TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DISSERTATION

PATTERNS IN LABELLED COMBINATORIAL OBJECTS

Ausgeführt zum Zwecke der Erlangung des akademischen Grades einer Doktorin der technischen Wissenschaften unter der Anleitung von

AO. UNIV. PROF. DR. ALOIS PANHOLZER
E104, Institut für
Diskrete Mathematik und Geometrie

eingereicht an der Technischen Universität Wien
Fakultät für Mathematik und Geoinformation

von

MARIE-LOUISE BRUNER
Matrikelnummer 0525370
Josefstädterstraße 43-45/2/4
A-1080 Wien

Wien, am 13. Mai 2015

Marie-Louise Bruner

Alois Panholzer

Vincent Vatter

PATTERNS IN LABELLED COMBINATORIAL OBJECTS

MARIE-LOUISE BRUNER

Institute of Discrete Mathematics and Geometry
Vienna University of Technology

May 2015

Marie-Louise Bruner: *Patterns in labelled combinatorial objects,* May 2015

Dedicated to the memory of my mother, Ingela Bruner.

1952 – 2014

Thank you for always believing in me. $(\mathrm{I}\ \mathrm{U}\ 2)^{\infty}$

DECLARATION

I herewith declare that I have completed the present thesis independently, making use only of the specified literature and aids. Sentences or parts of sentences quoted literally are marked as quotations; identification of other references with regard to the statement and scope of the work is quoted. The thesis in this form has not been submitted to an examination body and has not been published. This thesis draws however on previous publications of the author. For a complete list of my relevant scientific articles, I refer to page xi.

Vienna, May 2015

Marie-Louise Bruner

ABSTRACT

This thesis is concerned with various types of patterns and structural restrictions in labelled combinatorial objects. We say that a combinatorial object A is contained in another combinatorial object B as a pattern if A can be obtained by deleting parts of B. Moreover, the order structure of the labels must be preserved.

A first part of this thesis addresses patterns in permutations. A question that is of particular interest is the decision problem "Does the permutation τ contain the pattern π"? In general, this problem is NP-complete. We present an fpt-algorithm that solves this problem efficiently if τ has few alternating runs. We also analyse the computational complexity of this problem for several different types of permutation patterns. For a special permutation class arising in the context of the third part of this thesis, we solve the enumeration problem. Finally, the log-concavity of a combinatorial sequence related to permutation patterns is investigated.

A second part of this work is concerned with Cayley trees, i.e., rooted unordered trees and mappings, i.e., functions from a finite set to itself. First, we present a new bijective proof of Cayley's formula which will subsequently allow us to establish bijective correspondences. Next, we generalize the label patterns "ascents" and "ascending runs" from permutations to Cayley trees and mappings and investigate their distribution. Another topic treated in this part is the generalization of the concept of parking functions to Cayley trees and mappings. The asymptotic analysis of the numbers of these objects leads to an interesting phase transition behaviour.

A third part deals with so-called single-peaked elections that play an important role in social choice theory. We introduce the concept of configurations in elections and establish a correspondence with permutation patterns. For the special case of single-peaked profiles we answer the question how likely these are to occur when preferences are chosen at random.

Some of the results presented in this thesis have already been published in scientific articles by the present author. For a complete list of the papers this thesis is based on, we refer to page xi.

ZUSAMMENFASSUNG

Diese Dissertation beschäftigt sich mit unterschiedlichen Arten von Mustern und strukturellen Einschränkungen in markierten kombinatorischen Objekten. Ein kombinatorisches Objekt A ist in einem anderen größeren Objekt B als Muster enthalten, falls man A erhalten kann, indem Teile von B entfernt werden. Dabei muss auch die Ordnungsstruktur der Marken (engl. "labels") erhalten bleiben.

Ein erster Teil dieser Arbeit befasst sich mit Mustern in Permutationen. Besondere Aufmerksamkeit wird in diesem Zusammenhang dem Entscheidungssproblem "Enthält die Permutation τ das Muster π?" gewidmet. Dieses Problem ist bekanntlich NP-vollständig. Wir stellen einen fpt-Algorithmus für dieses Problem vor, der besonders effizient ist, wenn die Permutation τ nur wenige "alternating runs" aufweist. Außerdem analysieren wir die Komplexität dieses Problems für verschiedene Typen von Permutationsmustern. Für eine besondere Permutationsklasse, die durch vier vermiedene Muster definiert ist und im dritten Teil dieser Arbeit auftaucht, konnten wir das Abzählproblem lösen. Weiters wird die Log-Konkavität einer kombinatorischen Zahlenfolge, die mit Mustervermeidung zusammenhängt, untersucht.

Ein zweiter Teil dieser Dissertation beschäftigt sich mit Cayley Bäumen, das sind gewurzelte ungeordnete Bäume, und mit Mappings, das sind Abbildungen einer endlichen Menge auf sich selbst. Zunächst präsentieren wir einen neuen bijektiven Zusammenhang zwischen Cayley Bäumen und Mappings, der uns im Folgenden erlauben wird, bijektive Resultate zu beweisen. Wir verallgemeinern die für Permutationen definierten Muster "Aufstiege" und "aufsteigende Runs" und untersuchen deren Verteilung für Cayley Bäume und Mappings. Ein weiteres Thema sind Parkfunktionen, die wir auf Cayley Bäume und Mappings verallgemeinern. Das asymptotische Verhalten der Abzählformeln weist dabei ein interessantes Phasenübergangsverhalten auf.

Ein dritter Teil behandelt sogenannte "Single-peaked elections", die in der sozialen Wahltheorie eine wichtige Rolle spielen. Wir führen den allgemeinen Begriff von Konfigurationen in Präferenzen ein und stellen eine Verbindung zu Permutationsmustern her. Für die spezielle Single-peaked-Konfiguration beantworten wir außerdem die Frage, wie wahrscheinlich es ist, dass diese in zufälligen Präferenzen auftaucht.

Die in dieser Dissertation vorgestellten Resultate sind zum Teil schon in wissenschaftlichen Artikeln der Autorin publiziert worden. Eine Auflistung der Arbeiten, auf denen diese Dissertation aufgebaut ist, findet sich auf Seite xi.

PUBLICATIONS

This thesis is based on the following publications and preprints:

[41] Marie-Louise Bruner and Martin Lackner. A fast algorithm for permutation pattern matching based on alternating runs. In Fedor V. Fomin and Petteri Kaski, editors, *SWAT*, volume 7357 of *Lecture Notes in Computer Science*, pages 261–270. Springer, 2012.

[42] Marie-Louise Bruner and Martin Lackner. The computational landscape of permutation patterns. *Pure Mathematics and Applications*, 24(2)(2):83–101, 2013.

[30] Miklós Bóna and Marie-Louise Bruner. Log-concavity, the Ulam distance and involutions. *arXiv preprint*, arXiv:1502.05438, 2015.

[44] Marie-Louise Bruner and Alois Panholzer. Parking functions for trees and mappings. *arXiv preprint*, arXiv:1504.04972, 2015.

[43] Marie-Louise Bruner and Martin Lackner. The likelihood of structure in preference profiles. In *Proceedings of the 8th Multidisciplinary Workshop on Advances in Preference Handling (MPref 2014)*, 2014.

ACKNOWLEDGMENTS

First and foremost I wish to thank my supervisor Alois Panholzer. He was the one who, in his lecture on "Discrete Methods", first sparked my interest for discrete mathematics and combinatorial problems. During my PhD studies, I greatly valued the substantial freedom Alois gave me to pursue my own research directions. This allowed me to explore various aspects of the combinatorial world. Also, he made it possible for me to participate in many conferences and workshops and to spend a month at the University of Florida. It has meant a lot to me personally to have his support in emotionally difficult times and never having to feel under pressure. I am also most thankful that he is not only enabling but also encouraging me to pursue my plan of hiking, kayaking and bicycling from Oslo back home this summer.

I would also like to thank all the other great researchers whom I met and interacted with during my studies. Thank you for making me feel part of the Analysis of Algorithms and the Permutation Patterns community. Thank you for welcoming me into your world and for encouraging me. In particular, my thanks go to Cyril Banderier, Miklós Bóna, Michael Drmota, Danièle Gardy, Antoine Genitrini, Bernhard Gittenberger, Markus Kuba, Adeline Pierrot and Vincent Vatter.

To Vincent Vatter I also owe my thanks for accepting the task of reviewing my thesis. In addition, I would like to thank him very much for inviting me for a second research stay at the University of Florida this past winter.

During my daily work in Vienna, it was my colleagues as well as my (current and former) roommates who made my days as a PhD student very enjoyable. These are in particular: Danièle Gardy, Veronika Kraus, Benoît Loridant, Johannes Morgenbesser, Adeline Pierrot, Georg Seitz and Michael Wallner. My special thanks also go to our secretary Barbara Dolezal-Rainer whose open ear for all sorts of questions and good sense of humour always created a warm-hearted and welcoming atmosphere at our department.

The program WINA+ was started rather recently to support young researchers at the Vienna University of Technology. The regular meetings with colleagues from various disciplines and our coach Stephan Faatz were very encouraging and helped to solve small and big problems. Furthermore, the wonderful seminars on scientific writing led by Katherine Tiede were very inspiring and taught me the joy of writing.

I would also like to thank Gerhard Bruner, Martin Lackner, Diana Newton-Smith and Michael Wallner who helped me by proof-reading my thesis.

If there is one thing that my mother's illness taught me, it is how much my family and friends mean to me. I am very grateful for the time we were able spend together and for the many strong memories I have of my mother. After her passing away last spring, my family and friends were a tremendous support. Especially I want to thank my father Gerhard Bruner for his generosity, patience, cheerful disposition, trust in me and helpfulness. Also, I would like to thank my "new" family, the Lackners, for including me in their lives and making me feel so much at home in their company.

My greatest support throughout the last years has been my partner Martin Lackner with whom I can share not only mathematical research but also crazy plans for outdoor adventures. Thank you for being my rain boots when it's pouring, the hot cup of tea on a cold winter day, the shelter when it's stormy and for being by my side to enjoy the sunny days.

Last but not least, my thanks go to the Austrian Science Fund FWF that supported my research throughout most of the duration of my thesis via the projects P25337-N23 "Restricted labelled combinatorial objects: new enumerative, statistical, asymptotic and complexity theoretical aspects" and 9608 "Combinatoric analysis of data structures and tree-like structures".

CONTENTS

1

INTRODUCTION

The notion of patterns or substructures is omnipresent within discrete mathematics and especially within combinatorics. Given a combinatorial object A we say that it contains a combinatorial object B if we can obtain B by removing some parts of A. Let us mention a couple of well-known examples:

Within graph theory, the notion of *induced subgraphs* is a very basic one. Given a graph $G = (V, E)$, we say that it contains $H = (V', E')$ as an induced subgraph if $V' \subseteq V$, $E' \subseteq E$ and E' contains all edges from E that have both endpoints in V'. That is, if we start with the graph G and successively remove all vertices in $V \setminus V'$ together with their adjacent edges, we end up with the graph H. Many properties of graphs or graph classes can be described with the help of (forbidden) subgraphs. For instance, consider the class of *split graphs*. These are graphs for which the vertex set V can be partitioned into two subsets V_1 and V_2 for which the following holds: the subgraph induced by V_1 is a clique, i.e., all possible edges between vertices in V_1 are present, and the subgraph induced by V_2 is an independent set, i.e., no two vertices in V_2 are connected by an edge. Split graphs arise amongst others in the context of skew-merged permutations; the permutation graph, i.e., the graph in which an edge (i, j) is present for every inversion in the permutation, of a skew-merged permutation is a split graph. If we want to check whether a graph is a split graph or not, it is not necessary to check every partition of V into two sets. Indeed, a split graph can also be characterized with the help of forbidden subgraphs: A graph is a split graph if and only if it has no cycle of length four or five and no pair of disjoint edges as induced subgraph.

Another simple example is that of *cycles* in permutations. A permutation π of length n is a bijective function from the set $\{1, 2, \ldots, n\}$ to itself. A permutation can be decomposed in a unique way into its cycles. We say that π contains a cycle of length k if there is some element i in $\{1, 2, \ldots, n\}$ such that $\pi^k(i) = i$ and $\pi^j(i) \neq i$ for all $1 \leq j < k$. That is, if we apply the permutation π k times to the element i we end up at i again. The cycle containing the element i then consists of the integers $i, \pi(i), \pi^2(i), \ldots, \pi^{k-1}(i)$. If we remove all the other elements from the set $\{1, 2, \ldots, n\}$ and restrict the function π to the set $\{i, \pi(i), \ldots, \pi^{k-1}(i)\}$, we obtain a function that is also a bijection and that consists of a single cycle of length k. We can thus say that π contains a cycle of length k as a pattern. Cycles can for instance be used to define *involutions*. An involution is a permutation π that is self-inverse, that is $\pi \circ \pi = \pi^2 = \mathrm{id}$. If we have the cycle structure

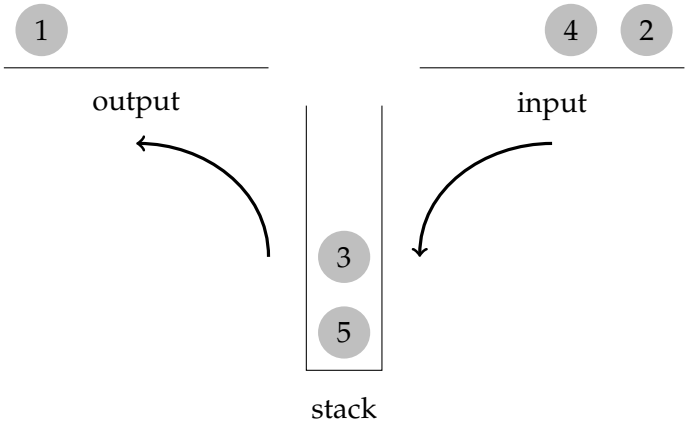


Figure 1: Attempting to sort the permutation $\pi = 53142$ with the help of a stack. As we can see, the subsequence 342 causes trouble and cannot be sorted.

of a permutation π at hand, it is very easy to say whether it is an involution or not. Indeed, since $\pi^2(i) = i$ has to hold for all elements i of the permutation, all cycles need to be of length one or two. Thus, involutions can be characterized as those permutations that do not contain any k-cycles for $k \geq 3$ as a pattern.

The next example is also related to permutations. Let us consider the following sorting device called a *stack*. A stack is a linear list for which all insertions and deletions are made at one end of the list. That is, the last element inserted from the input into the stack, will be the first to be deleted and placed in the output. Stacks are thus also called LIFO-(Last-In-First-Out-)lists. For a graphical representation of a stack, see Figure 1. The question is now: Which permutations π can be sorted with the help of a stack? In other words: For which permutations π as input is it possible to obtain the identity permutation π as output? Let us consider the permutation $\pi = 53142$ in Figure 1 as an example. As we can see, we get stuck as soon as the element 3 is placed in the stack and it is not possible to output the element 2 before the element 4. Within the permutation π, it is the subsequence 342 that cannot be sorted. This observation can be generalized and one can prove that a permutation π is stack-sortable if and only if there are no three elements ℓ, m, n in the permutations with $\ell < m < n$ and m stands to the left of n and n to the left of ℓ. That is, π may not contain a subsequence of length three in which the elements are in the same order as in the permutation 231. One also says that a permutation is stack sortable if and only if it avoids the pattern 231.

The first two examples of patterns or substructures have a point in common that distinguishes them from the third example. The patterns were defined by the underlying graph or cycle structure and the specific labels of the involved vertices or the elements of the per-

mutation were of no importance. For instance, when considering a graph containing a cycle of length 4, it is of no relevance whether the elements forming this cycle are labelled $1, 2, 3, 4$ or $7, 2, 1, 5$. This is in contrast to the third example where we introduced 231-avoiding permutations. Here, the involved elements of the permutation play a crucial role since they need to be in a specific order to form a 231-pattern.

This leads us to the concept of *label patterns* in combinatorial objects that is at the heart of this thesis. First, all combinatorial objects that we will consider are labelled ones. This means that the "atoms" – these are for instance the vertices in a graph or the elements in a permutation – have positive integers attached to them. We call these integers the *labels*. Second, we say that a labelled combinatorial object A contains a labelled combinatorial object B if we can obtain B by removing some parts of A. Additionally, the labels of B have to be order-isomorphic to the labels of the substructure of A that we obtained after deleting some parts. Returning to the example of stack-sortable permutations, we can formalize the definition of 231-avoiding permutations as follows: A permutation π contains the pattern 231 if and only if there are three positions $i < j < k$ in π such that $\pi(k) < \pi(i) < \pi(j)$, i.e., the subsequence $\pi(i)\,\pi(j)\,\pi(k)$ in π is order-isomorphic to 231.

In this thesis, we will study different types of label patterns in various combinatorial objects. Our research is devoted to a varied set of problems that concern algorithmical, enumerative, analytic and probabilistic questions. Also, the methods employed throughout this thesis are diverse: We employ techniques from analytic combinatorics, complexity theory, probability theory as well as bijective proofs. Figure 2 provides an overview of the combinatorial objects studied in this thesis and the methods that were used. A detailed introduction to the objects studied in this thesis is given in the Preliminaries, starting on page 13.

Part i of this thesis is devoted to permutation patterns. The history of permutation patterns can be traced back to the beginning of the 20th century and MacMahon's study [120] of permutations that can be partitioned into two decreasing subsequences, i.e., 123-avoiding permutations. Also, the studies by Erdős and Szekeres showing that a permutation of length n always contains an increasing subsequence of length $\sqrt{n-1}$ or a decreasing subsequence of length $\sqrt{n-1}$ [67] along with Schensted's study of longest increasing subsequences in permutations [139] can be seen as studies of the monotone patterns $12\ldots k$ for some integer k. However, the birth of permutation patterns is mostly attributed to Knuth who published the first volume of *The Art of Computer Programming* [111] in 1968. There, in an exercise, he asks which permutations can be sorted with the help of a stack and his question naturally leads to permutation patterns as

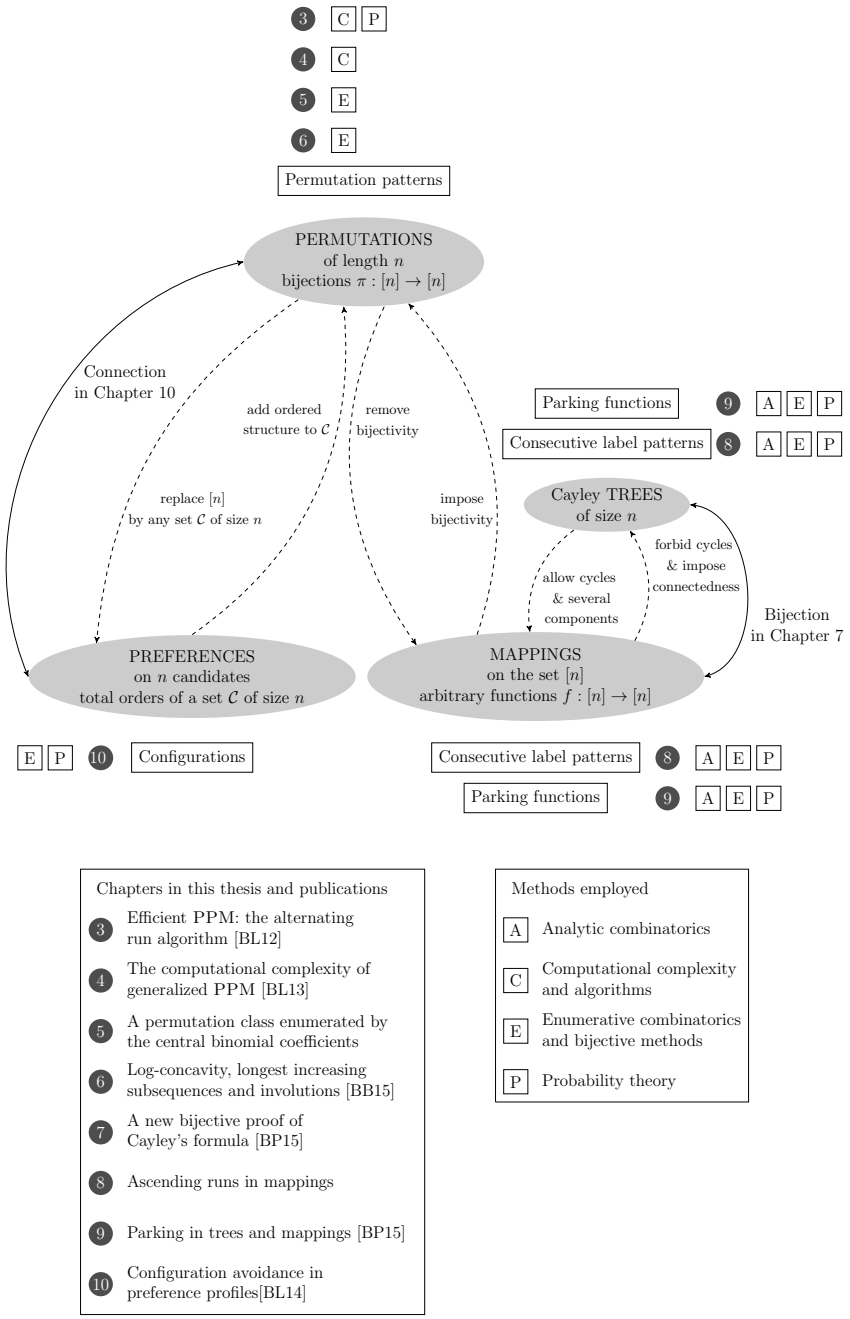


Figure 2: An overview of the objects studied and the methods employed in this thesis. Dashed lines mark connections between combinatorial objects and solid lines mark connections that are proved within this thesis.

we have seen above. The first systematic study of patterns in permutations was not performed until 1985 by Simion and Schmidt [140]. Since then, the field of permutation patterns has become a vastly growing part of combinatorics. This is evidenced by the attention attributed to this topic in various monographs: several chapters in Bóna's *Combinatorics of Permutations* are concerned with permutation patterns[25]; a comprehensive presentation of results related to various types of patterns in permutations and words can be found in Kitaev's monograph *Patterns in Permutations and Words* [108]; and a survey of permutation classes and recent research directions can be found in Vatter's Chapter 12 in the *Handbook of Enumerative Combinatorics* [29]. Moreover many applications of permutation patterns have been discovered: their relation to stack and deque sorting, genome sequences in computational biology, statistical mechanics and in general their numerous connections to other combinatorial objects [108].

So far, computational aspects of permutation patterns have received far less attention than enumerative ones. The computational problem of detecting permutation patterns, the Permutation Pattern Matching (PPM) problem, can be formulated as follows: Does a permutation τ of length n (the text) contain a permutation π of length $k \leq n$ (the pattern)? In Chapters 3 and 4 we will take the viewpoint of computational complexity and consider this algorithmical problem.

PPM has been shown to be NP-complete [31]. This implies that, unless $\mathsf{P} = \mathsf{NP}$, we cannot hope for a polynomial time algorithm for PPM. This is however not the end of the story: even though PPM is NP-complete in general, one can hope to find polynomial time algorithms for special cases of the problem. For instance, if π is the identity $12\ldots k$, PPM consists of looking for an increasing subsequence of length k in the text – this is a special case of the Longest Increasing Subsequence problem. This problem can be solved in $\mathcal{O}(n \log n)$-time for sequences in general [139] and in $\mathcal{O}(n \log\log n)$-time for permutations [48, 122]. Another example is that of *separable* patterns. These are permutations avoiding both patterns 3142 and 2413. In this case, PPM can be solved in $\mathcal{O}(k \cdot n^4)$ time [99]. Another possibility that allows to circumvent the computational hardness of PPM is to confine the combinatorial explosion to a certain parameter of the input (τ, σ). This is the approach of *parameterized complexity theory*, a rather new branch of complexity theory that has, so far, mostly been used for problems on graphs [59]. In Chapter 3 we employ parameterized complexity theory and construct an algorithm that has a worst-case runtime of $\mathcal{O}(1.79^{\mathsf{run}(\tau)} \cdot n \cdot k)$ where $\mathsf{run}(\tau)$ denotes the number of alternating runs in τ. This algorithm is particularly well-suited for instances where τ has few alternating runs, i.e., few ups and downs. Moreover, since $\mathsf{run}(\tau) < n$, this can be seen as a $\mathcal{O}(1.79^n \cdot n \cdot k)$ algorithm that is the first to beat the exponential 2^n exponential runtime of brute-force search. Furthermore, we prove that under standard com-

plexity theoretic assumptions such a fixed-parameter tractability result is not possible for $\mathsf{run}(\pi)$.

In Chapter 4 we consider variations of PPM. In the last years, different types of patterns in permutations have been studied: vincular, bivincular and mesh patterns, just to name a few. Every type of permutation pattern naturally defines a corresponding computational problem. The goal of Chapter 4 is to draw a map of the computational landscape of permutation pattern matching with different types of patterns. We provide a classical complexity analysis and investigate the impact of the pattern length on the computational hardness.

Chapter 5 is devoted to the enumeration of a specific permutation class. The set of permutations avoiding the patterns 2431, 4231, 1432 and 4132 simultaneously arises in the context of single-peaked elections that are studied in Chapter 10. Using two bijections, we can show that this class in enumerated by the central binomial coefficients. Thus this permutation class can join the list of classes that have "nice" enumeration formulæ.

An important property of combinatorial sequences is that of *log-concavity*. A sequence $(a_n)_{n\in\mathbb{N}}$ is called log-concave if the property $a_{n-1} \cdot a_{n+1} \leq a_n^2$ holds for all n. In Chapter 6 we formulate our conjecture that the sequence $\ell_{n,k}$ counting permutations of length n with a longest increasing subsequence of length k is log-concave. Alternatively, $\ell_{n,k}$ counts permutations of length n that contain the pattern $12\dots k$ but avoid the pattern $12\dots k(k+1)$. We prove our conjecture in some special cases, i.e., for certain subsets of permutations of length n. One tool in our proofs is a technique that allows us to turn injections between sets of involutions into injections between sets of permutations. In addition, we use several consequences of the well-known Robinson-Schensted correspondence (see e.g. Chapter 14 in [29]). The sequence $\ell_{n,k}$ can also be interpreted as counting the number of permutations of length n that have *Ulam-distance* $n-k$ to the identity permutation $12\dots n$. This distance was introduced by Ulam as an "evolutionary distance" in the context of biological sequences [150]. The Ulam distance $U(\sigma,\tau)$ of two permutations σ and τ is the minimal number of steps needed to obtain τ from σ where each step consists in taking an element from the current permutation and placing it at some other position. If τ is the identity permutation id of length n, then it is easy to see that $n - U(\sigma,\tau)$ is equal to the length $\ell(\sigma)$ of the longest increasing subsequence in σ. Our results can thus be seen as a contribution to the study of notions of distance for permutations and biologically motivated sorting algorithms that has received increased interest in the last years. A collection of these results can be found in [73].

Part ii is concerned with Cayley trees and mappings. *Cayley trees* are one of the simplest tree models in discrete mathematics and are a fundamental data structure in computer science: they are rooted la-

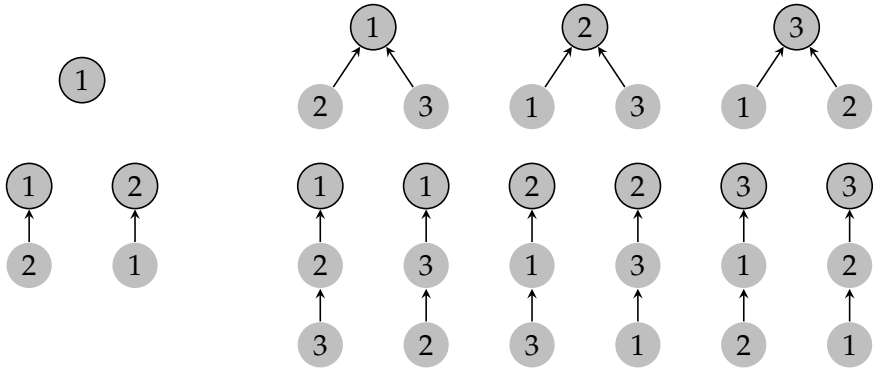


Figure 3: All Cayley trees of size up to $n = 3$

belled unordered trees. "Rooted" means that there is one designated node in the tree that we call the root (this node can be any one of the nodes) and we consider all the edges to be oriented toward the root. "Labelled" means that every node carries a label, which we assume to be an integer between 1 and n, where n is the size of the tree and every label occurs exactly once. "Unordered" means that there is no order among the children of a given node. Figure 3 provides a representation of all Cayley trees of size $n = 1, 2$ and 3. The roots are drawn on top and are marked by a black border. Cayley trees carry their name from the British mathematician Arthur Cayley who studied them in 1889 [47]. In this paper he showed that the number T_n of rooted unordered trees with n nodes is equal to n^{n-1} or, equivalently, that the number of unrooted unordered trees with n nodes is equal to n^{n-2}. Cayley was however not the first to have stated this result. An equivalent result had already been shown earlier by Borchardt (1860) and Sylvester (1857), but Cayley was the first to use graph theory terms. The theorem stating that $T_n = n^{n-1}$ has since been known as *Cayley's formula*. Since Cayley's paper, various proofs have been given for his formula, using bijections, double counting, recursions and even methods from linear algebra, see e.g. the collection in [2].

In Chapter 7 we present a new proof of Cayley's formula by constructing a bijection between pairs (T, w) where T is a Cayley tree of size n and w is a node of T and n-mappings. An *n-mapping* is a function from the set $\{1, 2, \ldots, n\}$ into itself. Clearly, there are n^n n-mappings and thus providing such a bijection is indeed a proof of Cayley's formula. The advantage of our bijection is that it can be applied to the objects studied in the subsequent two chapters and allows us to provide combinatorial explanations for facts that we obtain with the help of generating functions.

Besides this bijective correspondence between Cayley trees and mappings, these combinatorial objects are also structurally linked to each other. To see this, let us consider the functional digraph G_f associated with an n-mapping f. This graph is defined on the vertex set $\{1, 2, \ldots, n\}$ and a directed edge is present between vertex i and ver-

$$f : 1 \mapsto 3 \quad 4 \mapsto 3$$
$$2 \mapsto 2 \quad 5 \mapsto 2$$
$$3 \mapsto 6 \quad 6 \mapsto 3$$

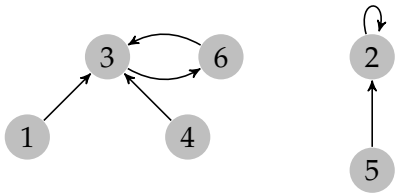


Figure 4: The functional graph of the 6-mapping f

tex j whenever $j = f(i)$. For an example, see the 6-mapping f depicted in Figure 4. The connected components of such a functional digraph are Cayley trees whose root nodes have been arranged in cycles. In our example, the connected component containing the node 1 has two cyclic elements: the node 3 and the node 6. The node 3 is the root node of a Cayley tree of size 3 and the node 6 is the root node of a Cayley tree of size 1. The second connected component has a single cyclic node and simply consists of a Cayley tree of size 2 where an additional loop edge has been attached to the root. Cayley trees with an additional edge from the root node to itself can thus be seen as a special case of mappings. This structural connection between Cayley trees and mappings translates easily into the language of generating functions using the symbolic method.

Random n-mappings, or simply random functions from the set $\{1, 2, \ldots, n\}$ into itself, appear in various applications: for instance in the birthday paradox, the coupon collector problem and occupancy problems, just to name a few. Structural properties of the functional digraphs of random mappings have widely been studied, see e.g. the work of Arney and Bender [8], Kolchin [112], Flajolet and Odlyzko [75]. For instance, it is well known that the expected number of connected components in a random n-mapping is asymptotically $1/2 \cdot \log(n)$, the expected number of cyclic nodes is $\sqrt{\pi n/2}$ and the expected number of terminal nodes, i.e., nodes with no preimages, is $e^{-1}n$. In the functional graphs corresponding to random mappings the labels of nodes play an important role and thus it is somewhat surprising that the occurrences of label patterns have, so far, not received more attention. In Chapter 8 we perform a first analysis of such a *label pattern* in random mappings. Namely, we generalize the notion of ascending runs from permutations to mappings and study the random variable counting the number of ascending runs in a random mapping. We obtain exact enumeration formulæ and limiting distribution results for the occurrences of ascending runs both in mappings and in Cayley trees. We also analyse the occurrence of ascents in ransom mappings and Cayley trees. Some recent research in this direction has been undertaken by Panholzer who studied *alternating mappings* in [130]. These are a generalization of the concept of alternating permutations to mappings. They can be defined as those mappings for which every iteration orbit forms an alternating sequence. Alternat-

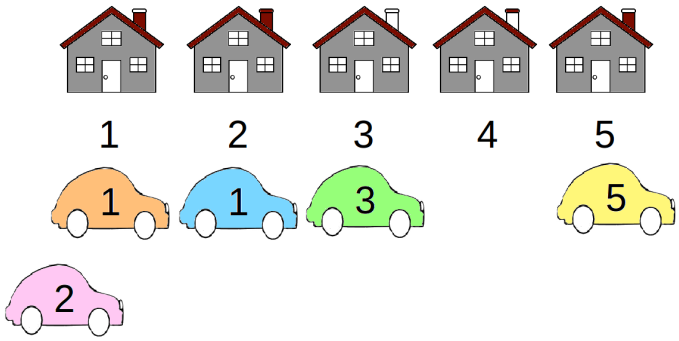


Figure 5: An example of a parking function: If the drivers want to park at the spaces $1, 5, 3, 1, 2$ everyone can park successfully.

ing mappings can thus also be seen as mappings that do not contain two consecutive ascents or descents, this being a characterization in terms of forbidden label patterns.

Chapter 9 is devoted to a generalization of parking functions to Cayley trees and mappings. Parking functions were introduced by Konheim and Weiss [113] in the context of linear probing hashing. An illustrative description of parking functions can be given as follows: Consider a one-way street with n parking spaces numbered from 1 to n and a sequence of m drivers with preferred parking spaces $s_1, s_2, \ldots, s_m$. The drivers arrive sequentially and each driver k, $1 \leq k \leq m$, tries to park at her preferred parking space with address $s_k \in [n]$. If it is free she parks. Otherwise she moves further in the allowed direction, thus examining parking spaces $s_k + 1, s_k + 2, \ldots$, until she finds a free parking space where she parks. If there is no such parking space she leaves the street without parking. A parking function is then a sequence $(s_1, \ldots, s_m) \in [n]^m$ of addresses such that all m drivers are able to park. For an example of a parking function with 5 parking spaces and 5 drivers, see Figure 5. The sequence $1, 5, 3, 1, 2$ of preferred parking spaces is a parking function. However, if the last driver decides that she wants to park at the fifth parking space she will not be able to park and thus the sequence $1, 5, 3, 1, 5$ is not a parking function.

The notion of parking functions has been generalized in various ways, yielding amongst others (a, b)-parking functions [159], bucket parking functions [23], x-parking functions [148] and G-parking functions [135]. We introduce a new generalization of parking functions by starting with the original definition of parking functions and applying it to rooted trees and mappings. Thus, we consider the nodes in a Cayley tree or a functional digraph of a mapping as parking spaces and the directed edges as one-way streets in a branched road net. The drivers first attempt to park at their preferred parking space. If this

one is not free, they continue their route following the directed edges until they find a free parking space. For this generalization of parking functions, we count the total number of tree-parking functions as well as the total number of mappings functions. Interestingly, it turns out that these numbers are directly linked to each other which can also be explained in a bijective manner. Furthermore, we perform an asymptotic analysis of these quantities using methods from analytic combinatorics. Subsequently, we observe an interesting phase transition behaviour at the point where the number m of drivers is equal to half the number of parking spaces.

Parking functions represent a structural restriction on integer sequences but cannot directly be interpreted in terms of label patterns. However, there does exist a correspondence between parking functions and certain types of patterns. In order to state this correspondence, consider the following sorting device, called a *priority queue*. In a priority queue, two possible operations can be performed: one can either insert the next element from the input to the queue or, whenever the queue is not empty, one can move the smallest element in the queue to the output. With such a sorting device, the possible output permutations depend on the input. For instance, the identity permutation $\mathsf{id} = 12\dots n$ will always lead to the same output, namely the identity again. However, if the input permutation is $n(n-1)\dots 21$, it can lead to various different output permutations. The pairs (σ, τ) of input-output pairs that can be obtained with a priority queue – these are called *allowable pairs* – can also be characterized in terms of forbidden patterns. Indeed, a pair (σ, τ) is an allowable pair if and only if it does not contain the pattern-pairs $(12, 21)$ and $(321, 132)$. This means that if there are two entries $\sigma(i)$ and $\sigma(j)$ with $i < j$ and $\sigma(i) < \sigma(j)$ in the input permutation, the elements $\sigma(i)$ and $\sigma(j)$ may not occur in inverse order in τ. Similarly, if there are positions $i < j < k$ in the input permutation with $\sigma(i) > \sigma(j) > \sigma(k)$, these elements may not occur in the order $\sigma(k), \sigma(i), \sigma(j)$ in the output permutation. This notion of pattern avoidance is connected with pattern avoidance in permutations but it requires patterns to occur simultaneously on the same elements in two permutations. Now, allowable pairs of size n are in bijective correspondence with parking functions with n parking spaces and n drivers. In [90], such a bijection is described where the output permutation of the allowable pair is equal to the output of the corresponding parking function. For a parking function, the output permutation associates to each driver the parking space where she eventually ends up parking. Thus parking functions have a nice connection to a certain notion of label patterns.

The last part of this thesis is devoted to so-called *single-peaked elections* and *configurations* in elections. The single-peaked restriction, introduced by Black [22], is an extensively studied concept in social choice theory. A collection of preferences, i.e., total orders on a set

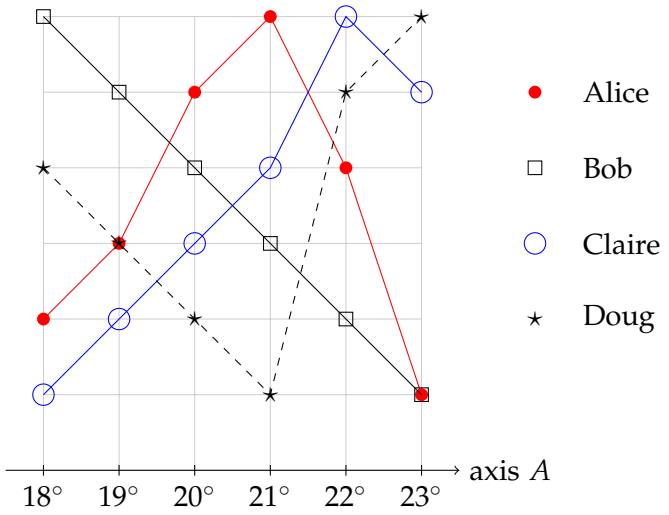


Figure 6: Alice's, Bob's and Claire's preferences form a single-peaked election with respect to the axis $A = 18°C < 19°C < 20°C < 21°C < 22°C < 23°C$. However, Doug's vote is not single-peaked with respect to A.

of candidates, is single-peaked if the candidates can be ordered linearly – on a so-called axis – so that each preference is either strictly increasing along this ordering, or strictly decreasing, or first increasing and then decreasing. For example, the preferences of voters Alice, Bob and Claire on the temperature in a class room are single-peaked with respect to the axis $18°C < 19°C < 20°C < 21°C < 22°C < 23°C$ as depicted in Figure 6. However, Doug's vote is not single-peaked with respect to this axis and one could argue with him whether his preferences make sense – or not.

Single-peaked elections play an important role in social choice theory since they have several nice properties. Probably their most important feature is that they they allow to circumvent Arrow's paradox [10]. Arrow's paradox states that as soon as there are more than two candidates, every possible way of aggregating individual preferences to a common preference in a way that this common preference fulfils two very basic properties, is a dictatorship. This means that there exists a voter whose preferences will always prevail. However, if we restrict ourselves to single-peaked elections, Arrow's paradox does not hold any longer and there do exist preference aggregation mechanisms with many desirable properties.

In Chapter 10 we perform the first combinatorial analysis of the single-peaked domain. Our aim is to establish results on the likelihood that an election is single-peaked for some axis. We consider three probability distributions for elections: the Impartial Culture (IC) assumption in which all total orders are equally likely and are chosen independently; the Pólya urn model which assumes a certain homogeneity among voters; and the Mallows model in which the proba-

bility of a vote depends on its Kendall-tau distance to a given reference vote. We also define the concept of *configurations* in elections and prove a general result on the probability that an election does not contain a configuration of size $(2, k)$.

Before we start with the actual results of this thesis, we gather the most important definitions of concepts that occur throughout this thesis in Chapter 2. In this Chapter, we also present the methods that will be employed in this thesis. In Chapter 10.6 we point out various directions for further research.

2

PRELIMINARIES

In this chapter, we introduce the objects and methods that will be used throughout this thesis. After some basic asymptotic notation we present the combinatorial objects that will be analysed throughout this thesis: permutations, Cayley trees, mappings and preference profiles. We continue by giving a brief introduction to the methods used to analyse these objects; these are algorithmic, analytic, probabilistic and combinatorial tools. Parts of this chapter are taken from the publications this thesis is based on, see page xii for a complete list.

Some standard notation that is used throughout this thesis is gathered in the Notation-Section on page 233 and following pages.

2.1 ASYMPTOTIC NOTATION

Throughout this thesis we will often be interested in the asymptotic growth of functions, both in the context of the runtime of algorithms and of the enumeration of combinatorial objects.

When comparing the growth rates of functions, we use the following standard asymptotic notation introduced by Bachmann and Landau. The presentation here follows the one in [78]. Let x_0 be some element of a set $\mathbb{X}$ on which some notion of neighbourhood exists. In this thesis this will be: $\mathbb{X} = \mathbb{N} \cup \{\infty\}$ and $x_0 = \infty$, $\mathbb{X} = \mathbb{R}$ and x_0 any real number or $\mathbb{X} = \mathbb{C}$ and $x_0 = 0$. Furthermore, let f and g be two real or complex functions defined on $\mathbb{X} \setminus \{x_0\}$.

- We write $f(x) = \mathcal{O}(g(x))$ as $x \to x_0$, if the ratio $f(x)/g(x)$ stays bounded as $x \to x_0$. That is, if there is a constant $C > 0$ and a neighbourhood $\mathcal{U}$ of x_0 such that

  $$|f(x)| \leq C \cdot |g(x)| \text{ for all } x \neq x_0, x \in \mathcal{U}.$$

  One says that "f is of order at most g" or "f is big-Oh of g".

- We write $f(x) = o(g(x))$ as $x \to x_0$, if the ratio $f(x)/g(x)$ tends to 0 as $x \to x_0$. That is, if for every $\epsilon > 0$ there is a neighbourhood $\mathcal{U}_\epsilon$ of x_0 such that

  $$|f(x)| \leq \epsilon \cdot |g(x)| \text{ for all } x \neq x_0, x \in \mathcal{U}_\epsilon.$$

  One says that "f is of order smaller than g" or "f is little-oh of g".

- We write $f(x) \sim g(x)$ as $x \to x_0$, if the ratio $f(x)/g(x)$ tends to 1 as $x \to x_0$. One says that "f and g are asymptotically equivalent".

The concepts above clearly depend very strongly on the point x_0 chosen. Most of the time it will be clear from the context what the choice of x_0 is and we will thus omit as $x \to x_0$ from the notation.

In order to obtain asymptotic expansions we will often use the well-known Taylor expansions of the following elementary functions:

$$(1+z)^\alpha = \sum_{n\geq 0} \binom{\alpha}{n} z^n, \qquad \frac{1}{(1-z)^{\alpha+1}} = \sum_{n\geq 0} \binom{n+\alpha}{n} z^n,$$
$$\exp z = \sum_{n\geq 0} \frac{z^n}{n!}, \qquad \log\left(\frac{1}{1-z}\right) = \sum_{n\geq 1} \frac{z^n}{n}.$$

Moreover the following two asymptotic expansions will be used:

1. *Stirling's formula*:

$$n! = \frac{n^n}{e^n}\sqrt{2\pi n}\cdot(1+\lambda_n)\,, \text{ where } 0 < \lambda_n < \frac{1}{12n}. \tag{1}$$

2. *The harmonic numbers*:

$$H_n := \sum_{i=1}^{n} \frac{1}{i} = \ln(n) + \gamma + \frac{1}{2n} + \mathcal{O}\left(\frac{1}{n^2}\right), \tag{2}$$

with the Euler-Mascheroni constant $\gamma \approx 0.5772\ldots$.

2.2 PERMUTATIONS, PATTERNS AND STANDARD YOUNG TABLEAUX

Permutations

A permutation π is a bijective function from a finite set onto itself. We will denote by π^{-1} the inverse function of π, i.e., the unique function that fulfils $\pi \circ \pi^{-1} = \pi^{-1} \circ \pi = id$. A permutation that is self-inverse, i.e., for which it holds that $\pi \circ \pi = id$, is called an *involution*.

If π is defined on a set of size n, we speak of a permutation of length n or of an n-permutation. For the definition of permutation patterns that will be given in Section 2.2 it is crucial that we have the natural order defined by $<$ on the elements of a permutation. We will thus assume that an n-permutation is always defined on the set $[n]$.

An n-permutation π can thus also be seen as the sequence $\pi(1)$, $\pi(2), \ldots, \pi(n)$ in which every one of the integers in $[n]$ appears exactly once. We will also use the notation $\pi_1, \pi_2, \ldots, \pi_n$ to refer to this sequence. This representation of a permutation is also known as the *one-line notation*. Given an integer $m \in [n]$ we may refer to m as an *element* of π and say that its *position* in π is i if $\pi^{-1}(m) = i$. Every n-permutation π defines a total order on $[n]$ that we will denote by $\prec_\pi$. We write $k \prec_\pi m$ if $\pi^{-1}(k) < \pi^{-1}(m)$, i.e., the element k lies to the left of the element m in the sequence π. In this case, we say that k is left of m or that m is right of k. We also use the non-strict total

order $\preceq_\pi$ associated to π: $k \preceq_\pi m$ iff $k \prec_\pi m$ or $k = m$. In this case, we say that k is not right of m or that m is not left of k.

Viewing permutations as sequences allows us to speak of *subsequences* of a permutation. We speak of a *contiguous subsequence* of π if the sequence consists of contiguous elements in π or, equivalently, if the corresponding positions form an interval of $[n]$. Given a set $S \subseteq [n]$, we write $\pi|_S$ to denote the subsequence of π consisting exactly of the elements of S.

Graphically, a permutation π on $[n]$ can be represented with the help of a plot in the integer plane in which dots are placed at the positions $(i, \pi(i))$. This representation thus corresponds to the function graph of π when viewing permutations as bijective maps. For examples, see Figure 7 on page 17 where the two permutations $\pi_{ex} = 2\,3\,1\,4$ and $\tau_{ex} = 1\,8\,12\,4\,7\,11\;6\,3\,2\,9\,5\,10$ which will serve as running examples throughout this section are represented with the help of plots.

We denote by $\pi^r := \pi(n)\pi(n-1)\dots\pi(1)$ the *reverse* of π and by $\pi^c := (n-\pi(1)+1)(n-\pi(2)+1)\dots(n-\pi(n)+1)$ its *complement*. From the plot corresponding to π one obtains π^r's plot by reflecting across $x = n/2$ and π^c's plots by reflecting across $y = n/2$. Also, the plot of π^{-1} is obtained by reflecting across the line $y = x$.

For other possible definitions and representations of permutations we refer to Stanley's *Enumerative combinatorics* [146], especially the section *Geometric representations of permutations*. Furthermore, for a extensive treatment of permutations as combinatorial objects, we refer to Bóna's monograph *Combinatorics of Permutations* [25].

Ups and downs in permutations: various statistics

In this section, we gather various definitions related to the up-down structure of permutations. In this thesis, three different notions of monotone subsequences in permutations will appear: alternating runs, ascending runs and longest increasing subsequences.

We discern two types of local extrema in permutations: valleys and peaks. A *valley* of a permutation π is an element $\pi(i)$ for which it holds that $\pi(i-1) > \pi(i)$ and $\pi(i) < \pi(i+1)$. If $\pi(i-1)$ or $\pi(i+1)$ is not defined, we still speak of valleys. Similarly, a *peak* denotes an element $\pi(i)$ for which it holds that $\pi(i-1) < \pi(i)$ and $\pi(i) > \pi(i+1)$.

Valleys and peaks partition a permutation into contiguous monotone subsequences, so-called *alternating runs*. The first run of a given permutation starts with its first element (which is also the first local extremum) and ends with the second local extremum. The second run starts with the following element and ends with the third local extremum. Continuing in this way, every element of the permutation belongs to exactly one alternating run. Observe that every alternating run is either increasing or decreasing. We therefore distinguish

between *runs up* and *runs down*. Note that runs up always end with peaks and runs down always end with valleys. The parameter $\mathsf{run}(\pi)$ counts the number of alternating runs in π. Hence, $\mathsf{run}(\pi)+1$ equals the number of local extrema in π. These definitions can be analogously extended to subsequences of permutations.

Example 2.1. In the permutation $\tau_{ex} = 1\,8\,12\,4\,7\,11\,6\,3\,2\,9\,5\,10$ the valleys are 1, 4, 2 and 5 and the peaks are 12, 11, 9 and 10. A decomposition into alternating runs is given by: $1\,8\,12|4|7\,11|6\,3\,2|9|5|10$ and is graphically represented in Figure 7. ⊣

A permutation in which all alternating runs except the first one are of length one only is called an *alternating permutation*. If an alternating permutation π starts with a run up it is also called an *up-down*-permutation and its elements have to satisfy $\pi(1) < \pi(2) > \pi(3) < \ldots$. Analogously, alternating permutations starting with a run down are called *down-up*-permutations and have to satisfy $\pi(1) > \pi(2) < \pi(3) > \ldots$. Note that a slightly different notation is sometimes used in the literature, calling alternating permutations in our sense *zigzag* permutations and only referring to up-down permutations as alternating permutations.

Example 2.2. The permutation $\pi_{ex} = 2\,3\,1\,4$ is an alternating permutation. To be more precise, it is an up-down permutation. ⊣

A different way of characterizing alternating permutations is with the help of *ascents* and *descents*. An ascent of a permutation π is a position i for which it holds that $\pi(i) < \pi(i+1)$. Similarly, a descent of a permutation π is a position i for which it holds that $\pi(i) > \pi(i+1)$. Thus, an alternating permutation is a permutation where every ascent is directly followed by a descent and vice-versa.

The descents of a permutation π partition it into so-called *ascending runs*. These are increasing contiguous subsequences of maximal length. The following holds: if π has k descents, it is the union of $k+1$ ascending runs. In order to make the difference between ascending and alternating runs clear, let us note the following: an alternating run that is a run up is always also an ascending run; however, a run down is the union of multiple ascending runs of length one.

Example 2.3. In our running example τ_{ex} the five descents are the following elements: 12, 11, 6, 3, 9. The decomposition into six ascending runs is given as follows : $1\,8\,12|4\,7\,11|6|3|2\,9|5\,10$. ⊣

Another important concept in permutations is that of *longest increasing* and *longest decreasing subsequences*. A longest increasing subsequence of a permutation π is a monotonically increasing subsequence that is of maximal length. Note that the longest increasing or decreasing subsequence need not be unique and in general are not contiguous subsequences.

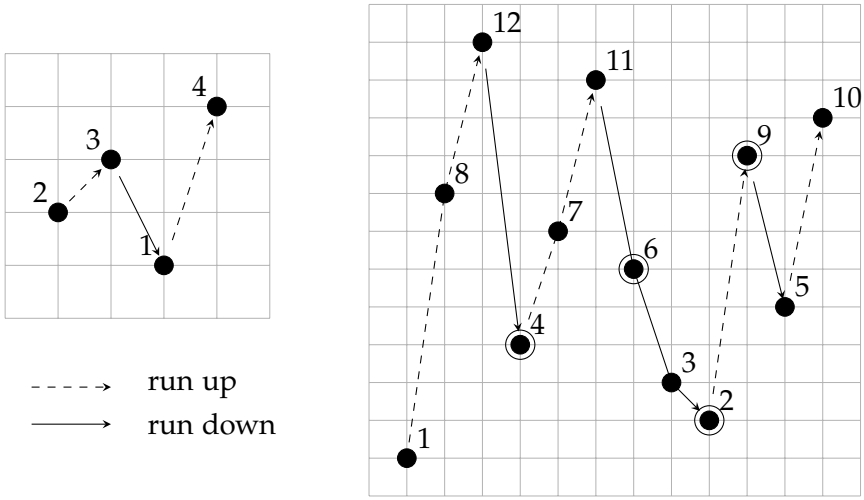


Figure 7: The pattern $\pi_{ex} = 2\,3\,1\,4$ (left-hand side) is contained in the permutation $\tau_{ex} = 1\,8\,12\,4\,7\,11\;6\,3\,2\,9\,5\,10$ (right-hand side).

Example 2.4. In our running example τ_{ex} the two increasing subsequences of maximal length are: 146910 and 147910. ⊣

Permutation patterns

Classical permutation patterns, or simply permutation patterns, are at the heart of Part i of this thesis.

Definition 2.5. *Let π be a permutation of length k and $\tau = \tau(1)\ldots\tau(n)$ a permutation of length n. An* occurrence of the permutation pattern π *in τ is a subsequence $\tau(i_1)\,\tau(i_2)\,\ldots\,\tau(i_k)$ of τ that is order-isomorphic to π. The subsequence $\tau(i_1)\,\tau(i_2)\,\ldots\,\tau(i_k)$ is order-isomorphic to π if $\tau(i_a) < \tau(i_b)$ holds iff $\pi(a) < \pi(b)$. If such a subsequence exists, one says that τ* contains *π or that there is a* matching *of π into τ. If there is no such subsequence one says than τ* avoids *the pattern π. If π is contained in τ we write $\pi \leq \tau$ and if τ avoids π we write $\pi \not\leq \tau$.*

Matching π into τ thus consists in finding a monotonically increasing map $M : [k] \to [n]$ that maps elements in π to elements in τ in such a way that the sequence $M(\pi)$, defined as $M(\pi(1))$, $M(\pi(2))$, $\ldots, M(\pi(k))$, is a subsequence of τ. We then refer to such a function M as a *matching*.

Example 2.6. The pattern π_{ex} is contained in the permutation τ_{ex} as witnessed by the subsequence 4629. The matching corresponding to this occurrence of π_{ex} is $M(1) = 2$, $M(2) = 4$, $M(3) = 6$, $M(4) = 9$. However, τ_{ex} avoids the pattern $\sigma = 1\,2\,3\,4\,5\,6$ since the length of the longest increasing subsequence of τ_{ex} is 5. For an illustration see again Figure 7. ⊣

Representing permutations with the help of plots allows for a simple interpretation of pattern containment respectively avoidance in

permutations. Indeed, the pattern π is contained in the permutation τ iff the plot corresponding to π can be obtained from the one corresponding to τ by deleting some columns and rows. Moreover, it is easy to see that the following statements are all equivalent:

- $\pi \leq \tau$
- $\pi^{-1} \leq \tau^{-1}$
- $\pi^r \leq \tau^r$
- $\pi^c \leq \tau^c$.

The pattern containment relation $\leq$ defines a partial order on the set of all permutations. This leads to the following definition:

Definition 2.7. *A* permutation class *is a downset of permutations in the containment order. That is, if $\mathcal{C}$ is a permutation class that contains a permutation τ, then every π with $\pi \leq \tau$ is also contained in $\mathcal{C}$.*

A permutation class can be defined by pattern avoidance and we use the following notation:

$$\begin{aligned} \mathrm{Av}(B) &:= \{\tau : \pi \not\leq \tau \text{ for all } \pi \in B\} \\ \mathrm{Av}_n(B) &:= \{\tau \in \mathrm{Av}(B) : \tau \text{ is of length } n\} \\ S_n(B) &:= |\mathrm{Av}_n(B)| \end{aligned}$$

If the set B is an *antichain*, i.e., no element of B is contained in another on, then it is called the *basis* of the permutation class.

For an introduction to permutation patterns and a survey of the main results of this field we refer to the Chapters 4, 5 and 8 in Bóna's *Combinatorics of Permutations* [25]. A comprehensive presentation of results related to various types of patterns in permutations and words can be found in Kitaev's monograph *Patterns in Permutations and Words* [108]. A survey of permutation classes and recent research directions can be found in Vince Vatter's Chapter 12 in [29].

Standard Young tableaux

The famous *Robinson-Schensted correspondence* establishes a bijective correspondence between permutations and pairs of standard Young tableaux of the same shape. Let us start by defining these:

Definition 2.8. *Let $\lambda = \lambda_1, \ldots, \lambda_k$ be a partition of n, i.e., $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_1 \geq 1$ and $\sum_{i=1}^{k} \lambda_i = n$. A* Young diagram *of shape λ consists of n square boxes that are arranged in k left-justified rows such that the i-th row contains λ_i boxes. A* Young tableau *is obtained by filling in the boxes of a Young diagram with elements from some ordered set. A* standard Young Tableau, *short SYT, of size n is a Young tableau on n boxes filled in with the elements in $[n]$ in such a way that every element appears exactly once and that the rows and columns are increasing when going to the right and down.*

Note that we label boxes in a Young diagram in the same way as one labels entries in matrices, i.e., we say that a box is at position (i, j) if it lies in the i-th row and the j-th column.

Example 2.9. Two SYT of size 12 and of the same shape are displayed in Figure 9. The underlying partition for both Young diagrams is $\lambda = (5, 2, 2, 2, 1)$. ⊣

We can now define the Robinson-Schensted correspondence. In a step-by-step procedure, it associates to every n-permutation σ a pair $(P(\sigma), Q(\sigma))$ of SYT of size n of the same shape. In order to alleviate notation, we will write (P, Q) instead of $(P(\sigma), Q(\sigma))$ whenever no confusion is possible. The procedure constructs a sequence (P_0, Q_0), $(P_1, Q_1), \cdots, (P_n, Q_n) = (P, Q)$ of pairs of Young tableaux of the same shape, where $P_0 = Q_0$ are empty tableaux. The intermediate tableaux P_i and Q_i for $i < n$ are in general not SYT since they don't necessarily contain all elements in $[i]$. However, they do have the property that every element occurs only once and that columns and rows are increasing. Having constructed P_{i-1}, P_i is formed by inserting $\sigma(i)$ into P_{i-1} according to the *Schensted insertion* that will be explained in a moment. The tableaux P_i are thus referred to as *insertion tableaux*. At the same time, Q_i is created by adding the element i to Q_{i-1} in the box that is added to the shape by the insertion of $\sigma(i)$ into P_{i-1}. The tableaux Q_i are referred to as *recording tableaux*.

