

Freie Universität  Berlin

Freie Universität Berlin / Institut für Mathematik

Erstbetreuer: Prof. Matthias Beck

Zweitbetreuer: Univ.-Prof. Dr. Klaus Altmann

Parkfunktionen und ihre relevanten Objekte

Bachelorarbeit

Eun Bee Lee

Matrikelnummer: 5195289

Mathematik / Philosophie im 8. Semester

Abgabedatum: 21. September 2021

Inhaltsverzeichnis

1	Einleitung	1
2	Parkfunktionen	3
2.1	Eigenschaften von Parkfunktionen	5
2.2	Anzahl von Parkfunktionen der Länge n	10
3	Markierte Bäume	15
3.1	Allgemeines zur Graphentheorie	16
3.2	Parkfunktionen und markierte Bäume	18
4	Prüfer Codes	20
4.1	Bijektion zwischen Prüfer Codes und markierten Bäumen	24
5	Bijektion zwischen Parkfunktionen und Prüfer Codes	28
6	Fazit	31
	Literatur	32

1 Einleitung

Parkfunktionen sind gewisse endliche Folgen natürlicher Zahlen und wurden erstmals im Jahr 1966 von Konheim und Weiss eingeführt [[7], S.1266-1274]. Sie haben als kombinatorisches Objekt Anwendungen in vielen Bereichen; unter anderem in der Kombinatorik, Gruppentheorie und Informatik. Wie ihr Name es bereits andeutet, lassen sie sich am besten durch eine bestimmte Parksituation erklären: Es haben $n \in \mathbb{N}$ Autos auf einer Einbahnstraße mit n Parkplätzen jeweils eine bestimmte Parkpräferenz, und die Präferenzen aller Autos lassen sich als eine Folge darstellen. Beginnend mit dem ersten Auto fahren alle Autos nacheinander zu ihrem gewünschten Parkplatz. Da sich die Autos auf einer Einbahnstraße befinden, ist das Zurückfahren nicht möglich. Ist der präferierte Parkplatz belegt, fährt es, falls möglich, zum nächsten freien Parkplatz. Bekommen am Ende unter Berücksichtigung dieser Gegebenheiten alle Autos einen Parkplatz, wird diese Parkpräferenz als Parkfunktion der Länge n bezeichnet.

In dieser Arbeit werden wir uns mit den *Parkfunktionen* befassen. Dabei interessiert uns vor allem ihre Anzahl, weshalb wir den Schwerpunkt darauf legen werden. Im ersten Teil dieser Arbeit werden wir die Anzahl der Parkfunktionen der Länge n bestimmen. Dieser Teil wird in Kapitel 2 ausführlich behandelt. Im zweiten Teil dieser Arbeit lernen wir zwei weitere mathematische Objekte kennen, die auf dem ersten Blick nichts mit den Parkfunktionen gemeinsam haben: Die *markierten Bäume* und die *Prüfer Codes*. Diese drei Objekte haben gemeinsam, dass sie die selbe Anzahl besitzen. Nachdem wir die markierten Bäume in Kapitel 3, und die Prüfer Codes in Kapitel 4 eingeführt haben, werden wir zeigen, dass ihre Anzahl tatsächlich identisch ist.

Wenn sie die selbe Anzahl besitzen, bedeutet es aber auch, dass es eine Bijektion zwischen ihnen geben muss. Im letzten Teil der Arbeit wollen wir uns genau mit diesen Bijektionen befassen. Die Bijektionen sind besonders interessant, weil wir durch ihnen eine engere

Verbindung zwischen den Objekten aufzeigen können. Somit könnte eine Parkfunktion einem Prüfer Code, und dieser könnte wiederum einem markierten Baum zugeordnet werden. Mit diesen Bijektionen befassen wir uns in Kapitel 4 und 5. Es ist dabei anzumerken, dass wir nicht alle Bijektionen in Ausführlichkeit behandeln werden. Außerdem werden wir zwar eine Bijektion zwischen Parkfunktionen und Prüfer Codes, und zwischen Prüfer Codes und markierten Bäumen herstellen; eine direkte Bijektion zwischen Parkfunktionen und markierten Bäumen jedoch nicht. Um bei den vielen Bijektionen nicht den Überblick zu verlieren, wurde dies in Abbildung 1 bildlich zusammengefasst.

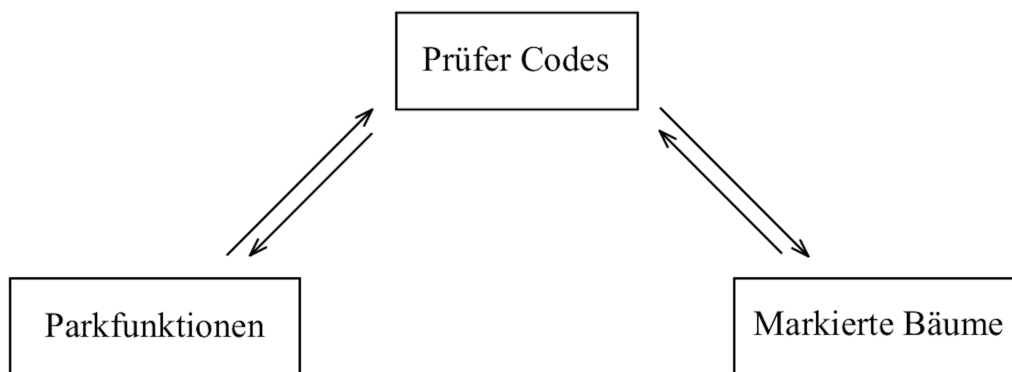


Abbildung 1: Bijektion zwischen Parkfunktionen, Prüfer Codes und markierten Bäumen.

2 Parkfunktionen

Ein Hauptbestandteil dieser Arbeit wird sein, die Anzahl der Parkfunktionen zu bestimmen. Daher wollen wir in diesem Kapitel erstmal wichtige Eigenschaften von Parkfunktionen feststellen, um dann eine angemessene Definition für die Parkfunktion geben zu können. Zunächst wollen wir uns erstmal ein Bild von den Parkfunktionen machen. Um ein intuitives Verständnis von Parkfunktionen zu bekommen, sollen sie an einem situativen Beispiel erläutert werden. Dabei seien $n \in \mathbb{N}$ Autos auf einer Einbahnstraße mit n Parkplätzen gegeben, wobei die Autos und Parkplätze jeweils folgendermaßen nummeriert werden sollen: Das vorderste Auto bekommt die Nummer 1, und mit aufsteigender Nummerierung bekommt das letzte Auto die Nummer n . Ähnlich erfolgt die Nummerierung der Parkplätze; der in der Fahrtrichtung nächste Parkplatz bekommt die Nummer 1, und mit aufsteigender Nummerierung bekommt der letzte Parkplatz die Nummer n [[3], S. 1]. Die Verbildlichung der Parksituation ist in Abbildung 2 dargestellt.

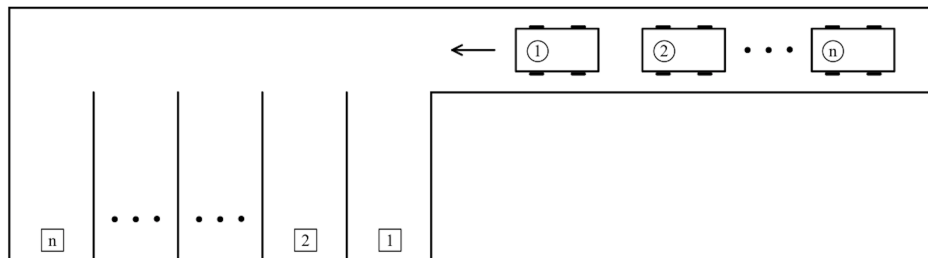


Abbildung 2: Einbahnstraße mit n Autos und n Parkplätzen.

Alle Autos möchten zu einem Parkplatz, wobei jedes Auto jeweils eine eigene Parkpräferenz hat. Die Parkpräferenzen der jeweiligen Autos lassen sich als eine Folge (p_1, \dots, p_n) in $[n]^n$ mit $[n]$ darstellen. Hierfür stehe der Index i für die Nummer des Autos und das Folgeglied p_i für die Parkpräferenz des i -ten Autos. Beginnend mit dem ersten Auto fahren alle

Autos nacheinander zu ihrem gewünschten Parkplatz; ist er belegt, fährt es zum nächsten freien Parkplatz. Da sich die Autos auf einer Einbahnstraße befinden, ist das Zurückfahren nicht möglich. Somit kann die Nummer des gewünschten Parkplatzes niemals größer als die des tatsächlichen Parkplatzes sein. Bekommen am Ende unter Berücksichtigung der Parkpräferenz alle Autos einen Parkplatz, wird die Folge dieser Parkpräferenz als *Parkfunktion der Länge n* bezeichnet. Wir bezeichnen PF_n als Menge aller Parkfunktionen der Länge n .

