Preferential and k-Zone Parking Functions

Parneet Gill, Christopher Soto, Pamela Vargas

Faculty Mentors: Rebecca E. Garcia, Pamela E. Harris, Dwight A. Williams II
Graduate Mentors: Carlos Martinez, Casandra Monroe

October 16, 2021

MSRI

Overview

Parking Functions

- Imagine n cars enter a one-way street consisting of n parking spots and a list of parking preferences p.
- Each car entering has a preferred spot.
- If that spot is empty, then the car parks.
- If the spot is taken, then there are 5 variances of parking rules that the car can follow.

Motivating Questions

- What are the relationships between different parking functions?
- Are there any connections to other combinatorial objects?

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately k spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available spot.

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately k spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available spot.

Classical Parking Functions

Definition

- Each car has a preferred spot which it goes to when entering the street.
- If parking spot is empty, car parks.
- Otherwise it continues down the street until it finds an empty spot to park in.

Consider the following parking preference vector $p = (2, 3, 1, 4)$:

i	p_i	Configuration			
1	2	__	c_1	__	__
2	3	__	c_1	c_2	__
3	1	c_3	c_1	c_2	__
4	4	c_3	c_1	c_2	c_4

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately k spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available spot.

k-Naples Parking Functions

Definition

- Each car prefers a spot (p_i), in which it attempts to park in.
- If spot empty, car parks.
- If occupied, car backs up checking k spots behind it's preferred spot one at a time and parks in first available.
- If there are no empty spots between $p_i - k$ and p_i, then car continues down the street and parks in first available.

Consider the following parking preference vector $p = (4, 4, 3, 2, 4)$:

i	p_i	k	Configuration				
1	4	2	___	___	___	c_1	___
2	4	2	___	___	c_2	c_1	___
3	3	2	___	c_3	c_2	c_1	___
4	2	2	c_4	c_3	c_2	c_1	___
5	4	2	c_4	c_3	c_2	c_1	c_5

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately **k** spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available. spot.

k-Zone Parking Functions

Definition

- Each car prefers a spot (p_i), in which it attempts to park in.
- If spot is empty, car parks.
- If occupied, car backs up immediately k spots behind it's preferred spot and then moves down the street if spot $p_i - k$ is taken.
- Parks in first available spot.

Consider the following parking preference vector $p = (4, 4, 3, 2, 4)$:

i	p_i	k	Configuration
1	4	2	__ __ __ c_1 __
2	4	2	__ c_2 __ c_1 __
3	3	2	__ c_2 c_3 c_1 __
4	2	2	c_4 c_2 c_3 c_1 __
5	4	2	c_4 c_2 c_3 c_1 c_5

k-Zone vs. k-Naples

Conjecture

k-Naples is a subset of k-Zone.

For $k = 2$:

n	k-Naples	k-Zone
1	1	1
2	4	4
3	27	27
4	240	244
5	2,731	2,808
6	38,034	39,416
7	627,405	654,302

Non-Increasing Preference Vectors

Consider the following table of the total number of parking functions formulated from non-increasing preference vector of length n.

For $k = 2$:

n	k-Naples	k-Zone
1	1	1
2	3	3
3	10	10
4	34	34
5	117	117
6	407	407
7	1,430	1,430

Findings From the Table

Theorem

Let $p = (p_1, \ldots, p_n) \in [n]^n$ be a non-increasing preference vector. Then p is a k-Naples if and only if p is a k-Zone.

Proof

We prove this by induction on the total number of cars, from last to first, that switch their parking rule from k-Naples to k-Zone and vice versa.

Enumerating *k*-Zone

Consider the table enumerating the number of k-Zone parking functions of length n and fixed parameter k.

n	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$
1	**1**							
2	3	**4**						
3	16	24	**27**					
4	125	203	244	**256**				
5	1,296	2,225	2,808	3,065	**3,125**			
6	16,807	30,067	39,416	44,424	46,296	**46,656**		
7	262,144	484,071	654,302	757,919	805,543	821,023	**823,543**	
8	4,782,969	9,057,316	12,553,351	14,880,368	16,110,376	16,613,896	16,757,056	**16,777,216**

Findings From the Table (Continued)

Conjecture

If $n \geq 2$ and $0 \leq k \leq n-1$, then $|\mathrm{ZPF}(n, k-1)| - |\mathrm{ZPF}(n, k-2)|$ is equal to:

- *The order of the alternating group A_{n+1}*
- *Number of Hamiltonian cycles on the complete graph, K_n*
- *Number of necklaces with n distinct beads for $n!$ bead permutations.*

Formula:

$$\frac{n!}{2} \tag{1}$$

$$1, 3, 12, 60, 360, 2520, 20160, 181440, \ldots$$

coinciding with the OEIS sequence A001710.

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately k spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available spot.

Preferential Parking Functions

Definition

- Each car prefers a spot (p_i), in which it attempts to park in.
- If spot is empty, car parks.
- Otherwise car checks $n - i$ spots behind p_i, one by one.
- If all the $n - i$ spots preceding p_i are taken, car continues down the street until it finds an available spot to park in.

Consider the following parking preference vector $p = (6, 6, 4, 3, 3, 3)$:

i	p_i	$n - i$	Configuration					
1	6	5	___	___	___	___	___	c_1
2	6	4	___	___	___	___	c_2	c_1
3	4	3	___	___	___	c_3	c_2	c_1
4	3	2	___	___	c_4	c_3	c_2	c_1
5	3	1	___	c_5	c_4	c_3	c_2	c_1
6	3	0	___	c_5	c_4	c_3	c_2	c_1

Types of Parking Rules

1. **Classical Parking Functions:** Checks only forward for available spots.
2. **k-Naples Parking Functions:** Checks k spots backwards one at a time.
3. **k-Zone Parking Functions:** Checks back immediately k spots for availability.
4. **Preferential Parking Functions:** Checks $n - i$ spots back for an available spot.
5. **Inverse Preferential Parking Functions:** Checks $i - 1$ spots back for an available spot.

Inverse Preferential Parking Functions

Definition

- Each car prefers a spot (p_i), in which it attempts to park in.
- If spot is empty, car parks.
- If occupied, car <u>checks $i-1$</u> spots behind p_i, one by one.
- If all $i-1$ spots behind p_i are taken, car continues down the street until it finds an available spot to park in.

Consider the following parking preference vector $p = (6, 6, 4, 3, 3, 3)$:

i	p_i	$i-1$	Configuration					
1	6	0	__	__	__	__	__	c_1
2	6	1	__	__	__	__	c_2	c_1
3	4	2	__	__	__	c_3	c_2	c_1
4	3	3	__	__	c_4	c_3	c_2	c_1
5	3	4	__	c_5	c_4	c_3	c_2	c_1
6	3	5	c_6	c_5	c_4	c_3	c_2	c_1

Preferential Parking Function Findings

Lemma

If $n \geq 2$, then $(n, \ldots, n) \in [n]^n$ is not a preferential parking function.

Proof

By contradiction assuming that all cars can park.

Other Results

Lemma

All preference vectors are inverse preferential parking functions.

Proof

Direct proof.

Acknowledgments

Funding provided by:

Alfred P. Sloan FOUNDATION

NSF

MSRI Mathematical Sciences Research Institute

Thank you to the 2021 GCURS organizers for the opportunity to present our research at GCURS!

Thank you to the audience for their curiosity!