The Schensted insertion or *row insertion* of an element $\sigma(i)$ into a Young tableau P_{i-1} can now be described as follows. If P_{i-1} is empty, we place a box at position $(1, 1)$ containing the element $\sigma(i)$. Otherwise, proceed recursively:

1. Find the leftmost element j in the first row that is larger than $\sigma(i)$. If there is no such element, we place $\sigma(i)$ at the end of the first row and are done. Otherwise, we replace the element j by $\sigma(i)$ and proceed to the next step.

2. Using the Schensted insertion, insert the element j into the Young tableau that is obtained by deleting the first row of P_{i-1}. Place the resulting tableau under the row obtained in the first step.

Example 2.10. For the permutation $\tau_{ex} = 1\,8\,12\,4\,7\,11\,6\,3\,2\,9\,5\,10$, the intermediate insertion tableau P_6 is displayed on the left-hand side of Figure 8. The next element to be inserted is $\tau_{ex}(7) = 6$. The intermediate steps of the Schensted insertion of the element 6 into P_6, leading to P_7, are shown in the same figure. The pair (P, Q) of SYT corresponding to τ_{ex} is displayed in Figure 9. ⊣

In the following, we list some important properties of the Robinson-Schensted correspondence that will be of use to use:

- If the permutation σ corresponds to the pair (P, Q) of SYT, the inverse permutation σ^{-1} corresponds to the pair (Q, P). Thus,

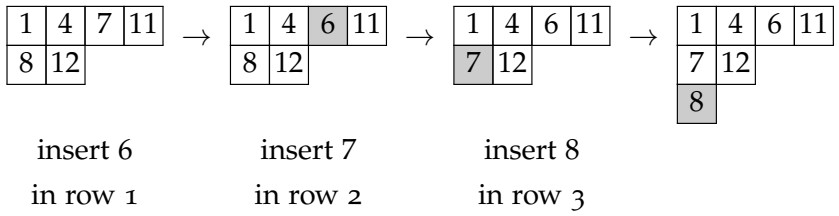


Figure 8: The Schensted insertion of the element 6 into an insertion tableau

$$P = \begin{array}{|c|c|c|c|c|}\hline 1 & 2 & 5 & 9 & 10 \\ \hline 3 & 6 \\ \cline{1-2} 4 & 11 \\ \cline{1-2} 7 & 12 \\ \cline{1-2} 8 \\ \cline{1-1}\end{array} \qquad Q = \begin{array}{|c|c|c|c|c|}\hline 1 & 2 & 3 & 6 & 12 \\ \hline 4 & 5 \\ \cline{1-2} 7 & 10 \\ \cline{1-2} 8 & 11 \\ \cline{1-2} 9 \\ \cline{1-1}\end{array}$$

Figure 9: The pair (P, Q) of SYT corresponding to τ_{ex} under the Robinson-Schensted correspondence

if σ is an involution, it corresponds to a pair (P, P) and can be identified with the single SYT P.

- The length of the first row in $P(\sigma)$ and $Q(\sigma)$ corresponds to the length of the longest increasing subsequence in σ.

The number of SYT of size n and of a given shape can be determined with the help of the so-called hooklength formula.

Definition 2.11. *Let b be a box in a Young diagram. Then the hook H_b of b consists of the box b itself, all boxes in the same row and to the right of b and all boxes in the same column and below b. The hook-length h_b is the number of cells in the hook H_b.*

Example 2.12. Let b be the box at position $(1, 2)$ in the SYT P displayed on the left-hand side of Figure 9, i.e., the box containing the element 2. Then $h_b = 3 + 3 + 1 = 7$. ⊣

Theorem 2.13. *Let D be a young diagram with n boxes. Then the number of SYT of shape D is equal to*

$$\frac{n!}{\prod_b h_b}$$

where the product is taken over all boxes b of D.

For an overview of the connections between permutations and SYT, see Chapter 7 in [25] and the references given therein. See also Chapter 14 of [29] for a recent survey by Ron Adin and Yuval Roichman on Standard Young Tableaux.

2.3 CAYLEY TREES AND MAPPINGS

Part ii is concerned with Cayley trees and with mappings, two closely related types of combinatorial objects that we introduce in the follow-

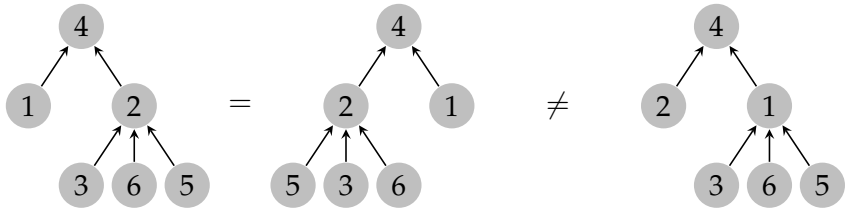


Figure 10: Examples of Cayley trees of size $n = 6$

ing. In this section, we only present the basic definitions of Cayley trees and mappings. For analytic properties of the corresponding generating functions, we refer to the examples in Section 2.6, in particular Examples 2.26, 2.27 and 2.29. For more background, we refer to [78].

Cayley trees

In a graph-theoretic sense, a *tree* is an acyclic undirected connected graph. A *rooted tree* is a tree in which one specific node is distinguished; it is called the *root*. In this case the edges can be oriented in a natural way, either towards or away from the root. Throughout this thesis, edges will always be oriented towards the root. Thus, the root node is always the unique node that has no outgoing edges. Nodes with no incoming edges are called *leaves*. Let i and j be nodes in a tree such that the directed edge (i, j) is present, i.e., j lies on the path from i to the root. Then j is called i's *parent* and i is a *child* of j. Furthermore, *unordered trees* are rooted trees in which there is no order on the children of any node. Such trees are sometimes also referred to as non-plane trees.

The main family of trees studied in this thesis is the following:

Definition 2.14. *A* Cayley tree *of size n is a unordered labelled tree with n nodes. In a labelled tree of size n every node carries a distinct integer from the set $[n]$ as a label.*

In the following, we will simply refer to Cayley trees as *trees* and indicate if we speak about other types of trees. Throughout this thesis we will always identify nodes with their labels.

Example 2.15. Three examples of Cayley trees of size $n = 6$ can be found in Figure 10. The tree to the left hand side and the one in the middle are the same since they can be obtained from another by changing the order of some subtrees. However, the tree on the right hand side is actually different from the other two trees. ⊣

The number of Cayley trees of size n, that is denoted by t_n throughout this thesis, is given by the following simple formula

$$t_n = n^{n-1}. \tag{3}$$

This formula is attributed to Arthur Cayley and thus also referred to as "Cayley's formula". It is sometimes also stated in the following form: the number of (unrooted) labelled trees of size n is equal to n^{n-2}. Many proofs of this formula have been given since Cayley's proof in 1889, using various different methods. See for instance the collection of proofs in *Proofs from THE BOOK* [2]. In Chapter 7 we will present a new bijective proof of Cayley's formula.

Mappings

Definition 2.16. *A function f from the set $[n]$ into itself is called an n-*mapping*. Given an n-mapping f, its* functional digraph *G_f is defined to be the directed graph on the vertex set $[n]$ where a directed edge (i, j) is drawn whenever f maps i to j.*

In this thesis we will always identify a mapping with its functional digraph and will not make any distinction between the objects f and G_f. Sometimes, mappings are also referred to as *functional graphs* or *functional digraphs* in the literature.

Note that an n-mapping can alternatively also be interpreted as a sequence $(f_i)_{i\in[n]}$ of length n over the alphabet $[n]$ by setting $f_i = f(i)$.

The structure of mappings is simple and is well described in [78]: the weakly connected components of their digraphs are simply cycles of Cayley trees. That is, each connected component consists of rooted labelled trees whose root nodes are connected by directed edges such that they form a cycle. In this context, we will call a node j that lies on a cycle in a mapping f, i.e., for which there exists a $k \geq 1$ such that $f^k(j) = j$, a *cyclic node*.

How this structural connection to trees can be translated into the language of generating functions using the symbolic method is described in Example 2.27 on page 30.

Example 2.17. For the 19-mapping corresponding to the sequence $7, 1, 10, 14, 17, 7, 11, 10, 16, 10, 17, 7, 17, 4, 7, 4, 1, 13, 13$, see Figure 11. This mapping consists of three connected components: one of size 4 with a cycle length 2, another one of size 12 with a cycle length 4 and a last one of size 3 with a cycle length 1. ⊣

2.4 PREFERENCES AND SOCIAL CHOICE THEORY

One of the central goals in Social Choice Theory is to design methods that allow to combine individual preferences in order to obtain a collective decision. In order to formalize the notion of "preferences", let us start by gathering some notation on sets and orders.

Let S be a finite set. A relation on S is total if for every $a, b \in S$, either the pair (a, b) or (b, a) is contained in the relation. A *total order*

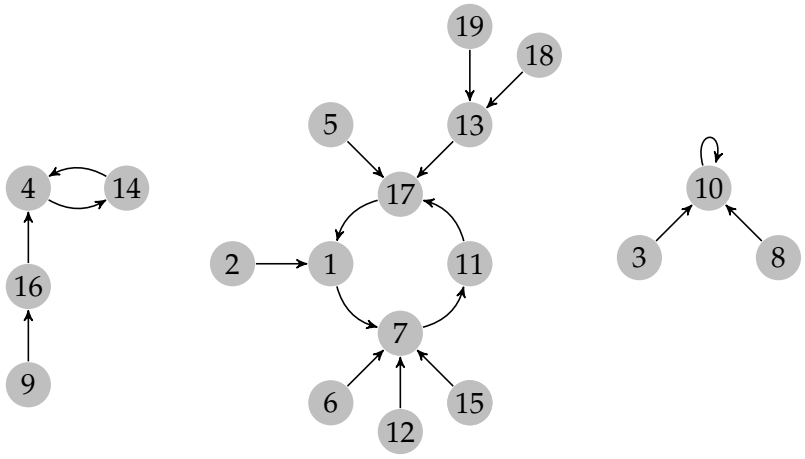


Figure 11: Functional graph of a 19-mapping

on S is a reflexive, antisymmetric, transitive and total relation. Let T be a total order of S. Instead of writing $(a, b) \in T$, we write $a \leq_T b$ or $b \geq_T a$. We write $a <_T b$ or $b >_T a$ to state that $a \leq_T b$ and $a \neq b$. As a short form, we write $T : s_1 s_2 s_3 \dots s_i$ instead of $s_1 >_T s_2 >_T s_3 >_T \dots >_T s_i$ for $s_1, s_2, \dots, s_i$ in S. We write $T(i)$ to denote the i-th largest element with respect to T.

Every pair (T_1, T_2) of total orders on a set with m elements can be identified with the m-permutation $p(T_1, T_2)$, which is defined as follows: i is the i-th largest element in T_1 maps to the j-th largest element in T_2. For $T_1 : bac$ and $T_2 : cab$ we have $p(T_1, T_2) = 321$. Note that $p(T_1, T_2) = p(T_2, T_1)^{-1}$.

We can now define elections:

Definition 2.18. *An (n, m)-election $(C, \mathcal{P})$ consists of a size-m set C and an n-tuple $(V_1, \dots, V_n)$ of total orders on C. The set C is referred to as the* candidate set. *The total orders $V_1, \dots, V_n$ are votes or* preferences.

We write $V \in \mathcal{P}$ to denote that there exists an index $i \in [n]$ such that $V = V_i$. Given a vote $V_i \in \mathcal{P}$ with $V_i : c_i c_j$, this means that the i-th voter prefers candidate c_i to candidate c_j.

When counting elections we do not care about the specific names that candidates have. Thus, if the candidate set C consists of m elements, we assume that the candidates are $c_1, c_2, \dots, c_m$. This implies that the number of (n, m)-elections is $(m!)^n$.

The most fundamental result in Social Choice Theory is an impossibility result. Namely, it states that there is no way of aggregating individual preference in a way that three very basic and desirable properties are fulfilled simultaneously. In order to state the theorem, let us introduce the notion of a *social welfare function*: a social welfare function is a function that associates to every election $(C, \mathcal{P})$ a single preference on the set C. The idea is that this single preference is an aggregation of all the preferences that are present in the election.

Theorem 2.19 (Arrow's paradox [10])**.** *Let f be a social welfare function and $(C, \mathcal{P})$ an election with preferences $V_1, \ldots, V_n$. As soon as $|C| > 2$ the following three conditions are incompatible:*

- *unanimity, or* Pareto efficiency*: If $V_i : ab$ holds for all preferences, then $f((C, \mathcal{P})) : ab$.*
- independence of irrelevant alternatives*: Consider a second election $(C, \mathcal{Q})$ with preferences $W_1, \ldots, W_n$. If for all voters i it holds that candidates a and b have the same order in V_i as in W_i, candidates a and b have the same order in $f((C, \mathcal{P}))$ as in $f((C, \mathcal{Q}))$*
- non-dictatorship*: There is no voter i whose preferences always prevail. That is, there is no $i \in [n]$ such that for all elections $(C, \mathcal{P})$ it holds that $f((C, \mathcal{P})) = V_i$.*

If we make the task easier and do not ask for a total ordering of all candidates but only for a winner of the election, this leads to the concept of *voting rules*. However, this only seemingly decreases the difficulty of the problem since the Gibbard-Satterthwaite theorem [89, 137] states a similar impossibility result as Arrow's paradox.

2.5 ALGORITHMS AND COMPLEXITY THEORY

This thesis classifies several permutation pattern matching problems by proving membership in complexity classes, see Chapters 4 and 3. We thus give a brief introduction to classical and parameterized complexity theory in the following.

Classical complexity theory

The two fundamental classes in classical complexity theory are P and NP. The class P contains all decision problems that can be solved in polynomial time on a deterministic Turing machine. A decision problem is a computational problem that takes as input some string over an alphabet Σ and outputs either "YES" or "NO". A subset $L \subseteq \Sigma^*$ defines a decision problem in the following sense: the strings in L are the YES-instances of the problem and all other strings in Σ^* are NO-instances. If there exists an algorithm that outputs the correct answer for any input string of length n in $\mathcal{O}(n^c)$ steps, where c is a constant that is independent of the input, then one says that the problem can be solved in polynomial time. Many natural problems are contained in P. For instance, consider the following decision problem related to permutation patterns (see Definition 2.5):

PERMUTATION ORDER-ISOMORPHISM	
Instance:	Two sequences of integers σ and τ of the same length.
Question:	Are σ and τ order-isomorphic?

This question can be answered by sorting the sequence σ using any sorting algorithm, for instance bubble-sort. While we sort σ, we perform exactly the same operations on τ and obtain a sequence $\tilde{\tau}$. This can be done in $\mathcal{O}(n^2)$-time. It remains to check whether $\tilde{\tau}$ is a non-decreasing sequence which can be done in linear time. The sequences σ and τ are order-isomorphic if and only if this is the case.

The class NP contains all problems that can be solved in polynomial time on a *non-deterministic* Turing machine. This means that a problem is contained in NP if the instances where the answer is "YES" have polynomial time verifiable proofs of the fact that the answer is indeed "YES". For instance, the following problem has this property:

CLIQUE	
Instance:	A graph $G = (V, E)$ and a positive integer k.
Question:	Is there a subset of vertices $S \subseteq V$ of size k such that S forms a clique, i.e., the induced subgraph $G[S]$ is complete?

Here, a proof of the fact that the input graph G contains a clique of size k would be a set $S = \{s_1, \ldots, s_k\}$ containing k of the vertices. In order to verify that the induced subgraph $G[S]$ indeed is a clique we need to check for all subsets $\{s_i, s_j\}$ of S of size two whether the corresponding edge is contained in E. This can clearly be done in $\mathcal{O}(k^2) = \mathcal{O}(n^2)$-time and thus CLIQUE is contained in NP.

The CLIQUE problem does not only have the property to be contained in NP, it is also NP-hard and thus NP-complete. Informally speaking, this means that CLIQUE is among the hardest problems in NP: if CLIQUE can be solved in polynomial time, this can be done for any other problem in NP as well. In order to formalize these notions, we need to introduce the concept of *polynomial time many-one reductions*. A polynomial-time many-one reduction from a problem L_1 to a problem L_2 transforms inputs to the first problem into inputs to the second problem in polynomial time and in such way that the output of the transformed problem is the same as for the original problem.

Definition 2.20. *Let $L_1, L_2 \subseteq \Sigma^*$ be two decision problems. A* polynomial many-one time reduction *from L_1 to L_2 is a mapping $G : \Sigma^* \to \Sigma^*$ such that*

- *$I \in L_1$ if and only if $G(I) \in L_2$.*
- *G is computable by a polynomial time-algorithm.*

A problem is called NP-hard if every problem in NP can be reduced to it by a polynomial time reduction. Note that this can also be the case for decision problems that are not in NP. Furthermore, a problem is NP-complete if it is contained in NP and NP-hard. The CLIQUE decision problem was one of the 21 problems shown to be NP-complete by Richard Karp in 1972 [106]. Under standard complexity-theoretical assumptions, namely $\mathsf{P} \neq \mathsf{NP}$, this result implies that CLIQUE cannot be solved in polynomial time.

For a formal definition of Turing machines and the complexity classes mentioned above, see e.g. [97]. A detailed introduction to complexity theory can be found in the monographs by Papadimitriou [132], Goldreich [91], and Arora and Barak [9].

Parameterized complexity theory

Many problems that have practical relevance are known to be NP-complete. Thus, unless $\mathsf{P} = \mathsf{NP}$, there is no hope for polynomial time algorithms for a large class of problems. If such problems need to be solved efficiently anyhow, methods such as approximation algorithms or heuristics can be used. Another, relatively new approach that originated in the work of Downey and Fellows [59] is to try to confine the combinatorial explosion to a *parameter* of the input. For instance, if the treewidth of the input graph is bounded, CLIQUE can be solved in $\mathcal{O}(n^c)$-time with a constant c that is independent of the treewidth [53]. Finding such parameters that allow to handle NP-hard problems efficiently is the key idea of parametrized complexity theory.

In contrast to classical complexity theory, a parameterized complexity analysis studies the runtime of an algorithm with respect to an additional parameter and not just the input size $|I|$. Therefore every parameterized problem is considered as a subset of $\Sigma^* \times \mathbb{N}$. An instance of a parameterized problem consequently consists of an input string together with a positive integer p, the parameter.

Definition 2.21. *A parameterized problem is* fixed-parameter tractable *(or in* FPT*) if there is a computable function f and an integer c such that there is an algorithm solving the problem in $\mathcal{O}(|I|^c \cdot f(p))$ time.*

The algorithm itself is also called fixed-parameter tractable (fpt).

A central concept in parameterized complexity theory are *fixed-parameter tractable reductions*.

Definition 2.22. *Let $L_1, L_2 \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. An* fpt-reduction *from L_1 to L_2 is a mapping $R : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that*

- *$(I, k) \in L_1$ iff $R(I, k) \in L_2$.*
- *R is computable by an fpt-algorithm.*

- *There is a computable function g such that for $R(I,k) = (I',k')$, $k' \leq g(k)$ holds.*

Other important complexity classes in the framework of parameterized complexity are $\mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \ldots$, the W-hierarchy. For our purpose, only the class $\mathsf{W}[1]$ is relevant. It can be defined as follows:

Definition 2.23. *The class* $\mathsf{W}[1]$ *is the class of all problems that are fpt-reducible to the following parameterized version of the* Clique *problem.*

Clique	
Instance:	A graph $G = (V, E)$ and a positive integer k.
Parameter:	k
Question:	Is there a subset of vertices $S \subseteq V$ of size k such that S forms a clique, i.e., the induced subgraph $G[S]$ is complete?

It is conjectured (and widely believed) that $\mathsf{W}[1] \neq \mathsf{FPT}$. Therefore showing $\mathsf{W}[1]$-hardness can be considered as evidence that the problem is not fixed-parameter tractable.

Definition 2.24. *A parameterized problem is in* XP *if it can be solved in time $\mathcal{O}(|I|^{f(k)})$ where f is a computable function.*

All the aforementioned classes are closed under fpt-reductions. The following relations between these complexity classes are known:

$$\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \ldots \subseteq \mathsf{XP} \quad \text{as well as}$$

$$\mathsf{FPT} \subset \mathsf{XP}.$$

Further details can be found, for example, in the monographs by Downey and Fellows [59], Niedermeier [127] and Flum and Grohe [80].

2.6 SYMBOLIC METHOD AND ANALYTIC COMBINATORICS

For an extensive introduction to the field of *Analytic Combinatorics* and a thorough treatment of the underlying theory, we refer to the monograph be Flajolet and Sedgewick [78]. Also, for a vivid presentation of generating functions as a bridge between discrete mathematics and continuous analysis we refer to Herbert Wilf's *generatingfunctionology* [158].

Generating functions

Throughout this thesis we consider various combinatorial sequences $(a_n)_{n\geq 0}$, i.e., $a_n = |\mathcal{A}_n|$ is the number of objects of size n in some combinatorial class $\mathcal{A}$. The size of an object α in $\mathcal{A}$, denoted by $|\alpha|$,

is a non-negative integer assigned to the object α. Throughout this thesis, it will always be clear from the context how the concept of size is to be understood. For instance, the size of a Cayley tree is simply its number of nodes.

The *(ordinary) generating function* (OGF) of $\mathcal{A}$ is the (formal) power series

$$A(z) = \sum_{n\geq 0} a_n z^n = \sum_{n\geq 0} |\mathcal{A}_n| z^n = \sum_{\alpha\in\mathcal{A}} z^{\alpha}. \tag{4}$$

The combinatorial objects in this thesis are always labelled ones. That is, every object of size n of $\mathcal{A}$ consists of n "atoms" (e.g., the nodes in a tree) to which the integers in $[n]$ are associated in a bijective manner. In this context one considers the *exponential generating function* (EGF) associated with the class $\mathcal{A}$:

$$A(z) = \sum_{n\geq 0} a_n \frac{z^n}{n!} = \sum_{n\geq 0} |\mathcal{A}_n| \frac{z^n}{n!} = \sum_{\alpha\in\mathcal{A}} \frac{z^{|\alpha|}}{|\alpha|!}. \tag{5}$$

Note that the concepts presented here are also applicable to a more general setting where sequences $(a_n)_{n\geq 0}$ are merely required to be sequences of real numbers.

The symbolic method

The *symbolic method* is a powerful tool in the context of generating functions. It allows to translate certain combinatorial constructions directly to equations for generating functions. Since all objects analysed in this thesis are labelled ones, we present the symbolic method for the case of exponential generating functions only. For the case of ordinary generating functions and even more combinatorial constructions, we refer to Chapters I and II of [78].

In what follows we collect some of the so-called *admissible constructions* that will be used in this thesis and which make it possible to build complex combinatorial classes from simpler ones.

- $\dot\cup$ *Disjoint union:* If $\mathcal{A}$ and $\mathcal{B}$ are two disjoint labelled classes, then $\mathcal{A}\dot\cup\mathcal{B}$ denotes the union of $\mathcal{A}$ and $\mathcal{B}$ in the set-theoretic sense. However, $\dot\cup$ can also be defined for not necessarily disjoint classes: If $\mathcal{A}$ and $\mathcal{B}$ are two arbitrary labelled classes, then $\mathcal{A}\dot\cup\mathcal{B}$ denotes the union of two *disjoint copies* of $\mathcal{A}$ and $\mathcal{B}$. Such disjoint copies can, e.g., be constructed by choosing two distinct colours, and colouring the elements of $\mathcal{A}$ with the first colour, and the elements of $\mathcal{B}$ with the second one.

- $\star$ *Labelled product:* The labelled product $\beta\star\gamma$ of two labelled objects β and γ of respective sizes $|\beta| = m$ and $|\gamma| = n$ is a set of pairs, defined as follows

$$\beta\star\gamma := \big\{(\beta',\gamma') \,\big|\, \text{distinct labels in } [m+n] \text{ s.t. } \rho(\beta') = \beta, \rho(\gamma') = \gamma\big\},$$

where ρ relabels each labelled object of size k such that exactly the integers $\{1, \ldots, k\}$ are used, while preserving the relative order of all labels. The labelled product $\mathcal{B} \star \mathcal{C}$ of two labelled classes $\mathcal{B}$ and $\mathcal{C}$ is defined by

$$\mathcal{B} \star \mathcal{C} := \bigcup_{\beta \in \mathcal{B}, \gamma \in \mathcal{C}} (\beta \star \gamma).$$

SEQ *Sequences:* The sequence class of $\mathcal{B}$ is defined by

$$\textsc{Seq}(\mathcal{B}) := \{\epsilon\} \,\dot\cup\, \mathcal{B} \,\dot\cup\, \mathcal{B} \star \mathcal{B} \,\dot\cup\, \mathcal{B} \star \mathcal{B} \star \mathcal{B} \,\dot\cup\, \ldots = \bigcup_{k \geq 0} \mathcal{B}^k,$$

where ϵ denotes the empty sequence.

SET *Sets:* Let the equivalence relation $\mathbf{R}$ identify sequences of equal length whenever the components of one are a permutation of the components of the other. Then the set class of $\mathcal{B}$ is defined as the quotient $\textsc{Seq}(\mathcal{B}) / \mathbf{R}$.

CYC *Cycles:* Let the equivalence relation $\mathbf{S}$ identify sequences of equal length whenever one can be obtained from the other by cyclically shifting the components of the other, i.e., $\beta \mathbf{S} \gamma$ for $\beta = (\beta_1, \ldots, \beta_n)$ and $\gamma = (\gamma_1, \ldots, \gamma_m)$ iff $(\beta_k, \ldots, \beta_n, \beta_1, \ldots, \beta_{k-1}) = (\gamma_1, \ldots, \gamma_m)$ for some k. The cycle class of $\mathcal{B}$ is then defined as the quotient $(\textsc{Seq}(\mathcal{B}) \setminus \{\epsilon\}) / \mathbf{S}$.

The following theorem states how these admissible constructions allow for a direct translation into equations for the associated exponential generating functions:

Theorem 2.25 (Symbolic method [78])**.** *Let $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ be classes of labelled objects, and $A(z)$, $B(z)$, $C(z)$ their associated exponential generating functions. Then the following holds:*

$$\begin{aligned}
&\text{If } \mathcal{A} = \mathcal{B} \,\dot\cup\, \mathcal{C}, && \text{then} && A(z) = B(z) + C(z),\\
&\text{if } \mathcal{A} = \mathcal{B} \star \mathcal{C}, && \text{then} && A(z) = B(z) \cdot C(z),\\
&\text{if } \mathcal{A} = \textsc{Seq}(\mathcal{B}) \text{ and } \mathcal{B}_0 = \emptyset, && \text{then} && A(z) = \frac{1}{1 - B(z)},\\
&\text{if } \mathcal{A} = \textsc{Set}(\mathcal{B}) \text{ and } \mathcal{B}_0 = \emptyset, && \text{then} && A(z) = \exp B(z),\\
&\text{if } \mathcal{A} = \textsc{Cyc}(\mathcal{B}) \text{ and } \mathcal{B}_0 = \emptyset, && \text{then} && A(z) = \log\left(\frac{1}{1 - B(z)}\right).
\end{aligned}$$

Example 2.26. As defined in Section 2.3, a Cayley tree consists of a root node and a set of Cayley trees attached to it as children. If $\mathcal{T}$ denotes the labelled combinatorial class of Cayley trees its combinatorial construction can thus be described as follows

$$\mathcal{T} = \mathcal{Z} \star \textsc{Set}(\mathcal{T})$$

where $\mathcal{Z}$ is an atom, i.e., a single node carrying the label 1. We thus immediately obtain that $T(z)$, the EGF of $\mathcal{T}$, is implicitly defined by the following functional equation

$$T(z) = ze^{T(z)}. \tag{6}$$

This function is known as the *(Cayley) tree function* and will appear at several occasions in this thesis.

The tree function is closely related to the so-called (Lambert) W-function [51] that is defined by the following functional equation:

$$z = W(z)e^{W(z)}.$$

The simple connection between the Cayley tree function and the Lambert W-function is:

$$T(z) = -W(-z).$$

⊣

Example 2.27. Using the symbolic method, the structural connection between Cayley trees and mappings can also easily be taken to the level of generating functions. Indeed, if $\mathcal{T}$, $\mathcal{C}$ and $\mathcal{M}$ denote the combinatorial classes of Cayley trees, connected mappings and mappings, respectively, and if $T(z)$, $C(z)$ and $M(z)$ denote their exponential generating functions we have the following connection:

$$\mathcal{C} = \textsc{Cyc}(\mathcal{T}) \qquad \Longrightarrow \qquad C(z) = \log\left(\frac{1}{1-T(z)}\right)$$

$$\mathcal{M} = \textsc{Set}(\mathcal{C}) \qquad \Longrightarrow \qquad M(z) = \exp(C(z)) = \frac{1}{1-T(z)}$$

⊣

In this thesis we will often be interested in enumerating combinatorial objects of a class $\mathcal{A}$ not only according to size but also according to some parameter χ. A d-dimensional parameter $\chi = (\chi_1, \ldots, \chi_d)$ is a function that maps an object $\alpha \in \mathcal{A}$ to a tuple $\chi(\alpha)$ of non-negative integers. In our studies, we will mostly have $d = 1$.

Given a labelled combinatorial class $\mathcal{A}$ and a d-dimensional parameter χ on $\mathcal{A}$, the *multivariate exponential generating function* of the pair $\langle \mathcal{A}, \chi \rangle$ is defined by

$$A(z, \mathbf{u}) := \sum_{\alpha \in \mathcal{A}} \mathbf{u}^{\chi(\alpha)} \frac{z^{|\alpha|}}{|\alpha|!},$$

where $\mathbf{u} := (u_1, \ldots, u_d)$ and $\mathbf{u}^{(k_1,\ldots,k_d)} := u_1^{k_1} \cdots u_d^{k_d}$.

In case of a 1-dimensional parameter χ, the *bivariate exponential generating function* of the pair $\langle \mathcal{A}, \chi \rangle$ is defined by

$$A(z, u) := \sum_{n \geq 0} \sum_{k \geq 0} \frac{A_{n,k}}{n!} u^k z^n, \tag{7}$$

where $A_{n,k}$ is the number of combinatorial objects in $\mathcal{A}$ of size n and for which $\chi = k$. One says that the variable z in $A(z,u)$ marks the size whereas u marks χ.

When working with multivariate generating functions, one can use an extension of the symbolic method. This will not be detailed here and we refer again to [78] as well as to the new monograph *Analytic Combinatorics in Several Variables* by Pemantle and Wilson [133].

Extracting coefficients

Once we have computed the generating function $A(z)$ of some combinatorial class $\mathcal{A}$, we need to know how to get our hands on the coefficients A_n (or $A_n/n!$ for EGF) since this is what we are actually interested in. In this context, we will denote by $[z^n]$ the operator which extracts the coefficient of z^n from a (formal) power series, i.e. $[z^n]A(z) = a_n$, if $A(z) = \sum_{n\geq 0} a_n z^n$. Sometimes, the generating functions will take simple forms and extracting coefficients can be done easily by using the Taylor expansions of well-known functions such as those in Section 2.1.

If this is not the case, it can be helpful to apply *Cauchy's integral formula*. If the radius of convergence of $A(z)$ is positive the coefficient $[z^n]A(z)$ can be obtained as follows

$$[z^n]A(z) = \frac{1}{2\pi i}\oint \frac{A(z)}{z^{n+1}}dz, \tag{8}$$

where the integral is taken in counter-clockwise direction along a simple closed curve around the origin that lies completely inside the circle of convergence of $A(z)$.

Moreover, we will often deal with generating functions $F(z)$ which are implicitly given by a functional equation of the form $F(z) = z\varphi(F(z))$. This is for instance the case for the Cayley tree function defined in Equation (6) where $\varphi(u) = \exp(u)$. When extracting coefficients of such an implicitly defined function, the following theorem proves useful:

Theorem 2.28 (Lagrange's inversion formula [78])**.** *Let $F(z)$ and $\varphi(u)$ be formal power series which satisfy $F(z) = z\varphi(F(z))$ and $\varphi_0 = [u^0]\varphi(u) \neq 0$. Then one has*

$$[z^n]g(F(z)) = \begin{cases} \frac{1}{n}[F^{n-1}]g'(x)(\varphi(F))^n, & n > 0 \\ [F^0]g(F), & n = 0, \end{cases}$$

for every formal power series $g(x)$.

Example 2.29. Let us consider the Cayley tree function $T(z)$. Its functional equation is of the form described in Theorem 2.28 with $\varphi(u) = \exp(u) = \sum_{n\geq 0} u^n/n!$. Thus $\varphi_0 = 1$ and the conditions are fulfilled. If

we are interested in the coefficients of $T(z)$, i.e., $t_n/n!$ where t_n is the number of Cayley trees of size n, we have $g(x) = x$.

We thus obtain

$$\begin{aligned} t_n =& n! \cdot [z^n]T(z) = n! \cdot \frac{1}{n}[T^{n-1}](\exp(T))^n = n! \cdot \frac{1}{n}[T^{n-1}](\exp(T))^n \\ =& n!\frac{1}{n}[T^{n-1}] \sum_{k\geq 0} \frac{n^k T^k}{k!} = n^{n-1} \end{aligned}$$

and $t_0 = [F^0]F = 0$. We have thus rediscovered Cayley's formula, using the symbolic method and Lagrange inversion. ⊣

Singularity analysis

Singularity analysis is one of the most prominent techniques presented in Flajolet and Sedgewick's monograph *Analytic combinatorics* [78]. It allows to give asymptotic expansions for the coefficients of generating functions that have isolated singularities on the boundary of their disc of convergence. Singularity analysis is based on the following two statements:

> **First Principle of Coefficient Asymptotics**. The location of a function's singularities dictates the exponential growth of its coefficients.
>
> **Second Principle of Coefficient Asymptotics.** The nature of a function's singularities determines the associate subexponential factor.

In the following, let us assume that the generating function $f(z)$ has a unique dominant singularity ρ, i.e., ρ is the unique singularity that lies on the boundary of f's disc of convergence. Then one can of course make use of the scaling rule

$$[z^n]f(z) = \rho^{-n}[z^n]f(\rho z),$$

and hence consider the case where $\rho = 1$ only. In this case, the following theorem can be applied:

Theorem 2.30 (Big-Oh transfer [76])**.** *Let $R > 1$ and $0 < \phi < \frac{\pi}{2}$, and assume that $f(z)$ is analytic in a so-called* Δ-domain

$$\Delta = \Delta(\phi, R) := \{z \,|\, |z| < R, z \neq 1, |Arg(z-1)| > \phi\}.$$

Furthermore, assume that, as $z \to 1$ in Δ,

$$f(z) = \mathcal{O}\left(|1-z|^\alpha\right),$$

for some constant $\alpha \in \mathbb{R}$. Then, as $n \to \infty$,

$$[z^n]f(z) = \mathcal{O}\left(n^{-\alpha-1}\right).$$

As a consequence, one obtains the following:

Theorem 2.31 (Singularity analysis [76]). *Assume that $f(z)$ is analytic in the domain Δ from Theorem 2.30, and that, as $z \to 1$ in Δ,*

$$f(z) = \sum_{j=0}^{m} c_j (1-z)^{\alpha_j} + \mathcal{O}\left(|1-z|^A\right),$$

for real numbers $\alpha_0 \leq \alpha_1 \leq \ldots \leq \alpha_m < A$ and a sequence of complex numbers $(c_j)_{0 \leq j \leq m}$. Then, as $n \to \infty$,

$$[z^n] f(z) = \sum_{j=0}^{m} c_j \binom{n-\alpha_j-1}{n} + \mathcal{O}\left(n^{-A-1}\right).$$

In case f has a finite number of dominant singularities, a similar result can be applied and the contributions of each singularity are added up. We won't detail this here and refer to [78].

As we will often encounter generating functions that are defined by a functional equation of the type $F(z) = z \cdot \varphi\big(F(z)\big)$ as in Theorem 2.28, the following theorem will be useful:

Theorem 2.32 (Singular Inversion, Theorem VI.6 in [76]). *Let $F = F(z)$ be a function satisfying the functional equation*

$$F = z \cdot \varphi\big(F\big), \text{ and } F(0) = 0,$$

where $\varphi(u) = \sum_{n \geq 0} \varphi_n u^n$ with $\varphi_n \in \mathbb{R}$. Assume that $\varphi(u)$ fulfils the following conditions:

- *$\varphi(u)$ is analytic at $u = 0$, $\varphi_0 \neq 0$ and $\varphi(u) \not\equiv \varphi_0 + \varphi_1 u$*
- *Within the open disc of convergence of $\varphi(u)$, $|u| < R$, there exists a (then necessarily unique) solution to the so-called* characteristic equation

$$\exists \tau : 0 < \tau < R, \ \varphi(\tau) - \tau \cdot \varphi'(\tau) = 0. \tag{9}$$

- *$\varphi(u)$ is* aperiodic, *i.e., there does not exist an integer $p \geq 2$ such that $\varphi(u) = \psi(u^p)$ for some function ψ*

Then it follows that

$$\rho = \frac{\tau}{\varphi(\tau)} = \frac{1}{\varphi'(\tau)}$$

is the radius of convergence of F and f has a unique dominant singularity at $z = \rho$, where it admits a local expansion:

$$F(z) = \tau - \sqrt{\frac{2\varphi(\tau)}{\varphi''(\tau)}} \cdot \sqrt{1 - \frac{z}{\rho}} + \sum_{j \geq 2} (-1)^j d_j \left(1 - \frac{z}{\rho}\right)^{j/2},$$

where the d_j are some computable constants. Thus, $F(z)$ is analytic in a Δ-domain.

Example 2.33. The above theorem can easily be applied to the Cayley tree function (6) in order to obtain an asymptotic expansion of $T(z)$. Here, the function $\varphi(u)$ is simply $\exp(u)$ and clearly fulfils the three conditions of the theorem. The characteristic equation is

$$e^u - \tau \cdot e^u = 0,$$

implying that $\tau = 1$ and $\rho = e^{-1}$. Thus $T(z)$ has its unique dominant singularity at e^{-1} and admits the following local expansion:

$$T(z) = 1 - \sqrt{2} \cdot \sqrt{1 - ez} + d_2(1 - ez) + \mathcal{O}\left((1 - ez)^{3/2}\right), \qquad (10)$$

where the constant d_2 can be determined as follows. With help of the Taylor expansion of $T \exp(-T)$ around $T = 1$, we obtain:

$$z = \frac{1}{e} - \frac{1}{2e}(T-1)^2 + \frac{1}{3e}(T-1)^3 - \frac{1}{8e}(T-1)^4 + \dots$$

Solving for T gives

$$T - 1 = -\sqrt{2} \cdot \sqrt{1 - ez} + \frac{2}{3}(1 - ez) + \mathcal{O}\left((1 - ez)^{3/2}\right)$$

and thus $d_2 = 2/3$. A further expansion could be obtained using the same idea. ⊣

Saddle point method

The saddle point method allows to investigate the asymptotic behaviour of the coefficients of generating functions. It is especially useful when the generating function is free of singularities and other methods such as singularity analysis presented in the previous section cannot be applied. We shall present this method with the help of an example and will follow the presentation in Chapter VIII of *Analytic combinatorics* [78]. We also refer to [57] for an instructive exposition of this method.

A saddle point of a surface is a point that resembles the inner part of a saddle or of a pass between two mountains. If the surface is defined by the modulus of an analytic function, saddle points are simply the zeros of the derivative of the function.

The main idea of the saddle point method is to express the coefficients with the help of Cauchy's Integral formula and to choose a suitable integration contour passing through or at least close to the dominant saddle point, i.e., the saddle point lying closest to the origin. Thus, one chooses the contour in such a way that, locally around the saddle point, it follows the steepest descents and steepest ascent lines. This ensures that the main contribution of the integral comes from a small part of the curve containing the saddle point.

This technique is detailed and illustrated in the following example.

Example 2.34. In the following, we shall use the saddle point method in order to prove Stirling's formula (1) for the asymptotic equivalent of the factorials. The goal is thus to estimate $\frac{1}{n!} = [z^n]\exp(z)$. Using Cauchy's Integral formula (8), we obtain

$$\frac{1}{n!} = I_n = \frac{1}{2\pi i}\oint \frac{\exp(z)}{z^{n+1}}dz,$$

where the integration is performed along a circle centred at the origin. The saddle points can be found where the derivative of the integrand vanishes, i.e., at:

$$\left(\frac{\exp(z)}{z^{n+1}}\right)' = \frac{\exp(z)(z-(n+1))}{z^{n+2}} = 0 \Leftrightarrow z = n+1.$$

We thus expect to obtain an asymptotic estimate of $n!$ by adopting a circle passing through the saddle point or close to it. Thus, we pass to polar coordinates and set $z = ne^{i\phi}$:

$$I_n = \frac{1}{2\pi}\cdot\frac{e^n}{n^n}\int_{-\pi}^{\pi}\exp\left(n(e^{i\phi}-1-i\phi)\right)d\phi,$$

where we will denote by $h(\phi)$ the function $e^{i\phi}-1-i\phi$. We now need to split the integral in two parts: a central part that corresponds to the integral for a small angle $\phi \in [-\phi_0, \phi_0]$ and that will deliver the main contribution as well as the remaining tails that will be asymptotically negligible.

$$I_n^{(0)} = \int_{-\phi_0}^{\phi_0}\exp\left(nh(\phi)\right)d\phi \text{ and } I_n^{(1)} = \int_{\phi_0}^{2\pi-\phi_0}\exp\left(nh(\phi)\right)d\phi.$$

In order to satisfy the desired tail and central approximation, one can make use of the *saddle point dimensioning heuristic*. For the case of an integral of the form $\oint \exp(f(z))dz$ and a saddle point at $z = \zeta$, it states that the angle ϕ_0 has to satisfy the following:

$$f''(\zeta)(\phi_0)^2 \xrightarrow{n\to\infty} \infty \text{ and } f'''(\zeta)(\phi_0)^3 \xrightarrow{n\to\infty} 0$$

where the choice of ϕ_0 clearly depends on n. Here, ϕ_0 needs to satisfy

$$nh''(0)(\phi_0)^2 = -n(\phi_0)^2 \xrightarrow{n\to\infty} \infty \text{ and } nh'''(0)(\phi_0)^3 = -i\cdot n(\phi_0)^3 \xrightarrow{n\to\infty} 0$$

and we can choose $\phi_0 = n^\alpha$ where α is a number between $-1/2$ and $-1/3$. Let us thus fix

$$\phi_0 = n^{-2/5}.$$

In particular, this implies that the angle of the central region of the integral tends to zero when n tends to infinity.

The asymptotic approximation of I_n now consists of three parts: *tails pruning*, *central approximation* and *tails completion*.

1) *Tails pruning:* With $z = ne^{i\phi}$, one has $|\exp(z)| = \exp(n\cos(\phi))$. Since the cosine is a unimodal function on $[-\pi, \pi]$ with its peak at $\phi = 0$, we have:

$$\left|I_n^{(1)}\right| = \mathcal{O}\left(\exp(n\cos(\phi_0) - 1)\right) = \mathcal{O}\left(\exp\left(-Cn^{1/5}\right)\right) \qquad (11)$$

for some constant $C > 0$. The tail integral is thus exponentially small.

2) *Central approximation:* Around $\phi = 0$ we have the following expansion of $h(\phi)$:

$$h(\phi) = -\frac{1}{2}\phi^2 + \mathcal{O}(\phi^3).$$

For $\phi \in [-\phi_0, \phi_0]$, the integrand can thus be approximated as follows:

$$\exp(nh(\phi)) = \exp\left(-\frac{n}{2}\phi^2 + \mathcal{O}(n\phi^3)\right) = \exp\left(-\frac{n}{2}\phi^2\right)\left(1 + \mathcal{O}(n\phi^3)\right)$$

and the central integral becomes

$$\begin{aligned} I_n^{(0)} &= \int_{-n^{-2/5}}^{n^{-2/5}} \exp\left(-\frac{n}{2}\phi^2\right) d\phi \cdot \left(1 + \mathcal{O}(n^{-1/5})\right) \\ &= \frac{1}{\sqrt{n}} \int_{-n^{1/10}}^{n^{1/10}} \exp\left(-\frac{t^2}{2}\right) dt \cdot \left(1 + \mathcal{O}(n^{-1/5})\right), \end{aligned} \qquad (12)$$

where the latter expression is obtained by the change of variables $t = \phi\sqrt{n}$.

3) *Tails completion:* What remains to be done is to show that the tails can be completed back in the integral (12). For tails of Gaussian integrals we have for every $c > 0$ that

$$\int_c^{+\infty} \exp\left(-\frac{t^2}{2}\right) dt = \mathcal{O}\left(\exp\left(-\frac{c^2}{2}\right)\right).$$

It follows that, for $n \to \infty$ we have the following asymptotic equivalent:

$$I_n^{(0)} \sim \frac{1}{\sqrt{n}} \int_{-\infty}^{\infty} \exp\left(-\frac{t^2}{2}\right) dt = \sqrt{\frac{2\pi}{n}}.$$

Finally, putting (11) and (12) together, we obtain

$$I_n^{(0)} + I_n^{(1)} \sim \sqrt{\frac{2\pi}{n}} \text{ and thus } I_n \sim \frac{e^n}{n^n\sqrt{2\pi n}}.$$

Of course, using a refined Taylor expansion for $h(\phi)$, one could obtain further coefficients in the asymptotic expansion of $n!$ using the same method. ⊣

2.7 PROBABILISTIC TOOLS

In this section we collect some probabilistic concepts and techniques we will use throughout this thesis. For a detailed introduction to these topics as well as for proofs we refer to the monographs by Billingsley [21] and Klenke [109].

Basics

Let $\mathcal{A}$ be a labelled combinatorial class, $\mathcal{A}_n$ the subset of $\mathcal{A}$ consisting of objects of size n and A_n the cardinality of $\mathcal{A}_n$. If we speak about a *random* object in $\mathcal{A}_n$, we will always mean the *uniform probability distribution* over $\mathcal{A}_n$. That is, every object in $\mathcal{A}_n$ has the same probability, namely $1/A_n$. Whenever we consider other probability distributions this will be stated explicitly.

Often we are interested in some parameter χ of A, see the remark on bivariate generating functions in Section 2.6. For every $n \in \mathbb{N}$, it defines a discrete random variable X_n:

$$\mathbb{P}(X_n = k) = \frac{A_{n,k}}{A_n},$$

where $A_{n,k}$ denotes the number of objects in $\mathcal{A}_n$ for which $\chi = k$.

Let us recall the following two elementary definitions: The *expectation* or *mean* of the discrete random variable X is defined as

$$\mathbb{E}(X) = \mu = \sum_k \mathbb{P}(X = k) \cdot k$$

and is a linear functional. Note that From the second moment $\mathbb{E}(X^2)$, one obtains the *variance*

$$\mathbb{V}(X) = \mathbb{E}\left((X - \mu)^2\right) = \mathbb{E}(X^2) - \mu^2$$

and the *standard deviation* $\sigma = \sqrt{\mathbb{V}(X)}$.

Let X be a discrete random variable supported by $\mathbb{N}$ with a given distribution. If one needs to calculate the expected value of a function $g(X)$ of X when one does not explicitly know the distribution of $g(X)$, the *Law of the Unconscious Statistician* or *Change of Variables Theorem* comes in handy. It states that, if the expected value of $g(X)$ exists, then the following holds:

$$\mathbb{E}(g(X)) = \sum_{n \in \mathbb{N}} g(n)\mathbb{P}(X = n).$$

One of the most fundamental discrete probability distributions is the *Bernoulli distribution*. It is the distribution of a random variable X that takes the value 1 with probability p, where $0 \leq p \leq 1$, and the value 0 with probability $(1 - p)$; one writes $X \sim B(p)$. Clearly, $\mathbb{E}(X) = p$ and $\mathbb{V}(X) = p(1 - p)$.

The *normal distribution* or *Gaussian distribution* is a continuous probability distribution that occurs very commonly in various contexts. It will appear as limiting distribution of discrete random variables in this thesis. A random variable X is normally distributed with mean μ and standard deviation σ, noted as $X \sim \mathcal{N}(\mu, \sigma)$, iff it has the probability density function

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

The distribution function of the *standard normal distribution* $\mathcal{N}(0,1)$ is denoted by $\Phi(x)$.

Probability and moment generating functions

In the following we introduce two generating functions that are closely related to the distribution of X_n.

1. The *probability generating function* (PGF) of X_n is the function $p_{X_n}(v) := p_n(v) := \mathbb{E}(v^{X_n})$. It is the ordinary generating function of the sequence of probabilities:

$$p_n(v) = \sum_{k\geq 0} \mathbb{P}(X_n = k)v^k.$$

   If $A(z,v)$ is the bivariate generating function of the parameter χ over $\mathcal{A}$, i.e.,

$$A(z,v) = \sum_{n\geq 0}\sum_{k\geq 0} A_{n,k}\frac{z^n}{n!}v^k,$$

   then there is a close relation between $p_n(v)$ and the coefficients of $A(z,v)$. Indeed, it holds that

$$p_n(v) = \frac{n!}{A_n}[z^n]A(z,v) = \frac{[z^n]A(z,v)}{[z^n]A(z,1)}.$$

   If $p_n(v)$ exists in a neighbourhood of $v = 1$ it can be helpful in order to compute the *factorial moments*

$$\mathbb{E}(X^{\underline{r}}) = \mathbb{E}(X(X-1)\cdots(X-r+1))$$

   of $X = X_n$. The factorial moments can be obtained by r-fold differentiation followed by evaluation at $v = 1$:

$$\mathbb{E}(X^{\underline{r}}) = \left.\frac{\partial^r p_n(v)}{\partial v^r}\right|_{v=1}.$$

   If the random variable X can be written as a sum of independent random variables X^i, i.e., $X = \sum_{i=1}^r a_i X_i$ where the a_i are constants, the probability generating function has a particularly simple form:

$$p_X(v) = \prod_{i=1}^r p_{X^i}(v^{a_i}).$$

2. The *moment generating function* of X_n is the function $g_{X_n}(s) := g_n(s)$ is defined by :

$$g_n(s) := p_n\left(e^s\right) = \mathbb{E}(e^{sX_n}) = \sum_{k\geq 0}\mathbb{P}(X_n = k)e^{sk}.$$

It is the exponential generating function of the sequence of moments, i.e.,

$$g_n(s) = \sum_{k\geq 0} \mathbb{E}(X^k)\frac{s^k}{k!}$$

and thus the moments $\mathbb{E}(X^r)$ can be obtained by r-fold differentiation followed by evaluation at $v = 1$:

$$\mathbb{E}(X^r) = \frac{\partial^r g_n(s)}{\partial s^r}\bigg|_{s=0}.$$

Convergence in distribution

Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of random variables, and $(F_n(x))_{n\in\mathbb{N}}$ the respective distribution functions. Furthermore, let X be another random variable with distribution function $F(x)$. The sequence $(X_n)_{n\in\mathbb{N}}$ is said to *converge in distribution* to X iff

$$\lim_{n\to\infty} F_n(x) = F(x),$$

for every $x \in \mathbb{R}$ where F is continuous. One then also says that $(X_n)_{n\in\mathbb{N}}$ *converges weakly* to X, and writes

$$X_n \xrightarrow{(d)} X.$$

If the random variable X can be written as a sum of independent and identically distributed (short: iid) random variables X^i, i.e., $X = \sum_{i=1}^n X_i$ whit $\mathbb{E}(X_i) = \mu < \infty$ and $\mathbb{V}(X_i) = \sigma^2 < \infty$, then the *Central limit theorem* (CLT) implies that the standardized random variable converges in distribution to the standard normal distribution:

$$\frac{X - n\mu}{\sigma\sqrt{n}} \xrightarrow{(d)} \mathcal{N}(0,1).$$

In case the random variable X is the sum of independent but not necessarily identically distributed random variables the following stronger version of the CLT can be used:

Theorem 2.35 (Ljapunov CLT)**.** *Let $X_1, X_2, \ldots, X_n$ be a sequence of independent random variables, with finite mean μ_i and variance σ_i^2 each. Moreover, let $s_n^2 = \sum_{i=1}^n \sigma_i^2$. If there exists a $\delta > 0$ such that the* Ljapunov condition

$$\frac{1}{s_n^{2+\delta}} \sum_{i=1}^n \mathbb{E}\left(|X_i - \mu_i|^{2+\delta}\right) \to 0 \text{ as } n \to \infty \tag{13}$$

is fulfilled, then $X = \sum_{i=1}^n X_i$ converges to a normal distribution:

$$\frac{X - \sum_{i=1}^n \mu_i}{s_n} \xrightarrow{(d)} \mathcal{N}(0,1).$$

Often we will encounter sequences of random variables $(X_n)_{n\in\mathbb{N}}$ that are not independent. In order to show that such a sequence has linear mean and variance and that it is asymptotically normal distributed when suitably standardized, a useful theorem is the following:

Theorem 2.36 (Hwang's quasi-power theorem [98]). *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of discrete random variables supported by $\mathbb{N}_0$. Assume that the moment generating functions $(g_n(s))_{n\in\mathbb{N}}$ are analytic in a fixed complex neighbourhood of $s = 0$ in which they have an expansion of the form*

$$g_n(s) = \exp\left(\phi_n U(s) + V(s)\right) \cdot \left(1 + \mathcal{O}\left(\frac{1}{\kappa_n}\right)\right),$$

where $\phi_n, \kappa_n \to \infty$ and $U(s)$, $V(s)$ do not depend of n and are analytic at $s = 0$. Assume finally that $U(s)$ satisfies the so-called "variability condition" $U''(0) \neq 0$. Under these conditions, the mean and variance of X_n satisfy

$$\begin{aligned}\mathbb{E}(X_n) =& \phi_n U'(0) + V'(0) + \mathcal{O}(\kappa_n^{-1}),\\ \mathbb{V}(X_n) =& \phi_n U''(0) + V''(0) + \mathcal{O}(\kappa_n^{-1}).\end{aligned}$$

Furthermore, X_n is after standardization asymptotically normally distributed, with speed of convergence $\mathcal{O}(\kappa_n^{-1} + \phi_n^{-1/2})$:

$$\mathbb{P}\left\{\frac{X_n - \mathbb{E}(X_n)}{\sqrt{\mathbb{V}(X_n)}} \leq x\right\} = \Phi(x) + \mathcal{O}\left(\frac{1}{\kappa_n} + \frac{1}{\sqrt{\phi_n}}\right),$$

where $\Phi(x)$ is the distribution function of the standard normal distribution.

2.8 METHOD OF CHARACTERISTICS

The *method of characteristics* is a technique for solving linear partial differential equations. We will use this method in order to solve some of the differential equations that occur in this thesis. We thus give a brief introduction to it; for details and more background we refer to [69].

In what follows we consider homogeneous quasilinear first order PDEs for a function $f(x, y)$ in two variables. These are of the following form

$$a(x, y, f) f_x(x, y) + b(x, y, f) f_y(x, y) = c(x, y, f), \tag{14}$$

with an initial condition $f(x, 0) = u(x)$. The idea behind the method of characteristics is to convert this PDE to a system of ordinary differential equations along so-called *characteristic curves*.

Let us first note that the following holds:

$$(a, b, c) \cdot (f_x, f_y, -1) = 0.$$

This can be understood geometrically as follows: A plot of $z = f(x, y)$ corresponds to a surface in (x, y, z)-space. The vector $(f_x, f_y, -1)$ is normal to this solution surface and (a, b, c) is tangent.

Starting with any point of the initial curve $f(x,0) = u(x)$, we follow the vector (a,b,c) in order to construct a curve on the solution surface. Such a curve is then called a characteristic curve. Once we have found all the characteristic curves, the solution $f(x,y)$ is fully characterized.

We parametrize our characteristic curves by an auxiliary parameter t. Then, given a characteristic curve $(x(t), y(t), f(t))$ its tangent vector is

$$\left(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}, \frac{\partial f}{\partial t}\right)$$

However, as described above, we chose our curves in such a way that (a,b,c) is tangent. We thus obtain the following *system of characteristic equations*,

$$\begin{aligned} x' &= \frac{dx}{dt} = a(x,y,f), \\ y' &= \frac{dy}{dt} = b(x,y,f), \\ f' &= \frac{df}{dt} = c(x,y,f). \end{aligned} \tag{15}$$

The next step is then to find first integrals of the system (15), i.e., functions $\zeta(x,y,F)$ that are constant along any characteristic curve. This is usually done by considering the equations

$$\frac{\partial x}{\partial y} = \frac{a(x,y,f)}{b(x,y,f)}, \quad \frac{\partial x}{\partial f} = \frac{a(x,y,f)}{c(x,y,f)} \quad \text{or} \quad \frac{\partial y}{\partial f} = \frac{b(x,y,f)}{b(x,y,f)}.$$

If one has found two independent first integrals $\zeta_1(x,y,F)$ and $\zeta_2(x,y,F)$ of (15), the general solution is given by:

$$G(\zeta_1(x,y,f), \zeta_2(x,y,f)) = \text{const.},$$

where G is an arbitrary differentiable function in two variables. Using the initial condition $f(x,0) = u(x)$, this function G can then be specified.

2.9 LOG-CONCAVITY AND COMBINATORIAL SEQUENCES

Log-concavity

Log-concave sequences play an important role in Combinatorics; see the extensive surveys by Stanley [147] and Brenti [40] as well Chapter 7 of [29] for a recent survey by Petter Brändén on the subject.

Definition 2.37. *Let $a_1, a_2, \cdots, a_n$ be a sequence of positive real numbers. We say that the sequence is* log-concave *if for all indices k, the inequality $a_{k-1}a_{k+1} \leq a_k^2$ holds.*

Note that a sequence $(a_k)_{1\leq k\leq n}$ is log-concave if and only if the sequence $(b_k)_{1\leq k\leq n} := (\log(a_k))_{1\leq k\leq n}$ is concave.

Log-concavity implies another important property of combinatorial sequences:

Proposition 2.38. *If the sequence $a_1, a_2, \cdots, a_n$ is log-concave then it is unimodal, i.e., there exists an index $0 \leq k \leq n$ such that $a_1 \leq a_2 \leq \ldots \leq a_k$ and $a_n \leq a_{n-1} \leq \ldots \leq a_k$.*

The following property is even stronger than log-concavity:

Definition 2.39. *We say that the sequence $a_1, a_2, \cdots, a_n$ has* real roots only *or* real zeros only *if the corresponding polynomial $A(x) = \sum_{i=1}^{n} a_i x^i$ has real roots only.*

The following holds:

Proposition 2.40. *If the sequence $a_1, a_2, \cdots, a_n$ has real roots only then it is log-concave as well.*

Note that the converse of this statement does not hold. For instance the sequence $1, 1, 1$ is log-concave but the polynomial $1 + x + x^2$ has two complex roots.

Example 2.41. A combinatorial sequence for which it is straightforward to prove real-rootedness for every fixed n is the sequence of binomial coefficients:

$$b_{n,k} = \binom{n}{k} \text{ with } 0 \leq k \leq n.$$

It follows, that this sequence is log-concave and unimodal. Indeed, we know that the binomial coefficients $\binom{n}{k}$ have a unique peak at $k = n/2$ in case n is even and have two maximal elements at $k = (n-1)/2$ and $k = (n+1)/2$ in case n is odd.