Für $n = 3$ gilt beispielsweise $(1, 3, 2) \in PF_3$, da alle Autos am Ende einen Parkplatz bekommen: Das erste Auto fährt zum ersten Parkplatz, das zweite Auto zum dritten, und das dritte Auto zum zweiten Parkplatz. Hingegen gilt $(2, 2, 2) \notin PF_3$: Wenn alle Autos zum zweiten Parkplatz möchten, bekommt das dritte Auto keinen Parkplatz mehr; der zweite Parkplatz wurde nämlich vom ersten, und der dritte Parkplatz vom zweiten Auto belegt. Dieser Zusammenhang wird in Abbildung 3 dargestellt.

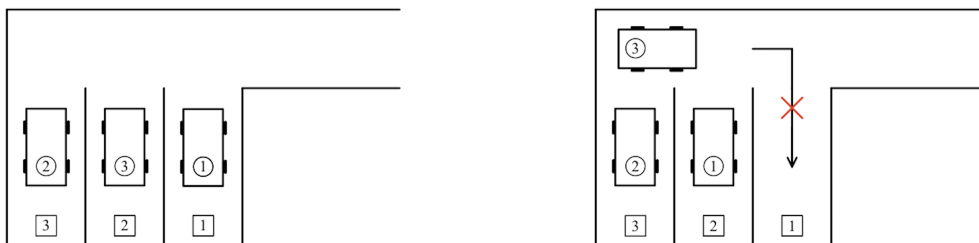


Abbildung 3: Visuelle Darstellung von $(1, 3, 2) \in PF_3$ (links) und $(2, 2, 2) \notin PF_3$ (rechts).

2.1 Eigenschaften von Parkfunktionen

Nachdem wir ein intuitives Verständnis von Parkfunktionen bekommen haben, wollen wir in diesem Abschnitt untersuchen, unter welcher Bedingung eine beliebige Folge $(p_1, \dots, p_n) \in [n]^n$ überhaupt Parkfunktion ist. Dafür wollen wir erste Eigenschaften von Parkfunktionen feststellen und schauen uns zunächst alle Folgen in $[n]^n$ für $n \in [3]$ an:

n	$[n]^n$
1	(1)
2	(1,1) (1,2) (2,1) (2,2)
3	(1,1,1) (1,1,2) (1,2,1) (2,1,1) (1,1,3) (1,3,1) (3,1,1) (1,2,2) (2,1,2) (2,2,1) (1,2,3) (1,3,2) (2,1,3) (2,3,1) (3,1,2) (3,2,1) (1,3,3) (3,1,3) (3,3,1) (2,2,2) (2,2,3) (2,3,2) (3,2,2) (2,3,3) (3,2,3) (3,3,2) (3,3,3)

Tabelle 1: Folgen in $[1]$, $[2]^2$ und $[3]^3$.

Zunächst fällt uns auf, dass einige Folgen Permutationen einer anderen sind. Diese Folgen haben wir bereits in der obigen Liste nebeneinander gestellt. In jeder Gruppe gibt es zudem eine Folge (p_1, \dots, p_n) mit der Eigenschaft $p_1 \leq p_2 \leq \dots \leq p_n$. In der Liste wurden

diese Folgen **fett** markiert. Folgen dieser Art wollen wir als *Repräsentanten* ihrer Gruppe bezeichnen; die Menge aller Repräsentanten nennen wir PR_n . Nun streichen wir alle Folgen aus Tabelle 1 durch, die keine Parkfunktionen sind:

n	$[n]^n$
1	(1)
2	(1,1) (1,2) (2,1) (2,2)
3	(1,1,1) (1,1,2) (1,2,1) (2,1,1) (1,1,3) (1,3,1) (3,1,1) (1,2,2) (2,1,2) (2,2,1) (1,2,3) (1,3,2) (2,1,3) (2,3,1) (3,1,2) (3,2,1) (1,3,3) (3,1,3) (3,3,1) (2,2,2) (2,2,3) (2,3,2) (3,2,2) (2,3,3) (3,2,3) (3,3,2) (3,3,3)

Tabelle 2: Parkfunktionen der Länge 1, 2 und 3.

Nachdem wir uns beispielhaft alle Parkfunktionen der Längen 1, 2 und 3 angeschaut haben, halten wir folgende Eigenschaften fest: Zum einen ist eine Folge $(p_1, \dots, p_n) \in \text{PR}_n$ genau dann eine Parkfunktion, wenn $p_i \leq i$ für jedes $i \in [n]$ gilt, und zum anderen sind alle Folgen aus der selben Gruppe Parkfunktionen, wenn ihr Repräsentant auch eine Parkfunktion ist. Diese Feststellung wollen wir nun allgemein für alle $n \in \mathbb{N}$ festhalten:

Satz 2.1. Für jedes beliebige $(p_1, \dots, p_n) \in \text{PR}_n$ und $i \in [n]$ gilt $(p_1, \dots, p_n) \in \text{PF}_n \Leftrightarrow p_i \leq i$.

Beweis. Seien $(p_1, \dots, p_n) \in \text{PR}_n$ und $i \in [n]$ beliebig. Zunächst wollen wir zeigen, dass wenn $p_i \leq i$ gilt, die Folge (p_1, \dots, p_n) dann eine Parkfunktion sein muss. Der Beweis erfolgt dabei über die vollständige Induktion, also ist die folgende Aussage für ein beliebiges n zu zeigen:

$A(n) = \text{„Wenn } p_i \leq i \text{ gilt, ist } (p_1, \dots, p_n) \text{ eine Parkfunktion.“}$

Sei hierzu $n_0 := 1$ und betrachte ein beliebiges $(p_1) \in \text{PR}_1$. Wenn $p_1 \leq 1$ gilt, kommt nur $p_1 = 1$ in Frage und wie wir auf dem obigen Beispiel gesehen haben, ist (1) eine Parkfunktion der Länge 1. Somit ist $A(n_0)$ wahr. Sei nun die Aussage $A(n)$ für ein beliebiges $n_0 \leq n$ wahr. Es ist zu zeigen, dass $A(n+1)$ dann auch wahr sein muss. Betrachte dafür ein beliebiges $(p_1, \dots, p_{n+1}) \in \text{PR}_{n+1}$ und nehme an, dass $p_i \leq i$ für jedes beliebige $i \in [n+1]$ gilt. Da $A(n)$ wahr ist, müssen die ersten n Autos jeweils genau einen der ersten n Parkplätze belegt haben. Somit hat das $(n+1)$ -te Auto unabhängig von seiner Parkpräferenz keine andere Wahl, als beim nächstfreien, also beim $(n+1)$ -ten Parkplatz, zu parken. Daher haben durch (p_1, \dots, p_n) alle Autos einen Parkplatz bekommen und es gilt $(p_1, \dots, p_{n+1}) \in \text{PF}_n$. Somit wurde $A(n+1)$ gezeigt.

Nun zeigen wir, dass wenn (p_1, \dots, p_n) eine Parkfunktion ist, dann $p_i \leq i$ für jedes $i \in [n]$ gelten muss. Hier führen wir einen Beweis per Kontraposition durch, also zeigen wir, dass wenn es in einer beliebigen Folge $(p_1, \dots, p_n) \in \text{PR}_n$ einen Index $i \in [n]$ mit der Eigenschaft $p_i > i$ gibt, diese Folge dann keine Parkfunktion sein kann.

Sei $(p_1, \dots, p_n) \in \text{PR}_n$ beliebig und nehme an, es gibt in dieser Folge mindestens einen Index, der kleiner als sein Folglied ist. Nenne ihn $l \in [n]$, für jeden m mit $m \geq l$ gilt dann $p_m > l$ und der l -te Parkplatz kann nicht vom m -ten Auto belegt werden. Da l der

kleinste Index ist, der kleiner als sein Folglied ist, gilt $p_k \leq k$ für jeden Index k mit $k < l$, also $k \in [l-1]$. Das bedeutet aber, dass der l -te Parkplatz vom k -ten Auto ebenfalls nicht belegt werden kann, denn die ersten $l-1$ Autos belegen nach dem obigen Beweis jeweils genau einen der ersten $l-1$ Parkplätze. Daher kann der l -te Parkplatz von keinem Auto belegt werden, was wiederum bedeutet, dass am Ende mindestens ein Auto keinen Parkplatz bekommen hat. Somit kann (p_1, \dots, p_n) keine Parkfunktion sein. \square

Satz 2.2. Sei $(q_1, \dots, q_n) \in [n]^n$ eine beliebige Permutation von $(p_1, \dots, p_n) \in \text{PR}_n$, dann gilt $(p_1, \dots, p_n) \in \text{PF}_n \Rightarrow (q_1, \dots, q_n) \in \text{PF}_n$.

