other statistics, as in the classical case.

We discussed partial (or incomplete) parking functions in Section 5, but we could extend that even further to $m \in \mathbb{N}_0$ drivers attempting to park and k failing to do so. In the classical setting, the number of such parking functions gives a proof of a special case of Abel's Binomial Identity, namely

$$(a+b)^b = \sum_{i=0}^{b} \binom{b}{i} \cdot a \cdot (a+i)^{i-1} \cdot (b-i)^{i-1}$$

when $a, b \in \mathbb{N}_0$. See [5] for details.

Finally, we may ask, given a digraph D, what an "average" or random parking function on D looks like. How many drivers are expected to prefer a given vertex or how many distinct preferences do we expect to see? See [7] for a treatment of similar questions on classical parking functions.

REFERENCES

[1] Joseph Abate and Ward Whitt, *Integer sequences from queueing theory*, Journal of Integer Sequences **13** (2010).

[2] Drew Armstrong, Nicholas A. Loehr, and Gregory S. Warrington, *Rational parking functions and catalan numbers*, Annals of Combinatorics **20** (2016), no. 1, 21–58.

[3] Nicolas Borie, *On the combinatorics of quadrant marked mesh patterns in 132-avoiding permutations*, Australasian Journal of Combinatorics **64** (2016), no. 1, 140–153.

[4] Steve Butler, Ron Graham, and Catherine H. Yan, *Parking distributions on trees*, European Journal of Combinatorics **65** (2017), 168 – 185.

[5] Peter J Cameron, Daniel Johannsen, Thomas Prellberg, and Pascal Schweitzer, *Counting defective parking functions*, Electronic Journal of Combinatorics **15** (2008).

[6] Emeric Deutsch, David Callan, M. Beck, D. Beckwith, W. Bohm, R. F. McCoart, and GCHQ Problems Group, *Another type of lattice path: 10658*, The American Mathematical Monthly **107** (2000), no. 4, 368–370.

[7] Persi Diaconis and Angela Hicks, *Probabilizing parking functions*, Advances in Applied Mathematics **89** (2017), 125 – 155.

[8] Paul H. Edelman and Rodica Simion, *Chains in the lattice of noncrossing partitions*, Discrete Mathematics **126** (1994), no. 1, 107 – 119.

[9] Richard Ehrenborg and Alex Happ, *Parking cars of different sizes*, The American Mathematical Monthly **123** (2016), no. 10, 1045–1048.

[10] Ira M. Gessel and Seunghyun Seo, *A refinement of cayley's formula for trees*, Electronic Journal of Combinatorics **11** (2006), no. 2.

[11] Westin King and Catherine H. Yan, *Parking functions on oriented trees*, Séminaire Lotharingien de Combinatoire **80B** (2018), 12 pp.

[12] _____, *Prime parking functions on rooted trees*, Journal of Combinatorial Theory, Series A **168** (2019), 1–25.

[13] Alan G. Konheim and Benjamin Weiss, *An occupancy discipline and applications*, SIAM Journal on Applied Mathematics **14** (1966), no. 6, 1266–1274.

[14] Marie-Louise Lackner and Alois Panholzer, *Parking functions for mappings*, Journal of Combinatorial Theory, Series A **142** (2016), 1 – 28.

[15] Alexander Postnikov and Boris Shapiro, *Trees, parking functions, syzygies, and deformations of monomial ideals*, Transactions of the American Math Society **142** (2004).

[16] Richard Stanley, *Hyperplane arrangements, interval orders, and trees*, Proceedings of the National Academy of Sciences of the United States of America **93** (1996), 2620–5.

[17] Richard P. Stanley and Jim Pitman, *A polytope related to empirical distributions, plane trees, parking functions, and the associahedron*, Discrete & Computational Geometry **27** (2002), no. 4, 603–602.

[18] R.P. Stanley, *Enumerative combinatorics*, vol. 2, Cambridge University Press, 1999.

[19] Catherine H. Yan, *Parking functions*, Handbook of Enumerative Combinatorics (Miklós Bóna, ed.), CRC Press, 2015, pp. 835–894.