Another prominent example of a combinatorial sequence that has real roots only is the sequence of *Eulerian numbers* $A(n, k)$. The Eulerian numbers count permutations of length n with exactly k ascents. Equivalently, $A(n, k)$ counts n-permutations that are the union of $(k+1)$ ascending runs. ⊣

Various combinatorial sequences appearing in this thesis

In the following Table 1 we give an overview of the most important combinatorial sequences appearing in this thesis and gather some of their properties.

Table 1 only contains univariate combinatorial sequences. A famous bivariate sequence that appears in this thesis in Chapter 8 is the sequence of *Stirling numbers of the second kind*. The sequence $S_{n,k}$ counts partitions of the set $[n]$ into m nonempty subsets and we will see that it is closely related to the number of n-mappings with m runs (see Chapter 8). The following holds for the Stirling numbers of the second kind:

$$\left\{ {n \atop m} \right\} := S_{n,k} = \frac{1}{m!} \sum_{\ell=0}^{m} \binom{m}{\ell} (-1)^{m-\ell} \ell^n$$

This sequence is the entry A008277 in Sloane's OEIS [128].

For a detailed introduction to these combinatorial sequences we refer the reader to the monographs by Bóna [27] and Stanley [146]. For the viewpoint of analytical combinatorics, we refer to Flajolet and Sedgewick's monograph [78].

Formula	Combinatorial objects counted by these numbers and appearing in this thesis	Exponential (EGF) or Ordinary (OGF) GF	Asymptotics	Entry in OEIS [128]
$p_n = n!$	n-permutations (Part i)	EGF: $P(z) = \frac{1}{1-z}$	$p_n \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$	A000142
$Cat_n = \frac{1}{n+1}\binom{2n}{n}$	sorted n-parking functions (Chapter 9), 321-avoiding n-permutations (Chapter 5)	OGF: $C(z) = \frac{1-\sqrt{1-4z}}{2z}$	$Cat_n \sim \frac{4^n}{n^{3/2}\sqrt{\pi}}$	A000108
$b_n = \binom{2n}{n}$	$(n+1)$-permutations avoiding 2431, 4231, 1432, 4132 (Chapter 5)	OGF: $B(z) = \frac{1}{\sqrt{1-4z}}$	$b_n \sim \frac{4^n}{\sqrt{n\pi}}$	A000984
$i_n = \sum_{k=0}^{\lfloor n/2 \rfloor} \frac{n!}{(n-2k)!2^k k!}$	involutions of length n (Chapter 6), SYT with n boxes (Chapter 6)	EGF: $I(z) = \exp\left(z + \frac{z^2}{2}\right)$	$i_n \sim \frac{n^{n/2}e^{\sqrt{n}-n/2-1/4}}{\sqrt{2}}$	A000085
$t_n = n^{n-1}$	Cayley trees with n nodes (Part ii)	EGF: $T(z) = ze^{T(z)}$	$t_n = n^{n-1}$	A000169
$m_n = n^n$	n-mappings (Part ii)	EGF: $M(z) = \frac{1}{1-T(z)}$	$m_n = n^n$	A000312
$f_n = (n+1)^{n-1}$	ordinary n-parking functions (Chapter 9)	EGF: $F(z) = \frac{T(z)}{z}$	$f_n = (n+1)^{n-1}$	A000272

Table 1: An overview of famous combinatorial sequences appearing in this thesis.

Part I

PERMUTATIONS

This part is concerned with various aspects of patterns in permutations. Chapters 3 and 4 focus on the computational problem of Permutation Pattern Matching, the first one with the design of an fpt algorithm for this problem and the second one dealing with the computational complexity of this problem for different types of patterns. Chapter 5 deals with a permutation class that arises within the study of domain restrictions and that has a nice enumeration formula. Chapter 6 deals with the conjectured log-concavity of the combinatorial sequence enumerating permutations by the length of their longest increasing subsequence. Especially, it provides two examples of permutation classes for which the conjecture holds.

3

EFFICIENT PERMUTATION PATTERN MATCHING: THE ALTERNATING RUN ALGORITHM

This chapter is based on joint work with Martin Lackner which has lead to the article *A Fast Algorithm for Permutation Pattern Matching Based on Alternating Runs* which has recently been accepted for publication in *Algorithmica*. A preliminary version of this paper appeared in the proceedings of the 13th Scandinavian Symposium and Workshops on Algorithm , SWAT 2012 [41].

The main goal of this chapter is the design of an efficient algorithm for the following computational problem:

Permutation Pattern Matching (PPM)	
Instance:	A permutation τ (the text) of length n and a permutation π (the pattern) of length $k \leq n$.
Question:	Is there a matching of π into τ?

Bose, Buss and Lubiw [31] showed that PPM is in general NP-complete. The trivial brute-force algorithm checking every subsequence of length k of τ has a runtime of $\mathcal{O}(2^n \cdot n)$. So far, no algorithm has been discovered that improves the exponential runtime to c^n for some constant $c < 2$. Improving exponential time algorithms is a major topic in algorithmics, as witnessed by the monograph of Fomin and Kratsch [81].

In this chapter we tackle the problem of solving PPM faster than $\mathcal{O}(2^n \cdot n)$ for arbitrary π and τ. We achieve this by exploiting the decomposition of permutations into alternating runs. Note that whenever we speak of runs in this Section, we will always mean alternating runs and not ascending runs (see Section 2.2 for a distinction of these two concepts). The number of alternating runs in a permutation π is denoted by $\mathsf{run}(\pi)$ throughout this chapter. Alternating runs are a fundamental permutation statistic and were studied already in the late 19th century by André [7]. Despite the importance of alternating runs within the study of permutations, the connection to PPM has so far not been explored.

This chapter is organized as follows: The main section, Section 3.1, describes the alternating run algorithm and is divided into the following subsections. Section 3.1 introduces matching functions. Section 3.1 describes the alternating run algorithm in detail. Section GetMatching contains proof details necessary to verify the correctness of the alternating run algorithm. Section GetMatching proves the corresponding runtime bounds. Our results concerning the parameter $\mathsf{run}(\pi)$ can be found in Section 3.2. The contributions of this chapter to the PPM problem are summarized in Section 3.3.

Throughout this chapter the pattern permutation $\pi_{ex} = 2\,3\,1\,4$ and the text permutation $\tau_{ex} = 1\,8\,12\,4\,7\,11\,6\,3\,2\,9\,5\,10$, as introduced in the Preliminaries, will serve as a running example. A graphical representation can be found in Figure 13 on page 51.

RELATED WORK. The most relevant paper is the recent breakthrough result by Guillemot and Marx [93] showing that PPM is FPT with respect to the length of the pattern. Their algorithm has a runtime of $2^{\mathcal{O}(k^2 \cdot \log k)} \cdot n$. This FPT result is anteceeded by algorithms with a runtime of $\mathcal{O}(n^{1+2k/3} \cdot \log n)$ [3] and $\mathcal{O}(n^{0.47k+o(k)})$ [1].

Although PPM is NP-complete in general, there are polynomial time algorithms if only certain permutations are allowed as patterns. The most important example are *separable* permutations: these are permutations that contain neither 3142 nor 2413. If the pattern is separable, PPM can be solved in polynomial time [3, 31, 99, 138]. In case P is the identity $1\,2\,\ldots\,k$, PPM consists of looking for an increasing subsequence of length k in the text T – this is a special case of the LONGEST INCREASING SUBSEQUENCE problem. This problem can be solved in $\mathcal{O}(n \log n)$-time for sequences in general [139] and in $\mathcal{O}(n \log \log n)$-time for permutations [48, 122]. An $\mathcal{O}(k^2 n^6)$-time algorithm is presented in [94] for the case that both the text and the pattern that do not contain 321. If the pattern is required to be matched to consecutive elements in the text, a $\mathcal{O}(n + k)$ algorithm has been found [114]. A similar result has been found independently in [107]. This work has been extended to the cases where some mismatches are tolerated [84]; also suffix trees have recently been generalized to be applicable in this setting [55].

The related LONGEST COMMON PATTERN problem is to find a longest common pattern between two permutations T_1 and T_2, i.e., a pattern P of maximal length that can be matched both into T_1 and T_2. This problem is a generalization of PPM since determining whether T_1 is the longest common pattern between T_1 and T_2 is equivalent to checking whether T_2 contains T_1 as a pattern. In [34] a polynomial time algorithm for the LONGEST COMMON PATTERN problem is presented for the case that one of the two permutations T_1 and T_2 is separable. A generalization of this problem, the so called LONGEST COMMON $\mathcal{C}$-PATTERN problem was introduced in [35]. This problem consists of finding the longest common pattern among several permutations belonging to a class $\mathcal{C}$ of permutations. For the case that $\mathcal{C}$ is the class of all separable permutations and that the number of input permutations is fixed, the problem was shown to be polynomial time solvable [35].

For a class of permutations X, the LONGEST X-SUBSEQUENCE (LXS) problem is to identify in a given permutation T its longest subsequence that is isomorphic to a permutation of X. Polynomial time algorithms for many classes X exist, but in general LXS is NP-hard [4].

NOTATION. To avoid confusion, we introduce the following notation for this chapter: Capital letters are used for functions, sets and lists; lower case letters for natural numbers. The letters i and j are exclusively used to denote positions or indices such as positions in permutations or indices in lists. We use greek letters only in the following cases: π (the pattern) and τ (the text) are permutations; $\kappa \in [k]$ as well as $\nu \in [n]$ are the main variables in the algorithm. In this chapter, tuples always have length $\mathsf{run}(\pi)$ and are denoted with bars, e.g., $\vec{x}$. The elements of $\vec{x}$ are $x_1, \ldots, x_{\mathsf{run}(\pi)}$.

3.1 THE ALTERNATING RUN ALGORITHM

We start with an outline of the alternating run algorithm. Its description consists of two parts. In Part 1 we introduce so-called *matching functions*. These functions map runs in π to sequences of adjacent runs in τ. The intention behind matching functions is to restrict the search space to certain subsequences of length k, namely to those where all elements in a run in π are mapped to elements in the corresponding sequences of runs in τ. In Part 2 a dynamic programming algorithm is described. It checks for every matching function whether it is possible to find a compatible matching. This is done by finding a small set of representative elements to which the element 1 can be mapped to, then – for a given choice for 1 – finding representative elements for 2, and so on.

Theorem 3.1. *The alternating run algorithm solves the* PPM *problem in time* $\mathcal{O}(1.79^{\mathsf{run}(\tau)} \cdot n \cdot k)$. *Therefore,* PPM *parameterized by* $\mathsf{run}(\tau)$ *is in* FPT.

Since $\mathsf{run}(\tau) < n$, we obtain as an immediate consequence:

Corollary 3.2. *The alternating run algorithm solves the* PPM *problem in time* $\mathcal{O}(1.79^{n} \cdot n \cdot k)$.

Before we start with the description of the alternating run algorithm, we introduce two functions which play an important role.

Definition 3.3. *Let* $u \in [k]$. *The* run predecessor $\mathsf{pre}(u)$ *denotes the largest element smaller than* u *that is contained in the same run as* u *in* π *(if such an element exists). Moreover, the* run index *function* ri *is defined as follows:* $\mathsf{ri}(u) = i$ *if* u *is contained in the* i*-th run in* π.

Note that both functions concern only the pattern π.

Matching functions.

We introduce the concept of matching functions. These are functions from the interval $[\mathsf{run}(\pi)]$ to sequences of adjacent runs in τ. For a

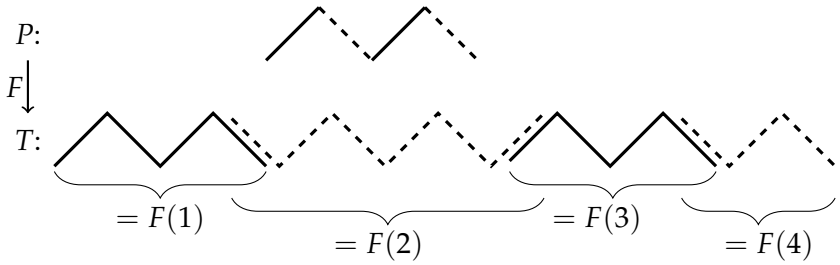


Figure 12: A sketch of a matching function and its M- and W-shaped subsequences

given matching function F the search space in τ is restricted to matchings where an element κ contained in the i-th run in π is matched to an element in $F(i)$. As we will see later on in Lemma 3.7, this restriction of the search space does not influence whether a matching can be found or not: if a matching exists, a corresponding matching function can be found. In addition, Lemma 3.27 will show that it is possible to iterate over all matching functions in fpt time. Thus, our algorithm verifies for all matching functions whether a compatible matching exists.

Let us now give a formal definition of matching functions.

Definition 3.4. *A* matching function F *maps an element of* $[\mathsf{run}(\pi)]$ *to a subsequence of* τ*. It has to satisfy the following properties for all* $i \in [\mathsf{run}(\pi)]$*.*

(P1) $F(i)$ *is a contiguous subsequence of* τ*.*

(P2) If the i*-th run in* π *is a run up (down),* $F(i)$ *starts with an element following a valley (peak) or the first element in* τ *and ends with a valley (peak) or the last element in* τ*.*

(P3) $F(1)$ *starts with the first and* $F(\mathsf{run}(\pi))$ *ends with the last element in* τ*.*

(P4) $F(i)$ *and* $F(i+1)$ *have one run in common:* $F(i+1)$ *starts with the leftmost element in the last run in* $F(i)$*.*

Property (P2) implies that every run up is matched into an M-shaped sequence of runs of the form up–down–up–...–up–down (if the run up is the first or the last run in π the sequence might start or end differently) and every run down is matched into a W-shaped sequence of runs of the form down–up–down–...–down–up (again, if the run down is the first or the last run in π, the sequence might start or end differently). These M- and W-shaped sequences are sketched in Figure 12.

Property (P4) implies that two adjacent runs in π are mapped to sequences of runs that overlap with exactly one run, as is also sketched

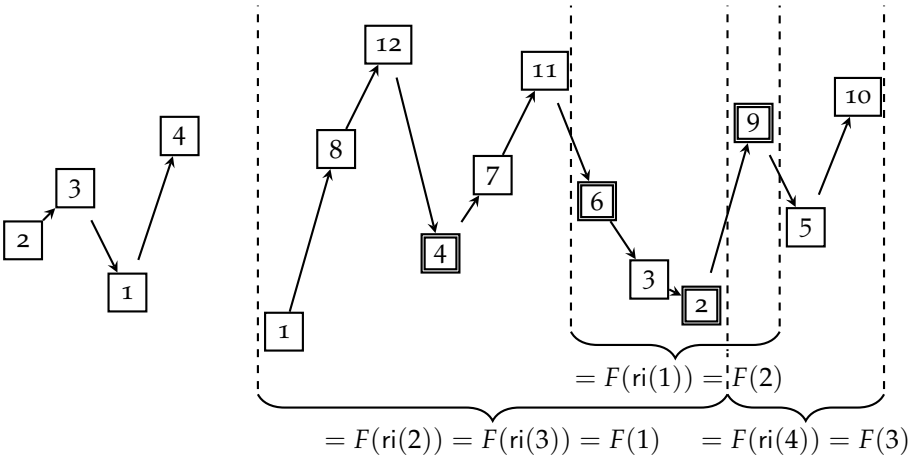


Figure 13: π_{ex} and τ_{ex} together with a matching function F and the compatible matching witnessed by the subsequence 4 6 2 9

in Figure 12. This overlap is necessary since elements in different runs in π may be matched to elements in the same run in τ. More precisely, valleys and peaks in π might be matched to the same run in τ as their successors (see the following example).

Example 3.5. In Figure 13, π_{ex} (left-hand side) and τ_{ex} (right-hand side) are depicted together with a matching function F. A matching compatible with F is given by 4 6 2 9. We can see that the elements 6 and 2 lie in the same run in τ_{ex} even though 3 (a peak) and 1 (its successor) lie in different runs in π_{ex}. ⊣

Note that there are no matching functions if $\mathsf{run}(\pi) > \mathsf{run}(\tau)$. This corresponds to the fact that in such a case no matching from π into τ exists either. The properties (P1)-(P4) guarantee that the number of functions we have to consider is less than $(\sqrt{2})^{\mathsf{run}(\tau)}$, as will be proven in Section GetMatching, Lemma 3.27. This allows us to iterate over all matching functions in fpt time.

Let us formalize what we mean by compatible matchings.

Definition 3.6. *A matching M is* compatible *with a matching function F if $M(\kappa) \in F(\mathsf{ri}(\kappa))$ for every $\kappa \in [k]$, i.e., M matches each element contained in the i-th run in π to an element in $F(i)$.*

Lemma 3.7. *For every matching M of π into τ there exists a matching function F such that M is compatible with F.*

The proof of this lemma can be found in Section GetMatching on page 61. We continue with the observation that, when searching for a compatible matching by looking for the possible values that $M(1), M(2)$ and so on can take, we do not have to remember *all* the previous choices we made. Let us have a look at an example first:

Example 3.8. In Figure 13, assume that we already have a partial matching: $M(1) = 2$ and $M(2) = 4$. We now have to decide where to map 3. There are two constraints that have to be satisfied: First, $M(3) > M(2)$. Second, $M(3)$ has to be to the right of $M(2)$, since $2 \prec_\pi 3$. Since our choices for $M(3)$ are limited to $F(\mathsf{ri}(3)) = F(1)$, we do not have to check whether $M(3)$ is left of $M(1)$ but only whether $M(3) > M(2)$. Later, when deciding where to map 4, we will only have to verify that $M(4) > M(3)$.

In more generality, we observe that given a matching function and a partial matching M defined on $[\kappa - 1]$, deciding where to map κ only requires the knowledge of $M(\kappa - 1)$ and of $M(\kappa')$, where κ' is the previous element in the same run as κ. ⊣

Let us now make this observation more precise:

Lemma 3.9. *Let F be a matching function. A function $M{:}[k] \to [n]$ is a matching of π into τ compatible with F if and only if for every $\kappa \in [k]$:*

1. $M(\kappa) \in F(\mathsf{ri}(\kappa))$,
2. $M(\kappa) > M(\kappa - 1)$ *and*
3. *if* $\mathsf{pre}(\kappa)$ *exists, then* $\mathsf{pre}(\kappa) \prec_\pi \kappa$ *if and only if* $M(\mathsf{pre}(\kappa)) \prec_\tau M(\kappa)$, *i.e., if κ is contained in a run up (down), then $M(\kappa)$ is right (left) of* $M(\mathsf{pre}(\kappa))$.

As we will see soon, this lemma is essential for our algorithm. Its proof can be found in Section GetMatching on page 62.

Algorithm description

Before we start explaining the actual fpt algorithm, let us consider a simple algorithm based on alternating runs. This simple algorithm (Algorithm 1) does not have fpt runtime but has the same basic structure as the fpt algorithm. In particular, this simple algorithm will already demonstrate the importance of Lemma 3.9.

From Lemma 3.7 we know that when checking whether τ contains π as a pattern, it is sufficient to test for all matching functions whether there exists a *compatible* matching. Let us fix a matching function F. We first find suitable elements to which 1 can be mapped, then suitable elements for 2, and so on. Observe that we can use Lemma 3.9 to verify what suitable elements are. In addition, Lemma 3.9 tells us that when finding suitable elements for $\kappa \in [k]$, we only require the values of $M(\kappa - 1)$ and $M(\mathsf{pre}(\kappa))$. This means in particular that we do not have to store all values of a possible partial matching $(M(1), \ldots, M(\kappa))$ but only the values of M for the largest element $\leq \kappa$ in each run in π. For example, when trying to match $\pi = 2357416$ into some text and looking for the possible elements for $\kappa = 4$, we only have to consider possibilities for $M(3)$ and $M(\mathsf{pre}(4)) = M(1)$.

Algorithm 1: A Simple Alternating Run Algorithm

```
1  X_0^F ← {(0,...,0)}          // The tuple (0,...,0) has run(π) elements.
2  foreach matching function F do
3      for κ ← 1...k do          // κ is the element to be matched.
4          X_κ^F ← ∅
5          foreach x⃗ ∈ X_{κ-1}^F do
6              R ← {ν ∈ [n] : ν ∈ F(ri(κ)) ∧ ν > x_{ri(κ-1)} ∧ (pre(κ) ≺_π κ ↔ x_{ri(pre(κ))} ≺_τ ν)}
                   // Conditions according to Lemma 3.9
7              foreach ν ∈ R do
8                  X_κ^F ← X_κ^F ∪ {(x_1,...,x_{ri(κ)-1}, ν, x_{ri(κ)+1},...,x_{run(π)})}
9      if X_k^F ≠ ∅ then
10         return "π can be matched into τ."
11 return "π cannot be matched into τ."
```

In this simple algorithm, we want to keep track of all possible partial matchings $(M(1), \dots, M(\kappa))$ for every $\kappa \in [k]$. Since such partial matchings can be described by storing a single value per run in π, every one of them can be stored as a tuple $\vec{x}$ of length $\mathsf{run}(\pi)$. The first element of $\vec{x}$ contains a possible choice for the largest element $\leq \kappa$ in the first run of π, the second element of $\vec{x}$ contains a possible choice for the largest element $\leq \kappa$ in the second run of π, etc. We formalize this notion of "tuples encoding partial matchings" as (κ, F)-matchings:

Definition 3.10. *Let κ be an integer in $[k]$. A tuple $\vec{x} = (x_1, x_2, \dots, x_{\mathsf{run}(\pi)})$ with $x_i \in [0, n]$ for all $i \in [\mathsf{run}(\pi)]$ is called a (κ, F)-matching of π into τ if the following holds: There exists a function $M : [\kappa] \to [n]$ that is a matching of $\pi|_{[\kappa]}$ into τ that is compatible with F and for which it additionally holds that for every $x_i \neq 0$, $M(\max\{\kappa' \leq \kappa : \mathsf{ri}(\kappa') = i\}) = x_i$, i.e., M maps the largest element $\leq \kappa$ in the i-th run of π to the i-th element of $\vec{x}$.*

The following lemma states that X_κ^F – as constructed by Algorithm 1 – indeed contains only tuples that are (κ, F)-matchings:

Lemma 3.11. *Let X_κ^F be the set of tuples as constructed by Algorithm 1. Then every $\vec{x} \in X_\kappa^F$ is a (κ, F)-matching.*

The proof can be found in Section GetMatching on page 63. As an immediate consequence of this lemma, we know that if $X_k^F \neq \emptyset$ then there exists a matching from π into τ that is compatible with F. Observe that X_k^F is always empty if a previous X_κ^F was empty. If for every F the set $X_k^F = \emptyset$, we know from Lemma 3.7 that π cannot be matched into τ.

Example 3.12. For our running example (π_{ex}, τ_{ex}) and $\kappa = 1$ the data structure is given as follows: $X_1^F = \{(0,6,0),(0,3,0),(0,2,0),(0,9,0)\}$. Given the choice $M(1) = 3$, we obtain 6 $(2,F)$-matchings, namely: $(8,3,0),(12,3,0)$, $(4,3,0),(7,3,0),(11,3,0)$ and $(6,3,0)$. In total X_2^F contains 19 elements. ⊣

As seen in this small example, the set R and consequently the set X_κ^F can get very large. In particular, it is not possible to bound the size of X_κ^F by a function depending only on $\mathsf{run}(\tau)$ and not on n – which is necessary for obtaining our fpt result. Thus, we have to further refine our algorithm.

We proceed by explaining how this simple algorithm can be improved in order to obtain an fpt algorithm based on alternating runs (Algorithm 2). This is the main algorithm described in this chapter. In the following description we fix F to be the current matching function under consideration. There are two modifications that have to be made in order to obtain fpt runtime. First, we have to restrict the set R to fewer, *representative* choices. Second, we have to change the data structure of X_κ^F from a set to an array of fixed size. In the array X_κ^F, every (κ,F)-matching has a predetermined position. Observe that if there are two (κ,F)-matchings $\vec{x}$, $\vec{y}$ where $\vec{x}$ leads to a matching only if $\vec{y}$ leads to a matching as well, the algorithm only has to remember $\vec{y}$. The position of a (κ,F)-matching will thus be assigned in such a way that one of two (κ,F)-matching sharing the same position is preferable in the above sense. We will now explain both modifications in detail.

Algorithm 2: The Alternating Run Algorithm

```
1  X_0^F ← [(0,...,0)]                    // (0,...,0) has run(π) elements.
2  foreach matching function F do
3      for κ ← 1...k do                   // κ is the element to be matched.
4          X_κ^F ← [ε,...,ε]              // X_κ^F is a fixed-size array.
5          foreach x⃗ ∈ X_{κ-1}^F with x⃗ ≠ ε do
6              R ← Rep(x⃗, κ, F)
7              foreach ν ∈ R do
8                  i ← Index(x_1,...,x_{ri(κ)-1}, ν, x_{ri(κ)+1},...,x_{run(π)})
9                  y⃗ ← X_κ^F(i)
10                 if y⃗ = ε or y_{ri(κ)} > ν then
11                     X_κ^F(i) ← (x_1,...,x_{ri(κ)-1}, ν, x_{ri(κ)+1},...,x_{run(π)})
12     if X_k^F ≠ [ε,...,ε] then          // Is X_k^F non-empty?
13         return "Matching found: GetMatching (X_1^F,...,X_k^F)"
14 return "π cannot be matched into τ."
```

Concerning the first modification, restricting the set R, we introduce the procedure $\mathsf{Rep}(\vec{x},\kappa,F)$. This procedure returns a set of representative elements to which κ can be mapped to. These choices have

to be compatible with previously chosen elements $(x_1, x_2, \ldots, x_{\mathsf{run}(\pi)})$ and the matching function F.

Procedure $\mathrm{Rep}(\vec{x}, \kappa, F)$

input : a (κ, F)-matching $\vec{x} = (x_1, x_2, \ldots, x_{\mathsf{run}(\pi)})$, $\kappa \in [k]$, a matching function F
output: R, the set of representative elements for $M(\kappa)$

1 $R \leftarrow F(\mathsf{ri}(\kappa))$
2 $R \leftarrow R \cap [x_{\mathsf{ri}(\kappa-1)} + 1, n]$
3 $R \leftarrow \mathtt{Valleys}(\tau|_R)$
4 **if** κ *is in a run up in* π **then**
5 **if** $x_{\mathsf{ri}(\kappa)\neq 0}$ **then**
6 $R \leftarrow \left\{\nu \in R : x_{\mathsf{ri}(\kappa)} \prec_\tau \nu\right\}$
7 **if** κ *is the largest element in its run* **then**
8 $R \leftarrow \{\min R\}$
9 **else**
10 $R \leftarrow \{\nu \in R : \exists \nu'$ with $\nu' \in F(\mathsf{ri}(\kappa)) \wedge \nu' > \nu \wedge \nu \prec_\tau \nu'\}$
11 **else**
12 **if** $x_{\mathsf{ri}(\kappa)\neq 0}$ **then**
13 $R \leftarrow \left\{\nu \in R : \nu \prec_\tau x_{\mathsf{ri}(\kappa)}\right\}$
14 **if** κ *is the largest element in its run* **then**
15 $R \leftarrow \{\min R\}$
16 **else**
17 $R \leftarrow \{\nu \in R : \exists \nu'$ with $\nu' \in F(\mathsf{ri}(\kappa)) \wedge \nu' > \nu \wedge \nu' \prec_\tau \nu\}$
18 **return** R

An element $\nu \in [n]$ is contained in $\mathtt{Rep}(\vec{x}, \kappa, F)$ if the following conditions are met:

(C1) [Line 1] It has to hold that $\nu \in F(\mathsf{ri}(\kappa))$ (cf. Condition 1 in Lemma 3.9).

(C2) [Line 2] It has to hold that $\nu > x_{\mathsf{ri}(\kappa-1)}$ (cf. Condition 2 in Lemma 3.9).

(C3) [Line 3] It is always preferable to choose elements that are as small as possible. To be more precise: If we consider the subsequence of τ containing all elements in the set R, we merely need to consider the valleys of this subsequence. The function $\mathtt{Valleys}(\tau|_R)$ returns exactly these valleys.

(C4) [Lines 6 and 13] It has to hold that if κ is contained in a run up (down), then ν has to be right (left) of $x_{\mathsf{ri}(\kappa)}$, i.e., the element to which the run predecessor of κ is mapped (cf. Condition 3 in Lemma 3.9).

(C5) [Lines 8 and 15] If κ is the largest element in its run, the optimal choice is the smallest possible element.

(C6) [Lines 10 and 17] If κ is not the largest element in its run, the choice of ν must not prevent finding elements for the next elements in its run. Thus, if κ is contained in a run up (down), then there has to be a larger element to its right (left) that is contained in $F(\mathsf{ri}(\kappa))$.

Since this smaller set R is a subset of the set R in the simple algorithm (Algorithm 1), we immediately obtain the following corollary of Lemma 3.11:

Corollary 3.13. *Let X_κ^F be the set of tuples as constructed by Algorithm 2. Then every $\vec{x} \in X_\kappa^F$ is a (κ, F)-matching.*

Example 3.14. Let us explain how the elements in $\mathtt{Rep}((4,2,0),3,F)$ are determined in our running example. The elements fulfilling Condition (C1) are: 1, 8, 12, 4, 7, 6, 3 and 2 (listed in the order they appear in τ). Among these, the elements larger than $x_{\mathsf{ri}(2)} = x_1 = 4$ are: 8, 12, 7, 11, 6 (cf. (C2)). If we consider this subsequence, its valleys are: 8, 7, and 6 (these are the elements fulfilling Condition (C3)). The element 3 is contained in a run up in τ, thus the element it is mapped to has to lie to the right of $x_{\mathsf{ri}(\mathsf{pre}(3))} = x_{\mathsf{ri}(2)} = 4$. The elements also fulfilling (C4) thus are 7 and 6. Since 3 is the largest element in its run in π, we only need to store the smallest possibility which is 6 (cf. (C5)). Condition (C6) does not apply here. If there were another, larger element in the same run as 3 in π, we would have to choose the element 7, since there are no larger elements in $F(\mathsf{ri}(3))$ to the right of 6. ⊣

If any matching of π into τ can be found that is compatible with F, it is also possible to find a matching that only involves representative elements. This statement is formalized and proven in Section Get-Matching (Definition 3.21 and Lemma 3.23). For the time being, let us convey the intuition behind this:

Example 3.15. In Figure 13, $\{2 \mapsto 4, 3 \mapsto 6, 1 \mapsto 3, 4 \mapsto 10\}$ is a matching of π_{ex} into τ_{ex} where the elements 3 and 10 are not representative: $3 \notin \mathtt{Rep}((0,0,0),1,F)$ and $10 \notin \mathtt{Rep}((6,3,0),4,F)$. This can be seen since 3 is not a valley in τ and 10 is not a valley in the subsequence consisting of elements larger than 6. However, this matching can be represented by the matching $\{2 \mapsto 4, 3 \mapsto 6, 1 \mapsto 2, 4 \mapsto 9\}$ that only involves representative elements (3 is represented by 2; 10 by 9) and that is compatible with the same matching function F. ⊣

This concludes our description of representative elements, our first modification of the simple alternating run algorithm. We proceed by explaining the data structure X_κ^F, which is changed from a set to an array of fixed size. In this array, every (κ, F)-matching $\vec{x}$ has a predetermined position which depends on the notion of *vales*.

Definition 3.16. *A subsequence of a permutation π consisting of a consecutive run down and run up (formed like a V) is called a* vale. *If π starts*

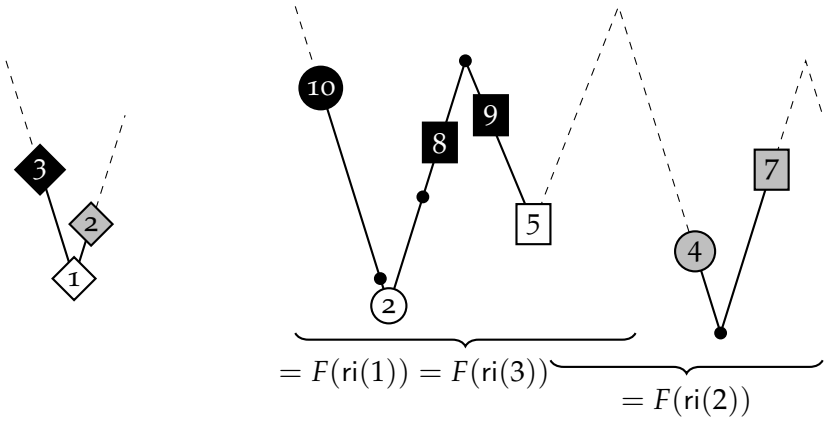


Figure 14: Schematic representation of the permutations occurring in Example 3.17: to the left the pattern π and to the right the text τ

with a run up, this run is also considered as a vale and analogously if π ends with a run down. Let $\mathsf{vale}(\pi)$ denote the number of vales in π. Finally, we define the vale index *function $\mathsf{vi}(u)$: given a matching function F and $u \in F(i)$, let $\mathsf{vi}(u) = j$ if u is contained in the j-th vale in $F(i)$. For notational convenience we set $\mathsf{vi}(0) = 1$.*

The main idea is the following: Two (κ, F)-matchings $\vec{x}$ and $\vec{y}$ in X_κ^F with $\mathsf{vi}(x_i) = \mathsf{vi}(y_i)$ for all $i \in [\mathsf{run}(\pi)]$ are comparable in the sense that one of these is less likely to lead to a matching. More precisely, the (κ, F)-matching containing the larger element at the $\mathsf{ri}(\kappa)$-th position (this is also the largest element of the entire tuple) leads to a matching only if the other one leads to a matching as well. Thus, the former (κ, F)-matching can be discarded and only the latter (κ, F)-matching has to be stored. The following example illustrates this notion of comparability:

Example 3.17. Consider the two permutations π and τ schematically represented in Figure 14. We are searching for representative elements for $\kappa = 3$ which lies in a run down in π. Which elements κ may be matched to depends on the choices for its run predecessor $\mathsf{pre}(3) = 1$ and for $\kappa - 1 = 2$. For the element 1, two representative elements are 2 (circle) and 5 (square), the valleys in $F(\mathsf{ri}(1))$ in τ. They lead to one representative element for 2 each: if 2 has been chosen then 4 is a representative element (circle) and if 5 has been chosen then 7 (square) is one. At this point, we have the following two $(2, F)$-matchings: $\vec{x} = (\ldots, 0, 2, 4, 0, \ldots)$ and $\vec{y} = (\ldots, 0, 5, 7, 0, \ldots)$. On the one hand, $\vec{x}$ seems to be preferable since it involves smaller elements than $\vec{y}$ and this leaves more possibilities for the following elements. On the other hand, $\vec{y}$ seems to be preferable since it involves 5 in $F(\mathsf{ri}(1))$, which is further to the right than 2. This is advantageous since $F(\mathsf{ri}(1))$ corresponds to a run down and this means that larger elements in the same run will have to be chosen to the left. All to-

gether we cannot say which of $\vec{x}$ and $\vec{y}$ is preferable and thus have to store both of them.

When we now turn to the element 3 in π, there are three representative elements: if we have chosen $\vec{x}$ the only possible choice is the element 10; if we have chosen $\vec{y}$ there are two possible choices namely 8 and 9. We thus obtain three $(3, F)$ matchings: $\vec{x}' = (\ldots, 0, 10, 4, 0, \ldots)$, $\vec{y}' = (\ldots, 0, 8, 7, 0, \ldots)$ and $\vec{y}'' = (\ldots, 0, 9, 7, 0, \ldots)$. We can now observe that we do not have to keep track of all three possibilities. Indeed, the two $(3, F)$-matchings $\vec{x}'$ and $\vec{y}'$ have coinciding vales and $\vec{x}'$ can be discarded in favour of $\vec{y}'$ since $\vec{x}'$ will only lead to a matching of π into τ if $\vec{y}'$ does. This is due to the fact that $x'_{\mathsf{ri}(3)} = 10 > 8 = y'_{\mathsf{ri}(3)}$ and can be seen as follows:

Let u be an element in the same run as 3 in π that is larger than 3 (which means that it lies to the left of 3). All elements to the left of and larger than 10 in $F(\mathsf{ri}(u))$ are clearly also to the left of and larger than 8. Thus, if there exists an element in $\mathtt{Rep}(\vec{x}', u, F)$ there also exists a smaller element in $\mathtt{Rep}(\vec{y}', u, F)$. This means that from the point of view of the run containing 3, $\vec{y}'$ is to be preferred over $\vec{x}'$. Now let $v > 3$ be an element in the same run in π as 2 (which means that it lies to the right of 2). Representative elements for v have to both lie to the right of the element chosen for 2 (4 or 7) and be larger than the element chosen for 3 (10 or 8). Since 4 and 7 lie in the same vale in τ there are no larger elements in between them. This implies that elements that are to the right of 4 in $F(\mathsf{ri}(2))$ and larger than 10 are automatically to the right of 7 and larger than 8. From the point of view of the run containing 2, $\vec{y}'$ it also to be preferred over $\vec{x}'$. The same argument also holds for any other element in π that is larger than 3.

To put this example in a nutshell: if we have two (κ, F)-matchings $\vec{x}$ and $\vec{y}$ with coinciding vales and $y_{\mathsf{ri}(\kappa)} \leq x_{\mathsf{ri}(\kappa)}$ we only need to store $\vec{y}$. For a formal proof of this statement, we refer to Lemma 3.25 on page 67 in Section GetMatching. ⊣

If we store only one (κ, F)-matching out of those with identical vales, the question arises how many vales there are in $F(i)$, $i \in [\mathsf{run}(\pi)]$. The answer is that at most $\lfloor \mathsf{run}(F(i))/2 \rfloor + 1$ exist: all vales but the two outermost consist of two runs and the two outermost may consist of only one run (cf. Definition 3.16). This would yield that we have to store at most $\prod_{i=1}^{\mathsf{run}(\pi)} \left(\lfloor \frac{\mathsf{run}(F(i)}{2} \rfloor + 1 \right)$ many (κ, F)-matchings. This number is still too large to show our desired runtime bounds. However, it suffices to distinguish between $\lfloor \mathsf{run}(F(i))/2 \rfloor$ many vales in $F(i)$ with $i \in [\mathsf{run}(\pi) - 1]$. This is achieved by not distinguishing between the first and the last vale in $F(i)$ for $i < \mathsf{run}(\pi)$. We only briefly mention that this is correct due to the Conditions (C5) and (C6); a formal proof will follow with Lemma 3.25 in Section GetMatching. For $i = \mathsf{run}(\pi)$, the last run in π, we still consider all vales occurring in $F(\mathsf{run}(\pi))$.

Recall that our goal is to assign a position in the array X_κ^F to every (κ, F)-matching $\vec{x}$. For every one of the $\mathsf{run}(\pi)$ values of the (κ, F)-matching there are at most $\lfloor \mathsf{run}(F(i)/2 \rfloor$ vales that need to be distinguished, except for the last one where we have to distinguish between $\lfloor \mathsf{run}(F(\mathsf{run}(\pi)))/2 \rfloor + 1$ vales. Thus, it is natural to use a mixed radix numeral system with bases $b_1 = \lfloor \mathsf{run}(F(1)/2 \rfloor$, $b_2 = \lfloor \mathsf{run}(F(2)/2 \rfloor, \ldots, b_{\mathsf{run}(\pi)-1} = \lfloor \mathsf{run}(F(\mathsf{run}(\pi) - 1)/2 \rfloor$ and $b_{\mathsf{run}(\pi)}$ is equal to the number of vales in $F(\mathsf{run}(\pi))$. Let `Index` be the function that assigns a position in the array to each (κ, F)-matching $\vec{x} = (x_1, \ldots, x_{\mathsf{run}(\pi)})$:

$$\texttt{Index}\left(x_1, \ldots, x_{\mathsf{run}(\pi)}\right) = 1 + \sum_{i=1}^{\mathsf{run}(\pi)} (\mathsf{vi}(x_i) - 1 \mod b_i) \cdot \prod_{j=1}^{i-1} b_j.$$

The mod operator is required since for $x \in F(i)$, $\mathsf{vi}(x) \in [b_i + 1]$ – as explained above.

Example 3.18. Let us discuss what the `Index` function looks like for our running example π_{ex} and τ_{ex} (cf. Figure 13). The subsequence $F(1)$ contains four runs. Thus, $b_1 = 2$. Since both $F(2)$ and $F(3)$ contain two runs, $b_2 = b_3 = 1$. Consequently, in our running example, X_κ^F contains at most two elements for every $\kappa \in [k]$. For example, $\texttt{Index}(8, 3, 10) = 1$, $\texttt{Index}(6, 3, 10) = 1$ and $\texttt{Index}(11, 3, 10) = 2$. ⊣

From the definition of the `Index`-function, it follows that the length of our array is $\prod_{i=1}^{\mathsf{run}(\pi)} b_i$. We will show in Lemma 3.28 that $\prod_{i=1}^{\mathsf{run}(\pi)} b_i = \mathcal{O}\left(1.2611^{\mathsf{run}(\tau)}\right)$. At this point, we see the huge advantage of this array data structure over the set data structure in the simple algorithm: the set X_κ^F has a potential size of $n^{\mathsf{run}\pi}$ – too large for an fpt algorithm.

This concludes the description of the array data structure. Let us now – once again – return to our running example and see how this would be dealt with by the alternating run algorithm.

Example 3.19. Let us demonstrate how the alternating run algorithm works. As before, consider τ_{ex}, π_{ex} and the matching function F as represented in Figure 13. We already know from the last example that X_κ^F has size 2, i.e., the `Index` function has range $\{1, 2\}$. We start with $X_0^F = \{(0, 0, 0)\}$. Refer to Table 2 for an overview. For the element 1 in π the only representative element is 2. Since $\texttt{Index}(0, 2, 0) = 1$, we store this $(1, F)$-matching at position 1 in X_1^F. Position 2 remains empty (symbolized by ϵ). For the element 2, we have more representative elements: $\texttt{Rep}((0, 2, 0), 2, F) = \{4, 8\}$. Note that 3 is not a representative element since there is no larger element to its right in $F(\mathsf{ri}(2)) = F(1)$ (cf. (C6)). Since $\texttt{Index}(8, 2, 0) = 1$ and $\texttt{Index}(4, 2, 0) = 2$, both $(2, F)$-matchings are stored in X_2^F. For placing the element 3, observe that 3 is the largest element in its run in π. Thus, Condition (C5) applies. We obtain $\texttt{Rep}((8, 2, 0), 3, F) = \min\{11, 12\} = \{11\}$ as well

	$\texttt{Index}(.,.,.) = 1$	$\texttt{Index}(.,.,.) = 2$
X_1^F	$(0,2,0)$	ϵ
X_2^F	$(8,2,0)$	$(4,2,0)$
X_3^F	$(6,2,0)$	$(11,2,0)$
X_4^F	$(6,2,9)$	ϵ

Table 2: The arrays $X_1^F, \ldots, X_4^F$ for our running example (cf. Figure 13)

as $\texttt{Rep}((4,2,0),3,F) = \min\{7,6\} = \{6\}$. Thus, we have two $(3,F)$-matchings to store in X_3^F: $(11,2,0)$ and $(6,2,0)$ with $\texttt{Index}(11,2,0) = 2$ and $\texttt{Index}(6,2,0) = 1$. Finally, we have to place the element 4. Since $\texttt{Rep}((11,2,0),4,F) = \emptyset$ the $(3,F)$-matching $(11,2,0)$ does not lead to a matching. However, $\texttt{Rep}((6,2,0),3,F) = \{9\}$. Thus, X_4^F contains the $(4,F)$-matching $(6,2,9)$. This $(4,F)$-matching corresponds to the matching $\{2 \mapsto 4, 3 \mapsto 6, 1 \mapsto 2, 4 \mapsto 9\}$. ⊣

Finally, it only remains to explain the `GetMatching` procedure. From

Procedure GetMatching($X_1^F, \ldots, X_k^F$)

input : k arrays $X_1^F, X_2^F, \ldots, X_k^F$ generated by Algorithm 2
output: M, a matching of π into τ that is compatible with F

```
for κ ← k ... 1 do
    if κ = k then
        x⃗ ← some element in X_k^F
    else
        x⃗ ← some element y⃗ in X_κ^F with x_i = y_i for all i ≠ ri(κ)
    M(κ) ← x_{ri(κ)}
return M = (M(1), M(2), ..., M(k))
```

Lemma 3.11 we know that if there is an element in X_k^F, a matching from π into τ that is compatible with F exists. However, we have not yet shown how a matching can be constructed from an element in X_k^F. This is what the `GetMatching` procedure does: it extracts an actual matching $M : [k] \to [n]$ out of the arrays $X_1^F, \ldots, X_k^F$. We construct M recursively: First, we pick some element $\vec{x} \in X_k^F$ and set $M(k) := x_{\mathsf{ri}(k)}$. Now, suppose the matching has been determined for $\kappa \in [k]$ and $M(\kappa) = x_{\mathsf{ri}(\kappa)}$ for some $\vec{x} \in X_\kappa^F$. Then there must exist an element $\vec{y} \in X_{\kappa-1}^F$ that has led to the element $\vec{x} \in X_\kappa^F$, i.e., $\vec{y}$ differs from $\vec{x}$ only at the $\mathsf{ri}(\kappa)$-th element. We define $M(\kappa - 1) := y_{\mathsf{ri}(\kappa-1)}$. This defines the function $M : [k] \to [n]$. It can easily be seen with the help of Lemma 3.9 that the function M returned by the `GetMatching` procedure is indeed a matching of π into τ that is compatible with F.

This concludes our description of the alternating run algorithm. We would like to remark that this description omits two minor details

necessary for obtaining the polynomial factor $\mathcal{O}(n \cdot k)$ of the desired runtime. The one detail concerns the calculation of the `Index` function. The second details concerns how data is stored in the array. These details are described in the proof of the runtime, Proposition 3.29.

Correctness

We start by providing the proof of Lemma 3.7, which states that for every matching M there exists a matching function F such that M is compatible with F.

Lemma 3.7. Given a matching M from π to τ, we will construct a matching function F such that M is compatible with F. In order to describe F, it is enough to determine the first (=leftmost) element $l_{F(i)}$ of every $F(i)$, where $i \in [\mathsf{run}(\pi)]$. In order to specify the last (=rightmost) element $r_{F(i)}$ of $F(i)$ for $i \in [\mathsf{run}(\pi)]$, we simply need to apply the properties (P3) and (P4): $r_{F(i)}$ is either the last element in τ or the leftmost valley (peak) in $F(i+1)$ in case that the i-th run is a run up (down). Clearly, $l_{F(1)} = \tau(1)$, the first element in τ – cf. (P3). When determining $l_{F(i)}$, let $l_{\pi,i}$ be the first element in the i-th run in π and $r_{\pi,i}$ be the last element in the i-th run in π. If the i-th run is a run up (down), $l_{F(i)}$ is the right-most element in τ not lying to the right of $M(l_{\pi,i})$ and following a valley (peak). This construction guarantees that F is a matching function.

In order to prove that M is compatible with F, we need to show for all $i \in [\mathsf{run}(\pi)]$ that $l_{F(i)} \preceq_\tau M(l_{\pi,i})$ and $M(r_{\pi,i}) \preceq_\tau r_{F(i)}$. The first statement holds by construction. For $i = \mathsf{run}(\pi)$, the second statement clearly also holds by construction. Let $i \in [\mathsf{run}(\pi) - 1]$. Let us assume that the i-th run is a run up – the proof for runs down is analogous. We distinguish between the following cases that are depicted in Figure 15:

- $M(r_{\pi,i})$ and $M(l_{\pi,i+1})$ lie in the same run in τ. Since we have assumed that the i-th run in π is a run up, $r_{\pi,i}$ is a peak in π. Hence, this case is only possible if $M(r_{\pi,i})$ is in a run down in τ and $r_{\pi,i} > l_{\pi,i+1}$. Thus, $l_{F(i+1)}$ is the first element in this run, which implies that $r_{F(i)}$ is the last element of this run and thus $M(r_{\pi,i}) \preceq_\tau r_{F(i)}$.

- $M(r_{\pi,i})$ and $M(l_{\pi,i+1})$ do not lie in the same run in τ and the element $M(l_{\pi,i+1})$ is in a run up in τ. In this case, $r_{F(i)}$ is the last element in the run down preceding this run and thus it clearly holds that $M(r_{\pi,i}) \preceq_\tau r_{F(i)}$.

- $M(r_{\pi,i})$ and $M(l_{\pi,i+1})$ do not lie in the same run in τ and the element $M(l_{\pi,i+1})$ is in a run down in τ. In this case, $r_{F(i)}$ is the last element in this run and again it clearly holds that $M(r_{\pi,i}) \preceq_\tau r_{F(i)}$.

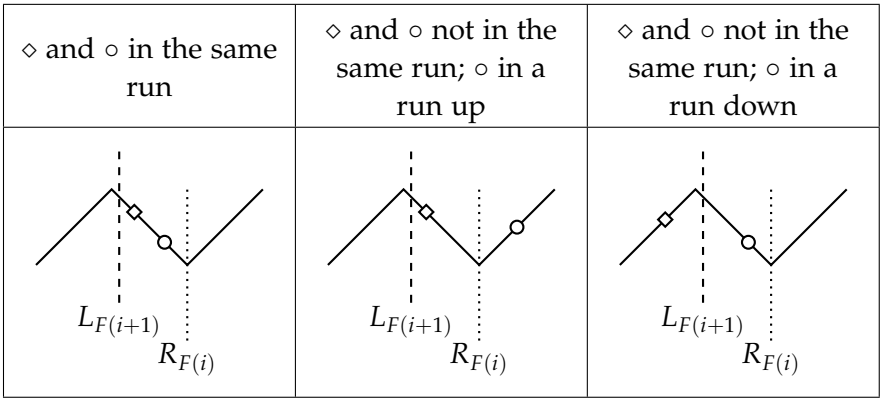


Figure 15: Three cases that have to be distinguished in the proof of Lemma 3.7 when showing that $M(r_{\pi,i}) \preceq_\tau r_{F(i)}$ for all $i \in [\mathsf{run}(\pi) - 1]$ under the assumption that the i-th run in π is a run up. The element $M(r_{\pi,i})$ is represented by a $\diamond$ and the element $M(l_{\pi,i+1})$ by a $\circ$.

□

Example 3.20. Constructing F as described in the proof of Lemma 3.7 for the matching 4629 of π_{ex} into τ_{ex} yields the matching function represented in Figure 13. ⊣

Next, we prove Lemma 3.9. This lemma states that a function M from the set $[k]$ into the set $[n]$ is a matching of π into τ compatible with F if and only if for every $\kappa \in [k]$:

1. $M(\kappa) \in F(\mathsf{ri}(\kappa))$,
2. $M(\kappa) > M(\kappa - 1)$ and
3. if $\mathsf{pre}(\kappa)$ exists, then $\mathsf{pre}(\kappa) \prec_\pi \kappa$ if and only if $M(\mathsf{pre}(\kappa)) \prec_\tau M(\kappa)$, i.e., if κ is contained in a run up (down), then $M(\kappa)$ is right (left) of $M(\mathsf{pre}(\kappa))$.

Lemma 3.9. Let $M{:}[k] \to [n]$ be a matching of π into τ that is compatible with F. Recall Definition 2.5 which states that M has to be a monotonically increasing function. This implies the second condition. Moreover, the sequence $M(\pi) = M(\pi(1)), M(\pi(2)), \ldots, M(\pi(k))$ has to be a subsequence of τ. This means nothing else than $M(\pi(1)) \prec_\tau M(\pi(2)) \prec_\tau \ldots \prec_\tau M(\pi(k))$. In particular it must hold that $M(u) \prec_\tau M(v)$, where u and v are two neighbouring elements in the same run in π with $u \prec_\pi v$. This implies the third condition. Finally, the first condition follows directly from the definition of compatibility (Definition 3.6).

Let $M{:}[k] \to [n]$ be a function fulfilling the three conditions stated above. The second condition implies that M is monotonically increasing. In order to show that M is indeed a matching of π into τ, we

have to show that $M(\pi) = M(\pi(1)), M(\pi(2)), \ldots, M(\pi(k))$ is a subsequence of τ. In other words, we have to show that for all $i \in [k-1]$ it holds that $M(\pi(i)) \prec_\tau M(\pi(i+1))$. We distinguish three cases:

- The elements $\pi(i)$ and $\pi(i+1)$ lie in the same run in π. Thus, for the case of a run up (down) we have $\pi(i) = \mathsf{pre}(\pi(i+1))$ ($\pi(i+1) = \mathsf{pre}(\pi(i))$). With $\kappa = \pi(i+1)$ ($\kappa = \pi(i)$) it follows from the third condition that $M(\pi(i)) \prec_\tau M(\pi(i+1))$ (in both cases).

- The elements $\pi(i)$ and $\pi(i+1)$ do not lie in the same run in π and $M(\pi(i))$ and $M(\pi(i+1))$ do not lie in the same run in τ. If $\pi(i)$ lies in the j-th run in π, the first condition implies that $M(\pi(i))$ lies in $F(j)$ and that $M(\pi(i+1))$ lies in $F(j+1)$ in τ. Then property (P4) of matching functions (the leftmost run of $F(j+1)$ is the rightmost run of $F(j)$) implies that $M(\pi(i))$ lies to the left of $M(\pi(i+1))$ in τ.

- The elements $\pi(i)$ and $\pi(i+1)$ do not lie in the same run in π but $M(\pi(i))$ and $M(\pi(i+1))$ lie in the same run in τ. By the definition of matching functions and since it holds that $M(\kappa) \in F(\mathsf{ri}(\kappa))$ for all $\kappa \in [k]$, this can only be possible if $M(\pi(i))$ is in the last run of $F(j)$ and $M(\pi(i+1))$ is in the first run of $F(j+1)$ for some $j \in [\mathsf{run}(\pi)]$. Thus, if $\pi(i)$ lies in a run up (down) in π both $M(\pi(i))$ and $M(\pi(i+1))$ are contained in a run down (up) in τ. On the other hand, if $\pi(i)$ is in a run up (down) it must be a peak (valley) and thus it holds that $\pi(i) > \pi(i+1)$ ($\pi(i) < \pi(i+1)$). The second condition then ensures that $M(\pi(i)) > M(\pi(i+1))$ ($M(\pi(i)) < M(\pi(i+1))$), which implies that $M(\pi(i))$ lies to the left of $M(\pi(i+1))$ in τ.

The function M is thus a matching of π into τ additionally fulfilling that $M(\kappa) \in F(\mathsf{ri}(\kappa))$ which means that M is a matching compatible with F. □

Lemma 3.11 states that in Algorithm 1, $\vec{x} \in X_\kappa^F$ is a (κ, F)-matching. This can be shown as follows:

Lemma 3.11. We prove this statement by induction over κ. For $\kappa = 1$ this is easy: An element $\vec{x} \in X_1^F$ looks as follows: $x_i = 0$ for all $i \neq \mathsf{ri}(1)$ and $x_{\mathsf{ri}(1)}$ is equal to some $u \in F(\mathsf{ri}(1))$. Thus, the function $M : [1] \to [n]$ with $M(1) = u$ is clearly a $(1, F)$-matching.

Now suppose we have proven the statement of Lemma 3.11 for $\kappa - 1$ and we want to prove it for κ. If $\vec{x} \in X_\kappa^F$, then there must exist an element $\vec{y} \in X_{\kappa-1}^F$ and an element $\nu \in [n]$ such that $\vec{x} = (y_1, \ldots, y_{\mathsf{ri}(\kappa)-1}, \nu, y_{\mathsf{ri}(\kappa)+1}, \ldots, y_{\mathsf{run}(\pi)})$ (see lines 5 to 8 in Algorithm 1). This element ν may not be any arbitrary element, it must fulfil the following conditions (see Algorithm 1, Line 6): $\nu \in F(\mathsf{ri}(\kappa))$, $\nu > x_{\mathsf{ri}(\kappa-1))}$ and $\mathsf{pre}(\kappa) \prec_\pi \kappa$ if and only if $x_{\mathsf{ri}(\mathsf{pre}(\kappa))} \prec_\tau \nu$. Since $\vec{y} \in X_{\kappa-1}^F$ it

is a $(\kappa-1, F)$-matching and thus there exists a function $M : [\kappa-1] \to [n]$ that is a matching of $\pi|_{[\kappa-1]}$ into τ that is compatible with F and for which it additionally holds that for every $y_i \neq 0$, $M(\max\{\kappa' \leq \kappa-1 : \mathsf{ri}(\kappa') = i\}) = y_i$.

We now define a function $\tilde{M} : [\kappa] \to [n]$ as follows: $\tilde{M}(u) = M(u)$ for all $u \in [\kappa-1]$ and $\tilde{M}(\kappa) = \nu$. We will see that this function $\tilde{M}$ is a witness for the fact that $\vec{x}$ is a (κ, F)-matching. For this purpose we have to check that the three conditions in Lemma 3.9 are fulfilled for every $u \in [\kappa]$. For $u < \kappa$ these conditions are necessarily fulfilled since we then have $\tilde{M}(i) = M(i)$ and M is a matching of $\pi|_{[\kappa-1]}$ into τ that is compatible with F. For $u = \kappa$, i.e., $\tilde{M}(u) = \nu$, these conditions are exactly those stated above that must be fulfilled by the element $\nu \in [n]$. The last condition in Definition 3.10, namely that for every $x_i \neq 0$, $\tilde{M}(\max\{\kappa' \leq \kappa : \mathsf{ri}(\kappa') = i\}) = x_i$, is fulfilled since M is a witness for the fact that $\vec{y}$ is a $(\kappa-1, F)$-matching and since we defined $\tilde{M}(\kappa)$ to be equal to $\nu = x_{\mathsf{ri}(\kappa)}$. Thus, $\vec{x}$ is a (κ, F)-matching. □

The next lemma shows that it is sound to consider elements returned by the Rep procedure only.

Definition 3.21. *Let F be a matching function and $\vec{x} = (x_1, x_2, \ldots, x_{\mathsf{run}(\pi)})$ be a (κ, F)-matching for some $\kappa \in [k]$. A matching M (κ, F)-*extends *$\vec{x}$ if M is compatible with F and if for every $x_i \neq 0$, $M(\max\{\kappa' \leq \kappa : \mathsf{ri}(\kappa') = i\}) = x_i$, i.e., M maps the largest element $\leq \kappa$ in the i-th run of π to the i-th element of $\vec{x}$.*

Definition 3.22. *Let $\vec{x} = (x_1, \ldots, x_{\mathsf{run}(\pi)})$. In the following, we write $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu)$ instead of $(x_1, \ldots, x_{\mathsf{ri}(\kappa)-1}, \nu, x_{\mathsf{ri}(\kappa)+1}, \ldots, x_{\mathsf{run}(\pi)})$.*

Lemma 3.23. *Let $\kappa \in [k]$ and $\vec{x} \in X_\kappa^F$. If there exists a matching M that (κ, F)-extends $\vec{x}$, then there exist an element $\nu \in \mathsf{Rep}(\vec{x}, \kappa+1, F)$ and a matching $\tilde{M}$ that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu)$.*

Proof. Let us first explicitly show how to pick the element ν. Then we will prove that it indeed holds that ν is in $\mathsf{Rep}(\vec{x}, \kappa+1, F)$. We define $\tilde{M}$ as follows: $\tilde{M}(\kappa+1) := \nu$ and $\tilde{M}(u) := M(u)$ otherwise. Finally, we will see that $\tilde{M}$ is a matching that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu)$.