Beweis. Sei (q_1, \dots, q_n) eine beliebige Permutation von $(p_1, \dots, p_n) \in \text{PR}_n$ und nehme an, dass (p_1, \dots, p_n) eine Parkfunktion ist. Zu zeigen ist, dass (q_1, \dots, q_n) dann auch eine Parkfunktion sein muss. Bekanntlich ist eine Folge genau dann eine Parkfunktion, wenn am Ende alle Parkplätze belegt sind. Für einzelne Parkplätze müssen dann folgende Bedingungen erfüllt sein:

Der 1. Parkplatz wird belegt, wenn es mindestens ein q_i mit $q_i = 1$ gibt. Diese Bedingung ist erfüllt; nach Satz 2.1 gilt $p_1 = 1$, da (p_1, \dots, p_n) eine Parkfunktion ist. Somit gibt es ein q_i mit $q_i = p_1 = 1$, wodurch der 1. Parkplatz mit Sicherheit belegt wird.

Der 2. Parkplatz wird unter der Annahme, dass der 1. Parkplatz bereits besetzt ist, belegt, wenn es mindestens ein q_i mit $q_i \leq 2$ gibt. Diese Bedingung ist erfüllt; nach Satz 2.1 gilt $p_2 \leq 2$, da (p_1, \dots, p_n) eine Parkfunktion ist. Somit gibt es ein q_i mit $q_i = p_2 \leq 2$, wodurch der 2. Parkplatz mit Sicherheit belegt wird.

Wir können die Aussage für einen beliebigen Parkplatz ausweiten: Sei dafür $k \in [n]$ beliebig, der k -te Parkplatz wird unter der Annahme, dass die ersten $k-1$ Parkplätze bereits besetzt sind, belegt, wenn es mindestens ein q_i mit $q_i \leq k$ gibt. Diese Bedingung ist erfüllt; nach Satz 2.1 gilt $p_k \leq k$, da (p_1, \dots, p_n) eine Parkfunktion ist. Somit gibt es ein q_i mit $q_i = p_k \leq k$, wodurch der k -te Parkplatz mit Sicherheit belegt wird. Somit bekommen

alle Autos durch die Parkpräferenz (q_1, \dots, q_n) einen Parkplatz, weshalb (q_1, \dots, q_n) eine Parkfunktion sein muss. □

Aus den Sätzen 2.1 und 2.2 können wir das folgende Korollar ableiten:

Korollar 2.1. Jede beliebige Permutation einer ist ebenfalls eine Parkfunktion.

2.2 Anzahl von Parkfunktionen der Länge n

Nachdem wir die Eigenschaften von Parkfunktionen festgestellt haben, wollen wir in diesem Abschnitt endlich ihre Anzahl bestimmen. Dabei benutzen wir die Beweisidee von Pollak, wo eine neue Variante der Parkfunktion, und zwar die *zirkulären Parkfunktionen* als Hilfsmittel eingeführt werden [[1], S.1; [6], S.13]. Seien dafür wieder n Autos gegeben; diese befinden sich aber dieses Mal nicht auf einer Einbahnstraße mit n Parkplätzen, sondern in einem einspurigen Kreisverkehr mit $n + 1$ Parkplätzen. Die Verbildlichung der Parksituation ist in Abbildung 4 dargestellt.

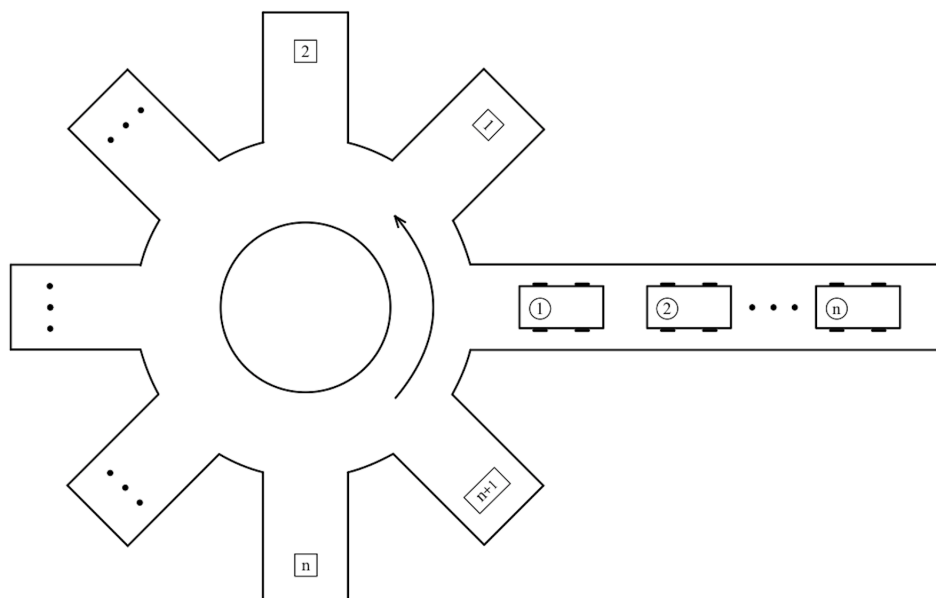


Abbildung 4: Einspuriger Kreisverkehr mit n Autos und $n + 1$ Parkplätzen.

Hier wollen ebenso alle Autos zu einem Parkplatz, wobei auch diesmal jedes Auto eine Parkpräferenz hat. Auch hier lässt sich die Parkpräferenz der n Autos als eine Folge (p_1, \dots, p_n) in $[n + 1]^n$ darstellen. Da sich die Autos in einem Kreisverkehr befinden, finden sie im Gegensatz zur Einbahnstraße in jedem Fall einen Parkplatz, denn falls der $(n + 1)$ -te

Parkplatz belegt ist, können sie nämlich wieder zum ersten Parkplatz fahren und von dort aus den nächstfreien Parkplatz suchen. Somit ist jede Parkpräferenz auch eine zirkuläre Parkfunktion und die Menge aller zirkulären Parkfunktionen der Länge n ist somit $[n + 1]^n$. Ihr Anzahl lässt sich leicht ableiten, es gibt $(n + 1)^n$ zirkuläre Parkfunktionen der Länge n .

Eine Eigenschaft der zirkulären Parkfunktionen ist, dass am Ende genau ein Parkplatz frei bleibt, da sich n Autos auf $n + 1$ Parkplätzen verteilen. Nenne $ZP_{k,n}$ für $k \in [n + 1]$ die Menge aller zirkulären Parkfunktionen der Länge n , die den k -ten Parkplatz frei lassen. Nun wollen wir sie unter Betrachtung dieser Eigenschaft analysieren. Hierzu listen wir wieder alle Funktionen der Länge n beispielhaft für $n \in [3]$ auf, wobei wir die Funktionen, die denselben Parkplatz freilassen, zusammensortieren:

Freier Parkplatz	$ZP_{k,n}$
1. Parkplatz ($ZP_{1,1}$)	(2)
2. Parkplatz ($ZP_{2,1}$)	(1)
1. Parkplatz ($ZP_{1,2}$)	(2,2) (2,3) (3,2)
2. Parkplatz ($ZP_{2,2}$)	(1,3) (3,1) (3,3)
3. Parkplatz ($ZP_{3,2}$)	(1,1) (1,2) (2,1)
1. Parkplatz ($ZP_{1,3}$)	(2,2,2) (2,2,3) (2,2,4) (2,3,2) (2,3,3) (2,3,4) (2,4,2) (2,4,3) (3,2,2) (3,2,3) (3,2,4) (3,3,2) (3,4,2) (4,2,2) (4,2,3) (4,3,2)
2. Parkplatz ($ZP_{2,3}$)	(1,3,3) (1,3,4) (1,4,3) (3,1,3) (3,1,4) (3,3,1) (3,3,3) (3,3,4) (3,4,1) (3,4,3) (3,4,4) (4,1,3) (4,3,1) (4,3,3) (4,3,4) (4,4,3)
3. Parkplatz ($ZP_{3,3}$)	(1,1,4) (1,2,4) (1,4,1) (1,4,2) (1,4,4) (2,1,4) (2,4,1) (2,4,4) (4,1,1) (4,1,2) (4,1,4) (4,2,1) (4,2,4) (4,4,1) (4,4,2) (4,4,4)
4. Parkplatz ($ZP_{4,3}$)	(1,1,1) (1,1,2) (1,1,3) (1,2,1) (1,2,2) (1,2,3) (1,3,1) (1,3,2) (2,1,1) (2,1,2) (2,1,3) (2,2,1) (2,3,1) (3,1,1) (3,1,2) (3,2,1)

Tabelle 3: Zirkuläre Parkfunktionen der Länge 1, 2 und 3.