In order to increase legibility, let $i \in [k]$ be the position for which $\pi(i) = \kappa+1$. Let us then consider the set S consisting of all elements in τ that lie to the right of $M(\pi(i-1))$ and to the left of $M(\pi(i+1))$, that are contained in $F(\mathsf{ri}(\kappa+1))$ and that are larger than $M(\kappa) = x_{\mathsf{ri}(\kappa)}$. More formally, we have

$$\begin{aligned} S :=& \{u \in [n] : M(\pi(i-1)) \prec_\tau u \prec_\tau M(\pi(i+1))\} \\ & \cap F(\mathsf{ri}(\kappa+1)) \cap [M(\kappa)+1, n]. \end{aligned}$$

This set is never empty: Especially, $M(\kappa+1)$ is contained in S since M is a matching that (κ, F)-extends $\vec{x}$. We now define $\nu := \min(S)$.

We have to check that it indeed holds that $\nu \in \mathsf{Rep}(\vec{x}, \kappa+1, F)$. We refer the reader to the definition of $\mathsf{Rep}(\vec{x}, \kappa+1, F)$ on page 55.

- (C1) is fulfilled by construction of S.
- (C2) is fulfilled since $\nu > M(\kappa-1) = x_{\mathsf{ri}(\kappa-1)}$.
- (C3) is fulfilled: ν is a valley in the subsequence of τ consisting of elements larger than $M(\kappa)$ by construction of S.
- (C4) If the run predecessor of $\kappa+1$ exists and $\kappa+1$ lies in a run up (down), $\mathsf{pre}(\kappa+1) = \pi(i-1)$ ($\mathsf{pre}(\kappa+1) = \pi(i+1)$). Moreover, note that $M(\mathsf{pre}(\kappa+1)) = x_{\mathsf{ri}(\kappa+1)}$ since M (κ, F)-extends $\vec{x}$. Since $S \subseteq \{u \in [n] : M(\pi(i-1)) \prec_\tau u \prec_\tau M(\pi(i+1))\}$, it is guaranteed that ν lies on the correct side of $x_{\mathsf{ri}(\kappa+1)}$.
- (C5) In case $\kappa+1$ is the largest element in its run in π, there is only a single element in $\mathsf{Rep}(\vec{x}, \kappa+1, F)$ which is exactly ν.
- (C6) In case $\kappa+1$ is not the largest element in its run in π and $\kappa+1$ lies in a run up (down), the element $M(\pi(i+1))$ ($M(\pi(i-1))$) is an element larger than ν that lies to the right (left) of ν in $F(\mathsf{ri}(\kappa+1))$ since M is compatible with F.

Now let us show that $\tilde{M}$ as defined above is a matching that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu)$. First we need to show that the function $\tilde{M}$ is a matching of π into τ that is compatible with F. Here Lemma 3.9 comes in handy since it tells us that we only have to check the following three conditions for all $u \in [k]$:

1. $\tilde{M}(u) \in F(\mathsf{ri}(u))$: For $u = \kappa+1$ this holds by construction of ν and for $u \neq \kappa+1$ this holds since we then have $\tilde{M}(u) = M(u)$ and M is a matching that is compatible with F.
2. $\tilde{M}(u+1) > \tilde{M}(u)$ for $u \neq k$: For $u \notin \{\kappa, \kappa+1\}$ this again holds since M is a matching.

   $u = \kappa$: By the construction of S, $\tilde{M}(\kappa+1) = \nu > M(\kappa) = \tilde{M}(\kappa)$.

   $u = \kappa+1$: Again by the construction of S we know that $\nu \leq M(\kappa+1)$. Since M is a matching $M(\kappa+1) < M(\kappa+2) = \tilde{M}(\kappa+2)$ it follows that $\nu = \tilde{M}(\kappa+1) < \tilde{M}(\kappa+2)$.
3. If $\mathsf{pre}(u)$ exists, then $\mathsf{pre}(u) \prec_\pi u$ if and only if $\tilde{M}(\mathsf{pre}(u)) \prec_\tau \tilde{M}(u)$: Since M is a matching, we only have to check this condition for $\kappa+1$ and its run predecessor $\mathsf{pre}(\kappa+1)$ as well as for $\kappa+1$ and κ', the next largest element in the same run in π (we could call this element the run successor of $\kappa+1$), i.e., $\mathsf{pre}(\kappa') = \kappa+1$. If $\kappa+1$ lies in a run up (down), we have $\mathsf{pre}(\kappa+1) = \pi(i-1)$ and $\kappa' = \pi(i+1)$ ($\mathsf{pre}(\kappa+1) = \pi(i+1)$ and $\kappa' = \pi(i-1)$). By construction of S we have that $M(\pi(i-1)) = \tilde{M}(\pi(i-1)) \prec_\tau \nu = \tilde{M}(\kappa+1) \prec_\tau \tilde{M}(\pi(i+1)) = M(\pi(i+1))$ and thus this condition is also fulfilled.

	possible for $i = 1$	possible for $i \in [1, \mathsf{run}(P) - 1]$
the i-th run in P is a run up	 b_1 = number of vales in $F(1) - 1$	 b_i = number of vales in $F(i) - 1$
the i-th run in P is a run down	 b_1 = number of vales in $F(1) - 1$	 b_i = number of vales in $F(i)$

Figure 16: Possible shapes that $F(i)$ can have in τ, where $i \neq \mathsf{run}(\pi)$. Runs that are drawn with dashed lines indicate that elements x lying in these runs fulfil $\mathsf{vi}(x) \equiv 1 \mod b_i$.

In order to show that $\tilde{M}$ $(\kappa+1, F)$-extends $\vec{y} := \vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu)$ it remains to show that for every $y_i \neq 0$, $\tilde{M}(\max\{\kappa' \leq \kappa : \mathsf{ri}(\kappa') = i\}) = y_i$. For $i \neq \mathsf{ri}(\kappa+1)$ this follows from the fact that $y_i = x_i$ and that M is a matching that (κ, F)-extends $\vec{x}$. For $i = \mathsf{ri}(\kappa+1)$ this hold by definition of $\tilde{M}$: we have $y_i = \nu$ and $\tilde{M}(\max\{\kappa' \leq \kappa+1 : \kappa' \text{ is in the same run as } \kappa+1\}) = \tilde{M}(\kappa+1) = \nu$. □

It remains to prove that the use of the array data structure and in particular the `Index` function do not cause that relevant (κ, F)-matchings are discarded. This is done by the following two lemmas.

Lemma 3.24. *Let $\vec{x}, \vec{y}$ be two (κ, F)-matchings, where $\kappa \in [k]$ and F is a matching function. If $\mathtt{Index}(\vec{x}) = \mathtt{Index}(\vec{y})$, then for all $i \in [\mathsf{run}(\pi)]$ it holds that*

- *x_i and y_i lie in the same vale in τ or*
- *the largest element in the i-th run in π is smaller or equal to κ.*

Proof. From the definition of the `Index` function it is clear that the following holds for all $i \in [\mathsf{run}(\pi)]$: if $\mathtt{Index}(\vec{x}) = \mathtt{Index}(\vec{y})$, then $\mathsf{vi}(x_i) \equiv \mathsf{vi}(y_i) \mod b_i$. Recall that for $i = \mathsf{run}(\pi)$, b_i corresponds exactly to the number of vales in $F(i)$ and thus $\mathsf{vi}(x_{\mathsf{run}(\pi)}) \equiv \mathsf{vi}(y_{\mathsf{run}(\pi)}) \mod b_i$ is only possible if $\mathsf{vi}(x_{\mathsf{run}(\pi)}) = \mathsf{vi}(y_{\mathsf{run}(\pi)})$ which means nothing else than that $x_{\mathsf{run}(\pi)}$ and $y_{\mathsf{run}(\pi)}$ lie in the same vale in τ.

For the case that $i \neq \mathsf{run}(\pi)$, this is not always that simple. Consider the four possible shapes that $F(i)$ can have, as depicted in Figure 16. Let us first take a look at the case that the i-th run in π is a run up. Here, $\mathsf{vi}(x_i) \equiv \mathsf{vi}(y_i) \mod b_i$ is possible if x_i and y_i lie in the same

vale in τ or if x_i lies in the first vale in $F(i)$ and y_i lies in the last run in $F(i)$ (or vice-versa). Now recall the definition of the Rep procedure: an element in the last run (which is always a run down) may only be chosen for the largest element in its run in π (Condition (C6)). This means that the largest element in the i-th run in π must be smaller or equal to κ. Now let us consider the case that the i-th run in π is a run down. Here, if x_i and y_i do not lie in the same vale in τ, $\mathsf{vi}(x_i) \equiv \mathsf{vi}(y_i) \mod b_i$ is only possible for $i = 1$ and if τ starts with a run up: x_i has to then lie in this first run of τ and y_i in the last vale of $F(1)$ (or vice-versa). Again, because of Condition (C6), this is only possible for the largest element in its run in π. Thus, we can again conclude that the largest element in the i-th run in π must be smaller or equal to κ. □

Lemma 3.25. *Let $\vec{x}, \vec{y}$ be two (κ, F)-matchings, where $\kappa \in [k]$ and F is a matching function. In addition to that, let $\nu_x \in \mathtt{Rep}(\vec{x}, \kappa+1, F)$ and $\nu_y \in \mathtt{Rep}(\vec{y}, \kappa+1, F)$. If*

$$\mathtt{Index}(\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)) = \mathtt{Index}(\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y))$$

and $\nu_y \leq \nu_x$ the following holds: if there exists a matching that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)$, then there exists a matching that $(\kappa+1, F)$-extends $\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y)$. Thus, the alternating run algorithm only has to keep track of the $(\kappa+1, F)$-matching $\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y)$.

Proof. Let M_x be a matching of π into τ that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)$. We shall construct a function $M_y : [k] \to [n]$ and show that it is a matching that $(\kappa+1, F)$-extends $\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y)$.

Since $\vec{y}$ is a (κ, F)-matching (Recall Definition 3.10) there exists a partial matching $M : [\kappa] \to [n]$ of $\pi|_{[\kappa]}$ into τ for which it additionally holds that for every $y_i \neq 0$, $M(\max\{\kappa' \leq \kappa : \mathsf{ri}(\kappa') = i\}) = y_i$. We define the function M_y as follows:

$$M_y(u) = \begin{cases} M(u), & \text{for } u \in [\kappa] \\ \nu_y, & \text{for } u = \kappa+1 \\ M_x(u), & \text{for } u \in [\kappa+2, k] \end{cases}$$

We now need to show that M_y is indeed a matching that $(\kappa+1, F)$-extends $\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y)$. As in the proof of Lemma 3.23, we shall use Lemma 3.9 to show that M_y is a matching that is compatible with F. We have to check the following three conditions for all $u \in [k]$:

1. $M_y(u) \in F(\mathsf{ri}(u))$: For $u = \kappa+1$ this holds since $\nu_y \in \mathtt{Rep}(\vec{y}, \kappa+1, F)$ (Condition (C1)) and for $u \neq \kappa+1$ this holds since M_x and M are matchings that are compatible with F.

2. $M_y(u+1) > M_y(u)$ for $u \neq k$: For $u \notin \{\kappa, \kappa+1\}$ this again holds since M_x and M are matchings.

a) $M_y(\kappa+1) > M_y(\kappa)$ or equivalently $\nu_y > M(\kappa) = y_{\mathsf{ri}(\kappa)}$: This holds since $\nu_y \in \mathsf{Rep}(\vec{y}, \kappa+1, F)$ (Condition (C2)).

b) $M_y(\kappa+2) > M_y(\kappa+1)$ or equivalently $M_x(\kappa+2) > \nu_y$: Since M_x is a matching that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)$ it has to hold that $M_x(\kappa+2) > M_x(\kappa+1) = \nu_x$. Since we have $\nu_y \leq \nu_x$, this condition is fulfilled.

3. If $\mathsf{pre}(u)$ exists, then $\mathsf{pre}(u) \prec_\pi u$ if and only if $M_y(\mathsf{pre}(u)) \prec_\tau M_y(u)$: Since M_x and M are matchings, this condition is fulfilled for all $u \in [k]$ such that both $u < \kappa+1$ and $\mathsf{pre}(u) < \kappa+1$ or such that both $u > \kappa+1$ and $\mathsf{pre}(u) > \kappa+1$. Thus, we only have to check this condition for $u = \kappa+1$ and for all $\kappa' \in [\kappa+2, k]$ that satisfy $\mathsf{pre}(\kappa') \leq \kappa+1$. Let K be the set of all such κ'. Observe that such a κ' is the smallest element in the $\mathsf{ri}(\kappa')$-th run in π that is strictly larger than $\kappa+1$. This means that $\mathsf{pre}(\kappa')$, if it exists, is the largest element in the $\mathsf{ri}(\kappa')$-th run in π that is smaller or equal to $\kappa+1$. We only consider the case that u is contained in a run up – the proof for the case that u lies in a run down works analogously. We have to check the condition for the following three situations:

a) $u = \kappa+1$: If $\mathsf{pre}(\kappa+1)$ exists it has to hold that $M_y(\mathsf{pre}(\kappa+1)) = y_{\mathsf{ri}(\kappa+1)} \prec_\tau \nu_y$. This condition is fulfilled since $\nu_y \in \mathsf{Rep}(\vec{y}, \kappa+1, F)$ (Condition (C4)).

b) $u = \kappa' \in K$ such that $\mathsf{pre}(\kappa') = \kappa+1$: If this element κ' exists we have to show that $\nu_y \prec_\tau M_y(\kappa') = M_x(\kappa')$. Since $\kappa+1$ is not the largest element in its run in π, we know from Lemma 3.24 that ν_x and ν_y lie in the same vale in τ. Moreover we know that $\nu_x \geq \nu_y$ – but what does this imply for the right-left order of ν_x and ν_y within this vale? Two cases may occur: ν_x may lie in the run up or in the run down of this vale. If ν_x lies in the run up, then it has to hold that $\nu_y \prec_\tau \nu_x$. Since M_x is a matching, it has to hold that $\nu_x = M_x(\kappa+1) \prec_\tau M_x(\kappa')$ and thus $\nu_y \prec_\tau M_x(\kappa')$. If ν_x lies in the run down, $\nu_x \prec_\tau \nu_y$ and all elements between ν_x and ν_y in τ are smaller than v_x. This implies that $M_x(\kappa')$ which is larger than ν_x and lies to the right of ν_x also has to lie to the right of ν_y in τ.

c) $u = \kappa' \in K$ with $\mathsf{pre}(\kappa') < \kappa+1$: We need to show that $y_{\mathsf{ri}(\mathsf{pre}(\kappa'))} = y_{\mathsf{ri}(\kappa')} = M(\mathsf{pre}(\kappa')) \prec_\tau M_x(\kappa')$. Since M_x is a matching that $(\kappa+1, F)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)$, we know that $M_x(\mathsf{pre}(\kappa')) = x_{\mathsf{ri}(\kappa')}$ and that $x_{\mathsf{ri}(\kappa')} \prec_\tau M_x(\kappa')$. Moreover, since $\mathtt{Index}(\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu_x)) = \mathtt{Index}(\vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y))$ and $\mathsf{pre}(\kappa')$ is not the largest element in its run in π we know from Lemma 3.24 that $x_{\mathsf{ri}(\kappa')}$ and $y_{\mathsf{ri}(\kappa')}$ lie in the same vale in τ. However, nothing is known about the relative positions of these two elements within this vale

and we have to distinguish two cases. If $y_{\mathsf{ri}(\kappa')} \prec_\tau x_{\mathsf{ri}(\kappa')}$ the statement follows easily since $y_{\mathsf{ri}(\kappa')} \prec_\tau x_{\mathsf{ri}(\kappa')} \prec_\tau M_x(\kappa')$. If $x_{\mathsf{ri}(\kappa')} \prec_\tau y_{\mathsf{ri}(\kappa')}$ we have to collect a few more arguments in order to prove that the condition holds. By transitivity and the condition checked in Point 2. of this proof we know that $y_{\mathsf{ri}(\kappa')} < \nu_y = M_y(\kappa+1) < M_x(\kappa')$. Now note that the elements that lie in τ between $x_{\mathsf{ri}(\kappa')}$ and $y_{\mathsf{ri}(\kappa')}$ are all smaller than $\max(x_{\mathsf{ri}(\kappa')}, y_{\mathsf{ri}(\kappa')})$ (since both are contained in the same vale). Thus, the element $M_x(\kappa')$ – that is to the right of $x_{\mathsf{ri}(\kappa')}$ and larger than $y_{\mathsf{ri}(\kappa')}$ – has to lie to the right of $y_{\mathsf{ri}(\kappa')}$. This is what we wanted to prove.

Let $\vec{y}' = \vec{y}(\mathsf{ri}(\kappa+1) \leftarrow \nu_y)$. It remains to show that for every $i \in [\mathsf{run}(\pi)]$ with $y'_i \neq 0$, $M_y(\max\{\kappa' \leq \kappa+1 : \mathsf{ri}(\kappa') = i\}) = y'_i$. This follows directly from the definition of $M_y(\kappa+1)$ and the fact that M is a witness for $\vec{y}$ being a (κ, F)-matching. □

Finally, we have gathered all necessary information to prove the correctness of the alternating run algorithm.

Proposition 3.26. *π can be matched into τ if and only if X_k^F is non-empty for some matching function F.*

Proof. ($\Rightarrow$) If there is a matching of π into τ, then there is at least one matching function F for which X_k^F is nonempty:
Since there exists a matching M, we know from Lemma 3.7 that there exists some matching function F such that M is compatible with F. Let us fix this F. We prove by induction over $\kappa \in [k]$ that there is an $\vec{x} \in X_\kappa^F$ and a matching M_κ that (κ, F)-extends $\vec{x}$. For $\kappa = 1$ this is easy. Let ν be the valley in τ that lies in the same vale as $M(1)$. It is clear that $\nu \in \mathsf{Rep}((0,\dots,0), 1, F)$. Consequently, the tuple $\vec{x}$ with $x_i = 0$ for $i \neq \mathsf{ri}(1)$ and $x_{\mathsf{ri}(1)} = \nu$ is contained in X_1^F. Observe that M_1 being defined by $M_1(u) = M(u)$ for $u \neq 1$ and $M_1(1) = \nu$ is a matching that $(1, F)$-extends $\vec{x}$.

Now, let $\kappa \in [k]$ and assume that $\vec{x} \in X_\kappa^F$ and M_κ κ-extends $\vec{x}$. We show that there exist an $\vec{x}' \in X_{\kappa+1}^F$ and a $M_{\kappa+1}$ that $(\kappa+1)$-extends $\vec{x}'$. By Lemma 3.23, there exists a $\nu \in \mathsf{Rep}(\vec{x}, \kappa+1, F)$ and a matching $M_{\kappa+1}$ that $(\kappa+1)$-extends $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu)$. At this point, we cannot be sure that $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu) \in X_{\kappa+1}^F$ since $X_{\kappa+1}^F$ may contain another (κ, F)-matching $\vec{y}$ with $\mathtt{Index}(\vec{x}) = \mathtt{Index}(\vec{y})$. However, this is only possible if $y_{\mathsf{ri}(\kappa+1)} \leq x_{\mathsf{ri}(\kappa+1)}$ (see Line 10 in Algorithm 2). By Lemma 3.25 we know that, in this case, there exists a matching that $(\kappa+1)$-extends $\vec{y}$. So, no matter whether $\vec{x}(\mathsf{ri}(\kappa+1) \leftarrow \nu) \in X_{\kappa+1}^F$ or not, we can conclude that there is an $\vec{x}' \in X_{\kappa+1}^F$ and a matching function $M_{\kappa+1}$ that $(\kappa+1)$-extends $\vec{x}'$. By induction, we have shown that $X_k^F \neq \emptyset$.
($\Leftarrow$) If there is a matching function F such that the corresponding X_k^F is non-empty, then a matching of π into τ can be found: This is an immediate consequence of Corollary 3.13. □

Finally, let us remark that the function M returned by the procedure $\texttt{GetMatching}(X_1^F, \ldots, X_k^F)$ is indeed a matching, as can easily be seen with the help of Lemma 3.9: The first condition in the lemma is satisfied because of Condition (C1) for representative elements. The second condition holds because of Condition (C2). The third condition corresponds to Condition (C4). Note that(C3), (C5) and (C6) are only required for improving the runtime.

Runtime

We are now going to prove the promised fpt runtime bounds. First, we bound the number of matching functions.

Lemma 3.27. *There are less than* $(\sqrt{2})^{\mathsf{run}(\tau)}$ *functions from* $[\mathsf{run}\pi]$ *to subsequences of* τ *that satisfy (P1) to (P4).*

Proof. A matching function F can be uniquely characterized by fixing the position of the first run up in every $F(i)$ for $i \in [\mathsf{run}(\pi)]$. This is because the last run of $F(i)$ is the first run of $F(i+1)$ for all $i \in [\mathsf{run}(\pi) - 1]$. Moreover the first run up in $F(1)$ is always the first run up in τ. Thus, the number of matching functions is equal to the number of possibilities of picking $\mathsf{run}(\pi) - 1$ runs (for the first run in π no choice has to be made) among the at most $\lceil \mathsf{run}(\tau)/2 \rceil$ runs up in τ. Hence, we obtain

$$\binom{\lceil \mathsf{run}(\tau)/2 \rceil}{\mathsf{run}(\pi) - 1} \leq 2^{\lceil \mathsf{run}(\tau)/2 \rceil - 1} < (\sqrt{2})^{\mathsf{run}(\tau)}.$$

The first inequality holds since $\binom{n}{k} < 2^{n-1}$ for all $n, k \in \mathbb{N}$ as can easily be proven by induction over n. □

Now we bound the size of X_κ^F, which is the main step to achieve the $1.79^{\mathsf{run}(\tau)}$ runtime bound.

Lemma 3.28. *For any given matching function* F *and every* $\kappa \in [k]$

$$|X_\kappa^F| \leq 2 \cdot \prod_{i=1}^{run(\pi)} \frac{\mathsf{run}(F(i))}{2} \leq 1.6 \cdot 1.261071^{\mathsf{run}(\tau)}.$$

Proof. Recall that each (κ, F)-matching in X_κ^F has a position as determined by the function $\texttt{Index}$, defined by

$$\texttt{Index}(x_1, \ldots, x_{\mathsf{run}(\pi)}) = 1 + \sum_{i=1}^{\mathsf{run}(\pi)} (\mathsf{vi}(x_i) \mod b_i) \cdot \prod_{j=1}^{i-1} b_j.$$

For $i \in [\mathsf{run}(\pi) - 1]$ we have $b_i = \lfloor \mathsf{run}(F(i)/2 \rfloor$ and for $i = \mathsf{run}(\pi)$ $b_{\mathsf{run}(\pi)} \leq \lfloor \mathsf{run}(F(\mathsf{run}(\pi))/2 \rfloor + 1$ since $b_{\mathsf{run}(\pi)}$ is equal to the number

of vales in $F(\mathsf{run}(\pi))$[1]. The range of `Index` is $\left\{1,\ldots,\prod_{i=1}^{\mathsf{run}(\pi)} b_i\right\}$. Since the function `Index` determines the positions in the array X_κ^F, we obtain

$$|X_\kappa^F| = \prod_{i=1}^{\mathsf{run}(\pi)} b_i \leq \prod_{i=1}^{\mathsf{run}(\pi)-1} \left\lfloor \frac{\mathsf{run}(F(i))}{2} \right\rfloor \cdot \left(\left\lfloor \frac{\mathsf{run}(F(\mathsf{run}(\pi)))}{2} \right\rfloor + 1 \right)$$

and consequently

$$|X_\kappa^F| \leq 2 \cdot \prod_{i=1}^{\mathsf{run}(\pi)} \frac{\mathsf{run}(F(i))}{2}. \tag{16}$$

We want to bound X_κ^F and thus want to know when the product in Equation (16) is maximal. The maximum of this product has to be determined under the condition that

$$\sum_{i=1}^{\mathsf{run}(\pi)} \mathsf{run}(F(i)) = \mathsf{run}(\tau) + \mathsf{run}(\pi) - 1, \tag{17}$$

since two subsequent $F(i)$'s have one run in common (cf. Definition 3.4). The inequality of geometric and arithmetic means implies that the product in Equation (16) is maximal if all $\mathsf{run}(F(i))$ are equal, i.e., for every $i \in \mathsf{run}(\pi)$, $\mathsf{run}(F(i)) = \frac{\mathsf{run}(\tau)+\mathsf{run}(\pi)-1}{\mathsf{run}(\pi)}$. Therefore, X_κ^F has at most $2 \cdot \left(\frac{\mathsf{run}(\tau)+\mathsf{run}(\pi)-1}{2\cdot\mathsf{run}(\pi)}\right)^{\mathsf{run}(\pi)}$ elements. To shorten the proof, we write in the following p for $\mathsf{run}(\pi)$ and t for $\mathsf{run}(\tau)$.

Thus, we want to determine the maximum of the function

$$g(p) = \left(\frac{t+p-1}{2p}\right)^p$$

(we omit the factor 2 for the calculation). The derivative of $g(p)$ is:

$$\begin{aligned} g'(p) =& \frac{1}{p}\left(2^{-p}\left(\frac{p+t-1}{p}\right)^{p-1} \cdot h(p)\right) \\ \text{where } h(p) =& (p+t-1)\left(\log\left(\frac{p+t-1}{p}\right) - \log(2)\right) - t + 1. \end{aligned}$$

And thus:

$$g'(p) = 0 \to h(p) = 0 \to \log\left(\frac{p+t-1}{2p}\right) = \frac{t-1}{p+t-1}.$$

The solutions are:

$$\begin{aligned} p_1(t) &= (-1+t)/(-1+2e^{1+W_0(-1/(2e))}) \\ p_2(t) &= (-1+t)/(-1+2e^{1+W_{-1}(-1/(2e))}), \end{aligned}$$

1 The reason why we do not set $b_{\mathsf{run}(\pi)} = \lfloor \mathsf{run}(F(\mathsf{run}(\pi))/2 \rfloor$ is a rather technical one: $F(\mathsf{run}(\pi))$ may end with a run up if the last run in π is a run up and may end with a run down if the last run in π is a run down. This would lead to unwanted collisions concerning the `Index` function and consequently would prohibit the proof of Lemma 3.24.

where W_0 is the principal branch of the Lambert function (defined by $x = W(x) \cdot e^{W(x)}$) and W_{-1} its lower branch. It holds that

$$\begin{aligned}(-1+t)/3.311071 &\leq p_1(t) \leq (-1+t)/3.311070\\ (-1+t)/-0.62663 &\leq p_2(t) \leq (-1+t)/-0.62664,\end{aligned}$$

The second solution $p_2(t)$ is negative and therefore of no interest to us. The first solution $p_1(t)$ is a local maximum as can be checked easily and yields

$$\begin{aligned}g(p_1) &\leq \left(\frac{t+(-1+t)/3.311070-1}{2(-1+t)/3.311071}\right)^{(-1+t)/3.311070}\\ &\leq 0.80 \cdot (1.261071)^t.\end{aligned}$$

It therefore holds that $|X_\kappa^F| \leq 1.6 \cdot 1.261071^{\mathsf{run}(\tau)}$. □

Proposition 3.29. *The runtime of the alternating run algorithm (Algorithm 2) is* $\mathcal{O}(1.784^{\mathsf{run}(\tau)} \cdot n \cdot k)$.

Proof. The main structure of the algorithm is the following: for every matching function F and for every $\kappa \in [k]$ the array X_κ^F is computed. There are $(\sqrt{2})^{\mathsf{run}(\tau)}$ matching functions (Lemma 3.27). The maximal number of elements in X_κ^F is $1.6 \cdot 1.2611^{\mathsf{run}(\tau)}$ (Lemma 3.28). Given a matching function and an element $\kappa \in [k]$, the algorithm has to execute Lines 6 to 11 for every $\vec{x} \in X_{\kappa-1}^F$. Once we have shown that the runtime of these lines is $\mathcal{O}(n)$, we obtain a total runtime of $\mathcal{O}\left((\sqrt{2})^{\mathsf{run}(\tau)} \cdot 1.2611^{\mathsf{run}(\tau)} \cdot k \cdot n\right) = \mathcal{O}(1.784^{\mathsf{run}(\tau)} \cdot k \cdot n)$.

So it remains to show that the runtime of the Lines 6 to 11 is $\mathcal{O}(n)$. First, observe that determining the set R with the help of the `Rep` procedure requires $\mathcal{O}(n)$ time. Second, for every element in R the Lines 8, 10 and 11 are executed. Since R only contains valleys (of some subsequence of τ), its size is less than $\mathsf{run}(\tau)$. Assuming unit cost for arithmetic operations, computing `Index` requires $\mathcal{O}(\mathsf{run}(\pi))$ time. However, note that it is not necessary to repeat all calculations for `Index` for every element v in R. Indeed, for a fixed $\vec{x} \in X_\kappa^F$, the elements for which `Index` is computed at Line 8 only differ at the $\mathsf{ri}(\kappa)$-th position. Assume that we have already computed $\texttt{Index}(\vec{x})$ for some $\vec{x}$. Computing $\texttt{Index}(\vec{y})$ for a $\vec{y}$ that is identical to $\vec{x}$ except at the $\mathsf{ri}(\kappa)$-th position can be done as follows:

$$\begin{aligned}\texttt{Index}(\vec{y}) =& \texttt{Index}(\vec{x})\\ &+ \left(\mathsf{vi}(y_{\mathsf{ri}(\kappa)}) - \mathsf{vi}(x_{\mathsf{ri}(\kappa)}) \mod b_{\mathsf{ri}(\kappa)}\right) \cdot \prod_{j=1}^{\mathsf{ri}(\kappa)-1} b_j.\end{aligned}$$

Consequently, Line 8 requires (amortized) constant time.

Checking the condition in Line 10 requires only constant time. However, Line 11 requires $\mathcal{O}(\mathsf{run}(\pi))$ time to write the (κ, F)-matching

to its position in X_κ^F. This is too much time to obtain the desired runtime bound – we can only afford amortized $\mathcal{O}(n)$ time per $\vec{x} \in X_{\kappa-1}^F$. This can be achieved by executing Line 11 at most once per $\vec{x} \in X_{\kappa-1}^F$. Let $\nu \in R$ be the first element for which the condition at Line 10 is fulfilled. For this element Line 11 is executed and a pointer p' to the position $\texttt{Index}(\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu))$ is created. (Recall the $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu)$ notation from Definition 3.22.) If the condition at Line 10 is fulfilled for the same $\vec{x}$ and some other $\nu' \in R$, we do not execute Line 11. Instead we only store the pointer p' and the element ν'. This is sufficient information since two (κ, F)-matchings in Line 11 that originate from the same $\vec{x}$ are identical except for the $\mathsf{ri}(\kappa)$-th element. It might be that Line 11 is executed for some other element $\vec{y} \in X_{\kappa-1}^F$ and $\nu_y \in \texttt{Rep}(\vec{y}, \kappa, F)$ at a later point. It is then possible that a (κ, F)-matching $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu)$ is overwritten that has other (κ, F)-matchings $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu')$ pointing to it. However, this can only happen in the following situation: $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu')$ is (κ, F)-extendable only if $\vec{y}(\mathsf{ri}(\kappa) \leftarrow \nu')$ is (κ, F)-extendable. (It holds that $\texttt{Index}(\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu')) = \texttt{Index}(\vec{y}(\mathsf{ri}(\kappa) \leftarrow \nu'))$. Lemma 3.25 shows that if $\vec{x}(\mathsf{ri}(\kappa) \leftarrow \nu')$ is (κ, F)-extendable, then so is $\vec{y}(\mathsf{ri}(\kappa) \leftarrow \nu')$. Strictly speaking Lemma 3.25 is not applicable since it is not guaranteed that $\nu' \in \texttt{Rep}(\vec{y}, \kappa, F)$ because ν' might not be a valley in the corresponding subsequence of τ (cf. Condition (C3)). However, all other conditions are satisfied and this suffices to prove Lemma 3.25.) Therefore, this modified array data structure is equivalent to the original data structure described in Section 3.1. Thus, we have shown that Lines 6 to 11 have a runtime of $\mathcal{O}(n)$, if we modify the array data structure to also allow for pointers. This concludes our proof. □

We conclude this section about the runtime of the alternating run algorithm by proving that an even smaller constant than 1.784 can be expected. Indeed, the following holds:

Theorem 3.30. *Let R_n be the random variable counting the number of alternating runs in an n-permutation chosen uniformly at random amongst all n-permutations. Then for $n \geq 2$ we have:* $\mathbb{E}\left(1.784^{R_n}\right) = \mathcal{O}\left(1.515^n\right)$.

Proof. In the following, let $R_{n,m}$ denote the number of n-permutations with exactly m alternating runs. Then the mean of R_n is given as follows:

$$\mathbb{E}(R_n) = \sum_{m \geq 1} m \cdot \frac{R_{n,m}}{n!}.$$

By the law of the unconscious statistician we then have that:

$$\mathbb{E}\left(1.784^{R_n}\right) = \sum_{m \geq 1} 1.784^m \cdot \frac{R_{n,m}}{n!}.$$

Let $R_n(u) = \sum_{m\geq 1} R_{n,m}u^m$ denote the generating function of alternating runs in n-permutations. Then $\mathbb{E}\left(1.784^{R_n}\right)$ can also be expressed as follows:

$$\mathbb{E}\left(1.784^{R_n}\right) = \frac{R_n(1.784)}{n!}.$$

A lot is known about the numbers $R_{n,m}$ as well as the associated generating functions: for instance $\mathbb{E}(R_n) = \frac{2n-1}{3}$ and $\mathbb{V}(R_n) = \frac{16n-29}{90}$ (see e.g.[117]). However we cannot get our hands on $R_n(1.784)$ directly, but we can do so by exploiting a connection to the well-studied Eulerian polynomials. The n-th Eulerian polynomial $A_n(u)$ enumerates n-permutations by their ascents and is defined as $A_n(u) = \sum_{m\geq 1} A_{n,m}u^m$, where $A_{n,m}$ is the number of n-permutations with exactly m ascents. Now, for the Eulerian polynomials, the following is known:

$$\sum_{n\geq 0} A_n(u)\frac{z^n}{n!} = \frac{1-u}{e^{(u-1)z} - u}. \tag{18}$$

Moreover, we have the following connection between $R_n(u)$ and $A_n(u)$ for all integers $n \geq 2$ (established in [56] and formulated more concisely by Knuth [111]):

$$\frac{R_n(u)}{n!} = \left(\frac{1+u}{2}\right)^{n-1}(1+w)^{n+1}A_n\left(\frac{1-w}{1+w}\right),$$

where $w = \sqrt{(1-u)/(1+u)}$. In order to evaluate $R_n(u)$ at $u = 1.784$, we thus only need to determine $A_n(u)$ at the corresponding value. As demonstrated in Example IX.12 in [78], it is easy to get asymptotics for the coefficients of z^n in $\sum_{n\geq 0} A_n(u)\frac{z^n}{n!}$ by a straightforward analysis of the singularities. Indeed, for $|u| < 2$, one has:

$$\frac{A_n(u)}{n!} = \left(\frac{u-1}{\log(u)}\right)^{n+1} + \mathcal{O}(2^{-n}). \tag{19}$$

Putting Equations (18) and (19) together, we finally obtain:

$$\begin{aligned}\mathbb{E}\left(1.784^{R_n}\right) =& \frac{R_n(1.784)}{n!}\\ =& \mathcal{O}\left(\left(\frac{2.784}{2}\cdot(1+w)\cdot\frac{\frac{1-w}{1+w}-1}{\log(\frac{1-w}{1+w})}\right)^n\right)\\ =& \mathcal{O}\left(c^n\right),\end{aligned}$$

where $w = \sqrt{(1-1.784)/(1+1.784)}$. Computations using any computer algebra system show that the constant c is strictly less than 1.515. Finally, we remark that the tempting approach $\mathbb{E}\left(1.784^{R_n}\right) = 1.784^{\mathbb{E}(R_n)}$ is not correct. □

Corollary 3.31. *The runtime of the alternating run algorithm can be expected to be in* $\mathcal{O}\left(1.514^{\mathsf{run}(\tau)}\cdot n\cdot k\right)$.

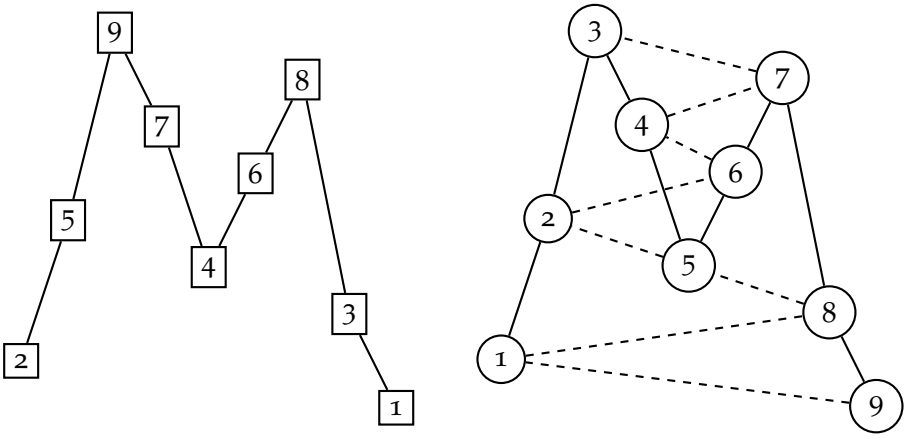


Figure 17: To the left is a graphical representation of the permutation π introduced in Example 3.33, to the right the corresponding incidence graph G_π.

3.2 THE PARAMETER run(π)

The aim of this section is twofold: First, we want to show that PPM can be solved in time $\mathcal{O}(n^{1+\mathsf{run}(\pi)})$. This result builds upon an algorithm by Ahal and Rabinovich [1] and a novel connection between the pathwidth of the *incidence graph of a permutation* [1] and the number of alternating runs in that permutation. Second, we show that this runtime cannot be improved to an fpt result unless FPT = W[1]. Let us start by defining incidence graphs:

Definition 3.32. *Given an m-permutation π, the incidence graph $G_\pi = (V, E)$ of π is defined as follows: The vertices $V := [m]$ represent positions in π. There are edges between adjacent positions, i.e., $E_1 := \{\{i, i+1\} \mid i \in [m-1]\}$. There are also edges between positions where the corresponding elements have a difference of 1, i.e., $E_2 := \{\{i, j\} \mid \pi(i) - \pi(j) = 1\}$. The edge set is defined as $E := E_1 \cup E_2$.*

Example 3.33. Consider the permutation

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 5 & 9 & 7 & 4 & 6 & 8 & 3 & 1 \end{pmatrix}$$

written in two-line representation. A graphical representation of π can be found on the left-hand side of Figure 17. The corresponding graph G_π is represented on the right-hand side of the same figure. The solid lines correspond to the edges in E_1 and the dashed lines to the ones in E_2. ⊣

Definition 3.34. *Let $G = (V, E)$ be a simple graph, i.e., E is a set of cardinality 2 subsets of V. A path decomposition of G is a sequence of subsets $S_i \subseteq V$ such that*

1. *Every vertex appears in at least one S_i.*

2. *Every edge is a subset of at least one S_i.*

3. *Let three indices $h < i < j$ be given. If a vertex is contained both in S_h and S_j then it is also contained in S_i.*

The width of a path decomposition is defined as $\max_i(|S_i|) - 1$. The pathwidth of a graph G, written $\mathsf{pw}(G)$, is the minimum width of any path decomposition.

In [1], Theorem 2.7 and Proposition 3.5, the authors present an algorithm that solves PPM in time $\mathcal{O}\left(n^{1+\mathsf{pw}(G_\pi)}\right)$. The following lemma relates $\mathsf{pw}(G_\pi)$ and the number of alternating runs in π.

Lemma 3.35. *For all permutations π, it holds that $\mathsf{pw}(G_\pi) \leq \mathsf{run}(\pi)$.*

Proof. Given an m-permutation π we will define a sequence $S_1, \ldots, S_m$. We then show that this sequence is a path decomposition of $G_\pi = (V, E)$ with width at most $\mathsf{run}(\pi)$.

In order to define the sequence $S_1, \ldots, S_m$ of subsets of V, we shall extend alternating runs to maximal monotone subsequences. This means that we add the preceding valley to a run up and the preceding peak to a run down. For any $i \in [\mathsf{run}(\pi)]$, R_i then denotes the set of elements in the i-th run in π together with the preceding valley or peak. Note that this implies that $|R_i \cap R_{i+1}| = 1$ for all $i \in [\mathsf{run}(\pi) - 1]$.

We define $S'_1 := \{1\}$ and for every $v \in [2, m]$,

$$S'_v := \left\{ \max(R_j \cap [v-1]) \mid j \in [\mathsf{run}(\pi)] \text{ and } R_j \cap [v-1] \neq \emptyset \right\} \cup \{v\},$$

i.e., S'_v contains v and the largest element of every run that is smaller than v. Since S_v should contain positions in π (and not elements), we define

$$S_v := \{\pi^{-1}(w) \mid w \in S'_v\}.$$

For an example of this construction, see Example 3.37. We now check that $S_1, \ldots, S_m$ indeed is a path decomposition.

1. The vertex i appears in $S_{\pi(i)}$.

2. First we consider edges of the form $\{i, i+1\}$. Without loss of generality let $\pi(i) < \pi(i+1)$. Then $\{i, i+1\}$ is a subset of $S_{\pi(i+1)}$. Clearly, $i+1 \in S_{\pi(i+1)}$. Since $\pi(i)$ and $\pi(i+1)$ are adjacent in π there has to be an $s \in [\mathsf{run}(\pi)]$ such that $\{\pi(i), \pi(i+1)\} \subseteq R_s$. It then holds that $\max(R_s \cap [\pi(i+1) - 1]) = \pi(i)$ since $\pi(i) \in R_s \cap [\pi(i+1) - 1]$ and $\pi(i)$ is the largest element in R_s smaller than $\pi(i+1)$. Consequently $i \in S_{\pi(i+1)}$.

   Second, every edge $\{i, j\} \in E$ with $\pi(i) - \pi(j) = 1$ is a subset of $S_{\pi(i)}$: As before $i \in S_{\pi(i)}$. Let s be any element of $[\mathsf{run}(\pi)]$ such that $j \in R_s$. Then $\max(R_s \cap [\pi(i) - 1]) = \max(R_s \cap [\pi(j)]) = \pi(j)$ and hence $j \in S_{\pi(i)}$.

   Only these two types of edges exists.

v	S'_v	S_v
1	1	9
2	12	91
3	123	918
4	234	185
5	2345	1852
6	3456	8526
7	34567	85264
8	3 5678	8 2647
9	5 789	2 473

Figure 18: The sets $S'_1, \ldots, S'_9$ and $S_1, \ldots, S_9$ for the permutation $\pi = 259746831$

3. Let $1 \leq u < v < w \leq m$ with $i \in S_u$ and $i \in S_w$. Let s be any element of $[\mathsf{run}(\pi)]$ such that $\pi(i) \in R_s$. Then either $\pi(i) \in R_s \cap [u-1]$ or $\pi(i) = u$. In both cases is $\pi(i) \in R_s \cap [v]$. Furthermore, since $\pi(i) < w$, $\pi(i) = \max(R_s \cap [w-1]) = \max(R_s \cap [v])$. Hence $\pi(i) \in S'_v$ and $i \in S_v$.

The cardinality of each S_i is at most $\mathsf{run}(\pi)+1$ and hence $\mathsf{pw}(G_\pi) \leq \mathsf{run}(\pi)$. □

Remark 3.36. This bound is tight: For $\pi = 123\ldots m$ the graph G_π is a path and hence $\mathsf{pw}(G_\pi) = \mathsf{run}(\pi) = 1$.

Example 3.37. Consider again π as defined in Example 3.33. The elements of the sets $S'_1, \ldots, S'_9$ and those of $S_1, \ldots, S_9$ as defined in the proof of Lemma 3.35 are given in Figure 18. It is easy to check that $S_1, \ldots, S_9$ indeed is a path decomposition of width $4 = \mathsf{run}(\pi)$. Note that in the given table, columns of equal numbers do not contain any gaps. This fact corresponds to the third condition in the definition of path decompositions. ⊣

Theorem 3.38. PPM *can be solved in time* $\mathcal{O}(n^{1+run(\pi)})$.

Proof. Lemma 3.35 tells us that $\mathsf{pw}(G_\pi) \leq \mathsf{run}(\pi)$. Thus the runtime of the $\mathcal{O}(n^{1+pw(G_\pi)})$ algorithm can be bounded by $\mathcal{O}(n^{1+run(\pi)})$. □

We continue with a corresponding hardness result. We prove that one cannot hope to substantially improve the XP results in Theorem 3.38: an fpt algorithm with respect to $\mathsf{run}(\pi)$ is only possible if FPT = W[1].

Theorem 3.39. PPM *is* W[1]*-hard with respect to the parameter* $\mathsf{run}(\pi)$.

Proof. We give an fpt-reduction from the W[1]-hard SEGREGATED PERMUTATION PATTERN MATCHING PROBLEM [42] to PPM. This problem is defined on page 90 in Section 4.3.

In this problem we are looking for matchings M where for all $u \leq p$ it holds that $M(u) \in [t]$ and for all $u > p$ it holds that $M(u) \in [t+1, n]$. Let (π, τ, p, t) be a SPPM instance, where $|\pi| = k \leq n = |\tau|$. We are going to construct a PPM instance $(\tilde{\pi}, \tilde{\tau})$ as follows:

$$\tilde{\pi} = (p + 0.5) \quad \underbrace{(k+1)(k+2)\dots(k+n+1)}_{=R_\pi} \quad \pi$$

$$\tilde{\tau} = (t + 0.5) \quad \underbrace{(n+1)(n+2)\dots\dots(2n+1)}_{=R_\tau} \quad \tau$$

Note that the increasing runs $R_{\{}\pi\}$ and R_τ both consist of $(n+1)$ elements. The element placed at the beginning of $\tilde{\pi}$, $p + 0.5$, is larger than p but smaller than $p+1$. Analogously, $t + 0.5$ in $\tilde{\tau}$ is larger than t but smaller than $t+1$. Note that $\tilde{\pi}$ and $\tilde{\tau}$ are not permutations in the classical sense, since they contain elements that are not integers. However, in order to obtain permutations on $[k+n+2]$ and $[2n+2]$, we simply need to relabel the respective elements order-isomorphically.

Given this construction of $\tilde{\pi}$ and $\tilde{\tau}$ the following holds: In every matching of $\tilde{\pi}$ into $\tilde{\tau}$ the element $p + 0.5$ has to be mapped to $t + 0.5$. Indeed, the increasing run of elements $R_\pi = (k+1)(k+2)\dots(k+n+1)$ in $\tilde{\pi}$ has to be mapped to the increasing run of elements $R_\tau = (n+1)(n+2)\dots\dots(2n+1)$ in $\tilde{\tau}$ and consequently π has to be matched into τ. This holds because of the following observation: If the element $(k+1)$ in $\tilde{\pi}$ is mapped to an element $(n+u)$ with $u > 1$ in $\tilde{\tau}$, some of the elements of R_π have to be matched into τ since R_π and R_τ have the same length. This is however not possible, since all elements in τ are smaller than $(n+u)$. If $(k+1)$ is instead mapped to one of the elements of τ, then all remaining elements of R_π also have to be matched into τ which is not possible since R_π is longer than τ. Therefore, the element $(k+1)$ in $\tilde{\pi}$ is always mapped to the element $(n+1)$ in $\tilde{\tau}$. Both in $\tilde{\pi}$ and in $\tilde{\tau}$ there is only one element lying to the left of $(k+1)$ and one to left of $(n+1)$: $(p+0.5)$ and $(t+0.5)$, respectively. Thus, $(p+0.5)$ has to be mapped to $(t+0.5)$. This implies that all elements smaller than $(p+0.5)$, i.e., elements in the interval $[p]$, in π have to be mapped to elements smaller than $t+0.5$, i.e., elements in the interval $[t]$, in τ. We have shown that (π, τ, p, t) is a YES-instance of SPPM if and only if $(\tilde{\pi}, \tilde{\tau})$ is a YES-instance of PPM.

It remains to show that this reduction can be done in fpt-time. Clearly $\mathsf{run}(\tilde{\pi}) = 2 + \mathsf{run}(\pi) = \mathcal{O}(k)$. Moreover the length of τ is bounded by a polynomial in the size of G since $|\tau| = n + 2 + |\tau| = 2n + 2 = \mathcal{O}(n)$. □

3.3 SUMMARY OF THE RESULTS

Let us sum up the contributions of this chapter:

- Our main result is a fixed-parameter algorithm for PPM with a runtime of $\mathcal{O}(1.79^{\mathsf{run}(\tau)} \cdot n \cdot k)$. Since the combinatorial explosion is confined to $\mathsf{run}(\tau)$, this algorithm performs especially well when τ has few alternating runs.
- Since $\mathsf{run}(\tau) \leq n$, the alternating run algorithm also solves PPM in time $\mathcal{O}(1.79^n \cdot n \cdot k)$. This is a major improvement over the brute-force algorithm with a runtime of $\mathcal{O}(2^n \cdot n)$.
- Since the number of runs in a random permutation is unlikely to be n, one can expect an even smaller constant than 1.79 on average. Indeed, we prove that the expected runtime of our algorithm is in $\mathcal{O}(1.52^n \cdot n \cdot k)$.
- We also show that an algorithm by Ahal and Rabinovich [1] has a runtime of $\mathcal{O}(n^{1+\mathsf{run}(\pi)})$. This is achieved by proving that the pathwidth of a certain graph generated by a permutation is bounded by the number of alternating runs of this permutation.
- Finally, we prove that – under standard complexity theoretic assumptions – no fixed-parameter algorithm exists with respect to $\mathsf{run}(\pi)$, i.e., we cannot hope for an algorithm with runtime $\mathcal{O}(c^{\mathsf{run}(\pi)} \cdot \mathit{poly}(n))$ for some constant c. Thus, the runtime of the aforementioned $\mathcal{O}(n^{1+\mathsf{run}(\pi)})$ algorithm cannot be substantially improved.

4

THE COMPUTATIONAL COMPLEXITY OF GENERALIZED PERMUTATION PATTERN MATCHING

This chapter is based on the publication *The computational landscape of permutation patterns* [42] which is joint work with Martin Lackner.

Permutation patterns, which we will refer to as *classical permutation patterns* in this chapter, have become well-established combinatorial objects and are widely studied in the combinatorial literature. In recent years, several other types of permutation patterns were introduced and have received increased interest, such as vincular [13], bivincular [33], mesh [36], boxed mesh [12] and consecutive patterns [62]. All these pattern types are introduced in Section 4.1.

Every type of permutation pattern naturally defines a corresponding computational problem. Let $\mathcal{C}$ denote any type of permutation pattern, i.e., let $\mathcal{C} \in \{\text{classical, vincular, bivincular, mesh, boxed mesh, consecutive}\}$.

$\mathcal{C}$ Permutation Pattern Matching ($\mathcal{C}$ PPM)	
Instance:	A permutation τ (the text) and a $\mathcal{C}$ pattern π
Question:	Does the $\mathcal{C}$ pattern π occur in τ?

In this chapter we study the classical, vincular, bivincular, mesh, boxed mesh and consecutive pattern matching problem. Often we abbreviate Classical Permutation Pattern Matching with PPM and the other problems with $\mathcal{C}$ PPM, where $\mathcal{C}$ is the corresponding pattern type.

While the combinatorial structure of permutation patterns is being extensively studied, the computational perspective has so far received less attention. Related work to computational aspects of permutation patterns has been gathered on page 48 in the preceding Chapter 3. This chapter draws a map of the computational landscape of permutation patterns and thus aims at paving the way for a detailed computational analysis of these problems.

The contents of this chapter are the following:

- In Section 4.1, we survey different types of permutation patterns. Our focus lies on classical, vincular, bivincular, mesh, boxed mesh and consecutive patterns. The hierarchy of these patterns with the most general one at the top is displayed in Figure 19.
- In Section 4.2, we study the computational complexity of every corresponding decision problem. We strengthen the result that

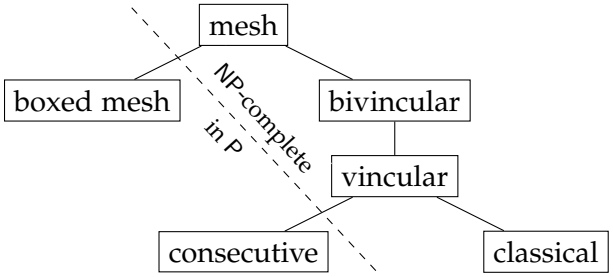


Figure 19: Hierarchy of pattern types

Classical Permutation Pattern Matching is NP-complete [31] and also show in which cases $\mathcal{C}$ PPM can be solved in polynomial time.

- In Section 4.3, we offer a more fine-grained complexity analysis by employing the framework of parameterized complexity. For most NP-complete problems we provide a more detailed complexity classification by showing W[1]-completeness with respect to the parameter length of π.
- Both the classical as well as the parameterized complexity results are summarized in Section 4.4 and in the two Tables 3 and 4 on pages 83 and 84, respectively.

4.1 TYPES OF PATTERNS

In this section we give an overview of several different types of permutation patterns that have been introduced in the last years and that will be of interest in this paper. Besides classical permutation patterns that were defined in the Preliminaries (see Section 2.2, we consider the following types of patterns: vincular, bivincular, mesh, boxed mesh and consecutive patterns. A schematic representation of their hierarchy can be found in Figure 19 and examples can be found in Tables 3 and 3. For details, we refer to the Chapters 1 and 5-7 in Kitaev's monograph *Patterns in Permutations and Words* [108].

Before we introduce non-classical types of patterns, we extend the definition of matchings from classical to other types of patterns:

Definition 4.1. *Let $\mathcal{C} \in \{$classical, vincular, bivincular, mesh, boxed mesh, consecutive$\}$. A* matching *of a $\mathcal{C}$ pattern π of length k into a permutation τ of length n is an increasing mapping $M : [k] \to [n]$ such that the sequence $M(P(1)), M(P(2)), \ldots, M(P(k))$ is an occurrence of the $\mathcal{C}$ pattern π in τ.*

Matchings are denoted by M throughout this chapter.

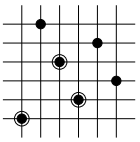
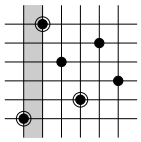
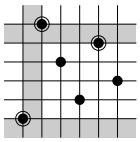
	Classical	Vincular	Bivincular
Pattern	$\pi = 132 =$ 	$\pi = \underline{13}2 =$  $\mathsf{cols}(\pi) = 1$	$\pi = \begin{matrix}\lceil 1\,\overline{2\,3} \\ \underline{1\,3}\,2\end{matrix} =$  $\mathsf{cols}(\pi) = 1$ $\mathsf{rows}(\pi) = 2$
Text			
Classical complexity	NP-complete [31]	NP-complete Corollary 4.8	NP-complete Corollary 4.8
Parameterized complexity	FPT [93]	W[1]-complete Theorem 4.15	W[1]-complete Theorem 4.16

Table 3: Examples of classical, vincular and bivincular permutation patterns

Vincular patterns

Let $\tau(i_1)\tau(i_2)\dots\tau(i_k)$ be an occurrence of the classical pattern π in the text τ. Then there are no requirements on the elements in τ lying in between $\tau(i_j)$ and $\tau(i_{j+1})$. It is however natural to ask for occurrences of patterns in which certain elements are forced to be adjacent in the text, i.e., $\tau(i_{j+1}) = \tau(i_j + 1)$. Vincular patterns are a generalization of classical patterns capturing these requirements on adjacency in the text. They were introduced under the name of *generalized patterns* in 2000 by Babson and Steingrímsson in [13], where it was shown that essentially all Mahonian permutation statistics in the literature can be written as linear combinations of vincular patterns. For a survey of this topic, see [149].

Here we use the name of *vincular patterns* as it was introduced by Kitaev in [108]. We also use the notation introduced there, since it is consistent with the notation for classical patterns.

Definition 4.2. *A* vincular pattern π *is a permutation in which certain consecutive entries may be underlined. An occurrence of* π *in a permutation* τ *is then an occurrence of the corresponding classical pattern for which underlined elements are matched to adjacent elements. To be more formal: An occurrence of* π *in* τ *corresponds to a subsequence* $\tau(i_1)\tau(i_2)\dots\tau(i_k)$ *of* τ *that is order-isomorphic to* π *and for which* $\tau(i_{j+1}) = \tau(i_j + 1)$ *whenever* π *contains* $\underline{\pi(j)\pi(j+1)}$*. Furthermore, if* π *starts with* $\underline{\lfloor\pi(1)}$ *an occurrence of* π *in* τ *must start with the first entry in* τ*, i.e.,* $\tau(i_1) = \tau(1)$*. Similarly, if* π *ends with* $\pi(k)\rfloor$ *it must hold that* $\tau(i_k) = \tau(n)$*.*

When using plots in a $[0, n+1] \times [0, n+1]$-grid to represent n-permutations, adjacency of positions clearly corresponds to adjacency of columns. In order to represent the underlined elements in vincular patterns in the corresponding grids, one shades the columns which

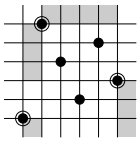
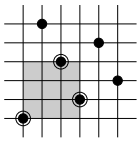
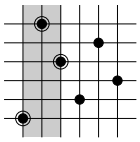
	Mesh	Boxed mesh	Consecutive
Pattern	$\pi = (\pi, R) =$  $\mathsf{cells}(\pi) = 5$	$\pi = \boxed{132} =$ 	$\pi = \underline{132} =$ 
Text			
Classical complexity	NP-complete Corollary 4.8	in P; Theorem 4.9	in P; Theorem 4.10
Parameterized Complexity	W[1]-complete Theorem 4.17	trivially FPT	trivially FPT

Table 4: Examples of mesh, boxed mesh and consecutive permutation patterns

may not contain any elements in a matching. For an example, see the middle column of Table 3. Matching the pattern $\underline{132}$ into the permutation τ, means that no elements may lie in the columns between $M(1)$ and $M(3)$ in τ.

In order to specify how many adjacency restrictions are made in the vincular pattern π, we define $\mathsf{cols}(\pi)$ to be the number of shaded columns in the grid corresponding to π.