Uns fällt für $n \in [3]$ auf, dass die Anzahl der zirkulären Parkfunktionen, die denselben Parkplatz frei lassen, unabhängig vom freistehenden Parkplatz immer gleich ist. Zudem stellt man nach der Umsortierung eine weitere Besonderheit fest: Funktionen in $ZP_{n+1,n}$ sind identisch mit Parkfunktionen der Länge n . Diese Feststellungen werden wir im Folgenden allgemein für alle n zeigen. Doch vorher wollen wir eine neue Notation einführen, die wir im weiteren Verlauf öfters benutzen werden.

Bemerkung 2.1. Bekanntlich berechnet der *Modulo* den Rest bei einer Division. Haben wir ganze Zahlen a, b, m und r mit $m \neq 0$ gegeben, so dass

$$a = b \cdot m + r,$$

dann wird der Rest r als $a \bmod m$ bezeichnet [[4], S.83]. Wir können ihn auch als eine Abbildung darstellen, der folgend gegeben ist:

$$\bmod : \mathbb{Z} \times (\mathbb{Z} \setminus \{0\}) \rightarrow \mathbb{Z}, (a, m) \mapsto a \bmod m = a - \left\lfloor \frac{a}{m} \right\rfloor \cdot m.$$

Beispielsweise haben wir $6 \bmod 3 = 0$ und es gilt $(5, 2, 6) \bmod 3 = (2, 2, 0)$. Wir führen nun eine andere Variante des Modulos ein, der als Rest m statt 0 angibt. Nennen wir ihn \bmod^* , dann ist er folgend gegeben:

$$\bmod^* : \mathbb{Z} \times (\mathbb{Z} \setminus \{0\}) \rightarrow \mathbb{Z}, (a, m) \mapsto a \bmod^* m = \begin{cases} m, & \text{falls } a \bmod m = 0 \\ a \bmod m, & \text{sonst.} \end{cases}$$

Somit haben wir anders als beim gewohnten Modulo dann $6 \bmod^* 3 = 3$ und es gilt $(5, 2, 6) \bmod^* 3 = (2, 2, 3)$.

Satz 2.3. Für ein beliebiges $k \in [n+1]$ gilt $|ZP_{k,n}| = (n+1)^{n-1}$.

Beweis. Sei $(p_1, \dots, p_n) \in ZP_{k,n}$ beliebig und weiter sei $\overline{k+a} := k + a \bmod^* n + 1$. Betrachte die Folge $(p_1 + a, \dots, p_n + a) \bmod^* n + 1$ für ein beliebiges $a \in [n+1]$, es gilt dann $(p_1 + a, \dots, p_n + a) \in ZP_{\overline{k+a},n}$: Durch die Verschiebung jeder Parkpräferenz um a

ebenfalls jeder belegte Parkplatz und demnach auch der am Ende frei bleibende Parkplatz um a verschoben wird.

Zudem ist die Abbildung $f : \text{ZP}_{k,n} \rightarrow \text{ZP}_{\overline{k+a},n}$ mit $f((p_1, \dots, p_n)) := (p_1 + a, \dots, p_n + a) \bmod^*(n+1)$ bijektiv: Betrachte die Abbildung $g : \text{ZP}_{\overline{k+a},n} \rightarrow \text{ZP}_{k,n}$ mit $g((p_1, \dots, p_n)) = (p_1 - a, \dots, p_n - a) \bmod^*(n+1)$, dann gilt $g(f((p_1, \dots, p_n))) = g((p_1 + a, \dots, p_n + a)) = (p_1, \dots, p_n)$, also $g \circ f = id$. Außerdem haben wir $f(g((p_1, \dots, p_n))) = g((p_1 - a, \dots, p_n - a)) = (p_1, \dots, p_n)$, also $f \circ g = id$. Somit gilt $|\text{ZP}_{k,n}| = |\text{ZP}_{\overline{k+a},n}|$. Da $\overline{k+a} \in [n+1]$ gilt und es $(n+1)^n$ zirkuläre Parkfunktionen gibt, können wir letztlich auf $|\text{ZP}_{k,n}| = \frac{(n+1)^n}{(n+1)} = (n+1)^{n-1}$ schließen. \square

Satz 2.4. Es gilt $\text{ZP}_{(n+1),n} = \text{PF}_n$.

Beweis. Jede Parkfunktion der Länge n ist auch eine Funktion in $\text{ZP}_{n+1,n}$, da durch ihnen die ersten n Parkplätze belegt werden und am Ende der $(n+1)$ -te Parkplatz freibleibt. Somit gilt $\text{PF}_n \subseteq \text{ZP}_{n+1,n}$. Zudem ist jede Folge $(p_1, \dots, p_n) \in \text{ZP}_{n+1,n}$ eine Parkfunktion der Länge n : Wenn durch (p_1, \dots, p_n) der letzte, also der $(n+1)$ -te Parkplatz freibleibt, bedeutet es, dass kein Auto eine weitere Runde drehen musste, um einen Parkplatz zu belegen, da nach den Eigenschaften der zirkulären Parkfunktionen nur dann eine weitere Runde gedreht wird, wenn bereits der letzte Parkplatz belegt ist. Somit bekommt jedes Autos durch (p_1, \dots, p_n) auch auf einer einspurigen Straße jeweils einen Parkplatz, weshalb $\text{ZP}_{n+1,n} \subseteq \text{PF}_n$ gelten muss. Aus $\text{PF}_n \subseteq \text{ZP}_{n+1,n}$ und $\text{ZP}_{n+1,n} \subseteq \text{PF}_n$ können wir auf $\text{ZP}_{n+1,n} = \text{PF}_n$ schließen. \square

Mithilfe dieser Sätze können wir nun endlich auf die Anzahl der Parkfunktionen der Länge n schließen:

Korollar 2.2. Die Anzahl der Parkfunktionen der Länge n ist $(n+1)^{n-1}$.

3 Markierte Bäume

Im letzten Kapitel haben wir die Anzahl der Parkfunktionen der Länge n bestimmt. Dazu war es zunächst notwendig zu verstehen, was eine Parkfunktion ist. Nachdem wir die Definition von Parkfunktionen formuliert haben, konnten wir letztlich die Anzahl der Parkfunktionen der Länge n bestimmen; ihre Anzahl ist $(n + 1)^{n-1}$.

In diesem Kapitel wollen wir ein weiteres mathematisches Objekt kennenlernen, und zwar die *markierten Bäume*. Markierte Bäume scheinen auf dem ersten Blick nichts mit den Parkfunktionen gemeinsam zu haben, doch wir bereits in der Einleitung erwähnt, ist ihre Anzahl mit der Anzahl der Parkfunktionen identisch. Doch bevor wir über ihre Anzahl sprechen, brauchen wir eine angemessene Definition von einem Baum. Dafür führen wir zunächst einige relevante Begriffe aus der Graphentheorie ein, um anschließend eine Definition von markierten Bäumen geben zu können.

3.1 Allgemeines zur Graphentheorie

Als Erstes beginnen wir mit dem Wesentlichen, und zwar mit den *Graphen*. Zunächst ist es zu erwähnen, dass es zwischen *gerichteten* und *ungerichteten Graphen* unterschieden wird. Jedoch wird die Unterscheidung in dieser Arbeit nicht aufgegriffen, da wir in unserer Arbeit nur ungerichtete Graphen behandeln werden. Somit meinen wir im Folgenden mit „Graphen“ stets „ungerichtete Graphen“. Die folgenden Definitionen stammen aus [[5], S. 2-17].

Definition 3.1. Sei V eine endliche Menge und E eine Teilmenge aller zweielementigen Teilmengen von V . Ein *Graph* G ist ein geordnetes Paar $G = (V, E)$, wobei V als eine Menge an *Knoten*, und E als eine Menge an *Kanten* zu verstehen ist.

Seien $v_i, v_j \in V$ beliebige Knoten, dann wird die Kante $\{v_i, v_j\} \in E$ als eine Verbindungslinie zwischen v_i und v_j dargestellt.

Als nächstes schauen wir uns weitere Begriffe aus der Graphentheorie an, die für den weiteren Verlauf wesentlich sein werden.

Definition 3.2. Sei ein Graph $G = (V, E)$ mit der Knotenmenge V und der Kantenmenge E gegeben. Ein *Kantenzug* Z in G ist eine Knotenfolge $Z = (v_1, v_2, \dots, v_k)$ mit $\{v_{i-1}, v_i\} \in E$ für $i \in [k] \setminus \{1\}$.

Definition 3.3. Sei ein Graph $G = (V, E)$ mit der Knotenmenge V und der Kantenmenge E gegeben. Ein *Kreis* K ist ein Kantenzug $K = (v_1, v_2, \dots, v_{k+1})$ in $G = (V, E)$ mit $k \geq 3$ und $v_1 = v_{k+1}$.

Schreibweise: $K = (v_1, v_2, \dots, v_k, v_1)$.

Definition 3.4. Ein nichtleerer Graph $G = (V, E)$ heißt *zusammenhängend*, wenn es für zwei beliebige Knoten $v_i, v_j \in V$ mit $v_i \neq v_j$ einen Kantenzug $Z = (v_i, v_1, v_2, \dots, v_k, v_j)$

gibt.

Nun haben wir alle relevanten Begriffe kennengelernt und können endlich eine kompakte Definition von einem *Baum* geben.

Definition 3.5. Ein *Baum* T ist ein zusammenhängender Graph $T = (V, E)$, der keinen Kreis enthält.

Es ist anzumerken, dass es in der Regel zwischen *markierten* und *nicht markierten Bäumen* unterschieden wird. In unserer Arbeit werden aber nur die markierten Bäume von Relevanz sein. Die Knoten eines markierten Baums werden bezeichnet und somit voneinander unterschieden. In der Regel, und insbesondere in dieser Arbeit, werden wir die Knoten beginnend mit 1 nummerieren. In Abbildung 5 wird ein markierter Baum dargestellt.

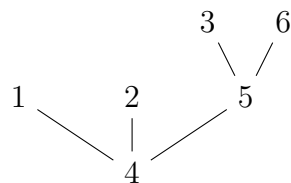


Abbildung 5: Markierter Baum $T = (V, E)$ mit der Knotenmenge $V = [6]$ und der Kantenmenge $E = \{\{1, 4\}, \{2, 4\}, \{3, 5\}, \{4, 5\}, \{5, 6\}\}$.

3.2 Parkfunktionen und markierte Bäume

Im letzten Abschnitt haben wir eine Definition für einen markierten Baum formuliert. Nun können ihn genauer untersuchen. Schauen wir uns dafür einige Bäume beispielhaft an und vergleichen diese mit den Parkfunktionen, um erste Eindrücke über ihre Gemeinsamkeiten gewinnen zu können. Dabei listen wir alle Parkfunktionen der Länge n und markierte Bäume mit $n + 1$ Knoten für $n \in [3]$ auf.

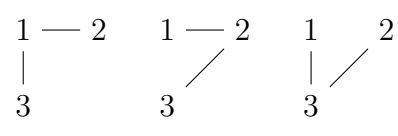
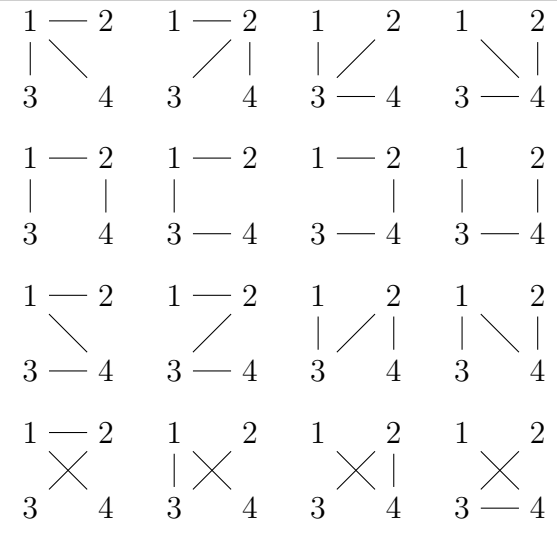
n	Parkfunktionen der Länge n	Bäume mit $n + 1$ Knoten
1	(1)	1 — 2
2	(1,1) (1,2) (2,2)	
3	(1,1,1) (1,1,2) (1,2,1) (2,1,1) (1,1,3) (1,3,1) (3,1,1) (1,2,2) (2,1,2) (2,2,1) (1,2,3) (1,3,2) (2,1,3) (2,3,1) (3,1,2) (3,2,1)	

Tabelle 4: Parkfunktionen der Länge n und markierte Bäume mit $n + 1$ Knoten für $n \in [3]$.

Uns fällt zumindest für $n \in [3]$ auf, dass die Anzahl der markierten Bäume mit $n + 1$ Knoten tatsächlich mit der Anzahl der Parkfunktionen der Länge n identisch ist. Nun wollen wir diese Feststellung auch für alle n zeigen.

Wir führen dafür ein weiteres mathematisches Objekt ein, und zwar die *Prüfer Codes*.

Prüfer Codes sind wie Parkfunktionen ebenfalls endliche Folgen natürlicher Zahlen und wir werden im nächsten Kapitel relativ schnell feststellen, dass die Anzahl der Prüfer Codes der Länge $n - 1$ auch $(n + 1)^{n-1}$ ist. Daher wollen wir im weiteren Verlauf eine Bijektion zwischen ihnen und den markierten Bäumen herstellen, um dann zeigen zu können, dass die Anzahl der markierten Bäume mit $n + 1$ Knoten ebenfalls $(n + 1)^{n-1}$ ist. Dafür müssen wir zunächst verstehen, was genau Prüfer Codes sind; daher werden wir die Prüfer Codes im nächsten Kapitel einführen und diese genauer untersuchen.

4 Prüfer Codes

Prüfer Codes wurden 1918 von Prüfer eingeführt, um die *Cayley-Formel* zu beweisen [[8], S.142-144]. Die Cayley-Formel besagt, dass n^{n-2} die Anzahl der markierten Bäume mit n Knoten ist. Dabei handelt es sich bei einem Prüfer Code der Länge $n - 2$ um eine Folge (c_1, \dots, c_{n-2}) in $[n]^{n-2}$. Dass n^{n-2} die Anzahl aller Prüfer Codes der Länge $n - 2$ ist, ist ohne weiteres nachvollziehbar. Durch die eindeutige Zuordnung eines markierten Baums mit einem Prüfer Code zeigt er ihre Bijektivität. Prüfer gibt dabei zwei Konstruktionsvorschriften vor; zum einen, wie man einen markierten Baum zu einem Prüfer Code umwandelt, und zum anderen ihre Rückrichtung. In diesem Kapitel werden wir uns beide Konstruktionsvorschriften anschauen und anschließend ihre Bijektivität zeigen, um die Cayley-Formel beweisen zu können. Ist die Cayley-Formel gezeigt, können wir daraus leicht ableiten, dass $(n + 1)^{n-1}$ die Anzahl der markierten Bäume mit $n + 1$ Knoten ist.

Bevor wir uns die beiden Konstruktionsvorschriften anschauen, klären wir zunächst einige Begriffe, die im weiteren Verlauf von Relevanz sein werden.

Definition 4.1. Sei $T = (V, E)$ ein Baum mit der Knotenmenge $V = [n]$ und der Kantenmenge E . Der *Grad eines Knoten v* gibt die Anzahl der Knoten an, die in T mit v verbunden sind.

Schreibweise: $\deg_T(v)$.

Definition 4.2. Sei $T = (V, E)$ ein beliebiger Baum mit der Knotenmenge $V = [n]$ und der Kantenmenge E . Als *Blätter* werden Knoten bezeichnet, die den Grad 1 haben. Die *Menge aller Blätter* bezeichnen wir als $B = \{v \in V \mid \deg_T(v) = 1\} \subseteq V$.

Als Beispiel schauen wir uns den Baum $T = (V, E)$ mit der Knotenmenge $V := [5]$ und der Kantenmenge $E := \{\{1, 2\}, \{1, 4\}, \{1, 5\}, \{2, 3\}\}$ an. In Abbildung 6 wurden die Blätter jeweils **fett** markiert.

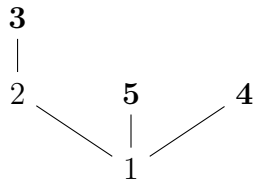


Abbildung 6: Markierter Baum $T = (V, E)$ mit der Blattmenge $B = \{3, 4, 5\}$.

Die Knoten 1 und 2 sind keine Blätter, denn ihr Grad ist jeweils $\deg_T(1) = 3$ und $\deg_T(2) = 2$. Aus Definition 4.2 leitet sich das folgende Korollar ab.

Korollar 4.1. Sei $T = (V, E)$ ein Baum mit der Knotenmenge $V = [n]$ und der Kantenmenge E . Für jedes beliebige Blatt $v \in B$ existiert genau ein Knoten $w \in V$, so dass $\{v, w\} \in E$.

Wir wollen nun einen beliebigen Baum mit der von Prüfer vorgeschlagenen Konstruktionsvorschrift in einen Prüfercode umwandeln.