Note that the operations *complement* and *reverse* may be performed on vincular patterns, leading to some (other) vincular pattern. Similarly as for classical patterns it then holds that π can be matched into τ iff π^c can be matched into τ^c and iff π^r can be matched into τ^r. The inverse of a vincular pattern is however not clearly defined. This leads to a larger class of patterns which is introduced below.

Bivincular patterns

Bivincular patterns generalize classical patterns even further than vincular patterns. Indeed, in bivincular patterns, not only positions but also values of elements involved in a matching may be forced to be adjacent. When Bousquet-Mélou, Claesson, Dukes and Kitaev introduced bivincular patterns in 2010 [33], the main motivation was to find a minimal superset of vincular patterns that is closed under the inverse operation. As mentioned in Section 4.1, the inverse of a vincular pattern is not well-defined – it is a bivincular, but not a vincular pattern.

Definition 4.3. *A* bivincular pattern π *is a permutation written in two-line notation, where some elements in the top row may be overlined and the bottom row is a vincular pattern as defined in Definition 4.2. An occurrence* $\tau(i_1)\tau(i_2)\dots(i_k)$ *of* π *in a permutation* τ *is an occurrence of*

the corresponding vincular pattern where additionally the following holds: $\tau(i_{j+1}) = (i_j) + 1$ *whenever the top row of* π *contains* $\overline{j(j+1)}$*. Furthermore, if the top row starts with* $\overline{1}$*, an occurrence of* π *in* τ *must start with the smallest entry in* τ*, i.e.,* $\tau(i_1) = 1$*. Similarly, if the top row ends with* $\overline{k}$*, it must hold that* $\tau(i_k) = n$*.*

This definition gets a lot less cumbersome when representing permutations with the help of grids: As remarked earlier, underlined elements in the bottom row are translated into forbidden columns in which no elements may occur in a matching. Similarly, overlined elements in the top row are translated into forbidden rows. For an example, see the right column in Table 3.

Again, in order to specify how many adjacency restrictions are made in the bivincular pattern π, we define – in addition to $\mathsf{cols}(\pi)$ – $\mathsf{rows}(\pi)$ to be the number of shaded rows in the grid corresponding to π.

Mesh patterns

A further generalization of bivincular patterns was given by Brändén and Claesson who introduced mesh patterns in [36] in 2011. Mesh patterns allow further restrictions on the relative positions of the entries in an occurrence of a pattern. Several permutation statistics can be formulated as the number of occurrences of certain mesh patterns [36].

Definition 4.4. *A* mesh pattern *is a pair* $\pi = (\pi, R)$ *where* π *is a permutation of length* k *and* $R \subset [0,k] \times [0,k]$ *is a relation. An occurrence of* π *in a permutation* τ *is an occurrence of the classical pattern* π *fulfilling additional restrictions defined by* R*. That is to say there is a subsequence* $\tau(i_1)\tau(i_2)\dots\tau(i_k)$ *of* τ *that is order-isomorphic to* π *and for which holds:*

$$(x,y) \in R \Longrightarrow \nexists i \in [n] : i_x < i < i_{x+1} \\ \wedge \tau\left(i_{\pi^{-1}(y)}\right) < \tau(i) < \tau\left(i_{\pi^{-1}(y+1)}\right).$$

This definition is again a lot easier to capture when representing permutations as grids. Indeed, the relation R can be translated very easily into the graphical representation of $\pi = (\pi, R)$, by shading the unit square with bottom left corner (x,y) for every $(x,y) \in R$. An occurrence of π in a permutation τ is then a classical occurrence of π in τ such that no elements of τ lie in the shaded regions of the grid.

Again, in order to specify how many adjacency restrictions are made in the mesh pattern π, we define $\mathsf{cells}(\pi)$ to be the number of shaded cells in the corresponding grid. Thus $\mathsf{cells}(\pi, R) := |R|$. For an example where $\pi = 132$ and $R = \{(1,0),(1,2),(2,3),(3,0),(3,1)\}$ see the left column in Table 4.

Boxed mesh patterns

A special case of mesh patterns, so called boxed mesh patterns, was very recently introduced by Avgustinovich, Kitaev and Valyuzhenich in [12].

Definition 4.5. *A* boxed mesh *pattern, or simply boxed pattern, is a mesh pattern $\pi = (\pi, R)$ where π is a permutation of length k and $R = [1, k-1] \times [1, k-1]$. π is then denoted by $\boxed{\pi}$.*

In the grid representing a boxed pattern all but the boundary squares are shaded. For an example, see the middle column of Table 4.

It is straightforward to see that the set of boxed patterns is closed under taking complements, reverses and inverses and that these operations are compatible with pattern containment. Interestingly, it was shown [12] that the statement "π can be matched into τ iff $\boxed{\pi}$ can be matched into τ" is only true if π is one of the following permutations: 1, 12, 21, 132, 213, 231, 312.

Consecutive patterns

Consecutive patterns are a special case of vincular patterns, namely those where *all* entries are underlined. In an occurrence of a consecutive pattern it is thus necessary that all entries are adjacent. Finding an occurrence of a consecutive pattern therefore consists in finding a contiguous subsequence of τ that is order-isomorphic to π. For an example, see the right column of Table 4.

Several well-known enumeration problems for permutations can be formulated in terms of forbidden consecutive patterns; Elizalde and Noy [62] provide examples. Chapter 5 in [108] is devoted to and gives an overview of different methods employed in the literature for the study of consecutive patterns.

4.2 THE POSSIBILITY OF POLYNOMIAL-TIME ALGORITHMS

The results of this Section are summarized in Figure 19 in which the studied problems are partitioned into those that are in P and those that are NP-complete.

NP-completeness

At the 1992 SIAM Discrete Mathematics meeting Herbert Wilf asked whether it is possible to solve the permutation pattern matching problem in polynomial time. The answer is no unless P=NP, as shown by the NP-completeness result of Bose, Buss and Lubiw [31]. This result immediately yields NP-hardness for all generalizations of classical permutation pattern matching. In this section we are going to show

that NP-hardness holds for these problems even in a more restricted case. Indeed,NP-hardness can be shown for instances in which all alternating runs are as short as possible, i.e., for alternating permutations.

Theorem 4.6. *Every* MESH PERMUTATION PATTERN MATCHING *instance* $(\pi, \tau) = ((\pi, R), \tau)$ *can be transformed into an instance* (π', τ') *with* $\pi' = (\pi', R)$ *and the following properties:* (π', τ') *is a yes-instance iff* (π, τ) *is yes-instance,* $|\pi'| = 2|\pi|$, $|\tau'| = 2|\tau|$ *and both* π' *and* τ' *are alternating permutations. This transformation can be done in polynomial time.*

Proof. Let $\pi = \pi_1 \dots \pi_k$ and $\tau = \tau_1 \dots \tau_n$. We define

$$\pi' = (k+1)\ \pi_1\ (k+2)\ \pi_2\ (k+3) \dots (2k)\ \pi_k$$
$$\tau' = (n+1)\ \tau_1\ (n+2)\ \tau_2\ (n+3) \dots (2n)\ \tau_n.$$

Clearly, $|\pi'| = 2|\pi|$, $|\tau'| = 2|\tau|$ and both permutations π' and τ' are alternating. We are now going to show that there is a matching from π into τ iff there is a matching from π' into τ'. Assume that M is a matching from π into τ, i.e., a map from $[k]$ to $[n]$. We extend this map to a map M' from $[2k+1]$ to $[2n+1]$ in the following way:

$$M'(i) = \begin{cases} M(i), & \text{if } i \in [k], \\ \tau(j), \text{ where } M(i-k) = \tau(j+1) & \text{if } i > k. \end{cases}$$

In other words, M' maps $(i+k)$ to the element in τ left of $M(i)$. For example if $M(\pi_3) = \tau_5$ then $\pi_3 \in \pi'$ is matched to $\tau_5 \in \tau'$ and $(k+3) \in \pi'$ is matched to $n+5 \in \tau'$ (which is the element in τ lying directly to the left of τ_5). Observe that the function M' is a matching from π' into τ'.

Now let us assume that M' is a matching from π' into τ'. If we restrict the domain of M' to $[k]$ then we obtain a matching from π into τ. □

Theorem 4.7. PPM *is NP-complete even when both* π *and* τ *are alternating permutations.*

Proof. We apply the transformation in Theorem 4.6 to show NP-hardness. NP-membership holds for this restricted class of input instances as well. □

Corollary 4.8. VINCULAR, BIVINCULAR *and* MESH PPM *are NP-complete even when both* π *and* τ *are alternating permutations.*

Proof. NP-hardness follows from Theorem 4.6 as well as from Theorem 4.7. NP-membership holds since checking whether the additional restrictions imposed by the vincular, bivincular or mesh pattern are fulfilled can clearly be done in polynomial time. □

Polynomial time algorithms

We have seen that polynomial time algorithms are unlikely to exist for PPM and its generalizations. However, this is not the case for the special cases of boxed mesh and consecutive pattern matching.

Theorem 4.9. BOXED MESH PERMUTATION PATTERN MATCHING *can be solved in* $\mathcal{O}(n^3)$ *time.*

Proof. Let π be a boxed pattern of length k and τ a permutation of length n. For every pair (i, j) where $i \in [n]$ and $i + k \leq j \leq n$ check whether there is a matching M of the *boxed* pattern π into τ where the smallest element in π is matched to i and the largest one to j, i.e., $M(1) = i$ and $M(k) = j$.

Checking whether such a matching exists can be done in the following way: From the permutation τ, construct the permutation $\tilde{\tau}$ by deleting all elements that are smaller than i and larger than j. Clearly, the matching that we are looking for must be contained in $\tilde{\tau}$, it could otherwise not be an occurrence of a boxed pattern. Moreover, it has to consist of k *consecutive* elements in $\tilde{\tau}$. Since the positions of the smallest and the largest element are fixed, the positions for all other elements of π are equally determined. Thus there is only one subsequence of τ that could possibly be a matching of π into τ with $M(1) = i$ and $M(k) = j$. Deleting the elements that are too small or too large and checking whether this subsequence actually corresponds to an occurrence of π in τ, i.e., whether it is order-isomorphic to π, can be checked in at most n steps. Note that this subsequence might consist of less than k elements in which case it clearly does not correspond to an occurrence.

In total, there are $(n-k+1) \cdot (n-k+2)/2 = \mathcal{O}(n^2)$ pairs (i, j) that have to be checked which leads to the runtime bound $\mathcal{O}(n^3)$. □

Theorem 4.10. CONSECUTIVE PERMUTATION PATTERN MATCHING *can be solved in* $\mathcal{O}((n-k) \cdot k)$ *time.*

Proof. Let π be a consecutive pattern of length k and τ a permutation of length n. For every $i \in [n-k+1]$ check whether there is a matching of π into τ where the first element of π is mapped to i. Since we are looking for an occurrence of a consecutive pattern, the only possible subsequence of τ then consists of the element i and the following $(k-1)$ elements of τ. Whether this sequence is order-isomorphic to π can be checked in k steps which leads to the runtime bound $\mathcal{O}((n-k) \cdot k)$. □

As was recently shown by Kubica et.al. in [114], this simple result can be improved by an algorithm with runtime $\mathcal{O}(n+k)$.

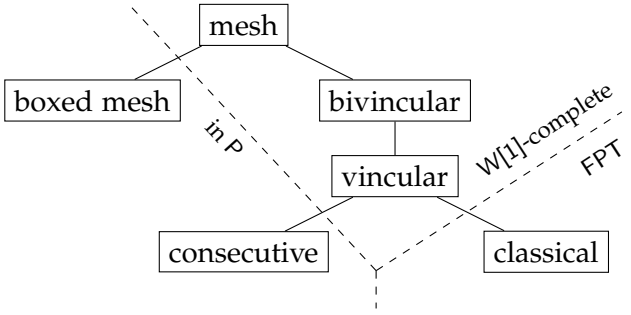


Figure 20: The influence of the pattern length on the computational hardness: parameterized complexity of permutation pattern matching

4.3 THE IMPACT OF THE PATTERN LENGTH

PPM can be solved in $\mathcal{O}(n^k)$ time by exhaustive search, where k is the length of π. This trivial upper bound has been improved first by Albert et al. to $\mathcal{O}(n^{1+2k/3} \cdot \log n)$ [3] and then to $\mathcal{O}(n^{0.47k+o(k)})$ by Ahal and Rabinovich [1]. In a recent breakthrough result, Guillemot and Marx have shown that PPM can be solved by an FPT algorithm [93]. Its runtime is $2^{\mathcal{O}(k^2 \cdot \log k)} \cdot n$. In this section we are going to show that such a result is likely not to be achievable for VINCULAR, BIVINCULAR and MESH PPM. This is done by showing W[1]-hardness with respect to the parameter k. First, we show that MESH PPM and therefore all other problems studied in this paper are contained in W[1].

All results in this section are summarized in Figure 20.

Theorem 4.11. MESH PERMUTATION PATTERN MATCHING *is contained in* W[1].

Proof. For showing membership we encode MESH PPM as a model checking problem of an existential first order formula. W[1]-membership is then a consequence of the fact that the following problem is W[1]-complete [79].

EXISTENTIAL FIRST-ORDER MODEL CHECKING	
Instance:	A structure $\mathcal{A}$ and an existential first-order formula φ
Parameter:	$\lvert\varphi\rvert$
Question:	Is $\mathcal{A}$ a model for φ?

Let $((\pi, R), \tau)$ be a MESH PPM instance. We compute a structure $\mathcal{A} = (A, <, \prec_\tau, E)$, where the domain set $A = \{1, \ldots, n\}$ represents indices in the text. The binary relation $\prec_\tau$ is defined as follows: $x \prec_\tau y$ holds iff $\tau(x) < \tau(y)$. E is a quaternary relation where $E(w, x, y, z)$ is true iff there are no elements in τ that are left of w, right of x, larger than y and smaller than z. Intuitively, w, x, y and z describe a *forbidden* rectangle in the permutation grid of τ which may not contain

any elements of τ. The relations $\prec_\tau$, E and $<$ can be computed in polynomial time. The formula φ we want to check is

$$\varphi = \exists x_1 \dots \exists x_k \quad x_1 < x_2 \ \wedge\ x_2 < x_3 \ \wedge\ \dots \ \wedge\ x_{k-1} < x_k \ \wedge \underbrace{\bigwedge_{\substack{P(i)<P(j) \\ \text{for } i,j\in[k]}} x_i \prec_T x_j}_{\varphi_1} \wedge \underbrace{\bigwedge_{\substack{P(i)>P(j) \\ \text{for } i,j\in[k]}} \neg(x_i \prec_T x_j)}_{\varphi_2} \wedge \underbrace{\bigwedge_{\substack{i,j\in[k] \text{ and} \\ R(i,j) \text{ is true.}}} E(x_i, x_{i+1}, x_j, x_{j+1})}_{\varphi_3}.$$

Observe that the length of φ is in $\mathcal{O}(k^2)$. The two sub-formulæ φ_1 and φ_2 are true exactly when a subsequence $\tau(x_1)\tau(x_2)\dots\tau(x_k)$ of τ can be found such that $\tau(x_i) < \tau(x_j)$ iff $\pi(i) < \pi(j)$. Thus $\varphi_1 \wedge \varphi_2$ is true iff there is a matching of the classical pattern π into τ. The sub-formula φ_3 encodes the relation R and is true iff no elements lie in the forbidden regions of τ, as can be seen by recalling Definition 4.4. Thus φ is true iff $((\pi, R), \tau)$ is a yes-instance of Mesh PPM. □

We now want to prove W[1]-hardness for vincular, bivincular and mesh pattern matching. For this purpose, we introduce Segregated Permutation Pattern Matching, a generalization of PPM. All subsequent hardness theorems use reductions from this problem.

Segregated Permutation Pattern Matching (SPPM)	
Instance:	A permutation τ (the text) of length n, a permutation π (the pattern) of length $k \leq n$ and two positive integers $p \in [k]$, $t \in [n]$.
Parameter:	k
Question:	Is there a matching M of π into τ such that $M(i) \leq t$ iff $i \leq p$?

Example 4.12. Consider the pattern $\pi = 132$ and the text $\tau = 53142$. As shown by the matching $M(2) = 3$, $M(1) = 1$ and $M(3) = 4$, the instance $(\pi, \tau, 2, 3)$ is a yes-instance of the SPPM problem. However, $(\pi, \tau, 2, 4)$ is a NO-instance, since no matching of π into τ can be found where $M(3) > 4$. ⊣

Theorem 4.13. Segregated Permutation Pattern Matching *is* W[1]*-hard with respect to the parameter* k.

Proof. We show W[1]-hardness by giving an fpt-reduction from the W[1]-complete Clique problem (see Section 2.5 in the Preliminaries) to SPPM.

The reduction has three parts. First, we will show that we are able to reduce a Clique instance to a pair (π', τ'), where π' and τ' are two

permutations on multisets, i.e., permutations in which elements may occur more than once. Applying Definition 2.5 to permutations on multisets means that in a matching repeated elements in the pattern have to be mapped to repeated elements in the text. In addition to repeated elements, π' and τ' contain so-called *guard elements*. Their function is explained below. Second, we will show how to get rid of repetitions. The method used in this step has already been used in the NP-completeness proof of PPM provided by Bose, Buss and Lubiw in [31]. Third, we implement the guards by using the segregation property and have thus reduced CLIQUE to SPPM.

Let (G, k) be a CLIQUE instance, where $V = \{v_1, v_2, \ldots, v_l\}$ is the set of vertices and $E = \{e_1, e_2, \ldots, e_m\}$ the set of edges. Both the pattern and the text consist of a single substring coding vertices ($\dot{\pi}$ resp. $\dot{\tau}$) and substrings coding edges ($\bar{\pi}_i$ resp. $\bar{\tau}_i$ for the i-th substring). These substrings are listed one after the other, with *guard elements* placed in between them. These guard elements have the function of separating substrings in a matching: guard elements will have to be mapped to guard elements and substrings embraced by two consecutive guard-elements will also have to be mapped to substrings embraced by two consecutive guard-elements. For the moment, we will simply write brackets to indicate where guard elements are placed. The meaning of these brackets is then the following: a block of elements enclosed by a $\langle$ to the left and a $\rangle$ to the right has to be matched into another block of elements between two such brackets. How the guard-elements are implemented as elements of a permutation is explained at the end of the proof after Claim 2.

We define the pattern to be

$$\begin{aligned}\pi' &:= \langle \dot{\pi} \rangle \langle \bar{\pi}_1 \rangle \langle \bar{\pi}_2 \rangle \langle \ldots \rangle \langle \bar{\pi}_{k(k-1)/2} \rangle \\ &= \langle 123 \ldots k \rangle \langle 12 \rangle \langle 13 \rangle \langle \ldots \rangle \langle 1k \rangle \langle 23 \rangle \langle \ldots \rangle \langle 2k \rangle \langle \ldots \rangle \langle (k-1)k \rangle.\end{aligned}$$

$\dot{\pi}$ corresponds to a list of (indices of) k vertices. The $\bar{\pi}_i$'s represent all possible edges between the k vertices (in lexicographic order).

For the text

$$\tau' := \langle \dot{\tau} \rangle \langle \bar{\tau}_1 \rangle \langle \bar{\tau}_2 \rangle \langle \ldots \rangle \langle \bar{\tau}_m \rangle$$

we proceed similarly. $\dot{\tau}$ is a list of the (indices of the) l vertices of G. The $\bar{\tau}_i$'s represent all edges in G (again in lexicographic order). Let us give an example:

Example 4.14. Let $l = 6$ and $k = 3$. Then the pattern permutation is given by

$$\pi' = \langle 123 \rangle \langle 12 \rangle \langle 13 \rangle \langle 23 \rangle.$$

Consider for instance the graph G with six vertices $v_1, \ldots, v_6$ and edge-set

$$\{\{1,2\}, \{1,6\}, \{2,3\}, \{2,4\}, \{2,5\}, \{3,5\}, \{4,5\}, \{4,6\}\}.$$

represented in Figure 21 (we write $\{i, j\}$ instead of $\{v_i, v_j\}$).

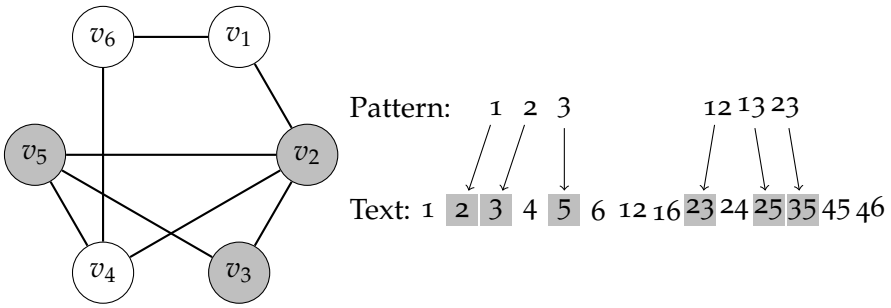


Figure 21: An example for the reduction of an Independent Set instance to a PPM instance

Then the text permutation is given by:

$$\tau' = \langle 123456\rangle\langle 12\rangle\langle 16\rangle\langle 23\rangle\langle 24\rangle\langle 25\rangle\langle 35\rangle\langle 45\rangle\langle 46\rangle.$$

⊣

Claim 1. *A clique of size k can be found in G iff there is a simultaneous matching of $\dot{\pi}$ into $\dot{\tau}$ and of every $\bar{\pi}_i$ into some $\bar{\tau}_j$.*

Example 4.14 (continuation)**.** In our example $\{v_2, v_3, v_5\}$ is a clique of size three. Indeed, the pattern π' can be matched into τ' as can be seen by matching the elements $1, 2$ and 3 onto $2, 3$ and 5 respectively. See again Figure 21 where the involved vertices respectively elements of the text permutation have been marked in gray. ⊣

Proof of Claim 1. A matching of $\dot{\pi}$ into $\dot{\tau}$ corresponds to a selection of k vertices amongst the l vertices of G. If it is possible to additionally match every one of the $\bar{\pi}$'s into a $\bar{\tau}$ this means that all possible edges between the selected vertices appear in G. This is because τ' only contains pairs of indices that correspond to edges appearing in the graph. The selected k vertices thus form a clique in G. Conversely, if for every possible matching of $\dot{\pi}$ into $\dot{\tau}$ defined by a monotone map $M : [k] \to [l]$ some $\bar{\pi}_i = xy$ cannot be matched into τ', this means that $\{M(x), M(y)\}$ does not appear as an edge in G. Thus, for every selection of k vertices there will always be at least one pair of vertices that are not connected by an edge and therefore there is no clique of size k in G. □

In order to get rid of repeated elements, we identify every variable with a real interval: 1 corresponds to the interval $[1, 1.9]$, 2 to $[2, 2.9]$ and so on until finally k corresponds to $[k, k + 0.9]$ (resp. l to $[l, l + 0.9]$). In $\dot{\pi}$ and $\dot{\tau}$ we shall therefore replace every element j by the pair of elements $(j + 0.9, j)$ (in this order). The occurrences of j in the $\bar{\pi}_i$'s (resp. $\bar{\tau}_i$'s) shall then successively be replaced by real numbers in the interval $[j, j + 0.9]$. For every j, these values are chosen one after the other (from left to right), always picking a real number that is larger than all the previously chosen ones in the interval $[j, j + 0.9]$.

Observe the following: The obtained sequence is not a permutation in the classical sense since it consists of real numbers. However, by replacing the smallest number by 1, the second smallest by 2 and so on, we do obtain an ordinary permutation. This defines π and τ (except for the guard elements).

Example 4.14 (continuation)**.** Getting rid of repetitions in the pattern of the above example could for instance be done in the following way:

$$\pi = \langle 1.9\ 1\ 2.9\ 2\ 3.9\ 3\rangle\langle 1.1\ 2.1\rangle\langle 1.2\ 3.1\rangle\langle 2.2\ 3.2\rangle$$

This permutation of real numbers is order-isomorphic to the following ordinary permutation:

$$\pi = \langle 4\,1\,8\,5\,12\,9\rangle\langle 2\,6\rangle\langle 3\,10\rangle\langle 7\,11\rangle.$$

⊣

Claim 2. *π can be matched into τ iff π' can be matched into τ'.*

Proof of Claim 2. Suppose that π' can be matched into τ'. When matching π into τ, we have to make sure that elements in π that were copies of some repeated element in π' may still be mapped to elements in τ that were copies themselves in τ'. Indeed this is possible since we have chosen the real numbers replacing repeated elements in increasing order. If i in π' was matched to j in τ', then the pair $(i+0.9, i)$ in π may be matched to the pair $(j+0.9, j)$ in τ and the increasing sequence of elements in the interval $[i, i+0.9]$ may be matched into the increasing sequence of elements in the interval $[j, j+0.9]$.

Now suppose that π can be matched into τ. In order to prove that this implies that π' can be matched into τ', we merely need to show that elements in π that were copies of some repeated element in π' have to be mapped to elements in τ that were copies themselves in τ'. Then returning to repeated elements clearly preserves the matching. Firstly, it is clear that a pair of consecutive elements $i+0.9$ and i in $\dot{\pi}$ has to be matched to some pair of consecutive elements $j+0.9$ and j in $\dot{\tau}$, since j is the only element smaller than $j+0.9$ and appearing to its right. Thus intervals are matched to intervals. Secondly, an element x in π for which it holds that $i < x < i+0.9$ must be matched to an element y in τ for which it holds that $j < y < j+0.9$. Thus copies of an element are still matched to copies of some other element.

Finally, replacing real numbers by integers does not change the permutations in any relevant way. □

It remains to implement the guards in order to ensure that substrings are matched to corresponding substrings. Let $\pi_{\max}$ and $\tau_{\max}$ denote the largest integer that is contained in π respectively τ at this point. We now replace all guards with integers larger than $\pi_{\max}$ respectively $\tau_{\max}$ and will choose the *segregating elements* p and t such that guards and "original" pattern/text elements are separated. We

insert the guard elements in the designated positions (previously marked by $\langle$ and $\rangle$) in the following order: $\pi_{\max}+2$ (instead of the first $\langle$), $\pi_{\max}+1$ (instead of the first $\rangle$), $\pi_{\max}+4$ (instead of the second $\langle$), $\pi_{\max}+3$ (instead of the second $\rangle$), $\ldots$, $\pi_{\max}+2i$ (instead of the i-th $\langle$), $\pi_{\max}+2i-1$ (instead of the i-th $\rangle$),..., and so on until we reach the last guard-position. The guard elements are inserted in this specific order to ensure that two neighbouring guard elements $\langle$ and $\rangle$ in π have to be mapped to two neighbouring guard elements $\langle$ and $\rangle$ in τ. We proceed analogously in τ. To ensure that guards in π are matched to guards in τ and pattern elements of π are matched to text elements in τ, we set p to $\pi_{\max}$ and t to $\tau_{\max}$.

This finally yields that (G,k) is a yes-instance of Clique iff (π,τ,p,t) is a yes-instance of SPPM. It can easily be verified that this reduction can be done in fpt-time. □

As can easily be seen, the reduction performed in the proof of Theorem 4.13 can be done in polynomial time. Thus this proof immediately yields NP-hardness for SPPM.

Now, that we have obtained this result, we are able to show W[1]-hardness for PPM with vincular, bivincular and mesh patterns. As before, the parameter is the length of the pattern.

Theorem 4.15. Vincular Permutation Pattern Matching *is* W[1]*-complete with respect to k. This holds even when restricting the problem to instances* (π,τ) *with* $\mathsf{cols}(\pi)=1$.

Proof. We reduce from Segregated PPM. Let (π,τ,p,t) be an SPPM instance. The Vincular PPM instance (π',τ') constructed from (π,τ) will have have an additional element in π' and an additional element in τ'. The new element in π, denoted by p', is $p+0.5$, i.e., p' is larger than p but smaller than $p+1$. Analogously, $t'=t+0.5$ is the new element in τ. We define $\pi'=\underline{p'}\pi$ and $\tau'=t'\,\tau$. In order to obtain a permutation π on $[k+1]$ and τ on $[n+1]$, we simply need to relabel the respective elements order-isomorphically. In every matching of π' into τ' the element p' has to be mapped to t'. Consequently, all elements larger than p' in π' have to be mapped to elements larger than t' in τ' and all elements smaller than p' have to be mapped to elements smaller than t'. This implies that (π,τ,p,t) is a Segregated PPM yes-instance iff (π',τ') is a Vincular PPM yes-instance. This reduction is done in linear time which proves W[1]-hardness of Vincular PPM. Membership follows from Theorem 4.11. □

Theorem 4.16. Bivincular Permutation Pattern Matching *is* W[1]*-complete with respect to k. This holds even when restricting the problem to instances* (π,τ) *with* $\mathsf{rows}(\pi)=1$.

Proof. As in the previous proof we reduce from Segregated PPM. Let (π,τ,p,t) be an SPPM instance. Identically to the previous proof, we define $p'=p+0.5$ and $t'=t+0.5$. The Bivincular PPM instance

consists of a permutation π' with elements in $[k+1] \cup \{p'\}$ and τ', a permutation on $[n+1] \cup \{t'\}$. The permutation π', written in two-line notation, is given as follows:

$$\pi' = \begin{pmatrix} 1 & 2 & 3 & \dots & p' & \dots & \overline{(k+1)} \\ p' & (k+1) & \pi(1) & & \dots & & \pi(k) \end{pmatrix}$$

and τ' is defined to be $t'(n+1)\tau$. In order to obtain permutations on $[k+2]$ respectively $[n+2]$ we again relabel the elements order-isomorphically.

In any matching of π' into τ' the element $(k+1)$ has to be mapped to $(n+1)$ and therefore p' has to be mapped to t'. Thus all elements larger than p' in π' have to be mapped to elements larger than t' in τ' and all elements smaller than p' have to be mapped to elements smaller than t'. This implies that (π, τ, p, t) is a Segregated PPM yes-instance iff (π', τ') is a Bivincular PPM yes-instance. Since this reduction can again be done in linear time, Bivincular PPM is W[1]-hard. Membership follows again from Theorem 4.11. □

Theorem 4.17. Mesh Permutation Pattern Matching *is* W[1]*-complete with respect to k. This holds even if* cells$(\pi) = 1$.

Proof. Let (π, τ, p, t) be a Segregated PPM instance. As before, we define $p' = p + 0.5$ and $t' = t + 0.5$. The Mesh PPM instance consists of a permutation π' with elements in $[k] \cup \{p'\}$ and τ', a permutation on $[n+1] \cup \{t'\}$. Again, permutations on $[k+1]$ respectively $[n+2]$ can be obtained by relabelling the elements order-isomorphically. We define $\pi' = p'\ \pi$ and $\tau' = t'\ (n+1)\ \tau$. Furthermore, we define: $R = \{(0, (k+1))\}$. This means that for every matching M of π' into τ' the following must hold: to the left of $M(p')$ in τ', there are no elements larger than $M(k)$. However, it surely holds that $M(k) \leq (n+1)$. Consequently, p' has to be mapped to t'. This implies that (π, τ, p, t) is a Segregated PPM yes-instance iff (π', τ') is a Mesh PPM yes-instance. Since this reduction can again be done in linear time, Mesh PPM is W[1]-hard. Membership follows from Theorem 4.11. □

These hardness results show that we cannot hope for a fixed-parameter tractable algorithm for Vincular/Bivincular/Mesh Permutation Pattern Matching.

4.4 SUMMARY OF THE RESULTS

Let us briefly sum up the contributions of this chapter. We analysed several versions of Permutation Pattern Matching, based on various types of permutation patterns that were recently introduced in the literature. This allows us to draw a picture of the computational landscape of generalized PPM:

We performed a classical complexity analysis and could show that the only cases where $\mathcal{C}$ PPM can be solved in polynomial time are Consecutive PPM and Boxed Mesh PPM. All other versions of PPM are NP-complete, even if the input instances are reduced to the case where both the pattern and the text are alternating permutations. This partition of the $\mathcal{C}$ PPM problems is represented in Figure 19 on page 82.

We offered a more fine-grained complexity analysis of the NP-hard versions of $\mathcal{C}$ PPM by employing the framework of parameterized complexity. We were able to show that Mesh, Bivincular and Vincular PPM are W[1]-complete with respect to the length of the pattern. This is in strong contrast to the case of classical PPM, that is fpt with respect to this parameter [93]. This partition of the $\mathcal{C}$ PPM problems that are NP-complete is represented in Figure 20 on page 89.

5

A PERMUTATION CLASS ENUMERATED BY THE CENTRAL BINOMIAL COEFFICIENTS

This short chapter is concerned with the enumeration of permutations avoiding the patterns 2431, 4231, 1432 and 4132 simultaneously. This permutation class arises in the context of *single-peaked elections* that are introduced and studied in Chapter 10 of this thesis. To be more precise, we can show that permutations of length n that avoid the four patterns above are in bijective correspondence with elections $\{V_1, V_2\}$ on a candidate set $C = \{c_1, c_2, \ldots, c_n\}$ that are single-peaked and where V_1 is the vote $c_1 > c_2 > \ldots > c_n$. This allows us to derive an upper bound on the number of single-peaked elections with an arbitrary number of votes. For definitions, the precise statements of our results and their proofs, we refer to Chapter 10 and in particular to Theorem 10.17 therein.

The study of this permutation class leads to a surprisingly simple exact enumeration formula, namely the central binomial coefficients:

Theorem 5.1. *For all $n \in \mathbb{N}$ the following holds:*

$$S_n(2431, 4231, 1432, 4132) = \binom{2 \cdot (n-1)}{n-1} = b_{n-1}.$$

In the following, let us denote the set of permutations avoiding the forbidden patterns 2431, 4231, 1432 and 4132 by $\mathcal{S}$ and the subset of $\mathcal{S}$ consisting of permutations of length n by $\mathcal{S}_n$.

We shall prove this theorem by identifying every permutation in $\mathcal{S}_n$ by a certain *code word* of length $(n-1)$. Later on we count these code words and show that they are precisely enumerated by the central binomial coefficients. This is done by identifying the non-initial segments of code words with certain lattice paths.

The set of code words corresponding to permutations in $\mathcal{S}$ is defined in the following way:

Definition 5.2. *$\mathcal{W} \subseteq (\{2, 3, \ldots\} \cup \{B, E\})^{\mathbb{N}}$ is the set of words $W = w_1 w_2 \ldots$ fulfilling the following conditions:*

1. *$w_i \in \{B, E\} \cup \{j \in \mathbb{N} : 2 \leq j \leq i\}$,*
2. *if $w_i \neq B, E$ then $w_{i+1} \neq B, E$,*
3. *if $w_i \neq B, E$ then $w_{i+1} \geq w_i$.*

We shall denote by $\mathcal{W}_n$ the set of all code words of length n and by W_n its cardinality.

In a code word, the letters B and E stand for *beginning* and *end*, respectively. Integers in a code word represent positions in the corresponding permutation. The precise meaning of this will become clear in the proof of the following Lemma 5.4.

Form the definition above it follows that code words contain an *initial segment* of letters B and E (conditions 1 and 3). This initial segment is followed by a non-decreasing sequence of not too large integers (conditions 1 and 2).

Example 5.3. The six code words of length two are: BB, BE, EB, EE, $B2$, $E2$. For an example of a code word of length 8, see Figure 22 on page 101. ⊣

Lemma 5.4. *There is a bijection between code words in $\mathcal{W}_{n-1}$ and permutations in $\mathcal{S}_n$.*

The approach used to prove the lemma above can be seen as a *generating trees* approach. This is a technique introduced in the study of Baxter permutations and systematized by Julian West [157] in the context of pattern avoidance. It is frequently used for the enumeration of permutation classes, see e.g. [32]. However, we decided not to introduce the generating trees terminology here: The following arguments and especially the enumeration of code words can be done in a bijective manner and do not require the use of rewriting rules and of the associated generating functions.

Proof. Given a permutation π in $\mathcal{S}_n$, we can associate with it the sequence $\pi|_{[1]}, \pi|_{[2]}, \dots \pi|_{[n-1]}, \pi|_{[n]}$, where $\pi|_{[i]}$ is the permutation π restricted to the elements in $[i]$–as defined in Section 2.2. Note that if π is in $\mathcal{S}$, all $\pi|_{[i]}$ are as well. Thus, a permutation π in $\mathcal{S}_n$ is created by inserting the element n in some *allowed position* in the permutation $\pi|_{[n-1]}$. In the following, we will use this observation in order to identify every permutation $\pi|_{[i]}$, $i \in [n]$, with an element of the alphabet $\{2, 3, \dots\} \cup \{B, E\}$ that encodes where the largest element of $\pi|_{[i]}$ was inserted in $\pi|_{[i-1]}$. This will be done in such a way that the resulting word bijectively encodes the permutation π.

For this purpose, let us consider a permutation π of length $n-1$ that lies in $\mathcal{S}$. Where are we allowed to insert the element n without creating one of the forbidden patterns?

- If π contains a 132-pattern on the elements abc, then the element n may not be inserted to the left of a since this would create a 4132-pattern and not between a and b since this would create a 1432-pattern. In other words, the element n must be inserted somewhere to the right of b.

- Similarly, if π contains a 231-pattern on the elements abc, the element n must be inserted somewhere to the right of b. Otherwise a 4231- or a 2431-pattern would be created.

Thus the position of the rightmost element b that plays the role of the 3 in a 132- or a 231-pattern in π tells us where the element n may be inserted without creating any of the forbidden patterns. Indeed, if the element b is at the i-th position in π, the element n can be inserted at any of the positions $i+1$ up to n.

Example 5.5. In the permutation $\pi = 2413$ the elements 241 form a 231-pattern and the elements 243 form a 132-pattern. In both patterns the element 4 plays the role of the 3 and thus all positions to the right of 4 are allowed. Indeed, we can insert 5 at the third, fourth and fifth position in π and obtain the following permutations in $\mathcal{S}_5$: 24513, 24153 and 24135. ⊣

Since this position will be the key to describing a permutation in $\mathcal{S}$ by a code word in a unique way, let us set up the following notation:

$$p(\pi) := \max\{i \in [n] : \exists j < i < k \text{ s.t. } \pi(j) < \pi(i) \text{ and } \pi(k) < \pi(i)\},$$

where we set $\max(\emptyset) := 0$.

Now, given a permutation π and its associated sequence of permutations $\pi|_{[2]}, \ldots, \pi|_{[n]}$, we identify it with a word $w = w_1 \ldots w_{n-1}$ on the alphabet $\{2, 3, \ldots\} \cup \{B, E\}$ in the following way. For all $i \in [n-1]$ we define:

$$w_i := \begin{cases} p(\pi|_{[i+1]}) & \text{if } p(\pi|_{[i+1]}) \neq 0 \\ B & \text{if } p(\pi|_{[i+1]}) = 0 \text{ and} \\ & (i+1) \text{ lies at the beginning of } \pi|_{[i+1]} \\ E & \text{if } p(\pi|_{[i+1]}) = 0 \text{ and} \\ & (i+1) \text{ lies at the end of } \pi|_{[i+1]} \end{cases}$$

Example 5.6. Consider the permutation $\pi = 245178396$ in $\mathcal{S}_9$. As shown in the table on the left hand side of Figure 22 on page 101, it can be identified with the word $w = BE233568$ in W_8. ⊣

Let us note that two distinct permutations can never be identified with the same code word. Indeed, if the permutation $\pi|_{[i]}$ is given and if we know that $p(\pi|_{[i+1]})$ is equal to some element in $\{2, 3, \ldots\} \cup \{B, E\}$, there is only a single possibility for placing the element $(i+1)$ in $\pi|_{[i]}$ in order to obtain $\pi|_{[i+1]}$. Thus the map that sends an element in $\mathcal{S}_n$ to a word of length $n-1$ is injective.

Now we need to show that this map actually associates permutations in $\mathcal{S}$ with *code words* as described in Definition 5.2. The first condition of code words is clearly fulfilled: First, $w_1 = B$ for $\pi|_{[2]} = 21$ and $w_1 = E$ for $\pi|_{[2]} = 12$. Second, if $w_i \neq B, E$ we have $2 \leq w_i \leq i$ since $2 \leq p(\pi|_{[i+1]}) \leq i$. In order to show that the second and third condition are also fulfilled we need to identify at which positions new 132- or 231-patterns can be created when the element i is inserted in a permutation in C_{i-1}.

- If the element i is inserted at some position j with $1 < j < i$ then it automatically creates a 132- or 231-pattern in which the element i plays the role of the 3. Thus, irrespective of the value of w_{i-1}, we have $p(\pi|_{[i]}) = j$. Since i can only be inserted into $\pi|_{[i-1]}$ to the right of the $p(\pi|_{[i-1]}$-th position, we always have that $p(\pi|_{[i]}) = w_i > p(\pi|_{[i-1]})$.
- If the element i is inserted at the beginning or at the end of the permutation, no new 132- or 231-patterns are created. Thus if there were no occurrences of such patterns before, we have that w_{i+1} is equal to B or E, depending on whether i is inserted at the beginning or at the end.

  However, if $p(\pi|_{[i-1]}) \neq 0$, that is if w_{i-1} is neither B nor E, we have $p(\pi|_{[i]}) = p(\pi|_{[i-1]}) + 1$ if the element i is inserted at the beginning and $p(\pi|_{[i]}) = p(\pi|_{[i-1]})$ if it is inserted at the end. Together with the fact that $w_i \neq B, E$ whenever i is not inserted at the first or last position, this implies that the second condition of code words is fulfilled. Furthermore, we also have in this case that $w_i \geq w_{i-1}$ whenever w_{i-1} is an integer and thus the third condition of code words is also fulfilled.

Finally, it remains to show that this map is surjective. Given a code word w of length $n-1$ it is straightforward how to construct the corresponding permutation π in a recursive manner. Start with the element 1 and then place the elements 2 up to n one after the other at the positions prescribed by w: if $w_i = B$ the element $i+1$ is placed at the beginning of the permutation, if $w_i = E$ it is placed at the end of the permutation and if $w_i = j$ for some integer $2 \leq j \leq i$ it is placed at the j-th position of the permutation. That the resulting permutation π is an element of $\mathcal{S}$ and avoids all forbidden patterns follows from the discussion at the beginning of this proof. □

Lemma 5.7. *For all $n \in \mathbb{N}$ the following statement holds:*

$$W_n = \binom{2n}{n}.$$

Proof. First of all, let us note that

$$W_n = \sum_{i=1}^{n} 2^i \cdot w(n-i, i)$$

where i corresponds to the length of the initial segment consisting of letters B and E and where $w(n-i, i)$ is the number of words $w = w_1 \dots w_{n-i}$ of length $(n-i)$ where $2 \leq w_j \leq j+i$ and $w_j \leq w_{j+1}$. These numbers $w(n-i, i)$ can be easily computed by viewing the words that they count as certain lattice paths.

First, let us remark that we can decrease every integer in such a word by 2 and counts the number of words of length $(n-i)$ where

i	$\pi\vert_{[i+1]}$	w_i
1	$\underline{2}1$	B
2	$21\underline{3}$	E
3	$2\underline{4}13$	2
4	$24\underline{5}13$	3
5	245136	3
6	$2451\underline{7}36$	5
7	$24517\underline{8}36$	6
8	$245178\underline{9}6$	8

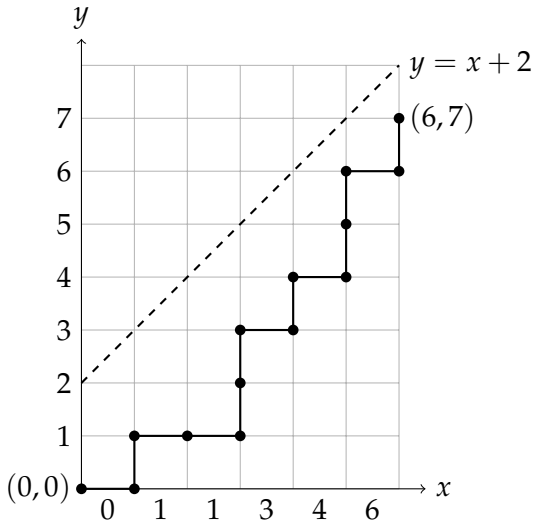


Figure 22: The permutation $\pi = 245178396$ in $\mathcal{S}_9$ can be identified with the word $w = BE233568$ in W_8 as shown in the table on the left hand side. The sequence of integers that remains when the initial segment BE has been removed from w and when all integers are decreased by 2 is 011346. It can be represented as the lattice path shown on the right-hand side that starts at $(0,0)$ and ends at $(6,7)$ without ever touching the dashed line.

$0 \leq w_j \leq j + i - 2$ and $w_j \leq w_{j+1}$. Now we can represent each such word by a lattice path starting at $(0,0)$, ending at $(n-i, n-1)$ and consisting of unit steps to the East or the North as follows: If $w_j = k$ for some integer k, the j-th step to the East is at height k, i.e., we take a step from $(j-1,k)$ to (j,k). Since it holds that $w_j \leq w_{j+1}$, the remaining gaps can simply be filled in with North-steps and there is no need for South-steps. The condition that $w_j \leq j + i - 2$ is easily translated into the condition that the lattice path must always stay below and never touch the line $y = x + i$.

It is now easy to convince oneself that this correspondence between words of length $(n-i)$ where $2 \leq w_j \leq j + i$ and $w_j \leq w_{j+1}$ and North-East-lattice paths from $(0,0)$ to $(n-i, n-1)$ that never touch the line $y = x + i$ is one-to-one.

Example 5.8. For an example of this correspondence, see the path corresponding to 011346 that is represented on the right-hand side of Figure 22. ⊣

Such lattice paths have been widely studied in the literature and it is shown, for instance in the first chapter of Mohanty's monograph on lattice path counting [125], that their number is:

$$w(n-i, i) = \binom{2n-i-1}{n-1} \cdot \frac{i}{n}$$

Since it is so simple and beautiful, let us briefly repeat the argument of the bijective proof here. It is known as Andrés *reflection principle*. First, let us remark that

$$w(n-i,i) = \binom{2n-i-1}{n-1} - r(n-i,i), \tag{20}$$

where the binomial coefficient counts all North-East-lattice paths from $(0,0)$ to $(n-i,n-1)$ and $r(n-i,i)$ is the number of such paths that do touch the line $y = x+i$ at some point. Given such a "bad" path from $(0,0)$ to $(n-i,n-1)$, we denote by X the point where it touches the forbidden line for the first time. From this path we now construct a new path as follows: The part to the left of X is reflected about the line $y = x+i$, thus turning North-steps into East-steps and vice-versa; the part to the right of X stays the same. The new path is then a path from $(-i,i)$ to $(n-i,n-1)$ and this transformation is a one-to-one correspondence to all such paths. Therefore, we have

$$w(n-i,i) = \binom{2n-i-1}{n-1} - \binom{2n-i-1}{n},$$

and thus the enumeration formula stated in Equation (20). Finally, this leads to:

$$\begin{aligned} W_n &= \sum_{i=1}^{n} 2^i \cdot \binom{2n-i-1}{n-i} \cdot \frac{i}{n} \\ &= \frac{1}{n} \cdot \sum_{j=0}^{n-1} 2^{n-j} \cdot \binom{n+j-1}{j} \cdot (n-j) \end{aligned}$$

In order to prove that $W_n = \binom{2n}{n}$ let us first remark that a more general statement holds. It holds that

$$\sum_{j=0}^{n-1} 2^{m-j} \cdot \binom{m+j-1}{j} \cdot (m-j) = n \cdot 2^{m-n+1} \binom{m+n-1}{n} \tag{21}$$

for all $m,n \in \mathbb{N}$ as can be showed very easily by induction over n and every fixed m. The induction start for $n = 1$ is trivially true since the left hand side of Equation (21) is equal to $2^m \cdot m$ and the right hand side to $2^m \cdot \binom{m}{1}$. For the induction step, let us assume that Equation (21) has been proven for n. For $n+1$, we have:

$$\begin{aligned} &\sum_{j=0}^{n} 2^{m-j} \cdot \binom{m+j-1}{j} \cdot (m-j) = \\ &= n \cdot 2^{m-n+1} \binom{m+n-1}{n} + 2^{m-n} \binom{m+n-1}{n} \cdot (m-n) \\ &= 2^{m-n} \binom{m+n-1}{n} \cdot (2n+m-n) \\ &= 2^{m-n} \binom{m+n}{n+1} \cdot \frac{n+1}{m+n} \cdot (m+n), \end{aligned}$$

which proves Equation (21) for $n+1$.

Setting $m = n$ in Equation (21) leads to the central binomial coefficients and finishes the proof. □

Let us very briefly sum up the results of this Chapter: We considered a permutation class that arises in the context of domain restrictions in preference profiles (see Chapter 10) and showed that its elements are enumerated by the central binomial coefficients. The proof used bijective methods and establishes a link to certain lattice paths.

6

LOG-CONCAVITY, LONGEST INCREASING SUBSEQUENCES AND INVOLUTIONS

This chapter in based on joint work with Miklós Bóna that has recently been submitted to a journal. A preprint of the article can be found on `arXiv.org` [30].

At the heart of this chapter lie two important concepts in combinatorics: First, the notion of log-concave sequences and second, that of longest increasing subsequences in permutations. Throughout this chapter we will denote by $\ell(\sigma)$ the length of the longest increasing subsequence in the permutation σ. Moreover, we will denote by $L_{n,k}$ the set of all permutations of length n that satisfy $\ell(\sigma) = k$. The cardinality of $L_{n,k}$ will be denoted by $\ell_{n,k}$.

The algorithmic question of determining the length of the longest increasing subsequence can be answered in $O(n\log(n))$-time for sequences in general [139] and in $O(n\log(\log(n)))$-time for permutations of length n [48]. The distribution of the parameter $\ell(\sigma)$ has been the subject of vigorous study for over 60 years. See [14] for strongest results on this subject, and see [6] for a history of the problem. However, it is still not known whether the sequence $\ell_{n,1}, \ell_{n,2}, \cdots, \ell_{n,n}$ is log-concave for each fixed n.

Supported by numerical evidence, we state our conjecture that this sequence indeed is log-concave for every fixed value of n (Section 6.1). Then, in Sections 6.2 and 6.3 we proceed to prove the conjecture in some special cases, that is, for certain subsets of permutations of length n, as opposed to the entire set of $n!$ permutations of length n. One tool in our proofs will be a technique that allows us to turn injections between sets of involutions into injections between sets of permutations. In addition, we will use several consequences of the well-known Robinson-Schensted correspondence. The results of this chapter are summed up in Section 6.4.

6.1 THE CONJECTURE AND A FIRST RESULT

Supported by the data that we computed for permutations of length up to $n = 15$, we conjecture the following:

Conjecture 6.1. *For every positive integer n the sequence $\ell_{n,k}$ where $1 \leq k \leq n$ is log-concave.*

Let $I_{n,k}$ denote the set of all involutions of length n with longest increasing subsequence of length k. The cardinality of $I_{n,k}$ will be denoted by $i_{n,k}$.

Theorem 6.2. *For every positive integer n the following holds: If the sequence $(i_{n,k})_{1\leq k\leq n}$ is log-concave, then so is the sequence $(\ell_{n,k})_{1\leq k\leq n}$.*

Proof. In order to show that the sequence $(\ell_{n,k})$ is log-concave it would suffice to find an injection from $L_{n,k-1} \times L_{n,k+1}$ to $L_{n,k} \times L_{n,k}$ for all $n \geq 1$ and $2 \leq k \leq n-1$.

Assume that the statement of log-concavity is true for involutions. Then there is an injection $f_{n,k}$ from $I_{n,k-1} \times I_{n,k+1}$ to $I_{n,k} \times I_{n,k}$ for all $n \geq 1$ and $2 \leq k \leq n-1$. Now let $\pi_1 \in L_{n,k-1}$ and $\pi_2 \in L_{n,k-2}$. Then, they correspond to the pairs (P_1, Q_1) and (P_2, Q_2) of Standard Young Tableaux. Define $F(\pi_1, \pi_2) = (\sigma_1, \sigma_2)$, where σ_1 is the permutation in $L_{n,k}$ whose pair of SYT is $f(P_1, P_2)$, and σ_2 is the permutation in $L_{n,k}$ whose pair of SYT is $f(Q_1, Q_2)$. Then F is injective since f is injective. □

In the following, we will apply this result to specific classes of permutations and show that our conjecture indeed holds there.

6.2 A CLASS OF PERMUTATIONS FOR WHICH THE CONJECTURE HOLDS

First we show that the conjecture holds for permutations whose corresponding SYT are *hooks*. This implies that the conjecture is true for the class of *skew-merged involutions*. Then we will define a generalization of hooks, introducing (l, m)-protected SYT. We will see that the conjecture holds for these much larger classes of SYT respectively permutations as well, for every pair (l, m) of non-negative integers.

Hook-shaped SYT and skew-merged involutions

Definition 6.3. *We call a SYT a* hook *if it consists of exactly one row and one column.*

In the following, let $H_{n,k}$ denote the set of all hooks of size n with first row of length k. The cardinality of $H_{n,k}$ will be denoted by $h_{n,k}$.

Theorem 6.4. *For every positive integer n the sequence $(h_{n,k})_{1\leq k\leq n}$ is log-concave.*

Proof. First, let us remark than it is straightforward to determine the numbers $h_{n,k}$. Indeed, in order to create a hook with n boxes with k of them in the first row, we simply need to choose the $(n-1)$ elements larger than 1 that will be in the first row. The remaining elements are then placed in increasing order in the first column. Thus

$$h_{n,k} = \binom{n-1}{k-1}$$

and it is immediately clear that the sequence $(h_{n,k})_{1\leq k\leq n}$ is log-concave. In the following we will provide a combinatorial explanation for this fact.

To give a combinatorial proof of the log-concavity of $(h_{n,k})_{1\leq k\leq n}$ we follow the same procedure as in [26]: The sequence $(h_{n,k})_{1\leq k\leq n}$ is log-concave if and only if $h_{n,k}\cdot h_{n,l}\leq h_{n,k+1}\cdot h_{n,l-1}$ for all $n\geq 1$ and $1\leq k\leq l-2\leq n-2$. We shall therefore inductively construct injections $\varphi_{n,k,l}$ from $H_{n,k}\times H_{n,l}$ to $H_{n,k+1}\times H_{n,l-1}$ for all $n\geq 3$ and $1\leq k\leq l-2\leq n-2$. First we construct the injections $\varphi_{n,k,l}$ for the smallest meaningful value of n, which is $n=3$. Since there is only a single element in $H_{3,1}\times H_{3,3}$, we shall also describe the functions $\varphi_{4,k,l}$ for all admissible values of k and l. Next, for the induction step, we use the assumption that the maps $\varphi_{n-1,k,l}$ exist for all admissible values of k and l to construct the maps $\varphi_{n,k,k+2}$. It is not necessary to construct the maps $\varphi_{n,k,l}$ for $k<l-2$ since the existence of the injective maps $\varphi_{n,k,k+2}$ implies the log-concavity of the sequence $(h_{n,k})_{1\leq k\leq n}$ which implies the existence of the maps $\varphi_{n,k,l}$ for $1\leq k<l-2\leq n-2$.

First note that the element n in a hook of size n will always lie in the last box of the first column or in the last box of the first row. This allows us to define the *type* of a hook-shaped Standard Young Tableau: it is of type $\downarrow$ if the element lies in the first column and of type $\rightarrow$ otherwise. Since we are dealing with pairs of SYT there are four possible types of pairs that can occur: $\downarrow\downarrow$, $\downarrow\rightarrow$, $\rightarrow\downarrow$ and $\rightarrow\rightarrow$. The maps $\varphi_{n,k,l}$ that we are going to construct are such that the type is preserved: the type of the image $\varphi_{n,k,l}(T_1,T_2)$ is the same as the type of (T_1,T_2). This will allow us to prove the injectivity of these maps.

Now let us start with the base step at $n=3$ of our induction proof. The map $\varphi_{3,1,3}$ is described in the top part of Figure 23. Since $\varphi_{3,1,3}$ is defined on a single element and this does not allow us to see what the functions $\varphi_{n,k,l}$ actually do, we have also included the description of the functions $\varphi_{4,k,l}$ for $(k,l)=(1,3)$, $(1,4)$ and $(2,4)$ in the bottom part of Figure 23.

Let us turn to the induction step and assume that the injective functions $\varphi_{n-1,k,l}$ for $1\leq k\leq l-2\leq n-3$ have already been constructed. The definition of the function $\varphi_{n,k,k+2}$ depends on the type of the pair of Tableaux it is applied to and we have two different rules:

1. Type $\downarrow\rightarrow$:

   This is the easy case: For a pair $(T_1,T_2)\in H_{n,k}\times H_{n,k+2}$ of type $\downarrow\rightarrow$ we can take the element n in T_1 and move it to the end of the first row in T_1 in order to obtain a Tableau U_1 where the first row has length $k+1$. Similarly, in T_2 we can take the element n and move it to the end of the first column in order to obtain a Tableau U_2 where the first row has length $k+1$. Now the type of (U_1,U_2) is $\rightarrow\downarrow$, so we define $\varphi_{n,k,l}(T_1,T_2)$ to be (U_2,U_1). For an example of the map $\varphi_{n,k,k+2}$ in this case for $n=5$, see Figure 24.

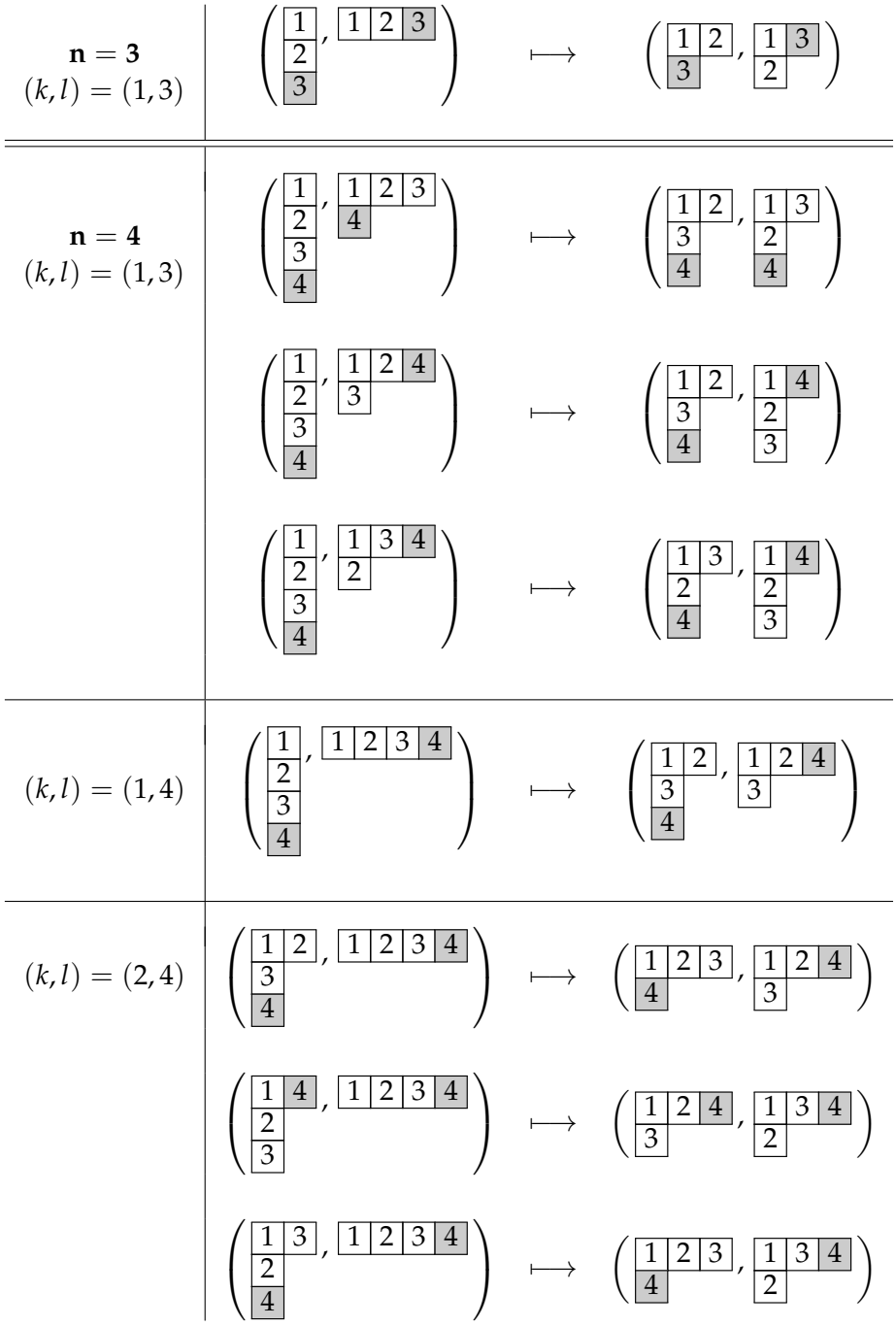


Figure 23: The base step of the induction in the proof of Theorem 6.4: the maps $\varphi_{n,k,l}$ for $n = 3$ and $n = 4$ and all allowed values for k and l. The box containing the largest element is marked in gray in every SYT and we can see that a pair of SYT of a given type is always mapped to a pair of SYT of the same type.

2. Other type:

   When the type is not $\downarrow\rightarrow$ it is less obvious how to define the map $\varphi_{n,k,k+2}$. Here we make use of the maps $\varphi_{n-1,k,l}$ for $1 \leq k \leq l-2 \leq n-3$ that exist by induction hypothesis. The function $\varphi_{n,k,k+2}$ is then defined as follows for a pair $(T_1, T_2) \in H_{n,k} \times H_{n,k+2}$: First we remove the element n both in T_1 and in T_2. If the type is $\downarrow\downarrow$ we obtain a pair $(t_1, t_2) \in H_{n-1,k} \times H_{n-1,k+2}$, if it is $\rightarrow\downarrow$ we obtain a pair $(t_1, t_2) \in H_{n-1,k-1} \times H_{n-1,k+2}$ and if it is $\rightarrow\rightarrow$ we obtain a pair $(t_1, t_2) \in H_{n-1,k-1} \times H_{n-1,k+1}$. In all three cases we can apply one of the maps $\varphi_{n-1,k,l}$ with $1 \leq k \leq l-2 \leq n-3$ for suitable values of k and l. We do so and obtain a pair (u_1, u_2) which is in $H_{n-1,k+1} \times H_{n-1,k+1}$ for type $\downarrow\downarrow$, in $H_{n-1,k} \times H_{n-1,k+1}$ for type $\rightarrow\downarrow$ and in $H_{n-1,k} \times H_{n-1,k}$ for type $\rightarrow\rightarrow$. Finally, we replace the element n in both Tableaux of the pair (u_1, u_2) according to its original positions in (T_1, T_2), thus creating a pair of Tableaux with n boxes of the same type as (T_1, T_2). For all three types, replacing the element n in its original position will lead to a pair $(U_1, U_2) \in H_{n,k+1} \times H_{n,k+1}$. For an example of the map $\varphi_{n,k,k+2}$ for a pair of type $\rightarrow\rightarrow$ and for $n = 5$, see Figure 24.

The construction described above ensures that the type of the image (U_1, U_2) under the map $\varphi_{n,k,k+2}$ will always be the same as the one of (T_1, T_2). Thus, in order to prove that the map $\varphi_{n,k,k+2}$ is injective, it suffices to prove that two distinct pairs (T_1, T_2) and (S_1, S_2) of SYT with n boxes that are of the same type cannot have the same image. First, let us take a look at the case of pairs of type $\downarrow\rightarrow$: If the image of two pairs (T_1, T_2) and (S_1, S_2) of type $\downarrow\rightarrow$ is the same, this means that the two pairs have to be the same if we remove the element n in all involved SYT. Since (T_1, T_2) and (S_1, S_2) are both of type $\downarrow\rightarrow$ it follows that $(T_1, T_2) = (S_1, S_2)$. Second, the case of pairs of other type: the argument is similar here. For two pairs (T_1, T_2) and (S_1, S_2) let us denote by (t_1, t_2) and (s_1, s_2) the corresponding pairs of SYT where the element n has been removed. If the image of (T_1, T_2) and (S_1, S_2) is the same, this means that the image of (t_1, t_2) and (s_1, s_2) has to be the same. The image of (t_1, t_2) and (s_1, s_2) is given by one of the maps $\varphi_{n-1,k,l}$ for suitable k and l. By the induction hypothesis these maps are injective and thus the image of (t_1, t_2) and (s_1, s_2) can only be the same if $(t_1, t_2) = (s_1, s_2)$ This in turn implies that $(T_1, T_2) = (S_1, S_2)$. This finishes the proof.

□

Now let us turn from SYT to permutations: Pairs of hooks of size n and with a first row of length k bijectively correspond to permutations of length n and with a longest increasing sequence of length k and a longest decreasing sequence of length $n-k+1$. That is, these permutations are the merge of an increasing and a decreasing subse-

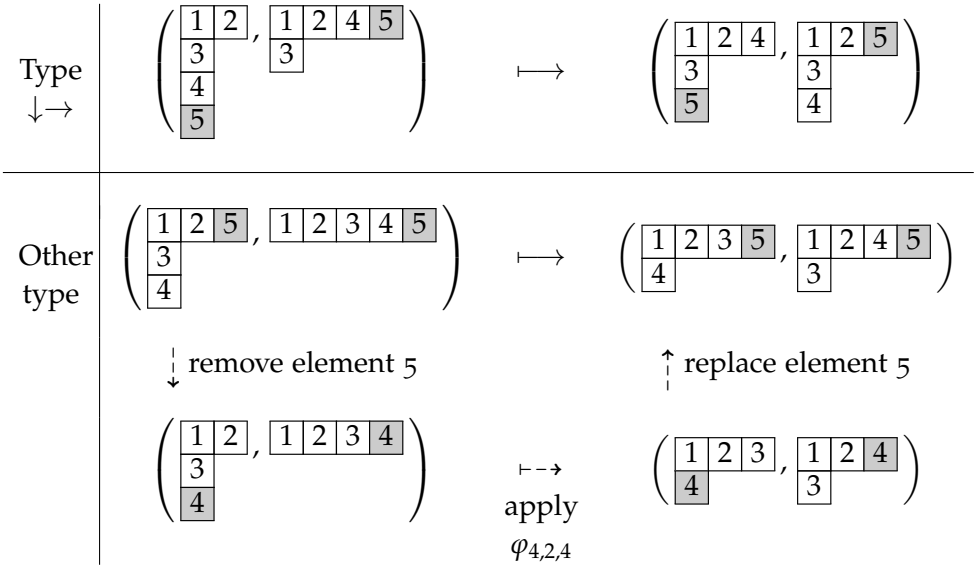


Figure 24: Two examples of the injective map $\varphi_{5,k,l}$ described in the proof of Theorem 6.4.

quence having one point in common. Let us denote these numbers by $m_{n,k}$. Then Theorems 6.2 and 6.4 lead to the following:

Corollary 6.5. *For every positive integer n the sequence $(m_{n,k})_{1\leq k\leq n}$ is log-concave.*

Permutations which are the merge of an increasing and a decreasing sequence are called *skew-merged* permutations. They have been shown to be exactly those permutations avoiding the two patterns 2143 and 3412 [145]. However, not all skew-merged permutations correspond to hook-shaped SYT. Indeed, a skew-merged permutation of length n can consist of a longest increasing subsequence of length k and of a longest decreasing subsequence of length $n-k$, i.e., the two sequences do not intersect. The shape of the corresponding tableau is then not a hook but a hook with an additional box at position $(2,2)$. Note however that not all pairs of SYT of this shape actually correspond to skew-merged permutations.

For skew-merged involutions the situation appears to be simpler and we can prove the following result:

Proposition 6.6. *The SYT associated to a skew-merged involution is always hook-shaped.*

From this we finally obtain the following result about skew-merged involutions:

Corollary 6.7. *The number of skew-merged involutions of length n and with longest increasing sequence of length k is:*

$$i_{n,k} = \binom{n-1}{k-1}.$$

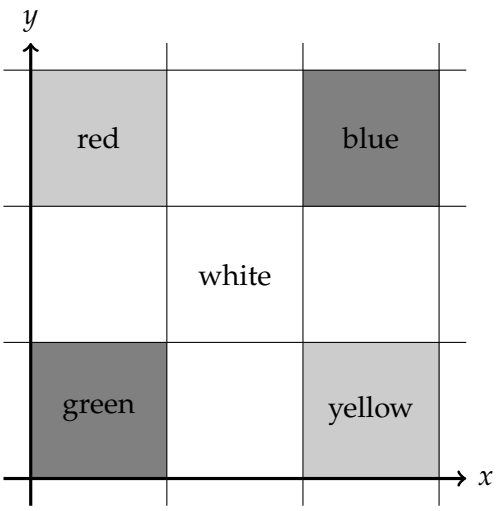


Figure 25: The disposition of colours in a skew-merged permutation due to Atkinson [11]

Thus the total number i_n of skew-merged involutions is simply 2^{n-1} and the sequence $(i_{n,k})_{1\leq k\leq n}$ is log-concave.

Proof of Proposition 6.6. In the following, we will use the notation introduced by Atkinson in [11] and will apply one of the intermediary Lemmas proven there.

Atkinson views a permutation σ as the set of points $(i, \sigma(i))$ in the plane. If σ is skew-merged the points can be partitioned into five (possibly empty) sets: there are red, blue, green, yellow and white points as represented in Figure 25. The red and yellow points are decreasing, the green and blue ones are increasing and the white points are either increasing or decreasing.

The points that are of particular interest to us are the white ones. White points can be defined as follows: Whenever one chooses two points (i, r) and (j, s) that are not of the same colour and are to the left (or to the right) of a white point (k, t), i.e. $i, j < k$ (or $k > i, j$), t is neither the largest nor the smallest among the elements r, s and t. Skew-merged permutations with at least one white point are exactly those which are the union of an increasing subsequence α and a decreasing subsequence β that have a common point. Thus skew-merged permutations with at least one white point correspond to hook-shaped SYT. Conversely skew-merged permutations with no white points are those where the associated SYT is not a hook but a hook with an additional box at position $(2, 2)$.

The goal of this proof is to show that a skew-merged permutation with no white elements cannot be an involution.

For this, we assume that the skew-merged involution σ has no white elements and construct a contradiction to the fact that σ is skew-merged.

In order to so, we will need a slightly weaker version of one of the two assertions in Lemma 11 in [11]. We shall state it here in the form we need it:

> **Lemma** [11]: *Suppose that σ is a skew-merged permutation of length n with no white points. Then there exist indices $1 \leq i < j < j+1 < k \leq n$ such that one of the following two statements holds:*
>
> - $\sigma(i)\sigma(j)\sigma(j+1)\sigma(k)$ *forms a 3142-pattern in σ*
> - $\sigma(i)\sigma(j)\sigma(j+1)\sigma(k)$ *forms a 2413-pattern in σ*

We will now show the following: If $\sigma(i)\sigma(j)\sigma(j+1)\sigma(k)$ forms a 3142-pattern then σ also contains the pattern 3412 or 2143 and is thus not skew-merged. In order to do so we distinguish three different cases. To alleviate notation, we will write *cadb* instead of $\sigma(i)\sigma(j)\sigma(j+1)\sigma(k)$ and will keep in mind that $a < b < c < d$.

1. $d \leq j+1$:

   This implies that $a \leq d-3 \leq j-2$ and that $b \leq d-2 \leq j-1$. Especially this means that a and b cannot be fixed points in σ. Since σ is an involution we thus have $\sigma(a) = j$ and $\sigma(b) = k$. Since $a < b < j < k$, we have *jkab* as a subsequence of σ that forms a 3412-pattern.

2. $a \geq j$:

   This implies that $c \geq a+2 \geq j+2$ and that $d \geq a+3 \geq j+3$. Especially this means that c and d cannot be fixed points in σ. Since σ is an involution we thus have $\sigma(c) = i$ and $\sigma(d) = j+1$. Since $i < j+1 < c < d$, we have $cda(j+1)$ as a subsequence of σ that forms a 3412-pattern.

3. $d > j+1$ and $a < j$:

   In this case a and d aren't fixed points and since σ is an involution we have $\sigma(a) = j$ and $\sigma(d) = j+1$. Since $a < j < j+1 < d$, we have $jad(j+1)$ as a subsequence of σ that forms a 2143-pattern.

Note that it is crucial in the arguments above that the elements a and d are adjacent in σ which is due to the fact that there are no white points in σ.

If the second statement of the Lemma above is fulfilled, that is if $\sigma(i)\sigma(j)\ \sigma(j+1)\sigma(k)$ forms a 2413-pattern, we consider the reversed permutation σ^r, i.e., the permutation σ read from right to left. The permutation σ^r then contains the pattern 3142 at the positions $(n+1-k), (n-j), (n+1-j)$ and $(n+1-i)$. This implies that σ^r also contains 3412 or 2143 as a pattern and is not skew-merged. Remark that a permutation is skew-merged exactly then when its reverse is

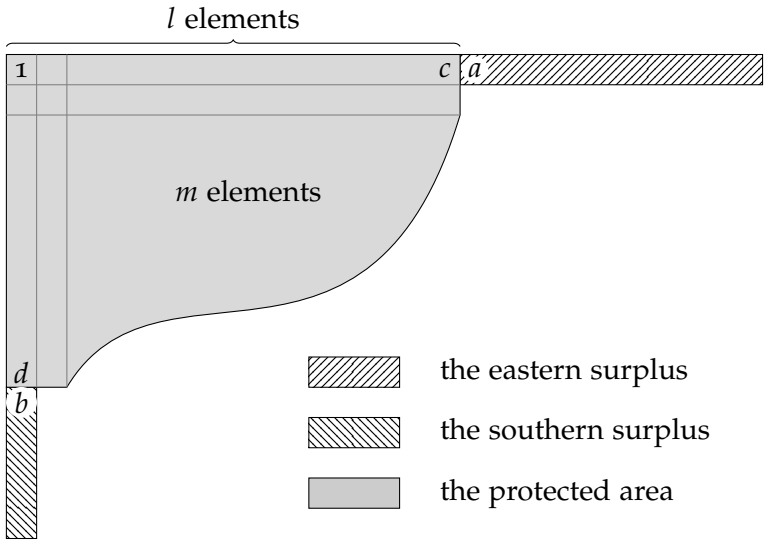


Figure 26: The decomposition of a SYT into its *surplus* and *protected area*. The protected area consists of m elements of which l are in the first row.

skew-merged. Thus we conclude that σ is not skew-merged in this case either.

We have thus proven that a skew-merged involution must contain at least one white point and the shape of its associated SYT is a hook. □

The set of (l, m)-protected SYT

Definition 6.8. *Given an arbitrary SYT T, it can be decomposed into two parts in a unique way as follows: The* protected area *of T is obtained from T by removing as many boxes as possible in the first row and in the first column of T without creating a shape that is no longer a Young diagram. The removed elements form the* surplus *of T. The elements that have been removed in the first row are referred to as the* eastern surplus *and the ones in the first column as the* southern surplus.

Example 6.9. For an illustration of this decomposition of a SYT into *protected area* and *surplus*, see Figure 26. Note that if T is a hook in the sense of Definition 6.3, then the protected area of T consists of the box containing the element 1 only. ⊣

We can now define the following class of SYT:

Definition 6.10. *Let T be a SYT in which all elements contained in its surplus are larger than all elements contained in the first row and first column of its protected area. If T has a protected area of size m of which l elements are contained in the first row it is called* (l, m)-protected.

Example 6.11. For an example of an (l, m)-protected SYT with $l = 4$ and $m = 12$, see the top part of Figure 27 where the protected areas have been marked in gray. ⊣

Let a, b, c and d be the elements of T as displayed in Figure 26, i.e., a is the smallest element in the eastern surplus and c is its left neighbour in T, b is the smallest element in the southern surplus and d is its top neighbour in T. Then the condition that all elements in the surplus are larger than those in the first row and first column of the protected area is equivalent to:

$$\min(a, b) > \max(c, d). \tag{22}$$

Before we tackle our conjecture for the set of (l, m)-protected SYT, let us consider the special case of $(2, 4)$-protected SYT and take a closer look at the set of these SYT. For $(2, 4)$-protected SYT, i.e., SYT whose shape is a hook with an additional box at the position $(2, 2)$, condition (22) is fulfilled if and only if the elements 1, 2, and 3 are contained in the protected area. Equivalently, the elements 1, 2, and 3 may not be contained in the same row or column of the SYT. Turning to involutions, this translates as follows: Involutions that correspond to $(2, 4)$-protected SYT have the property that the lengths of the longest increasing and of the longest decreasing sequence add up to the total length of the permutation and the elements 1, 2, and 3 do not form a 123- or a 321- pattern.

A question that is of interest here is the following: how many SYT whose shape is a hook with an additional box at the position $(2, 2)$ actually fulfil condition (22)? We will see that this is the case for roughly half of the SYT of this shape.

Proposition 6.12. *Let p_n denote the number of $(2, 4)$-protected SYT of size n and let b_n denote the number of SYT whose shape is a hook with an additional box at the position $(2, 2)$. Then the following holds for all $n \geq 4$:*

$$p_n = (n-3)2^{n-3},\ b_n = (n-4)2^{n-2} + 2 \text{ and thus } \lim_{n\to\infty} \frac{p_n}{b_n} = \frac{1}{2}.$$

Proof. Let us start by counting $(2, 4)$-protected SYT. We know that the elements 1, 2 and 3 have to be present in the protected area and there are two different possibilities of arranging them. Moreover we can choose the element i that will lie in the box at position $(2, 2)$ among the integers $4, \ldots, n$. The remaining entries are then inserted one after the other in increasing order. For every element we can choose to place it either in the eastern or in the southern surplus of the tableau which gives us a total of 2^{n-4} possibilities. In total we have:

$$p_n = 2 \cdot (n-3)2^{n-4} = (n-3)2^{n-3}.$$

Now let us count SYT whose shape is a hook with an additional box at the position $(2, 2)$ but that are not $(2, 4)$-protected. That is,

we determine $b_n - p_n$. This means that the elements 1, 2 and 3 are all contained in the first row or in the first column of the SYT. Let us concentrate on the case where 1, 2 and 3 are all contained in the first row; the second case can then be obtained by symmetry. Let us denote by i the element at position $(2,1)$, i.e., to the south of 1, and by j the element at position $(2,2)$, i.e., to the east of i and to the south of 2. The element i can be any integer in $\{4,\dots,n-1\}$ and j can be any integer in $\{i+1,\dots,n\}$. All elements that are smaller than i have to be placed in the eastern surplus. For the elements that are larger than i and not equal to j, we can choose whether to place them in the eastern or southern surplus. There are thus 2^{n-i-1} possibilities of placing these elements. In total we have:

$$b_n - p_n = 2 \cdot \sum_{i=4}^{n-1} (n-i) \cdot 2^{n-i-1},$$

which leads to:

$$\begin{aligned} b_n &= \sum_{i=3}^{n-1} (n-i) \cdot 2^{n-i} = 2^n \cdot \left(n \cdot \sum_{i=3}^{n-1} 2^{-i} - \sum_{i=3}^{n-1} i 2^{-i} \right) \\ &= 2^n \cdot \left(n \cdot \left(\frac{1}{4} - \frac{1}{2^n} \right) - \frac{2^n - 2 - n}{2^n} \right) = 2^{n-2}(n-4) + 2. \end{aligned}$$

We indeed obtain that p_n is roughly one half of b_n and that the fraction p_n/b_n tends to $1/2$ when n tends to infinity. □

Note that this result for the numbers b_n can of course also be obtained by applying the hooklength formula (see Chapter 14 in [29]). However this leads to rather tedious manipulations of sums involving fractions of binomial coefficients and the approach above is much faster.

Now let us turn to (l,m)-protected SYT for arbitrary integers l and m. In the following, for $1 \leq m \leq n$ and $1 \leq l \leq k$, we will denote by $P_{n,k}^{(l,m)}$ the set of all (l,m)-protected SYT of size n where the first row has length k. Whenever it is clear from the context, we will write $P_{n,k}$ instead of $P_{n,k}^{(l,m)}$. The cardinality of $P_{n,k}^{(l,m)}$ shall be denoted by $p_{n,k}^{(l,m)}$ or $p_{n,k}$.

Theorem 6.13. *For every positive integer n and every fixed pair (l,m), the sequence $(p_{n,k})_{1\leq k\leq n}$ is log-concave.*

Proof. Let us fix the integers l and m and omit them in the notation. In order to prove the log-concavity of the sequence $(p_{n,k})$ we need to construct injections $\psi_{n,k}$ from $P_{n,k-1} \times P_{n,k+1}$ to $P_{n,k} \times P_{n,k}$ for all $n \in \mathbb{N}$ and $1 \leq k \leq n$.

Let (T_1, T_2) be a pair of SYT in $P_{n,k-1} \times P_{n,k+1}$. The maps $\psi_{n,k}$ shall never affect the protected areas of T_1 and T_2 – thus this name. Let

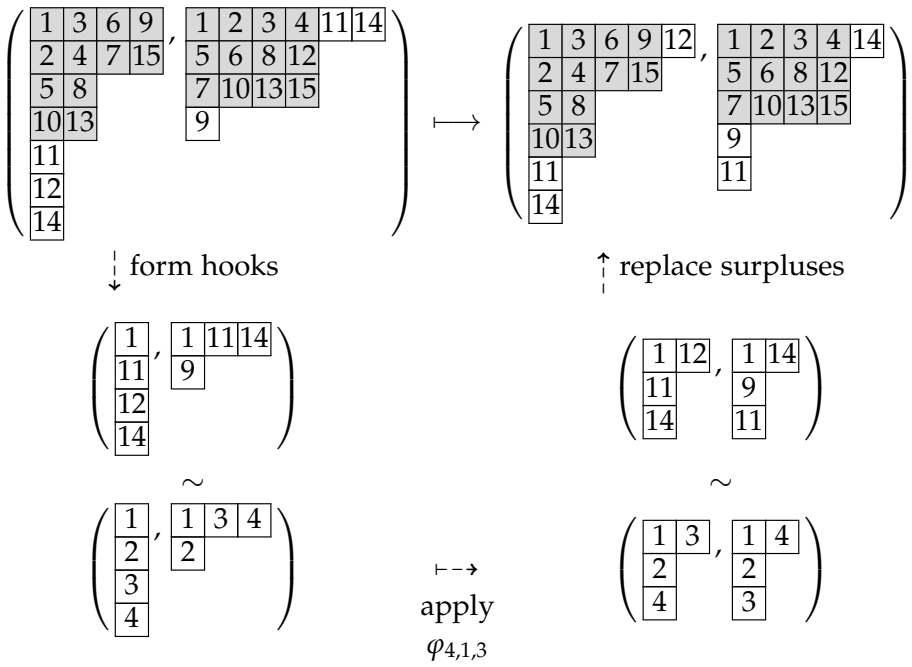


Figure 27: An example of the injective map $\psi_{15,5}$ described in the proof of Theorem 6.4. Here, the map is applied to a pair of $(4,12)$-protected SYT.

H_1 and H_2 be the hooks that one obtains from the surpluses of T_1 and T_2 as follows: Place the element 1 at position $(1,1)$; then attach the respective eastern surplus to the east of 1 and the southern surplus to the south of 1. The hooks H_1 and H_2 then both consist of $(n-m+1)$ boxes. Moreover, the first row of H_1 has $k-l$ elements and the first row of H_2 has $k-l+2$ elements. Let us denote by $\tilde{H}_i$ the SYT that we obtain from H_i for $i=1,2$ by replacing the entries order-isomorphically by the integers from 1 up to $(n-m+1)$. We can then apply the injection $\varphi_{n-m+1,k-l,k-l+2}$ defined in the proof of Theorem 6.4 to $(\tilde{H}_1,\tilde{H}_2)$ and obtain a pair (J_1,J_2) in $H_{n-m+1,k-l+1}\times H_{n-m+1,k-l+1}$.

In order to obtain (U_1,U_2), the image of (T_1,T_2) under $\psi_{n,k}$, we now do the following for $i=1,2$: First we create the hooks $\tilde{J}_i$ by order-isomorphically replacing the elements $2,\dots,n-m+1$ in J_i by the elements that occurred in the respective surpluses of T_i. Then we attach the eastern surplus of $\tilde{J}_i$ to the east of the first row of the protected area of T_i and the southern surplus of $\tilde{J}_i$ to the south of its first column.

The only change that has been made to the shapes of T_1 and T_2 was to remove one box at the end of the first column or row and to place it at the end of the first row or column. The shapes of U_1 and U_2 are thus as desired and the sizes of the protected areas are unchanged. We still need to check whether U_1 and U_2 are actually SYTs, i.e., whether the numbers in all rows and columns are increas-

ing. Clearly, we only need to do so for the first row and the first column since the other parts have not been affected. By definition of the maps $\varphi_{n-m+1,k-l,k-l+2}$, the elements in the surpluses of U_1 and U_2 are increasing. Moreover, the elements in the surpluses of U_1 and U_2 are all larger than the elements contained in the first row and first column of the corresponding protected areas. Thus we indeed obtain a pair of tableaux (U_1, U_2) in $P_{n,k} \times P_{n,k}$ with the same values for l and m as for (T_1, T_2). The injectivity of the maps $\psi_{n,k}$ clearly follows from the injectivity of the maps $\varphi_{n-m+1,k-l,k-l+2}$. □

Example 6.14. For an example of the map $\psi_{n,k}$ with $n = 15$, $k = 5$ and $(l, m) = (4, 12)$, see Figure 27. ⊣

6.3 LATTICE PATHS AND 321-AVOIDING PERMUTATIONS

In the following, let $A_{n,k}$ denote the set of all 321-avoiding involutions of length n with longest increasing sequence of length k. The cardinality of $A_{n,k}$ will be denoted by $a_{n,k}$. The Robinson-Schensted correspondence maps elements of $A_{n,k}$ into SYT that consist of at most two rows such that the length of the first row is $k \geq \lceil n/2 \rceil$. Then the hooklength formula yields that

$$a_{n,k} = \binom{n}{k} \frac{2k - n + 1}{k + 1}.$$

Using this formula, it is routine to prove that for any fixed n, the sequence $a_{n,k}$ is log-concave if $k \in [\lceil n/2 \rceil, n]$, but that proof is not particularly elucidating. In what follows, we provide a more elegant, injective proof.

There is a natural bijection $f = f_n$ between the set of such SYT and lattice paths using steps $(0, 1)$ and $(1, 0)$ that start at $(0, 0)$, consist of n steps, and never go above the diagonal $x = y$. We will refer to these steps as East and North steps. Indeed, if $i_1, i_2, \cdots, i_k$ are the numbers in the first row of the SYT T, and $j_1, j_2, \cdots, j_{n-k}$ are the numbers in the second row of T, then we can set $f(T)$ to be the lattice path starting at $(0, 0)$ whose East steps are in positions $i_1, i_2, \cdots, i_k$, and whose North steps are in positions $j_1, j_2, \cdots, j_{n-k}$. The fact that T is a SYT means that $i_t < j_t$ for all $t \leq n - k$, so $f(T)$ will indeed stay below the diagonal $x = y$ since its t-th North step will come some time after its t-th East step. Note that elements of $A_{n,k}$ will be mapped into paths that end in $(k, n - k)$.

Example 6.15. The SYT with elements $1, 3, 4, 5, 6, 7$ in the first row and the element 2 in the second row corresponds to the path Q consisting of the following steps: E, N, E, E, E, E, E. It is represented by circles on the left hand side of Figure 28. Similarly, the SYT with elements $1, 2, 4, 7$ in the first and elements $3, 5, 6$ in the second row corresponds to the path P : E, E, N, E, N, N, E. ⊣

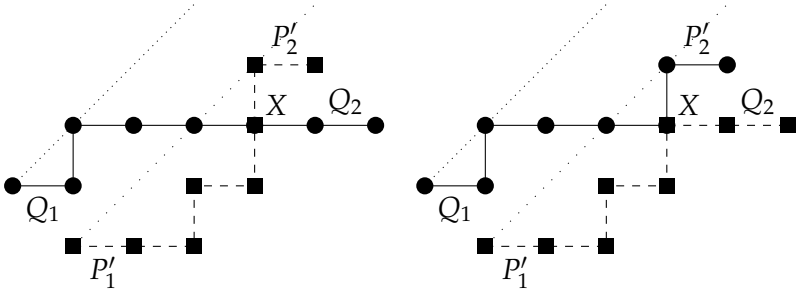


Figure 28: The main step in the construction of the map ϕ

Let $L(n,k)$ be the set of lattice paths using steps $(0,1)$ and $(1,0)$ that start at $(0,0)$, never go above the diagonal $x = y$, and end in $(k, n-k)$, where $n-k \leq k \leq n-2$. We define a map

$$\phi : L(n,k) \times L(n,k+2) \to L(n,k+1) \times L(n,k+1)$$

as follows.

Let $(P,Q) \in L(n,k) \times L(n,k+2)$. Let us translate P by the vector $(1,-1)$ to obtain the path P' that starts at $(1,-1)$, ends in $(k+1, n-k-1)$, and *never goes above the diagonal* $x-2 = y$. As Q starts West of P' and ends East of P', the paths P' and Q will have at least one point in common. Let X be the last such point. Note that X is not the endpoint of P' or Q, since those two paths do not end in the same point. We now "flip" the paths at X, which means the following. Let P_1' and Q_1 denote the parts of P' and Q that end in X, and let P_2' and Q_2 denote the parts of P' and Q that start in X. Then we define $\phi(P,Q)$ as $((P_1'Q_2)*, Q_1P_2')$, where $(P_1'Q_2)*$ denotes the path $P_1'Q_2$ translated back by the vector $(-1,1)$ so that it starts at $(0,0)$. See Figure 28 for an illustration.

Proposition 6.16. *The map ϕ described above indeed maps into the set $L(n,k+1) \times L(n,k+1)$ and is an injection from $L(n,k) \times L(n,k+2)$ into $L(n,k+1) \times L(n,k+1)$.*

Proof. It is a direct consequence of the definitions that both $(P_1'Q_2)*$ and Q_1P_2' will indeed start at $(0,0)$ and end in $(k+1, n-k+1)$. So, all we need to do in order to prove that $\phi(P,Q) \in L(n,k+1) \times L(n,k+1)$ is to show that neither $(P_1'Q_2)*$ and Q_1P_2' ever goes above the diagonal $x = y$.

- To show that $(P_1'Q_2)*$ does not go above the diagonal $x = y$ is equivalent to showing that $P_1'Q_2$ does not go above the diagonal $x-2 = y$. This is true for P_1' by its definition (it is a part of P'), and this is true for Q_2 since Q_2 is entirely below P_2', and P_2', by its definition (it is a part of P'), never goes above the diagonal $x-2 = y$.
- It is clear that Q_1P_2' never goes above the diagonal $x = y$, since neither Q_1 (a part of Q) nor P_2' (a part of P) do.

Finally, in order to prove that the map ϕ is injective, let (R, S) be in $L(n, k+1) \times L(n, k+1)$. If R and S have no points in common (other than their starting and ending points), (R, S) has no preimage under ϕ. Otherwise, we can recover X as the last point that R and S have in common other than their endpoint, and then reversing the "flipping" operation described in the construction of ϕ we can recover the unique preimage of (R, S). □

Corollary 6.17. *For any fixed n, the sequence $(a_{n,k})_{n/2 \leq k \leq n}$ is log-concave.*

Therefore, it follows by the principle that we used to prove Theorem 6.2 that we have an injective proof of the following corollary as well.

Corollary 6.18. *Let $b_{n,k}$ denote the number of 321-avoiding permutations of length n in which the longest increasing subsequence is of length k. Then for any fixed n, the sequence $(b_{n,k})_{n/2 \leq k \leq n}$ is log-concave.*

6.4 SUMMARY OF THE RESULTS

Let us briefly summarize the results of this chapter. First we stated our conjecture that the sequence $(\ell_{n,k})_{1 \leq k \leq n}$ is log-concave for fixed $n \in \mathbb{N}$ and showed that it suffices to prove this conjecture for involutions in order to obtain a result for permutations in general. Second, we proved that our conjecture holds for SYT whose shape is a hook which implies that it holds for skew-merged involutions. This result could be extended to a larger class of permutations, based on the concept of (l, m)-protected SYT. Third, we showed that our conjecture holds for 321-avoiding permutations. For all our results we provided combinatorial proofs and exploited the Robinson-Schensted correspondence between permutations and pairs of SYT.

Part II

CAYLEY TREES AND MAPPINGS

This part treats two seemingly unrelated topics on Cayley trees and mappings. First, Chapter 7 provides a new bijective proof of Cayley's formula that will be applied in the following chapters. Second, Chapter 8 deals with *ascending runs* in trees and mappings, analysing the occurrence of a specific pattern that involves the labels of the combinatorial object. Third, Chapter 9 generalizes the concept of parking functions to trees and to mappings. Even though these two topics have little in common at first sight, a similar decomposition approach leading to PDEs is fruitful for tackling the enumeration task.

7

A NEW BIJECTIVE PROOF OF CAYLEY'S FORMULA

We start this part by presenting a new bijective proof of Cayley's formula. This bijection will be used in the following two chapters that concern ascending runs in trees and mappings (Chapter 8) and a generalization of the concept of parking functions to trees and mappings (Chapter 9).

Theorem 7.1. *For each $n \geq 1$, there exists a bijection φ from the set of pairs (T, w), with $T \in \mathcal{T}_n$ a tree of size n and $w \in T$ a node of T, to the set of n-mappings. Thus*

$$n \cdot T_n = M_n, \quad \text{for } n \geq 1.$$

Proof. In the following, we will denote by $T(v)$ the parent of node v in the tree T. That is, for $v \neq \mathsf{root}(T)$, $T(v)$ is the unique node such that $(v, T(v))$ is an edge in T.

Given a pair (T, w), we consider the unique path $w \rightsquigarrow \mathsf{root}(T)$ from the node w to the root of T. It consists of the nodes $v_1 = w$, $v_2 = T(v_1), \ldots, v_{i+1} = T(v_i), \ldots, v_r = \mathsf{root}(T)$ for some $r \geq 1$. We denote by $I = (i_1, \ldots, i_t)$, with $i_1 < i_2 < \cdots < i_t$ for some $t \geq 1$, the indices of the right-to-left maxima in the sequence $v_1, v_2, \ldots, v_r$, i.e.,

$$i \in I \iff v_i > v_j, \quad \text{for all } j > i.$$

The corresponding set of nodes in the path $w \rightsquigarrow \mathsf{root}(T)$ will be denoted by $V_I := \{v_i : i \in I\}$. Of course, if follows from the definition that the root node is always contained in V_I, i.e., $v_r \in V_I$.

We can now describe the function φ by constructing an n-mapping f. The t right-to-left maxima in the sequence $v_1, v_2, \ldots, v_r$ will give rise to t connected components in the functional digraph G_f. Moreover, the nodes on the path $w \rightsquigarrow \mathsf{root}(T)$ in T will correspond to the cyclic nodes in G_f. We describe f by defining $f(v)$ for all $v \in [n]$, where we distinguish whether $v \in V_I$ or not.

(a) Case $v \notin V_I$: We set $f(v) := T(v)$.

(b) Case $v \in V_I$: Setting $T(v_{i_0}) := v_1 = w$, we define

$$f(v_{i_\ell}) := T\left(v_{i_{\ell-1}}\right).$$

This means that the nodes on the path $w \rightsquigarrow \mathsf{root}(T)$ in T form t cycles $C_1 := (v_1, \ldots, v_{i_1}), \ldots, C_t := (T(v_{i_{t-1}}), \ldots, v_r = v_{i_t})$ in G_f.

It is now easy to describe the inverse function φ^{-1}. Given a mapping f, we sort the connected components of G_f in decreasing order

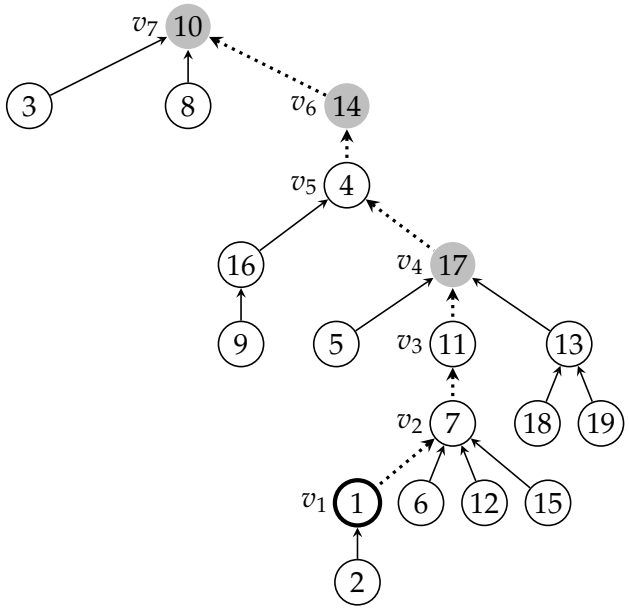


Figure 29: Taking the image of the pair $(T, 1)$ where T is depicted above leads to the mapping described in Figure 11 on page 23. Nodes in $\mathcal{S}$ are drawn in white.

of their largest cyclic elements. That is, if G_f consists of t connected components and c_i denotes the largest cyclic element in the i-th component, we have $c_1 > c_2 > \ldots > c_t$. Then, for every $1 \leq i \leq t$, we remove the edge (c_i, d_i) where $d_i = f(c_i)$. Next we reattach the components to each other by establishing the edges (c_i, d_{i+1}) for every $1 \leq i \leq t-1$. This leads to the tree T. Note that the node c_t is attached nowhere since it constitutes the root of T. Setting $w = d_1$, we obtain the preimage (T, w) of f. □

Example 7.2. Taking the image of the pair $(T, 1)$ where the tree T is depicted in Figure 29 leads to the mapping described in Figure 11 on page 23. We consider the unique path from the node labelled 1 to the root of T. It consist of the following nodes: $1, 7, 11, 17, 4, 14$ and 10. Within this sequence, the right-to-left maxima are $17, 14$ and 10 which are marked by gray nodes in the figure. When creating the image of $(T, 1)$ under the map φ, the edges $(17, 4)$ and $(14, 10)$ are removed and the edges $(17, 1)$, $(14, 4)$ and $(10, 10)$ are created. ⊣

For the bijection described in Theorem 7.1, the following properties that will turn out to be useful in Chapter 8 are fulfilled:

Proposition 7.3. *Consider the pair (T, w) where T is a Cayley tree and w is a node in T. Let f be the image of (T, w) under the bijection φ described in Theorem 7.1. Furthermore, for every node v that is not the root of T, let us denote by $T(v)$ the parent node of v in T . Then the following holds for every node v in T:*

1. $T(v) > v \iff f(v) > v$,

2. *There exists a node $x < v$ in T with $T(x) = v$*
   $\iff$ *There exists an element $x < v$ in f with $f(x) = v$.*

Proof. The main reason why these properties hold is the following: When creating the image of some (T, w) under the map φ, the only edges that are removed are decreasing ones. That is, the only edges that are present in T but not in G_f, are of the form (v, x) where $v > x$. The edges that are created instead in G_f are also decreasing ones. In the following, we prove the two statements of the theorem in more detail:

1. As in the proof of Theorem 7.1 we distinguish whether the node v in T is contained in V_I or not: If v is not contained in V_I, $f(v) = T(v)$ and thus the statement is clearly true. If $v \in V_I$, v is a right-to-left maximum in the unique path from w to the root of T. It thus holds $v = v_{i_\ell}$ for some ℓ and clearly $T(v) < v$. We need to show that this implies that $f(v) \leq v$. By definition, we have $f(v_{i_\ell}) = T\left(v_{i_{\ell-1}}\right)$. $T\left(v_{i_{\ell-1}}\right)$ is either equal to v_{i_ℓ} or it is not a right-to left maximum in which case it is clearly strictly smaller than v_{i_ℓ}. This proves the first statement of this proposition.

2. $(\Longrightarrow)$: If the node v has a child x with $x < v$, then x cannot belong to the set of nodes V_I. Thus $f(x) = T(x) = v$ and in the mapping f the node v has a preimage x with $x < v$.

   $(\Longleftarrow)$: Let x be an element with $x < v$ and $f(x) = v$. Suppose $x = v_{i_\ell}$ for some ℓ. Then $v = f(x) = T(v_{i_{\ell-1}}) \leq x$. Thus $x \notin V_I$ and we have $f(x) = T(x)$. Therefore x is a child of v in T that fulfils $x < v$.

□

8

ASCENDING RUNS IN MAPPINGS

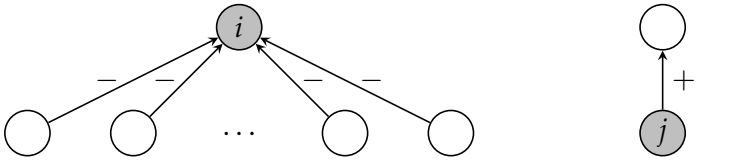
This chapter is based on joint work with Alois Panholzer.

Random mappings, i.e., random functions from the set $[n]$ into itself, arise in many applications and are very well-studied objects both from a combinatorial and a probabilistic point of view. For examples of situations where random mappings occur, see e.g. the introduction of [75]. Previous research has concentrated on structural parameters of the corresponding functional graphs. Structural parameters such as the number of connected components, the size of the largest tree, the number of terminal nodes, etc. are well-studied objects. Exemplarily, we would like to mention the work of Arney and Bender [8], Kolchin [112], Flajolet and Odlyzko [75], Drmota and Soria [61] (given in chronological order).

In the functional graphs corresponding to random mappings, the labels of nodes play an important role and thus it is somewhat surprising that, despite the many studies of mappings concerning structural quantities, the occurrences of label patterns have received far less attention so far.

Some recent research in this direction has been performed by Panholzer who studied *alternating mappings* in [130]. These are a generalization of the concept of alternating permutations to mappings; they can be defined as those mappings for which every iteration orbit forms an alternating sequence. Alternating mappings can thus also be seen as mappings that do not contain two consecutive ascents or descents; this is a characterization in terms of forbidden label patterns. Also, we would like to mention the PhD thesis of Okoth [129] who has very recently studied local extrema in trees (called sources and sinks there). His studies also lead to results for the corresponding quantities in mappings.

In this chapter, we study the occurrence of a specific label pattern in mappings by generalizing the concept of ascending runs from permutations to mappings. See Section 2.2 in the Preliminaries, for a definition and a distinction between ascending and alternating runs. Since the only types of runs encountered in this chapter are *ascending runs*, we will also use *runs* for short. As we will see in Section 32, counting runs in mappings is equivalent to counting the number of nodes that form the label pattern on the left-hand side in Figure 30. Our goal is to enumerate mappings by their size and by their number of ascending runs and to characterize the typical behaviour of this label pattern in random mappings. Since connected mappings can be



i is the start of a run *j* is an ascent

Figure 30: The label patterns in mappings studied in this chapter: Counting ascending runs is equivalent to counting the occurrences of the label pattern depicted on the left, counting ascents clearly corresponds to counting the occurrences of the pattern on the right.

described as cycles of Cayley trees, our studies of runs in mappings will also lead to corresponding results for runs in trees.

We start this chapter by studying ascents in mappings in Section 8.2. From a probabilistic point of view, this is a straightforward task. Counting ascents in mappings corresponds to counting nodes in mapping graphs forming the label pattern on the right-hand side in Figure 30. The results can then be transferred to ascents in trees by using the bijection presented in Section 7. The main part of this chapter is devoted to the exact and asymptotic study of ascending runs in mappings, which leads to results for Cayley trees as well. Ascending runs, which will formally be introduced in Section 8.2, are maximal ascending paths in the corresponding functional graphs. We use a combinatorial decomposition of the tree respectively mappings according to the smallest node. This leads to a recurrence relation for the studied numbers that can be translated into a partial differential equation for the corresponding generating functions. Solving this PDE allows for extraction of coefficients and for a characterization of the limiting distribution of the number of ascending runs in a random mapping or tree. Again, the bijection presented in Section 7 will allow to explain a simple connection between the results for trees and for mappings. Finally, we close this chapter by summarizing the results and setting them in relation with the corresponding results that have been obtained for permutations.

Note that throughout this chapter, we will identify a mapping with its functional graph.

8.1 A PROBABILISTIC WARM-UP: ASCENTS IN MAPPINGS AND TREES

In this introductory section, we consider ascents in mappings, i.e. nodes with label i such that $f(i) > i$. For an example, consider Figure 31 in which the ascents have been marked in our running example f_{ex}. Ascents in mappings are easily dealt with. Indeed, in a random n-mapping, the probability of i being an ascent is simply equal to $(n-i)/n$. Thus, if the random variable A_n^i is the indicator

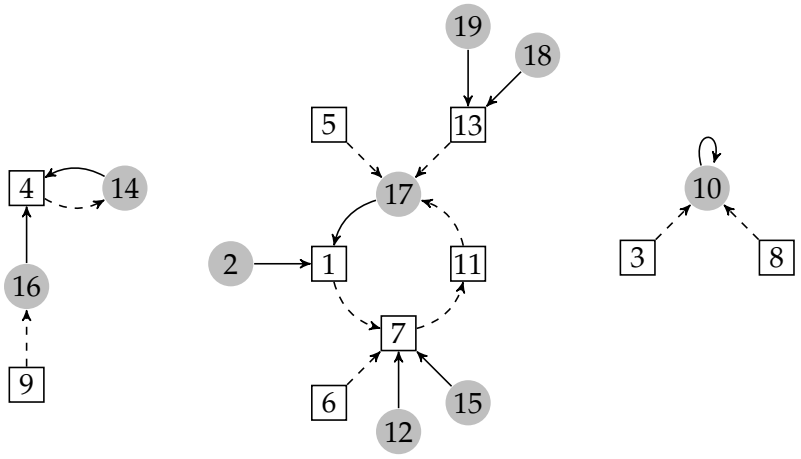


Figure 31: In our running example f_{ex}, 10 out of 19 nodes are ascents. These are marked by square nodes.

variable for the event "node i is an ascent in a random n-mapping" A_n^i follows a Bernoulli distribution with parameter $p = (n-i)/n$: $A_n^i \sim B((n-i)/n)$. Moreover, $A_n = \sum_{i=1}^n A_n^i$ and since the A_n^i's are independent random variables we easily obtain A_n's probability-generating function

$$\begin{aligned} m_{A_n}(t) &= \sum_{k=0}^{\infty} t^k \mathbb{P}\{A_n = k\} \\ &= \prod_{i=1}^{n} m_{A_n^i}(t) \\ &= \prod_{i=1}^{n} \frac{i + t(n-i)}{n}. \end{aligned}$$

Thus, A_n follows a so-called *Poisson binomial distribution* (see e.g. [21]). The number $A_{n,m}$ of n-mappings with exactly m ascents is thus simply given as:

$$\begin{aligned} A_{n,m} &= [t^m] \prod_{k=1}^{n} k + t(n-k) \\ &= \sum_{\substack{I \subseteq [n] \\ |I|=m}} \left(\prod_{i \in I} (n-i) \prod_{j \in I^c} j \right). \end{aligned}$$

Clearly, the sequence $(A_{n,m})_{m=0\ldots n}$ is symmetric: $A_{n,m} = A_{n,n-1-m}$. It is also easily shown to be unimodal since $m_{A_n}(t)$ has real roots only. Moreover, one readily sees that

$$\begin{aligned}\mathbb{E}(A_n) &= \sum_{i=1}^{n} \mathbb{E}(A_n^i) \quad \text{and} \quad \mathbb{V}(A_n) = \sum_{i=1}^{n} \left(1 - \mathbb{E}(A_n^i)\right) \mathbb{E}(A_n^i) \\ &= \sum_{i=1}^{n} \frac{n-i}{n} \qquad\qquad = \sum_{i=1}^{n} \frac{i}{n} \cdot \frac{n-i}{n} \\ &= \frac{n-1}{2} \qquad\qquad = \frac{n^2-1}{6n}.\end{aligned}$$

Using the Ljupanov-version of the Central Limit Theorem, Theorem 2.35, we can show even more:

Theorem 8.1. *Let A_n denote the random variable counting the number of ascents in a random n-mapping. Then the standardized random variable*

$$\frac{A_n - \mathbb{E}(A_n)}{\sqrt{\mathbb{V}(A_n)}}$$

converges in distribution to a standard normal distribution.

Proof. The result follows by a basic application of the Ljapunov CLT. Similar proofs can be found in any textbook on probability theory. We need to check whether there exists a $\delta > 0$ such that

$$\frac{1}{\sqrt{\mathbb{V}(A_n)}^{2+\delta}} \sum_{i=1}^{n} \mathbb{E}\left(|A_n^i - \mathbb{E}(A_n^i)|^{2+\delta}\right) \to 0 \text{ as } n \to \infty$$

is fulfilled. Let $\delta > 0$. Then the following holds:

$$\begin{aligned}1 \geq \frac{n-i}{n} \cdot \frac{i}{n} &= \mathbb{E}\left(\left(A_n^i - \frac{n-i}{n}\right)^2\right) \\ &= \mathbb{V}\left(A_n^i - \frac{n-i}{n}\right) \geq \mathbb{E}\left(\left(A_n^i - \frac{n-i}{n}\right)^{2+\delta}\right).\end{aligned}$$

Thus we obtain

$$\begin{aligned}&\frac{1}{\sqrt{\mathbb{V}(A_n)}^{2+\delta}} \mathbb{E}\left(\left(A_n^i - \frac{n-i}{n}\right)^{2+\delta}\right) \\ \leq &\frac{1}{\sqrt{\mathbb{V}(A_n)}^{2+\delta}} \mathbb{V}\left(A_n^i - \frac{n-i}{n}\right) \\ = &\frac{1}{\sqrt{\mathbb{V}(A_n)}^{\delta}} = \left(\frac{6n}{n^2-1}\right)^{\delta/2}\end{aligned}$$

and since the right hand side tends to zero as $n \to \infty$ we obtain the desired statement. □

These results themselves are maybe not particularly surprising, but what is less straightforward is that they can be transferred to the corresponding enumeration problem for trees. Indeed, as shown in Theorem 7.1 and Proposition 7.3, we can identify every pair (T, w) where T is a Cayley tree of size n with m ascents and w is a node in T with an n-mapping f with m ascents. We thus obtain the following:

$$A_{n,m} = n \cdot B_{n,m},$$

where $B_{n,m}$ denotes the number of trees with n nodes and exactly m ascents. Thus, if B_n is the random variable counting ascents in a tree of size n, A_n and B_n have exactly the same probability-generating function for every integer n.

Corollary 8.2. *Let B_n denote the random variable counting the number of ascents in a random Cayley tree of size n. Then the expectation and variance satisfy*

$$\mathbb{E}(B_n) = \frac{n-1}{2} \quad \textit{and} \quad \mathbb{V}(B_n) = \frac{n^2-1}{6n}.$$

Furthermore, the standardized random variable

$$\frac{B_n - \mathbb{E}(B_n)}{\sqrt{\mathbb{V}(B_n)}}$$

converges in distribution to a standard normal distribution.

A similar result for ascents in trees seems to have been obtained in [49]. However, this simple approach via the detour of ascents in mappings is, to the best of our knowledge, new.

8.2 ASCENDING RUNS IN MAPPINGS

We consider here the number of ascending runs of a mapping. Generalizing the definition of ascending runs for permutations (see Section 2.2), an ascending run in a mapping f is a maximal ascending sequence $i < f(i) < f^2(i) < \cdots < f^k(i)$, i.e., an ascending sequence which is not contained in a longer such sequence. The only type of "runs" studied in this chapter are ascending runs, we will thus also use the shorter term run when speaking of ascending runs. These are not to be confused with alternating runs as studied in Chapter 3 (see Section 2.2 for a distinction of these two concepts).

Given a specific mapping f and its functional graph G_f, the number of runs in f can be counted easily by considering the first element of a run: Indeed, an element j is the first element of a run iff each label i of a preimage of j is not smaller than j, i.e., $(f(i) = j) \Rightarrow (i \geq j)$. We thus need to count nodes in the mapping that form the label pattern depicted on the left-hand side of Figure 30. Note that elements with

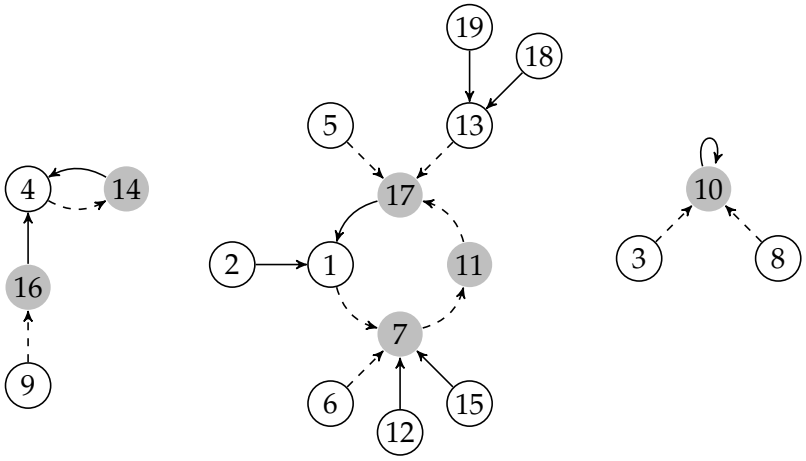


Figure 32: In f_{ex} there are 13 ascending runs. If an element is the first element in a run, its node is white; otherwise it is gray. Increasing edges, i.e., edges (i, j) with $i < j$, are marked by dashed lines.

no preimages at all, i.e., terminal nodes in the graph, are of course always at the beginning of a run.

Recall that for permutations the following simple connection between descents and ascending runs holds: if a permutation π has k descents, i.e., positions i for which it holds that $\pi(i) > \pi(i+1)$, it can be decomposed into $(k+1)$ ascending runs. Moreover, by taking the complement π^c of π one obtains a permutation with k ascents. Thus, studying the distribution of the following parameters in permutations is equivalent: ascents, descents, ascending runs and descending runs. This is clearly not the case for mappings: The presence of k descents does not yield a decomposition into $(k+1)$ ascending runs. For example, in f_{ex} there are 10 ascents and thus 9 descents but the number of ascending runs is equal to 13. Thus, the study of ascending runs has to be performed separately and requires, as we will see, far more involved techniques than the one of ascents.

Example 8.3. In Figure 32, our running example f_{ex} of a 19-mapping is depicted. In its functional graph, the nodes that correspond to first elements of an ascending run are marked by white nodes. There are 13 such nodes and thus our mapping has 13 ascending runs. Moreover, edges (i, j) where $i < j$ are marked with dashed lines and edges with $j > i$ are marked with full lines. An ascending run is thus a path of maximal length consisting of dashed edges only (or no edges at all). An element that is the start of an ascending run can be characterized as a node where no incoming edges are dashed. ⊣

In the following, R_n will denote the r.v. that counts the number of ascending runs in a random n-mapping. Because of the combinatorial structure of mappings as presented in Example 2.27, we will first analyse the corresponding random variable for trees before we study R_n.

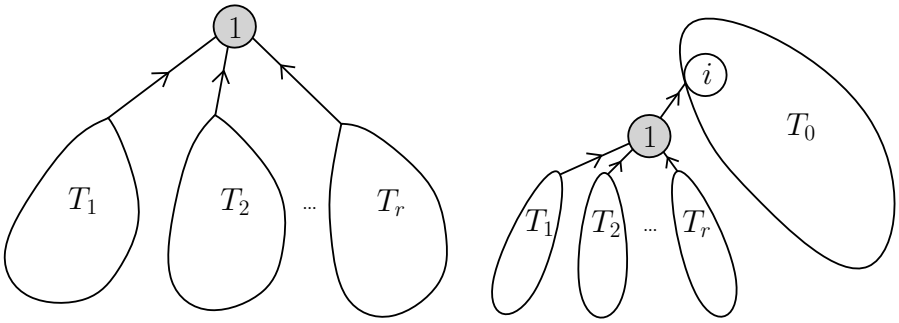


Figure 33: Decomposition of a tree with respect to its smallest node

Runs in trees

To study R_n we first consider the corresponding quantity in trees, where the edges are oriented towards the root node of the tree. Thus, we introduce the random variable F_n which counts the number of runs in a random Cayley tree of size n. Let us further introduce the generating function

$$\begin{aligned} F(z,v) &= \sum_{n\geq 1}\sum_{m\geq 0} \mathbb{P}\{F_n = m\} T_n \frac{z^n}{n!} v^m \\ &= \sum_{n\geq 1}\sum_{m\geq 0} F_{n,m} \frac{z^n}{n!} v^m, \end{aligned}$$

where $T_n = n^{n-1}$ is the number of trees of size n and $F_{n,m}$ is the number of trees of size n with exactly m ascending runs.

Clearly, $F_{1,m} = \delta_{m,1}$. In order to establish a recurrence relation for the numbers $F_{n,m}$, we decompose a tree of size $n \geq 2$ with respect to the node labelled 1. Two different cases, as shown in Figure 33, have to be considered:

Case (1): The node 1 is the root node. In this case a set of $r \geq 1$ subtrees $T_1, T_2, \ldots, T_r$ is attached to the root 1. Since the root has the smallest of all labels, it will always constitute a run of its own. Thus, if the subtrees T_i have respective sizes n_i and m_i runs each, the entire tree will consist of n nodes and have m runs iff $\sum_{i=1}^r n_i = n-1$ and $\sum_{i=1}^r m_i = m-1$. The contribution to $F_{n,m}$ is therefore:

$$F^{(1)}_{n,m} = \sum_{r\geq 1} \frac{1}{r!} \sum_{\substack{\sum n_i = n-1 \\ \sum m_i = m-1}} F_{n_1,m_1} \cdots F_{n_r,m_r} \cdot \binom{n-1}{n_1,\ldots,n_r},$$

where the multinomial coefficient reflects the number of possibilities of redistributing the labels in $[n-1]$ order-isomorphically to the subtrees.