Konstruktionsvorschrift 1. Sei ein Baum $T = (V, E)$ mit der Knotenmenge $V = [n]$ für $n \geq 3$ und der Kantenmenge E gegeben. Weiter sei c zunächst eine leere Folge. Beginne mit Schritt:

1. Sei $v := \min(B)$ und sei $w \in V$ der nach Korollar 4.1 eindeutig gegebene Knoten, wo $\{v, w\} \in E$ gilt. Wir nennen $c_1 := w$.
2. Setze $V' := V \setminus \{v\}$, $E' := E \setminus \{v, w\}$ und $B' := B \setminus \{v_i\}$, und ergänze c_1 zur Folge c .
3. Falls $|V'| \geq 2$, dann setze $V \leftarrow V'$, $E \leftarrow E'$, $B \leftarrow B'$, und $c_1 \leftarrow c_2$. Gehe anschließend zurück zum Schritt 1. Falls $|V'| = 2$, dann haben wir unseren Prüfer Code c .

Die Konstruktion soll nun an einem Beispiel durchgeführt werden. Betrachte dafür den Baum $T = (V, E)$ aus Abbildung 5, wir wollen ihn mithilfe der Konstruktionsvorschrift 1 in einen Prüfer Code umwandeln. Die Schrittweise Umwandlung ist in Tabelle 5 dargestellt.

	$1 = \min(B) \Rightarrow c_1 = 4$	$c = (4, c_2, c_3, c_4)$
	$2 = \min(B) \Rightarrow c_2 = 4$	$c = (4, 4, c_3, c_4)$
	$3 = \min(B) \Rightarrow c_3 = 5$	$c = (4, 4, 5, c_4)$
	$4 = \min(B) \Rightarrow c_4 = 5$	$c = (4, 4, 5, 5)$

Tabelle 5: Schrittweise Umwandlung eines markierten Baums zu einem Prüfer Code.

Wir haben den zugehörigen Prüfer Code gefunden, Folgendes fällt uns dabei auf: Zum einen kommen die Blätter, also 1, 2, 3 und 6, nicht im Prüfer Code vor, wohingegen 4 und 5, die jeweils mit drei Knoten verbunden sind, jeweils zweimal im Code vorkommen. Diese Feststellung wollen wir allgemeiner formulieren:

Satz 4.1. Sei $T = (V, E)$ mit $V = [n]$ ein Baum, dann kommt jeder beliebige Knoten v im zugehörigen Prüfer Code als Folgeglied genau $\deg_T(v) - 1$ Mal vor.

Beweis. Sei $T = (V, E)$ mit $V = [n]$ ein Baum und v ein beliebiger Knoten mit $\deg_T(v) := k \in [n - 1]$. Nach Definition 4.1 muss v mit k Knoten verbunden sein.

Nach Konstruktionsvorschrift 1 wird ein Knoten als Folgeglied in den Code genau dann aufgenommen, wenn der mit ihm verbundene Knoten vom Baum entfernt wird. Da dieser Schritt wiederholt wird, bis zwei Knoten übrig bleiben, gibt es für v zwei Fälle:

1. Der Knoten v gehört zu den zwei Knoten, die am Ende übrig bleiben.

Dann werden alle Knoten, bis auf der übrige, mit v verbundene Knoten, entfernt. Insgesamt sind es $k - 1$ Knoten, weshalb v als Folgeglied genau $k - 1$ Mal in den Prüfer Code aufgenommen wird.

2. Der Knoten v gehört nicht zu den zwei Knoten, die am Ende übrig bleiben.

Dafür müsste v nach endlichen Schritten selbst ein Blatt werden. Dieser Fall tritt auf, wenn $k - 1$ Knoten, die mit ihm verbunden sind, vom Baum entfernt werden. Daher kommt v als Folgeglied des Prüfer Codes genau $k - 1$ Mal vor.

Somit kommt v in jedem Fall $k - 1$ Mal als Folgeglied im Prüfer Code vor. □

Im nächsten Abschnitt werden wir eine Bijektion zwischen Prüfer Codes und den markierten Bäumen herstellen, um letztlich auf ihre gleiche Anzahl schließen zu können.

4.1 Bijektion zwischen Prüfer Codes und markierten Bäumen

Im Folgenden wollen wir zeigen, dass die im letzten Abschnitt vorgestellte Konstruktionsvorschrift 1 tatsächlich bijektiv ist. Wir zeigen das in mehreren Schritten, indem wir zunächst die Injektivität und anschließend die Surjektivität zeigen. Dabei benutzen wir die Beweisidee, die in[[2], S.3-4] beschrieben wird.

Satz 4.2. Die Konstruktionsvorschrift 1 ist injektiv.

Beweis. Die Injektivität gilt, wenn man durch Konstruktionsvorschrift 1 von verschiedenen Bäumen verschiedene Prüfer Codes bekommt. Daher wollen wir durch die vollständige Induktion zeigen, dass die folgende Aussage für ein beliebiges n wahr ist: $A(n) =$ „Verschiedene Bäume mit n Knoten haben verschiedene Prüfer Codes.“

Sei zunächst $n_0 := 3$, dann sehen wir in Tabelle 6, dass die Aussage $A(n_0)$ wahr ist:

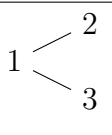
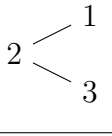
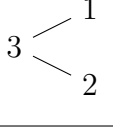
Baum	Prüfer Code
	(1)
	(2)
	(3)

Tabelle 6: Prüfer Codes der Länge $n - 2$ und markierte Bäume mit n Knoten für $n = 3$.

Gelte nun $A(n)$ für ein beliebiges $n \geq n_0$; wir wollen zeigen, dass die Aussage $A(n+1)$ dann auch wahr ist. Betrachte dafür zwei beliebige Bäume T_1 und T_2 mit der Knotenmenge $[n]$. Sei B_1 die Blattmenge von T_1 , und B_2 die Blattmenge von T_2 . Weiter sei (c_1, \dots, c_{n-1})

der zugehörige Prüfer Code von T_1 und (c'_1, \dots, c'_{n-1}) der zugehörige Code von T_2 . Dann sind zwei Fälle zu betrachten:

1. $\min(B_1) \neq \min(B_2)$

Ohne Beschränkung der Allgemeinheit können wir $\min(B_1) < \min(B_2)$ annehmen, sei dabei v das kleinste Blatt von B_1 . Dann gilt $v \notin B_2$, da sonst v das kleinste Blatt von B_2 wäre. Das bedeutet aber wiederum, dass der Grad von v in T_2 größer als 1 sein muss und v nach Satz 4.1 zwar in (c_1, \dots, c_{n-1}) , aber nicht in (c'_1, \dots, c'_{n-1}) enthalten ist. Somit gilt $(c_1, \dots, c_{n-1}) \neq (c'_1, \dots, c'_{n-1})$.

2. $\min(B_1) = \min(B_2)$

Nennen wir $v := \min(B_1) = \min(B_2)$, nach Korollar 4.1 gibt es für v einen eindeutigen Knoten, der mit ihm verbunden ist. Für T_1 müsste nach Konstruktionsvorschrift 1 c_1 der eindeutige, mit v verbundene Knoten sein; für T_2 wäre das c'_1 . Gilt $c_1 \neq c'_1$, dann gilt insbesondere auch $(c_1, \dots, c_{n-1}) \neq (c'_1, \dots, c'_{n-1})$. Gilt $c_1 = c'_1$, dann entferne von T_1 und T_2 jeweils ihr kleinstes Blatt v , und von T_1 die mit v verbundene Kante $\{v, c_1\}$, und von T_2 die Kante $\{v, c'_1\}$. Nennen wir die neuen Bäume T'_1 und T'_2 ; da $T_1 \neq T_2$ gilt und wir nur ihre gleichen Elemente entfernt haben, gilt weiterhin $T'_1 \neq T'_2$. Durch Konstruktionsvorschrift 1 können wir leicht ihre Prüfer Codes bestimmen. Der Prüfer Code von T' muss (c_2, \dots, c_{n-1}) , und der Code von T'_2 muss (c'_2, \dots, c'_{n-1}) sein. Nach $A(n)$ haben wir $(c_2, \dots, c_{n-1}) \neq (c'_2, \dots, c'_{n-1})$, insbesondere muss dann $(c_1, \dots, c_{n-1}) \neq (c'_1, \dots, c'_{n-1})$ gelten. Somit ist $A(n+1)$ gezeigt.

□

Satz 4.3. Die Konstruktionsvorschrift 1 ist surjektiv.

Beweis. Die Surjektivität gilt, wenn man durch Konstruktionsvorschrift 1 jeden Prüfer Code bekommt. Wir wollen durch die vollständige Induktion zeigen, dass die folgende

Aussage für jedes beliebige n gilt: $A(n) =$ „Für jeden Prüfer Code $(c_1, \dots, c_{n-2}) \in [n]^{n-2}$ gibt es einen zugehörigen Baum mit der Knotenmenge $[n]$.“

Sei zunächst $n_0 := 3$, dann können wir aus Tabelle 6 entnehmen, dass die Aussage $A(n_0)$ wahr ist. Gelte nun $A(n)$ für ein beliebiges $n \geq n_0$; wir wollen zeigen, dass die Aussage $A(n+1)$ dann auch wahr ist. Betrachte dafür einen beliebigen Code $(c_1, \dots, c_{n-1}) \in [n+1]^{n-1}$ und sei $v \in [n+1]$ die kleinste, nicht in (c_1, \dots, c_{n-1}) vorkommende Zahl. Zu zeigen ist, dass es für (c_1, \dots, c_{n-1}) einen zugehörigen Baum gibt.

Nun betrachte den Code $(c_2, \dots, c_{n-1}) \in [[n+1] \setminus \{v\}]^{n-2}$, dann gibt es nach $A(n)$ für (c_2, \dots, c_{n-1}) einen zugehörigen Baum $T' = (V', E')$ mit der Knotenmenge $V' = [n+1] \setminus \{v\}$. Betrachte als nächstes den Baum $T = (V, E)$ mit der Knotenmenge $V = [n+1]$. Alle Knoten in V , die nicht in (c_1, \dots, c_{n-1}) vorkommen, müssten nach Satz 4.1 ein Blatt sein. Da v der kleinste Knoten ist, der nicht in (c_1, \dots, c_{n-1}) vorkommt, muss er nach Konstruktionsvorschrift 1 als kleinstes Blatt mit c_1 verbunden sein. Daher ist $T = (V, E)$ mit $E = E' \cup \{(c_1, v)\}$ der zugehörige Baum von (c_1, \dots, c_{n-1}) . Somit haben wir $A(n+1)$ gezeigt. \square

Nun haben wir ihre Bijektivität gezeigt und können somit endlich auf die Anzahl der markierten Bäume schließen.

Korollar 4.2. Die Anzahl der Bäume mit n Knoten ist n^{n-2} .

Daraus können wir leicht ableiten, dass $(n+1)^{n-1}$ die Anzahl der markierten Bäume mit $n+1$ Knoten sein muss. Prüfer stellt ebenfalls eine Konstruktionsvorschrift vor, wie man einen Prüfer Codes zu einen markierten Baum umwandelt. Dass diese Konstruktionsvorschrift auch wirklich die Umkehrung von Konstruktionsvorschrift 1 ist, wird von Prüfer selbst gezeigt [[8], S.144]. Weil wir die Bijektivität bereits gezeigt haben, scheint die Umkehrabbildung in diesem Fall etwas redundant zu sein. Dennoch wäre es interessant

zu beobachten, wie sich ein Code zu einem markierten Baum umwandeln lässt, weshalb wir zum Ende dieses Abschnitts die Konstruktionsvorschrift skizzenhaft darlegen.

Konstruktionsvorschrift 2. Sei ein beliebiger Prüfer Code $c = (c_1, \dots, c_{n-2})$ mit $n \geq 3$, $C := \{c_1, \dots, c_{n-2}\}$ und $N := [n] \setminus C$ gegeben. Gesucht ist ein Baum $T = (V, E)$. Zunächst seien V und E leere Mengen, führe beginnend mit Schritt 1 die Anweisungen durch.

1. Sei $v_1 := \min(N)$, dann haben wir $v_i \in V$ und $\{v_i, c_1\} \in E$.
2. Sei $C' := C \setminus \{c_1\}$ und $N' := [n] \setminus (V \cup C')$.
3. Falls $C' \neq \emptyset$, dann setze $C \leftarrow C'$, $N \leftarrow N'$ und $v_1 \leftarrow v_2$; gehe anschließend zurück zum Schritt 1. Falls $C' = \emptyset$, dann ergänze die letzten übrigen $v_i, v_j \in N'$ zu V und setze $\{v_i, v_j\} \in E$. Dann ist $T = (V, E)$ der zugehörige Baum von c .

Wir versuchen den Prüfer Code vom letzten Beispiel mithilfe der Konstruktionsvorschrift 2 in einen Baum umzuwandeln.

$c = (4, 4, 5, 5), N = \{1, 2, 3, 6\}$	$1 = \min(N) \Rightarrow 1 \in V, \{1, 4\} \in E$	$1 - 4$
$c = (4, 5, 5), N = \{2, 3, 6\}$	$2 = \min(N) \Rightarrow 2 \in V, \{2, 4\} \in E$	$\begin{array}{c} 1 \quad 2 \\ \diagdown \quad / \\ 4 \end{array}$
$c = (5, 5), N = \{3, 4, 6\}$	$3 = \min(N) \Rightarrow 3 \in V, \{3, 5\} \in E$	$\begin{array}{c} 1 \quad 2 \quad 3 \\ \diagdown \quad / \quad \\ 4 \quad 5 \end{array}$
$c = (5), N = \{4, 6\}$	$4 = \min(N) \Rightarrow 4 \in V, \{4, 5\} \in E$	$\begin{array}{c} 1 \quad 2 \quad 3 \\ \diagdown \quad \quad / \\ 4 \quad 5 \end{array}$
$c = \emptyset, N = \{5, 6\}$	$5, 6 \in V, \{5, 6\} \in E$	$\begin{array}{c} 1 \quad 2 \quad 3 \quad 6 \\ \diagdown \quad \quad / \quad / \\ 4 \quad 5 \end{array}$

Tabelle 7: Schrittweise Umwandlung eines Prüfer Codes zu einem markierten Baum.

5 Bijektion zwischen Parkfunktionen und Prüfer Codes

Codes

Im letzten Abschnitt haben wir die Anzahl der markierten Bäume mit $n + 1$ Knoten bestimmt. Dafür haben wir zunächst die Cayley-Formel, dass n^{n-2} die Anzahl der markierten Bäume mit n Knoten ist, bewiesen, indem wir eine Bijektion zwischen der Menge aller markierten Bäume mit n Knoten und der Menge aller Prüfer Codes der Länge $n - 2$, also $[n]^{n-2}$ aufgezeigt haben. Anschließend konnten wir daraus die Anzahl der Bäume mit $n + 1$ Knoten ableiten.

Wenn die Anzahl der Prüfer Codes der Länge $n - 1$ ebenfalls $(n + 1)^{n-1}$ ist, bedeutet es, dass es auch eine Bijektion zwischen Parkfunktionen und Prüfer Codes geben muss. In diesem Kapitel wollen wir ihre Bijektion aufzeigen, so dass wir jede Parkfunktion einem Prüfer Code zuordnen können. Somit wäre es auch möglich, indirekt eine Parkfunktion einem markierten Baum zuzuordnen.

Foata und Riordan haben eine Abbildung formuliert, die jede Parkfunktion auf einen Prüfer Code abbilden soll [[6], S.12]. Sie ist gegeben durch:

$$\begin{aligned} \phi : \quad \text{PF}_n &\rightarrow [n + 1]^{n-1} \\ (p_1, \dots, p_n) &\mapsto (p_2 - p_1, \dots, p_n - p_{n-1}) \bmod^*(n + 1). \end{aligned}$$

Die Wohldefiniertheit von ϕ gilt genau dann, wenn für alle (p_1, \dots, p_n) und $(p'_1, \dots, p'_n) \in \text{PF}_n$ folgende Eigenschaften erfüllt sind:

1. $\phi((p_1, \dots, p_n)) \in [n + 1]^{n-1}$
2. $(p_1, \dots, p_n) = (p'_1, \dots, p'_n) \Rightarrow \phi((p_1, \dots, p_n)) = \phi((p'_1, \dots, p'_n))$

Die erste Eigenschaft gilt aufgrund der Definition von $c_i := p_{i+1} - p_i \bmod^* n + 1$, dass jedes c_i in $[n + 1]$ liegt. Somit liegt jedes $\phi((p_1, \dots, p_n)) = (c_1, \dots, c_{n-1})$ in $[n + 1]^{n-1}$. Die

zweite Eigenschaft gilt ebenfalls aufgrund der eindeutigen Definition von c_i . Somit ist ϕ wohldefiniert. Um die Abbildung ϕ besser nachvollziehen zu können, wollen wir beispielhaft eine Parkfunktion auf einen Prüfer Code abbilden. Nehmen wir dafür $(3, 1, 2) \in \text{PF}_n$, dann haben wir

$$\begin{aligned}\phi((3, 1, 2)) &= (1 - 3, 2 - 1) \bmod^* 4 \\ &= (2, 1).\end{aligned}$$