Case (2): The node 1 is not the root node. In this case the node 1 is attached to some node labelled i of a tree T_0 of size n_0 and with m_0 runs and has itself a set of $r \geq 0$ subtrees $T_1, T_2, \ldots, T_r$ attached to

it. Depending on where in T_0 the node 1 is attached, a new run is created or not. If 1 is attached at the begin of a run in T_0 no new run is created – there are m_0 possibilities of attaching 1 in this way. If 1 is however attached somewhere in the middle of a run, a new run will be created – for this case there are $(n_0 - m_0)$ possibilities. The contribution to $F_{n,m}$ is therefore:

$$F_{n,m}^{(2a)} = \sum_{\substack{1\le n_0\cdot\le n-1\\ 1\le m_0\le m}} m_0 \cdot \sum_{r\ge 0}\frac{1}{r!} \sum_{\substack{\sum n_i = n-n_0-1\\ \sum m_i = m-m_0}} F_{n_0,m_0}\cdots F_{n_r,m_r}\binom{n-1}{n_0,\ldots,n_r}$$

for the case that the node 1 is attached to the beginning of a run in T_0 and

$$F_{n,m}^{(2b)} = \sum_{\substack{1\le n_0\le n-1\\ 1\le m_0\le m}} (n_0 - m_0) \cdot \sum_{r\ge 0}\frac{1}{r!} \sum_{\substack{\sum n_i = n-n_0-1\\ \sum m_i = m-m_0-1}} F_{n_0,m_0}\cdots F_{n_r,m_r}\binom{n-1}{n_0,\ldots,n_r}$$

in the other cases.

The goal is now to translate this recurrence relation for $F_{n,m} = F_{n,m}^{(1)} + F_{n,m}^{(2a)} + F_{n,m}^{(2b)}$ into the language of generating functions. In order to alleviate notation we write F for $F(z,v)$, F_z and F_v for the partial derivatives of F with respect to z and v.

First, let us note that for $F_{n,m}^{(1)}$ we have the following:

$$\begin{aligned} F_{n,m}^{(1)} &= \sum_{r\ge 1}\frac{(n-1)!}{r!} \sum_{\substack{\sum n_i = n-1\\ \sum m_i = m-1}} [z^{n_1}v^{m_1}]F\cdots[z^{n_r}v^{m_r}]F \\ &= (n-1)![z^{n-1}v^{m-1}]\exp(F) \\ &= (n-1)![z^n v^m] zv\exp(F). \end{aligned}$$

Recall that $m_0 \cdot F_{n_0,m_0}/(n_0)! = [z^{n_0}v^{m_0}]vF_v$. We then obtain for $F_{n,m}^{(2a)}$:

$$\begin{aligned} F_{n,m}^{(2a)} &= (n-1)!\sum_{n_0,m_0}([z^{n_0}v^{m_0}]vF_v)\cdot\left([z^{n-n_0-1}v^{m-m_0}]\exp(F)\right) \\ &= (n-1)![z^n v^m] zvF_v\exp(F). \end{aligned}$$

Similarly, for $F_{n,m}^{(2b)}$:

$$F_{n,m}^{(2b)} = (n-1)![z^n v^m] zv\,(zF_z - vF_v)\exp(F).$$

Thus, we can write $F_{n,m}$ as follows:

$$\begin{aligned} F_{n,m} =& n![z^n v^m]F \\ =& (n-1)![z^n v^m] zv\exp(F)\,(1 + zF_z + (1-v)F_v)\,, \end{aligned}$$

or equivalently,

$$\sum_{n\ge 1}\sum_{m\ge 1} n[z^n v^m]F = zF_z = zv\exp(F)\,(1 + zF_z + (1-v)F_v)\,.$$

Thus the bivariate generating function $F(z,v)$ is defined by the following following partial differential equation:

$$\left(1 - zve^{F(z,v)}\right) F_z(z,v) = v(1-v)e^{F(z,v)}F_v(z,v) + ve^{F(z,v)},$$

with the initial condition $F(0,v) = 0$.

This PDE can be solved using the method of characteristics (see Section 2.8). Let us thus regard z, v and F as variables depending on t. We then obtain the following system of ordinary differential equations, the so-called system of characteristic differential equations:

$$\frac{dz}{dt} = 1 - zv \cdot e^{F(z,v)}, \tag{23a}$$

$$\frac{dv}{dt} = e^{F(z,v)} \cdot v(v-1), \tag{23b}$$

$$\frac{dF}{dt} = v \cdot e^{F(z,v)}. \tag{23c}$$

We are now looking for first integrals of the system (23), i.e., for functions $\zeta(z,v,F)$ that are constant along any solution curve – a so-called characteristic curve – of (23). From (23b) and (23c) one obtains the following differential equation

$$\frac{dv}{dF} = v - 1$$

which has the general solution

$$v = c_1 \cdot e^F + 1.$$

Thus a first integral of (23) is given by:

$$\zeta_1(z,v,F) = c_1 = e^{-F} \cdot (v-1). \tag{24}$$

From (23a) and (23c) one obtains, after substituting $v = c_1 \cdot e^F + 1$, the following differential equation

$$\frac{dz}{dF} = \frac{1}{(c_1 e^F + 1) \cdot e^F} - z.$$

The general solution of this first order linear differential equation is

$$z = \frac{F - \ln\left(c_1 \cdot e^F + 1\right) + c_2}{e^F}.$$

After backsubstituting $c_1 = e^{-F} \cdot (v-1)$ one obtains the following first integral of (23) which is independent of (24):

$$\zeta_2(z,v,F) = c_2 = z \cdot e^F - F + \ln(v). \tag{25}$$

The general solution of (23) is thus given by:

$$G(\zeta_1(z,v,F), \zeta_2(z,v,F)) = \text{const.},$$

where G is an arbitrary differentiable function in two variables. One can also solve this equation with respect to the variable z and obtains:

$$z = e^{-F} \cdot \left(F - \ln(v) + g\left(\frac{v-1}{e^F} \right) \right),$$

where g is an arbitrary differentiable function in one variable. In order to specify this function g, we plug the initial condition $F(0,v) = 0$ into this equation. This leads to:

$$g(v-1) = \ln(v),$$

and finally we obtain that the solution of (23) is given by the following functional equation:

$$z = \frac{\ln\left(\frac{e^{F(z,v)}-1+v}{v}\right)}{e^{F(z,v)}}. \tag{26}$$

Extracting coefficients can now be done by applying Lagrange's inversion formula with $\varphi(F) = \exp(F)F/\ln\left(\frac{e^F-1+v}{v}\right)$ and twice Cauchy's integral formula:

$$\begin{aligned}
[z^n]F(z,v) &= \frac{1}{n}[F^{n-1}]\frac{e^{nF}\cdot F^n}{\ln\left(\frac{e^F-1+v}{v}\right)^n} \\
&= \frac{1}{n\cdot 2\pi i}\oint \frac{e^{nF}}{\ln\left(\frac{e^F-1+v}{v}\right)^n} dF \\
&= \frac{1}{n\cdot 2\pi i}\oint \frac{(e^S\cdot v - 1 + v)^{n-1}\cdot e^S v}{S^n} dS \\
&= \frac{1}{n}[S^{n-1}](e^S\cdot v - 1 + v)^{n-1}\cdot e^S v \\
&= \frac{v}{n}\sum_{k=0}^{n-1}\frac{1}{(n-1-k)!}\sum_{\ell=0}^{n-1}\binom{n-1}{\ell}\frac{\ell^k v^\ell}{k!}(v-1)^{n-1-\ell} \\
&= \frac{v}{n}\sum_{\ell=0}^{n-1}\binom{n-1}{\ell} v^\ell (v-1)^{n-1-\ell}\frac{(l+1)^{n-1}}{(n-1)!}
\end{aligned}$$

When passing form the second to the third line above we use the substitution $S = \ln\left(\frac{e^R-1+v}{v}\right)$ for which it holds that $dR = \frac{e^R-1+v}{e^R}dS = \frac{ve^S}{ve^S-1+v}dS$.

What remains to do now is to extract the coefficients in v:

$$\begin{aligned}
F_{n,m} =& n![v^m z^n]F(z,v) \\
&= \sum_{\ell=0}^{n-1}\binom{n-1}{\ell}(\ell+1)^{n-1}[v^{m-\ell-1}](1-v)^{n-1-\ell} \\
&= \sum_{\ell=0}^{n-1}\binom{n-1}{\ell}(\ell+1)^{n-1}\binom{n-\ell-1}{m-\ell-1}(-1)^{m-\ell-1}
\end{aligned}$$

Finally, we obtain the following explicit solution for the numbers $F_{n,m} = T_n \mathbb{P}\{F_n = m\} = n![z^n v^m]F(z,v)$ of size-n trees with exactly m runs:

$$F_{n,m} = \binom{n-1}{m-1} \sum_{\ell=0}^{m-1} (\ell+1)^{n-1} (-1)^{m-1-\ell} \binom{m-1}{\ell}.$$

These numbers can also be described with the help of the Stirling numbers of second kind. Indeed, since the Stirling numbers of second kind $\left\{ {n \atop m} \right\}$ satisfy

$$\left\{ {n \atop m} \right\} = \frac{1}{m!} \sum_{\ell=0}^{m} \binom{m}{\ell} (-1)^{m-\ell} \ell^n, \tag{27}$$

we obtain the following for $n \geq 1$:

$$\begin{aligned}
F_{n,m} &= \binom{n-1}{m-1} \sum_{\ell=1}^{m} \ell^{n-1} (-1)^{m-\ell} \binom{m-1}{\ell-1} \\
&= \binom{n-1}{m-1} \sum_{\ell=1}^{m} \ell^{n-1} (-1)^{m-\ell} \cdot \left(\binom{m}{\ell} - \binom{m-1}{\ell} \right) \\
&= \binom{n-1}{m-1} \left(m! \left\{ {n-1 \atop m} \right\} + (m-1)! \left\{ {n-1 \atop m-1} \right\} \right) \\
&= \binom{n-1}{m-1} \cdot (m-1)! \left(m \left\{ {n-1 \atop m} \right\} + \left\{ {n-1 \atop m-1} \right\} \right) \\
&= \binom{n-1}{m-1} \cdot (m-1)! \left\{ {n \atop m} \right\} \\
&= (n-1)^{\underline{m-1}} \left\{ {n \atop m} \right\}
\end{aligned} \tag{28}$$

In our studies of runs in mappings and of the numbers R_n we will however not use these exact numbers, we only require the generating function $F(z,v)$ computed above in (26).

Runs in (connected) mappings

Having computed the generating functions $F(z,v)$ of ascending runs in trees in the previous section, we now turn to our actual problem, namely the enumeration of mappings by size and number of ascending runs. We introduce the following bivariate generating function

$$R(z,v) = \sum_{n \geq 0} \sum_{m \geq 0} \mathbb{P}\{R_n = m\} M_n \frac{z^n}{n!} v^m, \tag{29}$$

with $M_n = n^n$ the number of n-mappings. Actually, it is easier to study the quantity for connected mapping graphs first. We thus introduce the corresponding g.f. $C(z,v)$. Due to the combinatorial construction of mappings as sets of connected mappings (see Example 2.27), it holds that

$$R(z,v) = e^{C(z,v)}.$$

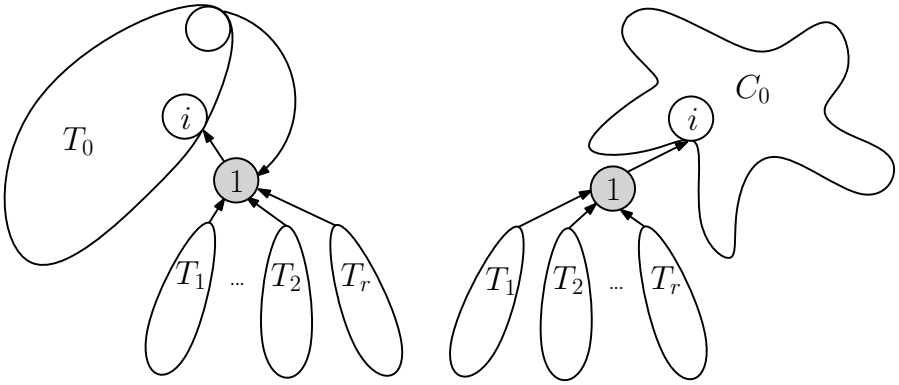


Figure 34: Decomposition of a connected mapping with respect to its smallest node

As for trees, we decompose a connected mapping graph according to the node labelled 1. Here, we have to consider three different cases, two of which are depicted in Figure 34.

Case (1): The node 1 is a cyclic node in a cycle of length one. In this case the mapping graph is simply a tree with root node 1 and an additional loop-edge $(1,1)$. We thus have exactly the same situation as in the first case for trees. See the left hand side of Figure 33. The contribution to $C_{n,m}$ is therefore:

$$C^{(1)}_{n,m} = \sum_{r\geq 1} \frac{1}{r!} \sum_{\substack{\sum n_i = n-1 \\ \sum m_i = m-1}} F_{n_1,m_1} \cdots F_{n_r,m_r} \cdot \binom{n-1}{n_1,\ldots,n_r},$$

Case (2): The node 1 is a cyclic node in a cycle containing more than one element; this case is depicted on the left hand side of Figure 34. The structure of such a mapping can be understood as follows: There is a (non-empty) tree T_0 in which the node 1 has been attached to some node i. Moreover, the root of T_0 is attached to the node 1, thus forming a cycle consisting of the path from i to the root of T_0 and i. Since 1 can have arbitrarily many children, we pick a set of $r \geq 0$ subtrees and attach them to 1. Whether the node 1 contributes to a new run or not depends on whether it is attached at the beginning or somewhere in the middle of a run (as in the second case for trees). Indeed, if T_0 is of size n_0 and has m_0 runs itself, there are m_0 possibilities of attaching 1 at the begin of a run in T_0 thus not creating a new run. Moreover, there are $(n_0 - m_0)$ possibilities of attaching 1 somewhere in the middle of a run and of creating a new run. The contribution to $C_{n,m}$ is therefore:

$$\begin{aligned} C^{(2)}_{n,m} = &\sum_{\substack{1\leq n_0\leq n-1 \\ 1\leq m_0\leq m}} m_0 \sum_{r\geq 0} \frac{1}{r!} \sum_{\substack{\sum n_i = n-n_0-1 \\ \sum m_i = m-m_0}} F_{n_0,m_0} \cdots F_{n_r,m_r} \binom{n-1}{n_0,\ldots,n_r} \\ &+ \sum_{\substack{1\leq n_0\leq n-1 \\ 1\leq m_0\leq m}} (n_0 - m_0) \cdot \sum_{r\geq 0} \frac{1}{r!} \sum_{\substack{\sum n_i = n-n_0-1 \\ \sum m_i = m-m_0-1}} F_{n_0,m_0} \cdots F_{n_r,m_r} \binom{n-1}{n_0,\ldots,n_r} \end{aligned}$$

Case (3): The node 1 is not a cyclic node; this case is depicted on the right hand side of Figure 34. Here, the node 1 is attached to some node i of a connected mapping C_0 and a set of $r \geq 0$ trees are attached to 1. Again, the node 1 contributes to a new run or not depending on where it is attached. Similarly as in the second case, we obtain that the contribution to $C_{n,m}$ is:

$$\begin{aligned} C^{(3)}_{n,m} =& \sum_{\substack{1\leq n_0\leq n-1\\ 1\leq m_0\leq m}} m_0 \sum_{r\geq 0} \frac{C_{n_0,m_0}}{r!} \sum_{\substack{\sum n_i = n-n_0-1\\ \sum m_i = m-m_0}} F_{n_1,m_1}\cdots F_{n_r,m_r} \binom{n-1}{n_0,\ldots,n_r} \\ &+ \sum_{\substack{1\leq n_0\leq n-1\\ 1\leq m_0\leq m}} (n_0-m_0)\cdot \sum_{r\geq 0} \frac{C_{n_0,m_0}}{r!} \sum_{\substack{\sum n_i = n-n_0-1\\ \sum m_i = m-m_0-1}} F_{n_1,m_1}\cdots F_{n_r,m_r} \binom{n-1}{n_0,\ldots,n_r} \\ =& (n-1)![z^n v^m] zv \exp(F)\left(zC_z + (1-v)C_v\right) \end{aligned}$$

Adding up the contributions of these three cases and summing over $n \geq 1$ and $m \geq 1$ we obtain the following:

$$C_z = v\exp(F)\cdot\left(1 + (1-v)F_v + zF_z + (1-v)C_v + zC_z\right)$$

Thus, the bivariate generating function $C(z,v)$ is defined by the following partial differential equation:

$$C_z\left(1 - zv\cdot\exp(F)\right) = v(1-v)\exp(F)C_v + F_z$$

where $F_z(z,v)$ is defined by Equation (26).

Using the auxiliary function $H(z,v)$ that is defined by

$$\exp(H) = \frac{\exp(F) - 1 + v}{v} \tag{30}$$

and that satisfies the following functional equation

$$z = \frac{H}{ve^H + 1 - v} \tag{31}$$

we can solve this PDE. We do not provide details here, since the approach is very similar to the one used for parking functions in Section 9.4 in Chapter 9. We find that the solution is given as follows:

$$C(z,v) = \ln\left(\frac{ve^{H(z,v)} + 1 - v}{ve^{H(z,v)}\big(1 - H(z,v)\big) + 1 - v}\right),$$

with $H(z,v)$ as given above. Checking that this function indeed is a solution to the partial differential equation can be done easily by using implicit differentiation in order to compute the partial derivatives.

We thus also obtain the solution for ascending runs in arbitrary mappings:

$$\begin{aligned} R(z,v) &= \frac{ve^{H(z,v)} + 1 - v}{ve^{H(z,v)}\big(1 - H(z,v)\big) + 1 - v} \\ &= \frac{H/z}{H/z - Hv\exp(H)} = \frac{1}{1 - zv\cdot\exp(H(z,v))} \end{aligned} \tag{32}$$

with $H(z,v)$ defined above.

Exact enumeration formula and asymptotic distribution

Having found the generating functions $F(z,v)$ and $R(z,v)$ for trees and mappings, respectively, we can prove the following interesting connection between ascending runs in trees and in mappings:

Theorem 8.4. *Let $F_{n,m}$ denote the number of Cayley trees of size n with exactly m ascending runs and $R_{n,m}$ the number of n-mappings with exactly m ascending runs. Then for all $n \geq 1$ and $m \geq 1$ the following identity holds:*

$$R_{n,m} = n \cdot F_{n,m}.$$

Proof. We will prove the assertion above at the level of generating functions. Combining Equation (30) and (32) we have:

$$R(z,v) = \frac{1}{1 - z \cdot (\exp(F(z,v) - 1 + v))}.$$

Moreover, implicit differentiation of (26) with respect to z leads to:

$$F_z(z,v) = \frac{\exp(F(z,v)) - 1 + v}{1 - z \cdot (\exp(F(z,v) - 1 + v))}.$$

Thus one easily sees that

$$R(z,v) = 1 + z \cdot F_z(z,v)$$

Extracting coefficients for $n \geq 1$ and $m \geq 1$ leads to

$$R_{n,m} = n![z^n v^m]R(z,v) = n![z^{n-1}v^m]F_z(z,v) = n \cdot F_{n,m}$$

□

A combinatorial proof of this result can be found in Section 8.2.

Combining this result with the formula obtained for $F_{n,m}$ in Equation (28) on page 137, we obtain the following.

Theorem 8.5. *The number $R_{n,m}$ of n-mappings with exactly m runs is given as follows:*

$$R_{n,m} = n^{\underline{m}} \left\{ {n \atop m} \right\} = \frac{n!}{(n-m)!} \left\{ {n \atop m} \right\}. \tag{33}$$

In order to show that the random variable R_n has linear mean and variance and that it is asymptotically normal distributed when suitably standardized, we apply H.-K. Hwang's quasi-power theorem [98]. We then obtain the following:

Theorem 8.6. *Let R_n denote the random variable counting the number of ascending runs in a random n-mapping. Then the expectation and variance satisfy*

$$\mathbb{E}(R_n) = (1 - e^{-1})n + \mathcal{O}(1) \text{ and}$$
$$\mathbb{V}(R_n) = (e^{-1} - 2e^{-2})n + \mathcal{O}(1).$$

Furthermore, the standardized random variable

$$\frac{R_n - \mathbb{E}(R_n)}{\sqrt{\mathbb{V}(R_n)}}$$

is asymptotically Gaussian distributed with a rate of convergence of order $\mathcal{O}(n^{-1/2})$.

Note that Theorem 8.4 implies that the random variable R_n counting the number of runs in n-mappings and the random variable F_n counting the number of runs in trees of size n follow exactly the same distribution. The theorem above thus holds for F_n as well.

Proof. Before we get our hands on the bivariate generating function $R(z, v)$ defined in Equation (29), we will need to take a closer look at the function $H(z, v)$ from Equation (31) that is involved in the functional equation (32) for $R(z, v)$. Once we have a singular expansion of $H(z, v)$, we can obtain an expansion of $R(z, v)$. Extracting coefficients $[z^n]$ in $R(z, v)$ will then then allow us to obtain an asymptotic expansion of the probability generating function $p_n(v)$ and the moment generating function $g_n(s) = p_n(e^s)$. This will finally allow us to apply the quasi-power theorem (Theorem 2.36) and to obtain the statement of this Theorem.

Asymptotic expansion of H. The function $H = H(z, v)$ is defined by a functional equation of the form $H = z\varphi(H)$ where the function φ is defined as $\varphi(u) = v \cdot e^u + 1 - v$. It is thus amenable to the Singular-Inversion theorem (Theorem 2.32). The first and third condition on the function $\varphi(u)$ can easily be checked. Since the constant term φ_0 is equal to 1 and $\varphi_1 = v$ they are clearly fulfilled. For the second condition, we need the characteristic equation (see Equation (9)). Since $\varphi'(u) = ve^u$ it is given as:

$$\begin{aligned} &\varphi(\tau) - \tau \cdot \varphi'(\tau) = 0 \\ \Leftrightarrow\ &ve^\tau + 1 - v - \tau \cdot ve^\tau = 0 \\ \Leftrightarrow\ &\tau = 1 + \frac{1 - v}{ve^\tau}. \end{aligned} \tag{34}$$

For $v = 1$ we have $\tau = 1$ and whenever v is close to 1 a unique solution to Equation (34) exists as can be seen by applying the Implicit Function Theorem. Note that τ is a function of v. This will however be omitted in the notation and in the following, τ will always denote the unique solution of Equation (34). Theorem 2.32 then implies that

the unique dominant singularity of H, seen as a function in z, is at $z = \rho$, with:

$$\rho = \frac{1}{\varphi'(\tau)} = \frac{1}{ve^{\tau}} \tag{35}$$

$$\overset{(34)}{=} \begin{cases} \frac{1}{e}, & \text{if } v = 1, \\ \frac{\tau-1}{1-v}, & \text{if } v \neq 1. \end{cases} \tag{36}$$

Before we continue by giving the asymptotic expansion of H, we want to characterize its unique dominant singularity ρ with the help of a functional equation. From (34) and $\rho = \tau/\varphi(\tau)$ it follows that:

$$\begin{aligned} \tau &= 1 + \rho(1-v) \text{ and} \\ \tau &= \rho \cdot (ve^{\tau} + 1 - v) \\ &= \rho \cdot \left(ve^{1+\rho(1-v)} + 1 - v \right). \end{aligned}$$

Dividing by ρ then leads to

$$\frac{1}{\rho} + 1 - v = ve^{1+\rho(1-v)} + 1 - v,$$

or, equivalently,

$$\rho = \frac{1}{v \cdot e \cdot e^{\rho(1-v)}}.$$

Thus

$$(1-v)\rho e^{\rho(1-v)} = \frac{1-v}{v \cdot e} \tag{37}$$

and ρ can be expressed with the help of the Lambert W function for $v \neq 1$ but in a neighbourhood of 1:

$$\rho = \frac{1}{1-v} W\left(\frac{1-v}{ev} \right).$$

Note that the above expression is not defined for $v = 1$. However, taking limits as v tends to 1, the right-hand side tends to $\exp(-1)$ as expected from Equation (36).

According to Theorem 2.32 the function H admits the following asymptotic expansion around its unique dominant singularity ρ, i.e, for $z \to \rho$:

$$\begin{aligned} H(z,v) =& \tau - \sqrt{\frac{2\varphi(\tau)}{\varphi''(\tau)}} \cdot \sqrt{1 - \frac{z}{\rho}} + K \cdot \left(1 - \frac{z}{\rho}\right) + \mathcal{O}\left(1 - \frac{z}{\rho}\right)^{3/2}, \\ =& \tau - \sqrt{2\tau}\sqrt{1 - \frac{z}{\rho}} + K \cdot \left(1 - \frac{z}{\rho}\right) + \mathcal{O}\left(1 - \frac{z}{\rho}\right)^{3/2}, \end{aligned}$$

where K is a computable constant that will not be needed explicitly in the following.

Asymptotic expansion of R. We can now use this result in order to describe the analytic behaviour of the function $R(z,v)$. Indeed, R inherits a unique dominant singularity at $z=\rho$ from H. This can be seen as follows: Let us first recall that $(R(z,v))^{-1} = 1 - zv \cdot \exp(H(z,v))$. Thus, R has a singularity whenever $zv \cdot \exp(H) = 1$ or equivalently, whenever $H = -\ln(zv)$. Using the functional equation for H given in Equation (31), this is equivalent to:

$$\begin{aligned} z &= \frac{-\ln(zv)}{v \cdot \exp(-\ln(zv)) + 1 - v} \\ \Leftrightarrow 1 &= \frac{-\ln(zv)}{1 + z - zv} \\ \Leftrightarrow z\exp(z(1-v)) &= \frac{1}{ev} \end{aligned}$$

which is exactly the same as Equation (37) for ρ. Thus $z=\rho$ is the unique dominant singularity of R.

In order to provide the asymptotic expansion of R for $z \to \rho$, let us write z as $(z-\rho)+\rho$ and H as $(H-\tau)+\tau$:

$$\begin{aligned} \frac{1}{R} &= 1 - v\rho e^{\tau} \cdot e^{H-\tau} + v\rho\left(1 - \frac{z}{\rho}\right) e^{\tau} \cdot e^{H-\tau} \\ &= 1 - e^{H-\tau} + \left(1 - \frac{z}{\rho}\right) \cdot e^{H-\tau} \end{aligned}$$

since $v\rho e^{\tau} = 1$ by definition of ρ (see Equation (35)). Using the power series expansion of exp and the asymptotic expansion for $H(z,v)$ we obtain the following (for $z \to \rho$):

$$\begin{aligned} e^{H-\tau} &= 1 + (H-\tau) + \mathcal{O}((H-\tau)^2) \\ &= 1 - \sqrt{2\tau}\sqrt{1-\frac{z}{\rho}} + \mathcal{O}(1-\frac{z}{\rho}) \end{aligned}$$

and thus

$$\frac{1}{R} = 1 - 1 + \sqrt{2\tau}\sqrt{1-\frac{z}{\rho}} + \mathcal{O}(1-\frac{z}{\rho}).$$

Finally, with $\tau = 1 + \rho(1-v)$, we obtain

$$\begin{aligned} R(z,v) &= \frac{1}{\sqrt{2\tau}\sqrt{1-\frac{z}{\rho}} + \mathcal{O}\left(1-\frac{z}{\rho}\right)} \\ &= \frac{1}{\sqrt{2}\sqrt{1+\rho(1-v)}\sqrt{1-\frac{z}{\rho}}} \cdot \left(1 + \mathcal{O}\left(\sqrt{1-\frac{z}{\rho}}\right)\right). \end{aligned} \tag{38}$$

Asymptotic expansion of $p_n(v)$. Let us first apply Theorem 2.31 (Singularity Analysis) in order to obtain the asymptotics of the coefficients of $R(z,v)$. It yields

$$\begin{aligned}[z^n]R(z,v) &= \frac{1}{\rho^n}[z^n]R(z\rho,v)\\ &= \frac{1}{\rho^n}\cdot\left[\frac{1}{\sqrt{2}\sqrt{1+\rho(1-v)}}\cdot\binom{n+\frac{1}{2}-1}{n}+\mathcal{O}\left(\frac{1}{n}\right)\right]\\ &= \frac{1}{\rho^n}\cdot\frac{1}{\sqrt{2}\sqrt{1+\rho(1-v)}}\cdot\frac{1}{\sqrt{\pi n}}\cdot\left(1+\sqrt{n}\mathcal{O}\left(\frac{1}{n}\right)\right).\end{aligned}$$

Since $p_n(v) = n!/n^n\cdot[z^n]R(z,v)$, we obtain the following expansion of the probability generating function by applying Stirling's formula:

$$\begin{aligned}p_n(v) =& \frac{n!}{n^n}\frac{1}{\rho^n}\cdot\frac{1}{\sqrt{2}\sqrt{1+\rho(1-v)}}\cdot\frac{1}{\sqrt{\pi n}}\cdot\left(1+\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)\\ =& \frac{n^n\sqrt{2\pi n}}{e^n n^n}\cdot\left(1+\mathcal{O}\left(\frac{1}{n}\right)\right)\frac{1}{\rho^n}\\ &\cdot\frac{1}{\sqrt{2}\sqrt{1+\rho(1-v)}}\cdot\frac{1}{\sqrt{\pi n}}\cdot\left(1+\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)\\ =& \frac{1}{(e\cdot\rho)^n\cdot\sqrt{1+\rho(1-v)}}\cdot\left(1+\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)\end{aligned}\tag{39}$$

Applying the quasi-power theorem. The expansion (39) now immediately gives an asymptotic expansion for the moment generating function:

$$\begin{aligned}g_n(s) =& \frac{1}{(e\cdot\rho)^n\cdot\sqrt{1+\rho(1-e^s)}}\cdot\left(1+\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)\\ =& \exp\left(-n\cdot\ln(e\cdot\rho)-\ln\left(\sqrt{1+\rho(1-e^s)}\right)\right)\\ &\cdot\left(1+\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)\end{aligned}$$

This is precisely the situation where the quasi-power theorem, Theorem 2.36, can be applied. The involved functions are here defined as follows:

$$\begin{aligned}&\phi_n = n, && U(s) = -\ln(e\cdot\rho),\\ &\kappa_n = \sqrt{n} \quad \text{and} && V(s) = -\ln\left(\sqrt{1+\rho(1-e^s)}\right).\end{aligned}$$

Note once again that ρ is not a constant but depends on e^s. We only need to check the variability condition $U''(0)\neq 0$; the other conditions are clearly fulfilled. For this purpose, let us use Equation (35) and write $U(s)$ as follows:

$$U(s) = -\ln\left(\frac{e}{e^s e^\tau}\right) = -1+s+\tau.$$

We thus obtain that the first and second derivative of $U(s)$ are given as follows:

$$\begin{aligned} U'(s) &= 1 + \tau'(e^s) \cdot e^s \\ U''(s) &= e^s \cdot \left(\tau''(e^s) + \tau'(e^s)\right), \end{aligned}$$

where τ is viewed as a function in $v = e^s$. In order to determine τ' and τ'', we will make use of implicit differentiation and the functional equation for τ given in (34):

$$\tau' = -\frac{1}{v^2 e^\tau} - \frac{1-v}{ve^\tau} \cdot \tau', \text{ implying } \tau' = \frac{1}{v \cdot (v - 1 - ve^\tau)}.$$

Furthermore,

$$\begin{aligned} \tau'' =& -\frac{(v - 1 - ve^\tau) + v \cdot (1 - e^\tau - ve^\tau \cdot \tau')}{v^2(v - 1 - ve^\tau)^2} \\ =& \frac{e^\tau}{v(v - 1 - ve^\tau)^3} + \frac{2ve^\tau + 1 - 2v}{v^2(v - 1 - ve^\tau)^2}. \end{aligned}$$

Recalling that $\tau = 1$ for $e^s = v = 1$, we obtain:

$$\begin{aligned} U'(0) &= 1 + \tau'(1) = 1 + \frac{1}{-e^{\tau(1)}} \\ &= 1 - e^{-1} \approx 0.632\ldots, \\ U''(0) &= \tau''(1) + \tau'(1) = \frac{e}{-e^3} + \frac{2e-1}{e^2} - e^{-1} \\ &= -2e^{-2} + e^{-1} \approx 0.0972\ldots. \end{aligned}$$

In a similar way, we could also determine $V'(0)$ and $V''(0)$. Since the calculations are rather tedious and we are not interested in the additive constants in $\mathbb{E}(R_n)$ and $\mathbb{V}(R_n)$), we omit this here. It suffices to note that $V'(0)$ and $V''(0)$ are both real constants. This finally gives us the desired result that

$$\mathbb{E}(R_n) = (1 - e^{-1})n + \mathcal{O}(1) \text{ and } \mathbb{V}(R_n) = (e^{-1} - 2e^{-2})n + \mathcal{O}(1),$$

and that the standardized random variable $(R_n - \mathbb{E}(R_n))/\sqrt{\mathbb{V}(R_n)}$ converges to a standard normal distribution. Moreover, since $\kappa_n^{-1} = n^{-1/2} = \phi_n^{-1/2}$, the speed of convergence is of order $\mathcal{O}(n^{-1/2})$. □

Bijective proofs

In the previous section, we proved two results about ascending runs in trees and mappings where a combinatorial, i.e., bijective explanation would be very desirable. First, the fact that $R_{n,m} = n \cdot F_{n,m}$ and second the fact that the numbers $R_{n,m}$ can very easily be expressed with the help of the Stirling numbers of the second kind. In this Section we will thus provide bijective proofs for these statements.

Theorem 8.7. *There is a bijection between pairs (T, w) where T is a tree of size n with exactly m ascending runs and w is a node in T and n-mappings f with exactly m ascending runs.*

Proof. This statement follows quite directly form Proposition 7.3. For a pair (T, w), let f be its image under the bijection described in Theorem 7.1. We need to show the following: A node v is at the start of a run in T if and only if v is at the start of a run in f. As remarked earlier, a node v is at the start of a run in a tree or mapping if and only if all children or preimages x of v have labels larger than x. Then the statement of this theorem follows by negating the second property in Proposition 7.3. □

Example 8.8. As described in Chapter 7, the pair $(T, 1)$ where T is depicted in Figure 29 is identified with the mapping f_{ex} as depicted in Figure 32. As can be seen very easily both T and f_{ex} have 13 ascending runs and the nodes that are at the start of a run are the following: $1, 2, 3, 4, 5, 6, 8, 9, 12, 13, 15, 18$ and 19. ⊣

Now let us turn to the second interesting fact that asks for a bijective proof, namely that the number $R_{n,m}$ of n-mappings with m ascending runs can be expressed with the help of the Stirling numbers of the second kind as in Theorem 33. We shall prove the following:

Theorem 8.9. *There is a bijection between the set of n-mappings with exactly m runs and the set of pairs (S, x), where S is a set-partition of $[n]$ into m parts and $x = (n_1, \ldots, n_m)$ is an integer sequence of length m. The set partition is given as $S = (S_1, S_2, \ldots, S_m)$ where the parts are ordered decreasingly according to the largest element in each part, i.e., it holds $\max(S_1) > \max(S_2) > \cdots > \max(S_m)$. The sequence x then has to fulfil the following restriction: $n_j \in [n] \setminus \left(\bigcup_{i=1}^{j-1} \min\{\ell \in S_i : \ell > \max(S_j)\} \right)$.*

The idea of the bijection is to successively decompose the mapping into ascending runs. This is done by starting with a run ending at the largest element of the mapping, then one ending at the next-largest element that has not been involved yet aso. The runs then correspond to blocks of the partition. In order to keep track of how these runs where "glued" together and to be able to reconstruct the mapping, we additionally store the image of the last element of each run in the sequence x.

Proof. First we remark that this indeed will prove that

$$R_{n,m} = n^{\underline{m}} \left\{ {n \atop m} \right\}$$

since the number of set-partitions of $[n]$ into m parts is given by $\left\{ {n \atop m} \right\}$ and the number of sequences x satisfying the restrictions is given by $n \cdot (n-1) \cdots (n-m+1) = n^{\underline{m}}$.

$S_1 = \{19\}$	$n_1 = 13$
$S_2 = \{18\}$	$n_2 = 13$
$S_3 = \{17, 13\}$	$n_3 = 1$
$S_4 = \{16, 9\}$	$n_4 = 4$
$S_5 = \{15\}$	$n_5 = 7$
$S_6 = \{14, 4\}$	$n_6 = 4$
$S_7 = \{12\}$	$n_7 = 7$
$S_8 = \{11, 7, 6\}$	$n_8 = 17$
$S_9 = \{10, 8\}$	$n_9 = 10$
$S_{10} = \{5\}$	$n_{10} = 17$
$S_{11} = \{3\}$	$n_{11} = 10$
$S_{12} = \{2\}$	$n_{12} = 1$
$S_{13} = \{1\}$	$n_{13} = 7$

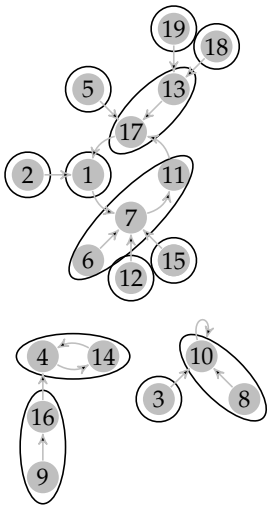


Figure 35: Example of the bijection described in the proof of Theorem 8.9 for the mapping depicted in Figure 11

To prove the theorem we consider an n-mapping with exactly m runs and iterate the following procedure, where we colour the elements of the mapping until all elements are coloured.

- In the j-th step we consider the largest element in the mapping which has not been coloured so far; let us denote it by $s_1^{(j)}$. Consider all preimages of $s_1^{(j)}$ with a label smaller than $s_1^{(j)}$ and, if there are such ones, take the one with largest label amongst them; let us denote this element by $s_2^{(j)}$. Then iterate this step with $s_2^{(j)}$, i.e., amongst all preimages of $s_2^{(j)}$ with a label smaller than $s_2^{(j)}$ take the one with largest label, which is denoted by $s_3^{(j)}$. After finitely many steps we arrive at an element $s_{k_j}^{(j)}$, which does not have preimages with a smaller label. We then define the set $S_j := \{s_1^{(j)}, \dots, s_{k_j}^{(j)}\}$. Note that in the mapping graph this corresponds to a path $s_{k_j}^{(j)} \to \cdots \to s_2^{(j)} \to s_1^{(j)}$ with increasing labels on it.
- Additionally we store in n_j the image of $s_1^{(j)}$. Clearly $s_1^{(j)}$ is in $[n]$. Due to the construction further restrictions hold: Indeed, if $i < j$, n_j cannot be the smallest element in S_i larger than $s_1^{(j)}$, since otherwise $s_1^{(j)}$ would have been chosen during the construction of the set S_i.
- Finally colour all elements of the mapping contained in S_j.

Since the mapping contains exactly m runs and the smallest element in each set S_j corresponds to the minimal element of a run the

procedure stops after exactly m steps. It thus defines a pair of a set partition $S = (S_1, \dots, S_m)$ and a sequence $x = (n_1, \dots, n_m)$ with the given restrictions.

If the pair (S, x) is given, the corresponding mapping can easily be reconstructed. Indeed, the partition S gives us a decomposition of the mapping into ascending runs and the sequence x tells us how these runs have to be linked to each other. The inverse of this bijection can therefore be defined in a straightforward way. □

Example 8.10. The construction of the partition S and the sequence x for the mapping described in Figure 11 can be found in Figure 35. Let us exemplarily explain how the set S_8 is constructed. At this point, the elements in $\cup_{i=1}^{7} S_i$, i.e., 19, 18, 17, 13, 16, 9, 15, 14, 4 and 12, have already been coloured. Thus, the largest element that has not been coloured so far is $11 = s_1^{(8)}$. For $s_2^{(8)}$, we consider the preimages of 11 that have a label smaller than 11. The only such element is 7 and thus $s_2^{(8)} = 7$. Next, the preimages of 7 are 6, 12 and 15 and thus $s_3^{(8)} = 6$. Since 6 does not have any preimages, we stop here and $\mathcal{S}_8 = \{11, 7, 6\}$. Since the image of 11 is 17, we set $n_8 = 17$. ⊣

8.3 SUMMARY OF THE RESULTS

Let us briefly discuss the results of this chapter.

We started with a study of ascents in mappings – which is easy since the probability of a node i being an ascent in a random n-mapping is simply $(n - i)/n$. The results obtained here can be translated to trees, where the study of ascents is less straightforward.

As can be seen very easily a permutation with k ascents consists of exactly $k + 1$ descending runs and thus the study of these two permutation statistics is equivalent. This is in contrast to the case of mappings, where results on ascents can not directly be translated to results on descending or ascending runs. We thus proceeded with the study of ascending runs in mappings which was the main focus of this chapter. Using a PDE we obtained exact enumeration formulæ involving the Stirling numbers of the second kind.

Both for ascents and for ascending runs in mappings (and trees), we could also obtain limiting distribution results and show convergence to a normal distribution. Let us compare this to the known results for the Eulerian numbers that enumerate permutations by their length and their number of ascents.

In [46] it is shown that the random variable X_n counting the number of ascents in a random permutation of length n satisfies a central limit theorem. Indeed, with $\mathbb{E}(X_n) = \frac{n-1}{2}$ and $\mathbb{V}(X_n) = \frac{n+1}{12}$ it holds that the standardized random variable $(X_n - \mathbb{E}(X_n))/\sqrt{\mathbb{V}(X_n)}$ converges in distribution to a standard normal distribution. Equivalently, the random variable $Y_n = X_n + 1$ counting the number of ascending

runs in a random permutation of length n satisfies a central limit theorem with $\mathbb{E}(Y_n) = \frac{n+1}{2}$ and $\mathbb{V}(Y_n) = \mathbb{V}(X_n)$. Thus, both for ascents and for ascending runs we have $\mathbb{E} \sim n/2$ and $\mathbb{V} \sim n/12 = 0.08\dot{3} \cdot n$.

Here, we could show similar results for ascending runs and ascents in trees and mappings. Indeed, we also obtained a linear mean and variance and convergence in distribution to a standard normal distribution of the standardized random variable. However, the involved coefficients are not the same:

- $\mathbb{E}(A_n) \sim n/2$ as well as $\mathbb{V}(A_n) \sim n/6$ for ascents and
- $\mathbb{E}(R_n) \sim (1 - e^{-1})n$ with $1 - e^{-1} \approx 0.632\ldots$ as well as $\mathbb{V}(R_n) \sim (e^{-1} - 2e^{-2})n$ with $e^{-1} - 2e^{-2} \approx 0.0972\ldots$ for ascending runs.

9

PARKING IN TREES AND MAPPINGS

This chapter is based on joint work with Alois Panholzer. A manuscript has recently been submitted to a journal and a preprint can be found on `arxiv.org` [44].

In this chapter, we apply the well-known concept of parking functions to Cayley trees and to mappings by considering the nodes as parking spaces and the directed edges as one-way streets: Each driver has a preferred parking space and starting with this node she[1] follows the edges in the graph until she either finds a free parking space or all reachable parking spaces are occupied. If all drivers are successful we speak about a parking function for the tree or mapping.

A more detailed introduction to the topic of this chapter, related literature and the organization of this chapter can be found in the following section.

9.1 INTRODUCTION

Parking functions are combinatorial objects originally introduced by Konheim and Weiss [113] during their studies of the so-called linear probing collision resolution scheme for hash tables. Since then, parking functions have been studied extensively and many connections to various other combinatorial objects such as forests, hyperplane arrangements, acyclic functions and non-crossing partitions have been revealed, see, e.g., [146].

An illustrative description of parking functions is as follows: consider a one-way street with n parking spaces numbered from 1 to n and a sequence of m drivers with preferred parking spaces $s_1, s_2, \ldots, s_m$. The drivers arrive sequentially and each driver k, $1 \leq k \leq m$, tries to park at her preferred parking space with address $s_k \in [n]$. If it is free she parks. Otherwise she moves further in the allowed direction, thus examining parking spaces $s_k + 1, s_k + 2, \ldots$, until she finds a free parking space where she parks. If there is no such parking space she leaves the street without parking. A parking function is then a sequence $(s_1, \ldots, s_m) \in [n]^m$ of addresses such that all m drivers are able to park. It has been shown already in [113] that there are exactly $P_{n,m} = (n+1-m) \cdot (n+1)^{m-1}$ parking functions for n parking spaces and $0 \leq m \leq n$ drivers.

The notion of parking functions has been generalized in various ways, yielding, e.g., (a, b)-parking functions [159], bucket parking functions [23], x-parking functions [148], or G-parking functions [135].

1 In the following we use "she" as a generic pronoun.

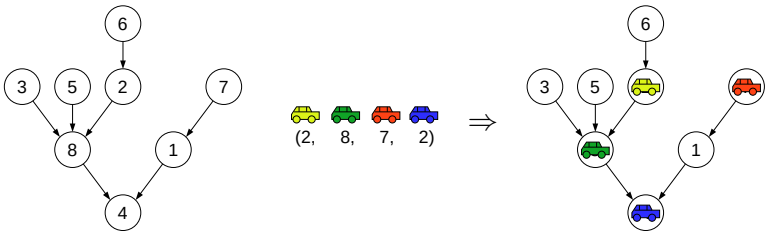


Figure 36: An example of a $(8,4)$-tree parking function

Another natural generalization that has however not been considered yet is the following: Starting with the original definition of parking functions as a vivid description of a simple collision resolution scheme, we apply it to other objects of interest, namely, to rooted trees and mappings.

First, when allowing branches in the road net, this collision resolution scheme leads to a natural generalization of parking functions to rooted labelled trees, i.e., Cayley trees. Consider a Cayley tree T of size $|T| = n$. As usual, we assume that the edges of the tree are oriented towards the root node, which we will often denote by $\mathrm{root}(T)$. We thus view edges as "one-way streets". Now, we consider a sequence of m drivers, where again each driver has her preferred parking space which in this case is a node in the tree, respectively its label. The drivers arrive sequentially and each driver k, $1 \leq k \leq m$, tries to park at her preferred parking space with address $s_k \in [n]$. If it is free she parks. Otherwise she follows the edges towards the root node and parks at the first empty node, if there is such one. If there is no empty node, she leaves the road net, i.e., the tree without parking. A sequence $s \in [n]^m$ of addresses is then called a parking function for the tree T, if all drivers are successful, i.e., if all drivers are able to find a parking space. More precisely, we will call a pair (T, s) an (n, m)-tree parking function, if T is a rooted labelled tree of size n and $s \in [n]^m$ is a parking function for T with m drivers.

Example 9.1. Consider the tree T of size 8 depicted in Figure 36. Moreover, consider the sequence $s = (2, 8, 7, 2)$ of addresses of preferred parking spaces for 4 drivers. All drivers are successful, thus (T, s) yields a $(8, 4)$-tree parking function. Conversely, the sequence $(2, 8, 8, 2)$ is not a parking function for T, since the fourth driver is not able to park. ⊣

Second, one can go a step further and consider structures in general for which the simple collision resolution scheme is applicable, i.e., for which a driver reaching an occupied parking space can move on in a unique way to reach a new parking space. This naturally leads to a generalization of parking functions to mappings: Consider the set $[n]$ of addresses and a mapping $f : [n] \to [n]$. If a driver reaches

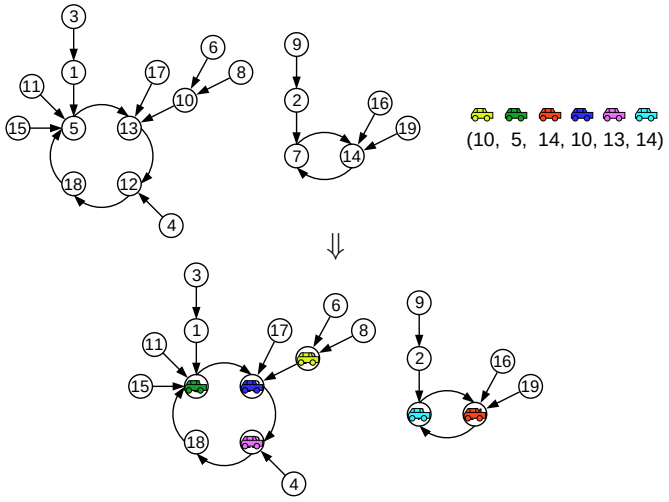


Figure 37: An example of a $(19, 6)$-mapping parking function

address i and is unable to park there, she moves on to the parking space with address $j = f(i)$ for her next trial. The road net is then the functional digraph G_f of the mapping f. Again, the drivers arrive sequentially and each driver has her preferred parking space which is a node in the graph. If it is empty she will park, otherwise she follows the edges and parks at the first empty node, if such one exists. Otherwise she cannot park since she would be caught in an endless loop. A sequence $s \in [n]^m$ of addresses is then called a parking function for the mapping f if all drivers are successful, i.e., all drivers find a parking space. A pair (f, s) is called an (n, m)-mapping parking function, if f is an n-mapping and $s \in [n]^m$ is a parking function for f with m drivers. Note that we will always identify mappings with their functional digraphs in the course of this chapter.

Example 9.2. Consider the mapping f of size 19 depicted in Figure 37 and the sequence $s = (10, 5, 14, 10, 13, 14)$ of addresses of preferred parking spaces for 6 drivers. All drivers are successful, thus (f, s) yields a $(19, 6)$-mapping parking function. When a new driver with preferred parking space 7 arrives, she is not able to park. Thus $(10, 5, 14, 10, 13, 14, 7)$ is not a parking function for f. ⊣

To each (n, m)-tree parking function (T, s) (or (n, m)-mapping parking function (f, s)), we associate the corresponding *output*-function $\pi := \pi_{(T,s)}$ (or $\pi := \pi_{(f,s)}$), with $\pi : [m] \to [n]$, where $\pi(k)$ is the address of the parking space (i.e., the label of the node) in which the k-th driver ends up parking. Of course, π is an injection and for the particular case $m = n$ a bijection; thus in the latter case one may speak about the output-permutation π. This notion will be useful in subsequent considerations describing characterizations and bijections for tree and mapping parking functions.

Example 9.3. Consider again the tree parking function represented in Figure 36. The parking positions of the drivers are given by the sequence $(2, 8, 7, 4)$ defining the output-function $\pi_{(T,s)}$. ⊣

Obviously, both concepts of parking functions generalize ordinary parking functions: first, each ordinary parking function on $[n]$ can be identified with a parking function for the linear tree (i.e., the chain) $1 - 2 - \cdots - n$ with root n, and second, each parking function for a tree T of size n can be identified with a parking function for the functional digraph which is obtained from T by adding a loop-edge to the root.

We start our studies of tree and mapping parking functions by giving some of their basic properties and characterizations in Section 9.2. This extends corresponding properties and characterizations for ordinary parking functions. As an application we can characterize the extremal values for the number of parking functions with $0 \leq m \leq n$ drivers amongst all size-n trees.

The main focus of this chapter lies on the exact and asymptotic enumeration of the total number of (n, m)-tree parking functions and (n, m)-mapping parking functions. This is done in Sections 9.3 and 9.4. We are interested in the exact and asymptotic behaviour of the quantities

$$F_{n,m} := |\{(T, s) \,:\, T \in \mathcal{T}_n, s \in [n]^m \text{ a parking function for } T\}|,$$
$$M_{n,m} := |\{(f, s) \,:\, f \in \mathcal{M}_n, s \in [n]^m \text{ a parking function for } f\}|,$$

counting the total number of (n, m)-tree parking functions and (n, m)-mapping parking functions.

In order to get exact enumeration results we use suitable combinatorial decompositions of the objects which give recursive descriptions of the quantities of interest. The recurrences occurring can be treated by a generating functions approach yielding partial differential equations. These differential equations allow for implicit characterizations of the generating functions. This treatment is divided into two main steps: First, in Section 9.3 we treat the important particular case $m = n$, i.e., we consider parking functions where the number of drivers is equal to the number of parking spaces. Second, the general case in which the number of drivers m is less than or equal to the number of parking spaces n is then treated in Section 9.4.

To give a complete picture of the asymptotic behaviour of $M_{n,m}$ (and thus also $F_{n,m}$) depending on the growth of m w.r.t. n requires a more detailed study using saddle point methods. In Section 9.4, we consider the probability $p_{n,m} := M_{n,m}/n^{n+m} = F_{n,m}/n^{n+m-1}$ that a randomly chosen pair (f, s) of an n-mapping f and a sequence s of m addresses is indeed a parking function and thus the probability that all drivers are successful.

In Section 9.5 we summarize the results of this chapter.

9.2 BASIC PROPERTIES OF PARKING FUNCTIONS FOR TREES AND MAPPINGS

In this section we will state and prove some basic facts on parking functions for trees and mappings. The following notation will turn out to be useful: Given an n-mapping f, we define a binary relation $\preceq_f$ on $[n]$ via

$$i \preceq_f j :\Longleftrightarrow \exists k \in \mathbb{N} : f^k(i) = j.$$

Thus $i \preceq_f j$ holds if there exists a directed path from i to j in the functional digraph G_f, and we say that j is a successor of i or that i is a predecessor of j. In this context a one-way street represents a total order, a tree represents a certain partial order, where the root node is the maximal element (to be precise, a partially ordered set with maximal element, where every interval is a chain – this is also called tree in set theory) and a mapping represents a certain pre-order (i.e., binary relation that is transitive and reflexive).

Changing the order in a parking function

For ordinary parking functions the following holds: changing the order of the elements of a sequence does not affect its property of being a parking function or not. This fact can easily be generalized to parking functions for mappings (which might also be trees).

Lemma 9.4. *A function $s : [m] \to [n]$ is a parking function for a mapping $f : [n] \to [n]$ if and only if $s \circ \sigma$ is a parking function for f for any permutation σ on $[m]$.*

Proof. Since each permutation σ on $[m]$ can be obtained by a sequence of transpositions of consecutive elements, i.e., $\sigma = \tau_r \circ \tau_{r-1} \circ \cdots \circ \tau_1$, with $\tau_i = (k_i\, k_i + 1)$, $1 \leq k_i \leq m-1$, $1 \leq i \leq r$, it suffices to prove the following: if s is a parking function for f, then $s \circ \sigma$ is a parking function for f for any transposition σ of consecutive elements, i.e., for any permutation σ on $[m]$ that swaps two consecutive elements and leaves the other elements fixed. The statement for general σ follows from this by iteration.

Thus, let $s : [m] \to [n]$ be a parking function for $f : [n] \to [n]$ and $s' = s \circ \sigma$, where $\sigma = (k\, k+1)$, with $1 \leq k \leq m-1$, i.e., $\sigma(k) = k+1$, $\sigma(k+1) = k$, and $\sigma(j) = j$ otherwise. In other words, s' is the parking sequence obtained from s by changing the order of the k-th and the $(k+1)$-th car.

In the following the mapping f is fixed and we denote by $\pi_s = \pi_{(f,s)}$ the output-function of the parking function s and consider the parking paths of the drivers: the path $y_j = s_j \rightsquigarrow \pi_s(j)$ denotes the parking path of the j-th driver of s in the mapping graph G_f starting with the preferred parking space s_j and ending with the parking position $\pi_s(j)$. In order to show that s' is still a parking function, we

have to show that all cars can successfully be parked using s'. In the following we do this and also determine the output-function $\pi_{s'}$ of s' (and thus the parking paths $y'_j = s'_j \rightsquigarrow \pi_{s'}(j)$, $1 \leq j \leq m$, of the drivers of s').

Clearly, the parking paths of the first $(k-1)$ cars are not affected by the swapping of the k-th and the $(k+1)$-th car and we have that $\pi_{s'}(1) = \pi_s(1), \ldots, \pi_{s'}(k-1) = \pi_s(k-1)$. For the k-th and the $(k+1)$-th car we will distinguish between two cases according to the parking paths $y_k = s_k \rightsquigarrow \pi_s(k)$ and $y_{k+1} = s_{k+1} \rightsquigarrow \pi_s(k+1)$.

(*a*) Case $y_k \cap y_{k+1} = \emptyset$: Since the parking paths y_k and y_{k+1} are disjoint, swapping the k-th and the $(k+1)$-th car simply also swaps the corresponding parking paths, i.e., $y'_k = y_{k+1}$ and $y'_{k+1} = y_k$, and in particular $\pi_{s'}(k) = \pi_s(k+1)$ and $\pi_{s'}(k+1) = \pi_s(k)$.

(*b*) Case $y_k \cap y_{k+1} \neq \emptyset$: Let us denote by v the first node in the path y_k that also occurs in y_{k+1}. Then, according to the parking procedure, the parking paths can be decomposed as follows:

$$\begin{aligned} y_k &= s_k \rightsquigarrow v \rightsquigarrow \pi_s(k) \quad \text{and} \\ y_{k+1} &= s_{k+1} \rightsquigarrow v \rightsquigarrow \pi_s(k) \rightsquigarrow \pi_s(k+1), \end{aligned}$$

i.e. $\pi_s(k+1)$ is a proper successor of $\pi_s(k)$, i.e., $\pi_s(k) \prec_f \pi_s(k+1)$. Thus, when swapping the k-th and the $(k+1)$-th car, both cars can also be parked yielding the parking paths

$$\begin{aligned} y'_k &= s_{k+1} \rightsquigarrow v \rightsquigarrow \pi_s(k) \quad \text{and} \\ y'_{k+1} &= s_k \rightsquigarrow v \rightsquigarrow \pi_s(k) \rightsquigarrow \pi_s(k+1). \end{aligned}$$

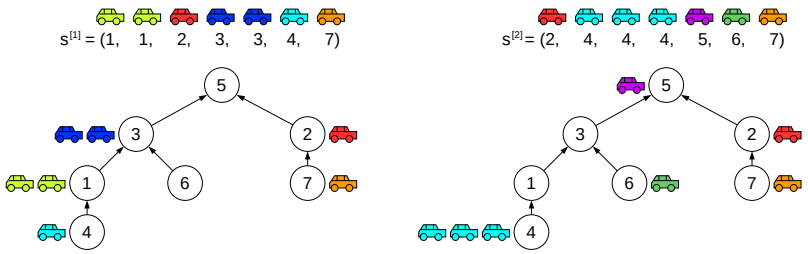
In particular, we obtain $\pi_{s'}(k) = \pi_s(k)$ and $\pi_{s'}(k+1) = \pi_s(k+1)$.

Thus, in any case we get $\{\pi_s(k), \pi_s(k+1)\} = \{\pi_{s'}(k), \pi_{s'}(k+1)\}$, and consequently swapping the k-th and the $(k+1)$-th car does not change the parking paths of the subsequent cars and we obtain $\pi_{s'}(j) = \pi_s(j)$, $k+2 \leq j \leq m$. So all drivers in the parking sequence s' are successful and s' is indeed a parking function for f. □

Alternative characterizations of parking functions

Using the fact that reordering the elements of a function does not have any influence on whether it is a parking function or not, one can obtain the following well-known simpler characterization of ordinary parking functions (see, e.g., [146]): A sequence $s \in [n]^n$ is a parking function if and only if it is a major function, i.e., the sorted rearrangement s' of the sequence s satisfies:

$$s'_j \leq j, \quad \text{for all } j \in [n].$$



j	1	2	3	4	5	6	7
$p(j)$	2	2	4	1	7	1	1
$q_1(j)$	3	2	5	1	7	0	1
$q_2(j)$	3	2	4	3	7	1	1

Figure 38: Exemplifying the characterization of generalized parking functions given in Lemma 9.5 for a tree T of size 7. The sequence $s^{[1]}$ represented on the left-hand-side does not give a parking function for T, whereas the sequence $s^{[2]}$ represented on the right-hand-side does.