Foata und Riordan haben eine weitere Abbildung formuliert; diesmal eine, die einen Prüfer Code auf eine Parkfunktion abbilden soll:

$$\begin{aligned}\psi : \quad [n + 1]^{n-1} &\rightarrow \text{PF}_n \\ (c_1, \dots, c_{n-1}) &\mapsto (p_1, p_1 + c_1, \dots, p_1 + c_{n-1} + \dots + c_1) \bmod^*(n + 1).\end{aligned}$$

Zu zeigen, dass sich p_1 allein anhand der Prüfer Codes eindeutig bestimmen lässt und dass $\psi(c_1, \dots, c_{n-1})$ wirklich eine Parkfunktion ist, ist sehr umfangreich. Es würde den Rahmen unserer Arbeit sprengen, wenn wir dies noch in unserer Arbeit aufgreifen würden. Für den Beweis wird auf Foata und Riordan verwiesen, die dies genau gezeigt haben und in dieser Arbeit nehmen wir die Wohldefiniertheit von ψ als gegeben an [[6], S.12-14]. Um auch ψ besser verstehen zu können, bilden wir nun den Prüfer Code $(2, 1)$ auf eine Parkfunktion ab:

$$\begin{aligned}\psi((2, 1)) &= (3 + 3 + 2, 3 + 1 + 2) \bmod^* 4 \\ &= (3, 1, 2).\end{aligned}$$

Das Beispiel lässt uns vermuten, dass ψ die Umkehrabbildung von ϕ ist. Im Folgenden wollen wir diese Vermutung bestätigen, indem wir die Bijektivität zeigen.

Satz 5.1. Die Abbildung $\phi : \text{PF}_n \rightarrow [n + 1]^{n-1}$ ist bijektiv.

Beweis. Wir zeigen die Bijektivität von ϕ , indem wir zeigen, dass ψ die Umkehrabbildung von ϕ ist. Zunächst zeigen wir $\psi \circ \phi = id$. Sei dafür $(p_1, \dots, p_n) \in \text{PF}_n$ mit $\phi((p_1, \dots, p_n)) =$

(c_1, \dots, c_{n-1}) beliebig, es gilt dann

$$\begin{aligned}\psi(\phi((p_1, \dots, p_n))) &= \psi((c_1, \dots, c_{n-1})) \\ &= (p_1, p_1 + c_1, \dots, p_1 + c_{n-1} + \dots + c_1) \bmod^*(n+1).\end{aligned}\tag{1}$$

Da $c_i := (p_{i+1} - p_i) \bmod^*(n+1)$ für jedes $i \in [n-1]$ gilt, haben wir

$$\begin{aligned}(1) &= (p_1, p_1 + (p_2 - p_1), \dots, p_1 + (p_n - p_{n-1}) + \dots + (p_2 - p_1)) \bmod^*(n+1) \\ &= (p_1, \dots, p_n) \bmod^*(n+1).\end{aligned}$$

Und weil $p_i \in [n]$ für alle $i \in [n]$ gilt, haben wir $(p_1, \dots, p_n) \bmod^*(n+1) = (p_1, \dots, p_n)$.

Und es gilt $\phi \circ \psi = id$. Nun zeigen wir $\phi \circ \psi = id$: Sei $(c_1, \dots, c_{n-1}) \in [n+1]^{n-1}$ mit $\psi((c_1, \dots, c_{n-1})) = (p_1, \dots, p_n)$, beliebig, dann gilt

$$\begin{aligned}\phi(\psi((c_1, \dots, c_{n-1}))) &= \phi((p_1, \dots, p_n)) \\ &= (p_2 - p_1, p_3 - p_2, \dots, p_n - p_{n-1}) \bmod^*(n+1).\end{aligned}\tag{2}$$

Weil $p_i = p_1 + c_{i-1} + \dots + c_1 \bmod^*(n+1)$ für alle $i = 2, \dots, n$ gilt und p_1 eindeutig bestimmt ist, haben wir

$$\begin{aligned}(2) &= ((p_1 + c_1) - p_1, \dots, (p_1 + c_{n-1} + \dots + c_1) - (p_1 + c_{n-2} + \dots + c_1)) \bmod^*(n+1) \\ &= (c_1, \dots, c_{n-1}) \bmod^*(n+1).\end{aligned}$$

Wir wissen, dass $c_i \in [n+1]$ für jedes $i \in [n-1]$ gilt; daher haben wir $(c_1, \dots, c_{n-1}) \bmod^*(n+1) = (c_1, \dots, c_{n-1})$ und es gilt $\psi \circ \phi = id$. Somit ist ϕ bijektiv. \square

6 Fazit

Ein Ziel dieser Arbeit war, die Anzahl der Parkfunktionen der Länge n zu bestimmen. In Kapitel 2 haben wir gezeigt, dass ihre Anzahl $(n + 1)^{n-1}$ ist. Anschließend haben wir zwei weitere mathematische Objekte eingeführt, und zwar die markierten Bäume und die Prüfer Codes. Es war zu zeigen, dass ihre Anzahl ebenfalls $(n + 1)^{n-1}$ ist. Die Anzahl der Prüfer Codes ließ sich leicht bestimmen; die Anzahl der markierten Bäume haben wir durch die Bijektion zwischen Prüfer Codes und markierten Bäumen bestimmt. Die Bijektion zwischen Prüfer Codes und markierten Bäumen waren als Konstruktionsvorschriften 1 und 2 gegeben, in Abbildung 7 wurden sie als [K1] und [K2] abgekürzt. Zum Schluss haben wir die Bijektion zwischen Parkfunktionen und Prüfer Codes aufgezeigt, die als ϕ und ψ bezeichnet wurden. Zwar haben wir keine direkte Bijektion zwischen Parkfunktionen und markierten Bäumen hergestellt, doch wie man in der Abbildung sehen kann, gibt es, auch wenn über einen Umweg, sehr wohl eine Bijektion zwischen ihnen. Somit können wir letztlich doch behaupten, dass wir eine Bijektion zwischen Parkfunktionen und markierten Bäumen aufgezeigt haben.

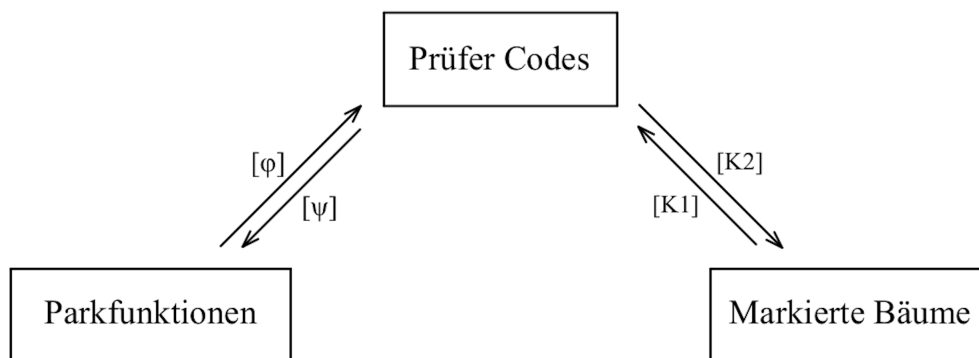


Abbildung 7: Bijektion zwischen Parkfunktionen, Prüfer Codes und markierten Bäumen.

Literatur

- [1] Mathfest circle – parking functions. <http://math.sfsu.edu/beck/314/parking.pdf>.
Letzter Abruf: 19.09.2021.
- [2] Prüfer codes and cayley’s formula. https://www3.nd.edu/~dgalvin1/40210/40210_F12/Prufer.pdf. Letzter Abruf: 17.09.2021.
- [3] Matthias Beck, Ana Berrizbeitia, Michael Dairyko, Claudia Rodriguez, Amanda Ruiz, and Schuyler Veeneman. Parking functions, Shi arrangements, and mixed graphs. *Amer. Math. Monthly*, 122(7):660–673, 2015. doi:10.4169/amer.math.monthly.122.7.660.
- [4] Albrecht Beutelspacher and Marc-Alexander Zschiegner. *Diskrete Mathematik für Einsteiger: Bachelor und Lehramt*. Springer Fachmedien Wiesbaden, Wiesbaden, 5., erw. Aufl. 2014 edition.
- [5] Reinhard Diestel. *Graphentheorie*. Springer Berlin Heidelberg, Wiesbaden, 2017.
- [6] Dominique Foata and John Riordan. Mappings of acyclic and parking functions. *Aequationes Math.*, 10:10–22, 1974. doi:10.1007/BF01834776.
- [7] Alan G Konheim and Benjamin Weiss. An occupancy discipline and applications. *SIAM journal on applied mathematics*, 14(6), 1966-11.
- [8] Heinz Prüfer. Neuer Beweis eines Satzes über Permutationen. *Archiv der Mathematischen Physik*, 27:142–144, 1918.