This statement can be reformulated in the following way: A sequence $s \in [n]^n$ is a parking function if and only if for every $j \in [n]$, it does not contain more than $(n-j)$ elements that are larger than j. Or again in other words, there must be at least j elements that are not larger than j:

$$|\{k \in [n] : s_k \leq j\}| \geq j, \quad \text{for all } j \in [n]. \tag{40}$$

Now, this characterization of parking functions can easily be generalized to parking functions for trees and mappings. Indeed, in (40) we merely need to replace the $\leq$ and $\geq$ relation which come from the order on the elements $1, 2, \ldots, n$ represented by the one-way street of length n by the binary relation given by the respective tree or mapping. The following characterization of (n,n)-mapping parking functions (which might also be trees) is now possible.

Lemma 9.5. *Given an n-mapping f and a sequence $s \in [n]^n$, let $p(j) := |\{i \in [n] : i \preceq_f j\}|$ denote the number of predecessors of j and $q(j) := |\{k \in [n] : s_k \preceq_f j\}|$ the number of drivers whose preferred parking spaces are predecessors of j. Then s is a mapping parking function for f if and only if*

$$q(j) \geq p(j), \quad \textit{for all } j \in [n].$$

Example 9.6. Consider the two pairs $(T, s^{[1]})$ and $(T, s^{[2]})$ represented in Figure 38. The sequence $s^{[1]} = (1,1,2,3,3,4,7)$ does not give a parking function for T, whereas the sequence $s^{[2]} = (2,4,4,4,5,6,7)$ represented on the right-hand-side does. This can be seen in the following way, where we denote by $q_1(j)$ and $q_2(j)$ the quantity $q(j)$ for $s^{[1]}$ and

$s^{[2]}$, respectively: for the sequence $s^{[1]}$ we have $q_1(6) = 0 < 1 = p(6)$ thus violating the condition $q(j) \geq p(j)$, whereas each element $p(j)$ is smaller or equal to the corresponding element $q_2(j)$. ⊣

Proof of Lemma 9.5. First, assume that $q(j) < p(j)$ holds for some $j \in [n]$. Let us denote by $P(j) := \{i \in [n] : i \preceq_j j\}$ the set of predecessors of j. Obviously, if $s_k \notin P(j)$ then the k-th driver will not get a parking space in $P(j)$. Thus, at most $q(j) = |\{k \in [n] : s_k \in P(j)\}|$ drivers are able to park in $P(j)$. In other words, at least $p(j) - q(j) > 0$ parking spaces in $P(j)$ remain free. Since there is the same number of cars and of parking spaces, this means that at least one driver will not be able to park successfully. Thus s is not a parking function for f.

Next, assume that $q(j) \geq p(j)$ holds for all $j \in [n]$. It will be sufficient to show the following: Let $s \in [n]^n$ be a parking sequence such that $q(j) \geq p(j)$ for some j, then node j will be occupied after applying the parking procedure. Due to the assumption above we may then conclude that all nodes will be occupied after applying the parking procedure and thus that all n drivers are successful, which means that s indeed is a parking function for f.

To prove the assertion above we distinguish between two cases:

(*a*) j is not a cyclic node: Then the set $P(j)$ of predecessors of j is a tree. If there is a driver k with preferred parking space $s_k = j$ then in any case node j will be occupied. Thus let us assume that $s_k \neq j$, for all $j \in [n]$. Let us further assume that $i_1, \ldots, i_r$ are the preimages of j, i.e., $f(i_t) = j$, $1 \leq t \leq r$. Since $q(j) \geq p(j)$, but no driver wishes to park at j, it holds that there exists a preimage i_ℓ, such that $q(i_\ell) > p(i_\ell)$. This means that there must be at least one driver appearing in $P(i_\ell)$ that is not able to get a parking space in $P(i_\ell)$ and thus, according to the parking procedure, she has to pass the edge (i_ℓ, j). Consequently, node j will be occupied.

(*b*) j is a cyclic node: If the edge $(j, f(j))$ is passed by some driver during the application of the parking procedure this necessarily implies that the node j is occupied. Thus let us assume that the edge $(j, f(j))$ will never be passed while carrying out the parking procedure. Then we may remove the edge $(j, f(j))$ from G_f without influencing the outcome of the parking procedure. By doing so, node j becomes a non-cyclic node and according to case (a), node j will be occupied.

□

Now let us turn to parking functions, where the number of drivers does not necessarily coincide with the number of parking spaces. It is well-known and easy to see that a parking sequence $s : [m] \to [n]$ on a one-way street is a parking function if and only if

$$| \{k \in [m] : s_k \geq j\} | \leq n - j + 1, \quad \text{for all } j \in [n]. \tag{41}$$

This characterization can be generalized to (n,m)-tree parking functions as follows.

Lemma 9.7. *Given a rooted labelled tree T of size $|T| = n$ and a sequence $s \in [n]^m$. Then s is a tree parking function for T if and only if*

$$|\{k \in [m] : s_k \in T'\}| \leq |T'|, \quad \text{for all subtrees } T' \text{ of } T \text{ containing } \mathsf{root}(T).$$

Proof. First, let us assume that there exists a subtree T' of T containing $\mathsf{root}(T)$, such that $q(T') := |Q(T')| := |\{k \in [m] : s_k \in T'\}| > |T'|$. Clearly, the possible parking spaces for any driver k with preferred parking space $s_k \in T'$ form a subset of T'. Here, the number of such drivers $q(T')$ exceeds the amount of parking spaces $|T'|$ and thus at least one of the drivers in $Q(T')$ will be unsuccessful. Thus s is not a parking function for T.

Next, let us assume that s is not a parking function for T. Let us further assume that $\ell \in [m]$ is the first unsuccessful driver in s when applying the parking procedure. We consider the situation after the first ℓ drivers: Define T' as the maximal subtree of T containing $\mathsf{root}(T)$ and only such nodes that are occupied by one of the first $\ell - 1$ cars. Of course, since the ℓ-th driver is unsuccessful, the $\mathsf{root}(T)$ has to be occupied by one of the first $\ell - 1$ cars, anyway. Due to the maximality condition of T', it holds that each driver ℓ that has parked in T' must have had her preferred parking space in T', thus $|\{k \in [\ell - 1] : s_k \in T'\}| = |T'|$. Since the ℓ-th driver is unsuccessful, her preferred parking space is also in T', yielding $|\{k \in [\ell] : s_k \in T'\}| > |T'|$. Of course, this implies $|\{k \in [m] : s_k \in T'\}| > |T'|$, for the subtree T' of T containing $\mathsf{root}(T)$. □

We remark that the characterization above could be also extended to mapping parking functions (where one has to consider connected subgraphs of G_f containing all cyclic nodes of the respective component). Since we will not make use of it in the remainder of this chapter we omit it here.

Extremal cases for the number of parking functions

Given an n-mapping $f : [n] \to [n]$ (which might be a tree), let us denote by $S(f,m)$ the number of parking functions $s \in [n]^m$ for f with m drivers. So far we are not aware of enumeration formulæ for the numbers $S(f,m)$ for general f. In Sections 9.3 and 9.4 however, we will compute the total number $F_{n,m} := \sum_{T \in \mathcal{T}_n} S(T,m)$ and $M_{n,m} := \sum_{f \in \mathcal{M}_n} S(f,m)$ of (n,m)-tree and (n,m)-mapping parking functions.

Before continuing, we first state the obvious fact that isomorphic mappings (or trees) yield the same number of mapping (or tree) parking functions, since one simply has to adapt the preferred parking spaces of the drivers according to the relabelling.

Proposition 9.8. *Let f and f' two isomorphic n-mappings, i.e., there exists a bijective function $\sigma : [n] \to [n]$, such that $f' = \sigma \circ f \circ \sigma^{-1}$. Then for $0 \leq m \leq n$ it holds*

$$S(f, m) = S(f', m).$$

Proof. First note that the corresponding functional digraphs $G_f = ([n], E)$ and $G_{f'} = ([n], E')$ are isomorphic in the graph theoretic sense, since

$$\begin{aligned} e = (i, j) \in E &\Leftrightarrow j = f(i) \Leftrightarrow \sigma(j) = \sigma(f(i)) \\ &\Leftrightarrow \sigma(j) = f'(\sigma(i)) \Leftrightarrow \sigma(e) = (\sigma(i), \sigma(j)) \in E'. \end{aligned}$$

It is then an easy task to show via induction that a function $s = (s_1, \ldots, s_m) \in [n]^m$ is a parking function for f if and only if $s' := \sigma \circ s = (\sigma(s_1), \ldots, \sigma(s_m))$ is a parking function for f'. □

In the following we consider the extremal cases of $S(f, m)$. Obviously, each surjective function $s \in [n]^m$ is a parking function for every mapping $f \in \mathcal{M}_n$, which yields the trivial bounds

$$n^{\underline{m}} \leq S(f, m) \leq n^m, \quad \text{for } f \in \mathcal{M}_n. \tag{42}$$

These bounds are actually tight. Indeed, for the identity $\mathsf{id}_n : j \mapsto j$, for $j \in [n]$, we have $S(\mathsf{id}_n, m) = n^{\underline{m}}$ since no collisions may occur. Moreover, for

$$\mathsf{cycle}_n : j \mapsto \begin{cases} j + 1, & \text{for } 1 \leq j \leq n - 1, \\ 1, & \text{for } j = n, \end{cases}$$

a cycle of length n, it holds that $S(\mathsf{cycle}_n, m) = n^m$.

The situation becomes more interesting when we restrict ourselves to trees. The following simple tree operation will turn out to be useful in order to identify the extremal cases. Let T be a rooted labelled tree and v a node of T. Furthermore, let U be a subtree of T attached to v such that $T \setminus U$ is still a tree, i.e., the graph consisting of all edges not contained in U has one connected component. For a node w not contained in U, we denote by $\mathsf{reallocate}\left(T \,\middle|\, \begin{smallmatrix} U \\ \downarrow \\ v \end{smallmatrix} \mapsto \begin{smallmatrix} U \\ \downarrow \\ w \end{smallmatrix}\right)$ the tree operation of first detaching the subtree U from v and then attaching it to w. See Figure 39 for an illustration.

Lemma 9.9. *Let T be a rooted labelled tree and $w \preceq_T v$, for two nodes $v, w \in T$. Furthermore, let U be a subtree of T attached to v that does not contain w such that $T \setminus U$ is still a tree. Let us denote by $\tilde{T}$ the tree which is obtained by reallocating U from v to w, i.e.,*

$$\tilde{T} = \mathsf{reallocate}\left(T \,\middle|\, \begin{smallmatrix} U \\ \downarrow \\ v \end{smallmatrix} \mapsto \begin{smallmatrix} U \\ \downarrow \\ w \end{smallmatrix}\right).$$

Then it holds that

$$S(\tilde{T}, m) \geq S(T, m).$$

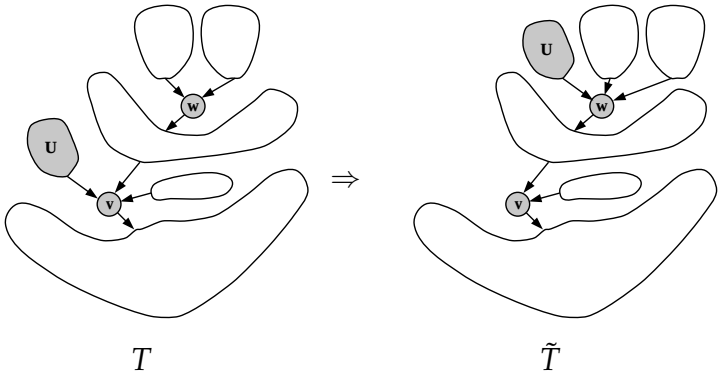


Figure 39: Illustrating the tree operation of reallocating the subtree U from v to w in T which yields the tree $\tilde{T}$. Here the nodes v and w satisfy $w \preceq_T v$, as required in the proof of Lemma 9.9.

Proof. By applying Lemma 9.5 we will show that each parking function $s \in [n]^m$ for T is also a parking function for $\tilde{T}$. For this purpose, let s be a parking function for T and consider a node j in T. We need to show that $q(j) \geq p(j)$ implies that $\tilde{q}(j) \geq \tilde{p}(j)$, where we denote by $\tilde{q}(j)$ the number of drivers whose preferred parking spaces are predecessors of j in $\tilde{T}$ and by $\tilde{p}(j)$ the number of predecessors of j in $\tilde{T}$.

We can assume that j lies somewhere on the path from w to v, since for all other nodes i it holds that $p(i) = \tilde{p}(i)$ and $q(i) = \tilde{q}(i)$. Now the following holds:

$$\tilde{q}(j) = q(j) + q(\mathsf{root}(U)),$$

where $q(\mathsf{root}(U))$ corresponds to the number of drivers that want to park in the subtree U. Similarly, we have:

$$\tilde{p}(j) = p(j) + p(\mathsf{root}(U)),$$

where $p(\mathsf{root}(U)) = |U|$ corresponds to the number of nodes in the subtree U.

Since it holds that $q(j) \geq p(j)$ and $q(\mathsf{root}(U)) \geq p(\mathsf{root}(U))$, the desired statement for $\tilde{T}$ holds. □

With this lemma we can easily obtain tight bounds on $S(T, m)$.

Theorem 9.10. *Let* star_n *be the rooted labelled tree of size* n *with root node* n *and the nodes* $1, 2, \ldots, n-1$ *attached to it. Furthermore let* chain_n *be the rooted labelled tree of size* n *with root node* n *and node* j *attached to node* $(j+1)$*, for* $1 \leq j \leq n-1$*. Then, for any rooted labelled tree* T *of size* n *it holds*

$$S(\mathsf{star}_n, m) \leq S(T, m) \leq S(\mathsf{chain}_n, m), \tag{43}$$

yielding the bounds

$$n^{\underline{m}} + \binom{m}{2}(n-1)^{\underline{m-1}} \leq S(T, m) \leq (n-m+1)(n+1)^{m-1}, \tag{44}$$

for $0 \leq m \leq n$.

Proof. Each tree T of size n can be constructed from a tree T_0, which is isomorphic to star_n, by applying a sequence of reallocations $T_{i+1} := \mathsf{reallocate}\left(T_i \,\middle|\, \begin{smallmatrix} u_i \\ \downarrow \\ v_i \end{smallmatrix} \mapsto \begin{smallmatrix} u_i \\ \downarrow \\ w_i \end{smallmatrix}\right)$, with $w_i \preceq_{T_i} v_i$, for $0 \leq i \leq k$, with $k \geq 0$. Furthermore, starting with $T =: \tilde{T}_0$, there always exists a sequence of reallocations $\tilde{T}_{i+1} := \mathsf{reallocate}\left(\tilde{T}_i \,\middle|\, \begin{smallmatrix} \tilde{u}_i \\ \downarrow \\ \tilde{v}_i \end{smallmatrix} \mapsto \begin{smallmatrix} \tilde{u}_i \\ \downarrow \\ \tilde{w}_i \end{smallmatrix}\right)$, with $\tilde{w}_i \preceq_{\tilde{T}_i} \tilde{v}_i$, for $0 \leq i \leq \tilde{k}$, with $\tilde{k} \geq 0$, such that the resulting tree is isomorphic to chain_n. Thus, equation (43) follows immediately from Lemma 9.9.

The upper bound in (44) is the well-known formula for the number parking functions in a one-way street (which corresponds to the number of tree parking functions for chain_n). For the lower bound one has to compute the number of parking functions with m drivers for star_n: there are only two possible cases, namely either s is injective or exactly two drivers have the same non-root node as preferred parking space, whereas all remaining drivers have different non-root nodes as preferred parking spaces. Elementary combinatorics yields the stated result. □

9.3 TOTAL NUMBER OF PARKING FUNCTIONS: THE NUMBER OF DRIVERS COINCIDES WITH THE NUMBER OF PARKING SPACES

In this section, we consider the total number of parking functions for trees and mappings for the case that the number of drivers m is equal to the number of parking spaces n. As for ordinary parking functions, this case is not only interesting in its own. It will also occur during the studies of the general case via initial values for recurrence relations.

Tree parking functions

We study the total number $F_n := F_{n,n}$ of (n,n)-tree parking functions, i.e., the number of pairs (T,s), with $T \in \mathcal{T}_n$ a Cayley tree of size n and $s \in [n]^n$ a parking sequence of length n for the tree T, such that all drivers are successful. To obtain a recursive description of the total number F_n of tree parking functions we use the decomposition of a Cayley tree $T \in \mathcal{T}_n$ w.r.t. the last empty node. We thus consider the situation just before the last driver starts searching a parking space.

Two different situations have to be considered for the last empty node: it might be the root node of the tree T as depicted in Figure 40 or a non-root node as in Figure 41.

Case (1): The last empty node is the root node. In this case the last driver will always find a free parking space regardless of the n possible choices of her preferred parking space.

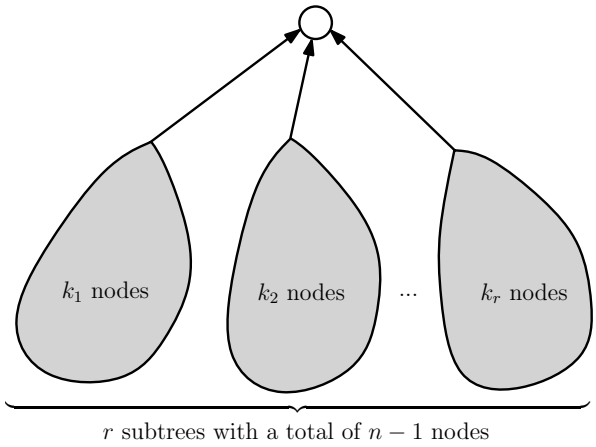


Figure 40: Schematic representation of the first case that can occur when considering parking functions with n drivers for a Cayley tree with n nodes. The last empty node which is marked in white in the tree is the root node of the tree.

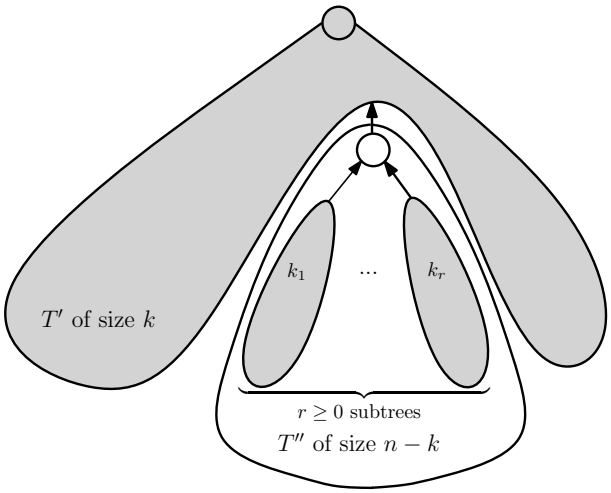


Figure 41: Schematic representation of the second case that can occur when considering parking functions with n drivers for a Cayley tree with n nodes. The last empty node which is marked in white in the tree is a non-root node.

Case (2): The last empty node is a non-root node. Here, the last driver will only find a free parking space if her preferred parking space is contained in the subtree (call it T'') rooted at the node corresponding to the free parking space. If we detach the edge linking this subtree T'' with the rest of the tree we get two unordered trees. Let us assume the tree containing the original root of the tree (denote it with T') has size k, whereas the remaining tree T'' has size $n-k$. Then there are $n-k$ choices for the preferred parking space of the last driver such that she is successful. Furthermore, it is important to take into account that, given T' and T'', the original tree T cannot be reconstructed, since there are always k different trees in $\mathcal{T}_n$ leading to the same pair (T', T''); in other words, given T' and T'', we have k choices of constructing trees $\tilde{T} \in \mathcal{T}_n$ by attaching the root of T'' to any of the k nodes of T'.

Taking into account the order-preserving relabellings of the subtrees and also the merging of the parking sequences for the subtrees, we obtain the following recursive description of F_n.

$$\begin{aligned} F_n = &\sum_{r\geq 1} \frac{1}{r!} \sum_{\substack{\sum k_i = n-1 \\ k_i \geq 1}} F_{k_1} \cdot F_{k_2} \cdot \cdots \cdot F_{k_r} n \binom{n}{k_1, k_2, \ldots, k_r, 1} \binom{n-1}{k_1, k_2, \ldots, k_r} \\ &+ \sum_{r\geq 0} \frac{1}{r!} \sum_{\substack{k+\sum k_i = n-1 \\ k\geq 1, k_i \geq 1}} F_k \cdot F_{k_1} \cdot F_{k_2} \cdot \cdots \cdot F_{k_r} k(n-k) \cdot \\ &\qquad \cdot \binom{n}{k, k_1, k_2, \ldots, k_r, 1} \binom{n-1}{k, k_1, k_2, \ldots, k_r}, \quad \text{for } n \geq 2, \end{aligned} \tag{45}$$

with initial value $F_1 = 1$. Here r denotes the number of subtrees of the free parking space (i.e., of the empty node), thus the factor $\frac{1}{r!}$ occurs, since each of the $r!$ orderings of the subtrees of the empty node represent the same tree. In order to treat this recurrence we introduce the following generating function

$$F(z) := \sum_{n\geq 1} \frac{F_n}{(n!)^2} z^n.$$

Then, after straightforward computations which are omitted here, (45) can be transferred into the following differential equation:

$$F'(z) = \exp(F(z)) \cdot \left(1 + zF'(z)\right)^2, \quad F(0) = 0. \tag{46}$$

This differential equation can be solved by standard methods and it can be checked easily that the solution of (46) is given as follows:

$$F(z) = T(2z) + \ln\left(1 - \frac{T(2z)}{2}\right), \tag{47}$$

where $T(z)$ denotes the Cayley tree function.

We shall not extract coefficients from (47) at this point yet, since we will soon see in Theorem 9.12 that the total number F_n of parking

functions for trees of size n is directly linked to the total number M_n of parking functions for mappings of size n. The latter quantity is treated in the next section and thus also yields exact and asymptotic enumeration formulæ for F_n.

Mapping parking functions

Now we study the total number $M_n := M_{n,n}$ of (n,n)-mapping parking functions, i.e., the number of pairs (f,s), with $f \in \mathcal{M}_n$ an n-mapping and $s \in [n]^n$ a parking sequence of length n for the mapping f, such that all drivers are successful. Recall the structure of functional graphs of mappings: the connected components are cycles of Cayley trees. It is thus natural to introduce connected mappings as auxiliary objects and study parking functions for them first. It will then be easy to treat the general situation for mappings when we have obtained results for connected mappings.

Whereas the relation between mappings and connected mappings can be translated immediately into connections between parking functions for these objects, this is not the case for connected mappings and trees. Indeed, the decomposition of connected mappings $\mathcal{C}$ into Cayley trees $\mathcal{T}$ is not consistent with the parking procedure. Instead of using this composition, we will therefore apply a decomposition of connected mappings w.r.t. the last empty node in the parking procedure. So, let us introduce the total number C_n of parking functions of length n for connected n-mappings, i.e., the number of pairs (f,s), with $f \in \mathcal{C}_n$ a connected n-mapping and $s \in [n]^n$ a parking sequence of length n for f, such that all drivers are successful. We will then obtain a recursive description of C_n in which the quantity F_n counting the number of (n,n)-tree parking functions which was introduced in Section 9.3 appears.

Here, we have to distinguish three different situations. To treat these cases only slight adaptions to the considerations made in Section 9.3 have to be done:

Case (1): The last empty node is the root node of the Cayley tree that forms a length-1 cycle. This situation is the same as the first case for parking functions for trees and has been depicted in Figure 40.

Case (2): The last empty node is the root node of a Cayley tree lying in a cycle of at least two trees. This case is depicted in Figure 42. Here, the last driver will always find a free parking space regardless of the n possible choices of her preferred parking space. Let us denote by T'' the tree whose root node is the last free parking space. When we detach the two edges linking T'' with the rest of the mapping graph, we cut the cycle and the graph decomposes into two trees: the tree T'' and the unordered tree (call it T'), which we may consider rooted at the former predecessor of the free parking space in the cycle of the original graph. Let us assume that T' has size k, whereas T'' has

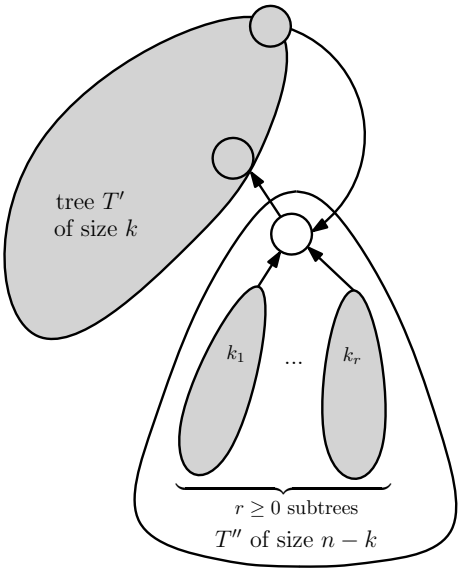


Figure 42: Schematic representation of the second situation that might occur when considering parking functions with n drivers for a connected n-mapping. The last empty node is a cyclic node in a cycle of length at least two.

size $n-k$. Then, given T' and T'', there are k different choices of constructing graphs in $\mathcal{C}_n$ by adding an edge from the root of T' to the root of T'' and attaching the root of T'' to any of the k nodes of T'.

Case (3): The last empty node is not a cyclic node, i.e., it is not one of the root nodes of the Cayley trees forming the cycle. This case is depicted in Figure 43. Here the last driver will only find a free parking space if her preferred parking space is contained in the subtree (call it T'') rooted at the node corresponding to the free parking space. If we detach the edge linking this subtree T'' with the rest of the graph, a connected mapping graph remains (call it C'). Let us assume that C' has size k, whereas the tree T'' has size $n-k$. Then there are $n-k$ possibilities of preferred parking spaces for the last driver such that she is successful. Again, given C' and T'', there are k different choices of constructing graphs in $\mathcal{C}_n$ by attaching the root of T'' to any of the k nodes of C'.

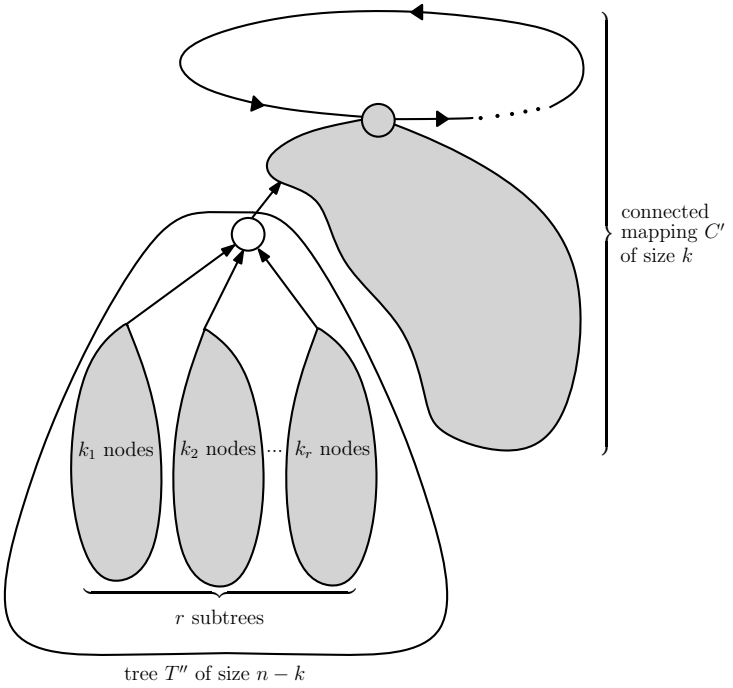


Figure 43: Schematic representation of the third situation that might occur when considering parking functions with n drivers for a connected n-mapping. The last empty node is a non-cyclic node.

Again, taking into account the order-preserving relabellings of the substructures and also the merging of the parking sequences for them, we obtain the following recursive description of C_n, valid for all $n \geq 1$:

$$\begin{aligned} C_n &= \sum_{r\geq 0} \frac{1}{r!} \sum_{\sum k_i = n-1} F_{k_1} F_{k_2} \cdots F_{k_r} \cdot n \binom{n}{k_1, k_2, \ldots, k_r} \binom{n-1}{k_1, k_2, \ldots, k_r} \\ &\quad + \sum_{r\geq 0} \frac{1}{r!} \sum_{k+\sum k_i = n-1} F_k F_{k_1} F_{k_2} \cdots F_{k_r} \cdot kn \binom{n}{k, k_1, \ldots, k_r} \binom{n-1}{k, k_1, \ldots, k_r} \\ &\quad + \sum_{r\geq 0} \frac{1}{r!} \sum_{k+\sum k_i = n-1} C_k F_{k_1} F_{k_2} \cdots F_{k_r} \cdot k(n-k) \cdot \\ &\qquad \cdot \binom{n}{k, k_1, \ldots, k_r} \binom{n-1}{k, k_1, \ldots, k_r}. \end{aligned} \tag{48}$$

Now we introduce the generating function

$$C(z) := \sum_{n\geq 1} \frac{C_n}{(n!)^2} z^n.$$

Then, recurrence (48) yields the following differential equation for $C(z)$,

$$\begin{aligned} &C'(z) \cdot \left(1 - z\exp(F(z)) - z^2 F'(z)\exp(F(z))\right) \\ &= \left(\left(1 + zF'(z)\right)^2 + zF'(z) + z^2 F''(z)\right)\exp(F(z)), \quad C(0) = 0, \end{aligned} \tag{49}$$

where $F(z)$ denotes the generating function of the number of tree parking functions given in (47). This differential equation has the following simple solution:

$$C(z) = \ln\left(\frac{1}{1-\frac{T(2z)}{2}}\right), \tag{50}$$

as can be checked easily by using the functional equation of the tree function $T(z)$ (Equation (6)).

Extracting coefficients from (50) gives the following auxiliary result.

Lemma 9.11. *The total number C_n of parking functions of length n for connected n-mappings is, for $n \geq 1$, given as follows:*

$$C_n = n!(n-1)! \sum_{j=0}^{n-1} \frac{(2n)^j}{j!}.$$

Proof. Using the functional equation of the tree function $T(z)$ in Equation (6), a standard application of the Lagrange inversion formula yields

$$\begin{aligned}[z^n]\ln\left(\frac{1}{1-\frac{T(2z)}{2}}\right) &= 2^n[z^n]\ln\left(\frac{1}{1-\frac{T(z)}{2}}\right) = \frac{2^n}{n}[T^{n-1}]\frac{e^{nT}}{2(1-\frac{T}{2})} \\ &= \frac{2^{n-1}}{n}\sum_{j=0}^{n-1}\frac{n^{n-1-j}}{2^j(n-1-j)!} = \frac{1}{n}\sum_{j=0}^{n-1}\frac{(2n)^j}{j!},\end{aligned}$$

and further

$$C_n = (n!)^2[z^n]C(z) = (n!)^2[z^n]\ln\left(\frac{1}{1-\frac{T(2z)}{2}}\right) = n!(n-1)!\sum_{j=0}^{n-1}\frac{(2n)^j}{j!}.$$

□

Now we are in the position to study the total number M_n of (n,n)-mapping parking functions. Again we introduce the generating function

$$M(z) := \sum_{n\geq 0} M_n \frac{z^n}{(n!)^2}.$$

Since the functional digraph of a mapping can be considered as the set of its connected components and furthermore a parking function for a mapping can be considered as a shuffle of the corresponding parking functions for the connected components, we get the following simple relation between the generating functions $M(z)$ and $C(z)$ of parking functions for mappings and connected mappings:

$$M(z) = \exp(C(z)).$$

Thus, by using (50), the generating function $M(z)$ is given as follows:

$$M(z) = \frac{1}{1 - \frac{T(2z)}{2}}. \tag{51}$$

Next, we remark that the following relation between $M(z)$ and $F(z)$, the generating functions for the number of parking functions for mappings and trees, holds:

$$\begin{aligned} 1 + zF'(z) &= 1 + \frac{T(2z)}{1 - T(2z)} \cdot \left(1 - \frac{1}{2 - T(2z)}\right) \\ &= 1 + \frac{\frac{T(2z)}{2}}{1 - \frac{T(2z)}{2}} = M(z), \end{aligned}$$

where we used $T'(z) = T(z)/(z \cdot (1 - T(z)))$ obtained by differentiating Equation (6). At the level of coefficients, this immediately shows the following somewhat surprising connection between F_n and M_n.

Theorem 9.12. *For all $n \geq 1$ it holds that the total numbers F_n and M_n of (n, n)-tree parking functions and (n, n)-mapping parking functions, respectively, satisfy:*

$$M_n = n \cdot F_n.$$

Since it also holds that the number of mappings of size n is exactly n times the number of Cayley trees of size n, this implies that the average number of parking functions per mapping of a given size is exactly equal to the average number of parking functions per tree of the same size. Later, in Section 9.3 we establish a combinatorial explanation for this interesting fact.

Extracting coefficients from the generating function solution (51) of $M(z)$ easily yields exact formulæ for M_n and, due to Theorem 9.12, also for F_n.

Theorem 9.13. *The total number M_n of (n, n)-mapping parking functions is for $n \geq 1$ given as follows:*

$$M_n = n!(n-1)! \cdot \sum_{j=0}^{n-1} \frac{(n-j) \cdot (2n)^j}{j!}.$$

Corollary 9.14. *The total number F_n of (n, n)-tree parking functions is for $n \geq 1$ given as follows:*

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

Proof of Theorem 9.13. Again, using (6) and the Lagrange inversion formula, we obtain

$$\begin{aligned} [z^n] \frac{1}{1 - \frac{T(2z)}{2}} &= 2^n [z^n] \frac{1}{1 - \frac{T(z)}{2}} = \frac{2^n}{n} [T^{n-1}] \frac{e^{nT}}{2\left(1 - \frac{T}{2}\right)^2} \\ &= \frac{2^{n-1}}{n} \sum_{k=0}^{n-1} \frac{(k+1)n^{n-1-k}}{2^k (n-1-k)!} = \frac{1}{n} \sum_{j=0}^{n-1} \frac{(n-j)(2n)^j}{j!}, \end{aligned}$$

and thus

$$M_n = (n!)^2[z^n]M(z) = (n!)^2[z^n]\frac{1}{1-\frac{T(2z)}{2}}$$
$$= n!(n-1)!\sum_{j=0}^{n-1}\frac{(n-j)\cdot(2n)^j}{j!}.$$

□

The asymptotic behaviour of the numbers M_n and F_n for $n \to \infty$ could be deduced from these exact formulæ; however, it seems easier to start with the generating function solution (51) of $M(z)$. Using the asymptotic expansion of the tree function $T(z)$ in a complex neighbourhood of its unique dominant singularity $\frac{1}{e}$ (as derived in Example 2.33),

$$T(z) = 1 - \sqrt{2}\sqrt{1-ez} + \frac{2}{3}(1-ez) + \mathcal{O}\left((1-ez)^{\frac{3}{2}}\right), \qquad (52)$$

one immediately obtains that $M(z)$ inherits a singularity from $T(z)$ at $\rho = \frac{1}{2e}$. According to (51), there might be another singularity at the point z_0 where $T(2z_0) = 2$. Due to the functional equation of $T(z)$ (6), this would imply $2 = 2z_0e^2$, i.e. $z_0 = 1/e^2$. It is easy to check that $T(2/e^2) \approx 0.4 \neq 2$. Therefore, $M(z)$ has its unique dominant singularity at $\rho = \frac{1}{2e}$. Its local expansion in a complex neighbourhood of ρ can easily be obtained as follows:

$$M(z) = \frac{2}{2-T(2z)} = \frac{2}{1+\sqrt{2}\sqrt{1-2ez} - \frac{2}{3}(1-2ez) + \mathcal{O}((1-2ez)^{\frac{3}{2}})}$$
$$= 2\Bigg(1 - \sqrt{2}\sqrt{1-2ez} + \frac{2}{3}(1-2ez)$$
$$+ \left(-\sqrt{2}\sqrt{1-2ez} + \frac{2}{3}(1-2ez) + \mathcal{O}((1-2ez)^{\frac{3}{2}})\right)^2$$
$$+ \mathcal{O}((1-2ez)^{\frac{3}{2}})\Bigg)$$
$$= 2 - 2\sqrt{2}\sqrt{1-2ez} + \frac{16}{3}(1-2ez) + \mathcal{O}((1-2ez)^{\frac{3}{2}}).$$

A standard application of Theorem 2.31 shows the following asymptotic equivalent of the numbers M_n. We get

$$[z^n]M(z) \sim \frac{\sqrt{2}}{\sqrt{\pi}}\frac{(2e)^n}{n^{\frac{3}{2}}}$$

and the following corollary, which follows directly when applying Stirling's approximation formula for the factorials [78].

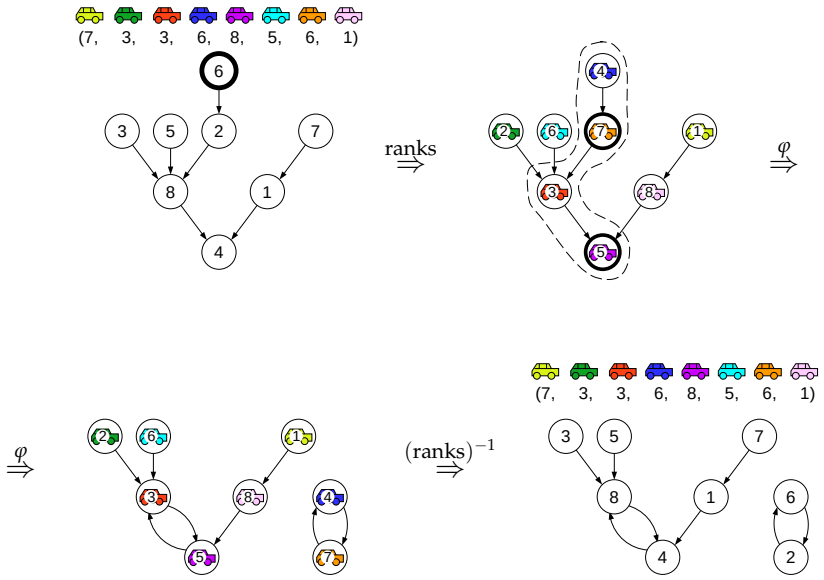


Figure 44: The bijection ψ described in Theorem 9.16 is applied to the triple (T, s, w) with T the tree depicted in the top left corner, $s = (7,3,3,6,8,5,6,1)$ a parking function for T and the node $w = 6$. It yields the mapping parking function (f, s) represented in the bottom right corner. The function φ is the bijection described in Theorem 7.1.

Corollary 9.15. *The total number M_n of (n,n)-mapping parking functions and the total number F_n of (n,n)-tree parking functions, respectively, are asymptotically, for $n \to \infty$, given as follows:*

$$M_n \sim \frac{\sqrt{2\pi}\, 2^{n+1} n^{2n}}{\sqrt{n}\, e^n}, \quad \text{and} \quad F_n \sim \frac{\sqrt{2\pi}\, 2^{n+1} n^{2n}}{n^{\frac{3}{2}}\, e^n}.$$

Bijective relation between parking functions for trees and mappings

The simple relation between the total number of parking functions of a given size for trees and mappings stated in Theorem 9.12 was proved by algebraic manipulations of the corresponding generating functions. This does not provide a combinatorial explanation of this fact.

Of course, the numbers $T_n := |\mathcal{T}_n| = n^{n-1}$ of Cayley trees of size n and the numbers $|\mathcal{M}_n| = n^n$ of n-mappings themselves satisfy such a relationship. However, standard constructions such as Prüfer codes do not seem to give a simple explanation why this carries over to the total number of parking functions. We thus present a bijective proof of this result in the following. An example of the bijection ψ is given after the proof in Example 9.18 and illustrated in Figure 44.

Theorem 9.16. *For each $n \geq 1$, there exists a bijection ψ from the set of triples (T, s, w), with $T \in \mathcal{T}_n$ a tree of size n, $s \in [n]^n$ a parking function for T with n drivers, and $w \in T$ a node of T, to the set of pairs (f, s) where $f \in \mathcal{M}_n$ is an n-mapping and $s \in [n]^n$ is a parking function for f with n drivers. Thus*

$$n \cdot F_n = M_n, \quad \text{for } n \geq 1.$$

Remark 9.17. The parking function s remains unchanged under the bijection φ. Thus, when denoting by $\hat{F}_n(s)$ and $\hat{M}_n(s)$ the number of trees $T \in \mathcal{T}_n$ and mappings $f \in \mathcal{M}_n$, respectively, such that a given $s \in [n]^n$ is a parking function for T and f, respectively, it holds:

$$n \cdot \hat{F}_n(s) = \hat{M}_n(s), \quad \text{for } n \geq 1.$$

Proof of Theorem 9.16. Let us start by defining the rank of a node v in T: the rank $k(v)$ is defined as $\pi^{-1}(v)$, where the output-function π of (T, s) is a bijection since s is a parking function for T with n drivers. That is, $k(v) = i$ if and only if the i-th car in the parking sequence ends up parking at node v in T. For an example, see the second picture in Figure 44.

The ranks associated to every node in T define a new tree $\tilde{T}$ that has the same shape as T but different labels. We now apply the bijection φ described in Theorem 7.1 to the pair $(\tilde{T}, k(w))$ in order to obtain a mapping $\tilde{f}$. From this mapping $\tilde{f}$ we construct the mapping f by associating to every node i in f the label j if and only if $k(j) = i$, i.e., $j = \pi(v)$.

What needs to be shown now is that s is a parking function for f as well. For this purpose, let us use the notation introduced in the proof of Theorem 7.1: $v_1, v_2, \ldots, v_r$ are the nodes in T constituting the unique path from w to the root of T and the ranks $k_1, \ldots, k_r$ are the labels of corresponding nodes in $\tilde{T}$. Furthermore, $i_1 < i_2 < \cdots < i_t$ are the positions of right-to-left maxima in the sequence $k_1, \ldots, k_r$.

Recall that the only edges that are deleted in $\tilde{T}$ are those between the nodes k_{i_ℓ} and their parents $T(k_{i_\ell})$ for $1 \leq \ell \leq t$. Equivalently, the edges that are removed in T are those between v_{i_ℓ} and their parents $T(v_{i_\ell})$ The crucial observation is that these edges $(k_{i_\ell}, T(k_{i_\ell}))$ are never used by any of the drivers of s. Since k_{i_ℓ} is a right-to-left maximum in the sequence $k_1, \ldots, k_r$, all nodes that lie on the path from k_{i_ℓ} to the root are already occupied when the k_{i_ℓ}-th driver parks at k_{i_ℓ}. Thus, no driver before k_{i_ℓ} (then she would have parked at k_{i_ℓ}) nor after k_{i_ℓ} (then she would not be able to park anywhere) could have reached and thus left the node k_{i_ℓ}. We may thus delete this edge and attach the node k_{i_ℓ} to an arbitrary node without violating the property that s is a parking function. Having defined the mapping f in this way, the sequence s is also a parking function for f and it holds that the parking paths of the drivers coincide for T and f. In particular, it holds that $\pi_{(f,s)} = \pi_{(T,s)}$ for the corresponding output-functions. □

Example 9.18. Consider the tree T of size 8 depicted in the top left corner of Figure 44. The sequence $s = 7,3,3,6,8,5,6,1$ is a parking function for T. In the top right corner of the figure, the ranks of every node in T can be found. With $w = 6$, the path from w to the root consists of the nodes $6,2,8,4$ with respective ranks $4,7,3,5$. Within this sequence of ranks, the right-to-left maxima are 7 and 5. The mapping f associated to the triple (T,s,w) thus consists of two connected components, as depicted in the bottom left corner of the figure. In the functional graph of f, the cyclic elements are $6,2,8,4$. It can easily be checked that s is a parking function for f as well. ⊣

9.4 TOTAL NUMBER OF PARKING FUNCTIONS: THE GENERAL CASE

In this section we study the exact and asymptotic behaviour of the total number of tree and mapping parking functions for the general case of n parking spaces and $0 \leq m \leq n$ drivers. In what follows we will always use $\tilde{m} := n - m$, i.e., $\tilde{m}$ denotes the number of empty parking spaces (i.e., empty nodes) in the tree or mapping graph after all m drivers have parked. The case $\tilde{m} = 0$ has already been treated in Section 9.3 and the results obtained there will be required here.

Tree parking functions

We analyse the total number $F_{n,m}$ of (n,m)-tree parking functions, i.e., the number of pairs (T,s), with $T \in \mathcal{T}_n$ a Cayley tree of size n and $s \in [n]^m$ a parking sequence of length m for the tree T, such that all drivers are successful. Furthermore, as introduced in Section 9.3, $F_n = F_{n,n}$ denotes the number of tree parking functions when the number of parking spaces n coincides with the number of drivers m.

Let us now consider tree parking functions for the case that $\tilde{m}$ parking spaces will remain free. In the following it is advantageous to use the abbreviation $\tilde{F}_{n,\tilde{m}} := F_{n,n-\tilde{m}}$, thus $\tilde{F}_{n,0} = F_n$. Let us assume that $1 \leq \tilde{m} \leq n$. To get a recursive description for the numbers $\tilde{F}_{n,\tilde{m}}$, we use the combinatorial decomposition of a Cayley tree $T \in \mathcal{T}_n$ w.r.t. the free node which has the largest label amongst all $\tilde{m}$ empty nodes in the tree.

Again, the two situations depicted in Figures 40 and 41 have to be considered. The argumentation given in Section 9.3 for the case $\tilde{m} = 0$ can be adapted easily:

Case (1): The root node is the empty node with largest label and we assume that the r subtrees of the root are of sizes $k_1,\ldots,k_r$ (with $\sum_i k_i = n-1$) and contain $\ell_1,\ldots,\ell_r$ (with $\sum_i \ell_i = \tilde{m}-1$) empty nodes, respectively.

Case (2): Here, a non-root node is the empty node with largest label. We denote by T'' the subtree of T rooted at this empty node. After de-

taching T'' from the remaining tree we obtain a tree T' that is of size k and has ℓ empty nodes for some $1 \le k \le n-1$ and $0 \le \ell \le \tilde{m}-1$. Furthermore, we assume that the r subtrees of the root of T'' are of sizes $k_1, \dots, k_r$ (with $k + \sum_i k_i = n-1$) and contain $\ell_1, \dots, \ell_r$ (with $\ell + \sum_i \ell_i = \tilde{m}-1$) empty nodes, respectively. In the latter case one has to take into account that there are k possibilities of attaching the root of T'' to one of the k nodes in T' yielding the same decomposition. The following recursive description of the numbers $\tilde{F}_{n,\tilde{m}}$ follows by considering the order-preserving relabellings of the subtrees and also the merging of the parking sequences for the subtrees. Moreover, one uses the simple fact that, when fixing an empty node v and considering all possible labellings of the $\tilde{m}$ empty nodes, only a fraction of $\frac{1}{\tilde{m}}$ of all labellings leads to v having the largest label amongst all empty nodes.

We then get the following recurrence for $1 \le \tilde{m} \le n$

$$\begin{aligned}\tilde{F}_{n,\tilde{m}} = \frac{1}{\tilde{m}} \sum_{r\ge 0} \frac{1}{r!} \sum_{\sum k_i = n-1} \sum_{\sum \ell_i = \tilde{m}-1} \tilde{F}_{k_1,\ell_1} \cdot \tilde{F}_{k_2,\ell_2} \cdots \tilde{F}_{k_r,\ell_r} \cdot \\ \cdot \binom{n}{k_1, k_2, \dots, k_r} \binom{n-\tilde{m}}{k_1-\ell_1, k_2-\ell_2, \dots, k_r-\ell_r} \\ + \frac{1}{\tilde{m}} \sum_{r\ge 0} \frac{1}{r!} \sum_{k+\sum k_i = n-1} \cdot \sum_{\ell + \sum \ell_i = \tilde{m}-1} \tilde{F}_{k,\ell} \tilde{F}_{k_1,\ell_1} \cdots \tilde{F}_{k_r,\ell_r} \cdot k \cdot \\ \cdot \binom{n}{k, k_1, \dots, k_r} \binom{n-\tilde{m}}{k-\ell, k_1-\ell_1, \dots, k_r-\ell_r},\end{aligned} \tag{53}$$

with initial values $\tilde{F}_{n,0} = F_n$. It is advantageous to introduce the generating function

$$\tilde{F}(z,u) := \sum_{n\ge 1} \sum_{\tilde{m}\ge 0} \tilde{F}_{n,\tilde{m}} \frac{z^n u^{\tilde{m}}}{n!(n-\tilde{m})!} = \sum_{n\ge 1} \sum_{0\le m\le n} F_{n,n-m} \frac{z^n u^{n-m}}{n!m!}. \tag{54}$$

The recurrence relation (53) then yields, after straightforward computations, the following partial differential equation for $\tilde{F}(z,u)$:

$$\tilde{F}_u(z,u) = z^2 \tilde{F}_z(z,u) \exp(\tilde{F}(z,u)) + z \exp(\tilde{F}(z,u)), \tag{55}$$

with initial condition $\tilde{F}(z,0) = F(z)$ and $F(z) = \sum_{n\ge 1} F_n \frac{z^n}{(n!)^2}$ given by (47). A suitable representation of the solution of this PDE as given next is crucial for further studies.

Proposition 9.19. *The generating function $\tilde{F}(z,u)$ defined in (54) is given by*

$$\tilde{F}(z,u) = Q \cdot (2 + u(1-Q)) + \ln(1-Q) = \ln\left(\frac{Q(1-Q)}{z}\right),$$

where the function $Q = Q(z,u)$ is given implicitly as the solution of the functional equation

$$Q = z \cdot e^{Q\cdot(2+u(1-Q))}. \tag{56}$$

Proof. Of course, once a solution is found, it can be checked easily after some computations that this solution indeed satisfies the PDE (55) as well as the initial condition $\tilde{F}(z,0) = F(z)$. However, we find it useful to carry out solving this first order quasilinear partial differential equation via the method of characteristics. To start with we assume that we have an implicit description of a solution $\tilde{F} = \tilde{F}(z,u)$ of (55) via the equation

$$g(z,u,\tilde{F}) = c = \text{const.},$$

with a certain differentiable function g. Taking derivatives of this equation w.r.t. z and u we obtain $g_z + g_{\tilde{F}}\tilde{F}_z = 0$ and $g_u + g_{\tilde{F}}\tilde{F}_u = 0$. After plugging these equations into (55) we get the following linear PDE in reduced form for the function $g(z,u,\tilde{F})$:

$$g_u - z^2 e^{\tilde{F}} g_z + z e^{\tilde{F}} g_{\tilde{F}} = 0. \tag{57}$$

To solve it we consider the following system of so-called characteristic differential equations,

$$\dot{u} = 1, \quad \dot{z} = -z^2 e^{\tilde{F}}, \quad \dot{\tilde{F}} = z e^{\tilde{F}}, \tag{58}$$

where we regard $z = z(t)$, $u = u(t)$, and $\tilde{F} = \tilde{F}(t)$ as dependent of a variable t, i.e., $\dot{z} = \frac{dz(t)}{dt}$, etc. Now we search for first integrals of the system of characteristic differential equations, i.e., for functions $\zeta(z,u,\tilde{F})$, which are constant along any solution curve (a so-called characteristic curve) of (58).

We may proceed as follows. The second and third equation of (58) yield the differential equation

$$\frac{dz}{d\tilde{F}} = -z,$$

leading to the general solution $z = c_1 e^{-\tilde{F}}$; thus, we get the following first integral of (58):

$$\zeta_1(z,u,\tilde{F}) = c_1 = z e^{\tilde{F}}.$$

To get another first integral (independent from this one) we consider the first and third differential equation of (58) and get, after the substitution $z = c_1 e^{-\tilde{F}}$, simply

$$\frac{du}{d\tilde{F}} = \frac{1}{c_1}.$$

The general solution $u = \frac{\tilde{F}}{c_1} + c_2$ yields, after backsubstituting $c_1 = z e^{\tilde{F}}$ the following first integral:

$$\zeta_2(z,u,\tilde{F}) = c_2 = u - \frac{\tilde{F}}{z e^{\tilde{F}}}.$$

Thus the general solution of (57) is given as follows:

$$\begin{aligned} g(z,u,\tilde{F}) &= H\left(\xi_1(z,u,\tilde{F}),\xi_2(z,u,\tilde{F})\right) \\ &= H\left(ze^{\tilde{F}}, u-\frac{\tilde{F}}{ze^{\tilde{F}}}\right) = \text{const.}, \end{aligned} \tag{59}$$

with H an arbitrary differentiable function in two variables. We can solve (59) w.r.t. the variable u and obtain that the general solution of the PDE (55) is implicitly given by

$$u = \frac{\tilde{F}(z,u)}{ze^{\tilde{F}(z,u)}} + h\left(ze^{\tilde{F}(z,u)}\right), \tag{60}$$

with $h(x)$ an arbitrary differentiable function in one variable. It remains to characterize the function $h(x)$ by adapting the general solution (60) to the initial condition $\tilde{F}(z,0) = F(z)$. First, we obtain

$$h(ze^{F(z)}) = -\frac{F(z)}{ze^{F(z)}},$$

with $F(z)$ given by (47). To get an explicit description of $h(x)$ we require some manipulations. Using the abbreviations $F = F(z)$, $T = T(2z)$ and introducing $R = R(z) := ze^{F(z)}$, we get

$$R = ze^{F} = ze^{T+\ln\left(1-\frac{T}{2}\right)} = ze^{T}\left(1-\frac{T}{2}\right) = \frac{T}{2}\left(1-\frac{T}{2}\right),$$

where we applied (3) for the last identity. Thus

$$T = 1-\sqrt{1-4R},$$

since $T(0) = R(0) = 0$ determines the correct branch for the solution. We can characterize the function $h(x)$ via

$$h(R) = -\frac{F}{R} = -\frac{T+\ln(1-\frac{T}{2})}{R} = -\frac{1-\sqrt{1-4R}+\ln(1-\frac{1-\sqrt{1-4R}}{2})}{R}.$$

Therefore, plugging this characterization of $h(x)$ into (60), the generating function $\tilde{F} = \tilde{F}(z,u)$ is given implicitly as follows:

$$uze^{\tilde{F}} - \tilde{F} + \left(1-\sqrt{1-4ze^{\tilde{F}}}\right) + \ln\left(1-\frac{1-\sqrt{1-4ze^{\tilde{F}}}}{2}\right) = 0. \tag{61}$$

To get a more amenable representation we introduce $Q = Q(z,u)$ via

$$Q := \frac{1-\sqrt{1-4ze^{\tilde{F}}}}{2}.$$

First, we get

$$e^{\tilde{F}} = \frac{Q(1-Q)}{z}, \tag{62}$$

and, after plugging this into (61),

$$\tilde{F} = uQ(1-Q) + 2Q + \ln(1-Q).$$

Exponentiating the latter equation shows then the functional equation characterizing Q,

$$Q = ze^{uQ(1-Q)+2Q},$$

finishing the proof. □

As for the case where the number of drivers coincides with the size of the tree, we do not extract coefficients at this point yet. We will see in Theorem 9.22 that the numbers $F_{n,m}$ are again linked directly to the numbers $M_{n,m}$ counting mapping parking functions and we shall therefore content ourselves with extracting coefficients for the corresponding generating function $\tilde{M}(z,u)$.

Mapping parking functions

We continue our studies on mapping parking functions by considering the total number $M_{n,m}$ of (n,m)-mapping parking functions, i.e., the number of pairs (f,s) with $f \in \mathcal{M}_n$ an n-mapping and $s \in [n]^m$ a parking sequence of length m for the mapping f, such that all drivers are successful.

As pointed out already in Section 9.3, it suffices to provide the relevant considerations for the subfamily $\mathcal{C}_n$ of connected n-mappings, since results for the general situation can then be deduced easily. Thus, let us introduce the total number $C_{n,m}$ of parking functions of length m for connected n-mappings, i.e., the number of pairs (f,s), with $f \in \mathcal{C}_n$ a connected n-mapping and $s \in [n]^m$ a parking sequence of length m for f, such that all drivers are successful. Additionally, we require the numbers F_n, C_n and $F_{n,m}$ as introduced in the Sections 9.3, 9.3 and 9.4, respectively.

Let us consider parking functions for connected mappings for the case that $\tilde{m} = n - m$ parking spaces remain free after all drivers have parked successfully. In what follows it is advantageous to define $\tilde{C}_{n,\tilde{m}} := C_{n,n-\tilde{m}}$ and also to use $\tilde{F}_{n,\tilde{m}} := F_{n,n-\tilde{m}}$ as done previously. Then it holds that $\tilde{C}_{n,0} = C_n$ and $\tilde{F}_{n,0} = F_n$. Let us assume that $1 \leq \tilde{m} \leq n$. To obtain a recursive description of the numbers $\tilde{C}_{n,\tilde{m}}$ we use the combinatorial decomposition of a connected mapping $f \in \mathcal{C}_n$ w.r.t. the free node which has the largest label amongst all $\tilde{m}$ empty nodes in the mapping graph.

Three situations may occur when using this decomposition: in the first case the empty node with largest label is the root node of the Cayley tree which forms a length-1 cycle (depicted in Figure 40), in the second case the empty node with largest label is the root node of a Cayley tree forming a cycle of at least two trees (depicted in Figure 42) and in the third case the empty node with largest label is

not a cyclic node (depicted in Figure 43). Analogous considerations to the ones given for tree parking function in Section 9.4 show the following recursive description of the number of parking functions for connected mappings for $1 \leq \tilde{m} \leq n$:

$$\begin{aligned}\tilde{C}_{n,\tilde{m}} = &\frac{1}{\tilde{m}} \sum_{r\geq 0} \frac{1}{r!} \sum_{\sum k_i = n-1} \sum_{\sum \ell_i = \tilde{m}-1} \tilde{F}_{k_1,\ell_1} \cdot \tilde{F}_{k_2,\ell_2} \cdots \tilde{F}_{k_r,\ell_r} \cdot \\ &\cdot \binom{n}{k_1, k_2, \ldots, k_r} \binom{n-\tilde{m}}{k_1-\ell_1, k_2-\ell_2, \ldots, k_r-\ell_r} \\ &+ \frac{1}{\tilde{m}} \sum_{r\geq 0} \frac{1}{r!} \sum_{k+\sum k_i = n-1} \sum_{\ell+\sum \ell_i = \tilde{m}-1} \tilde{F}_{k,\ell} \tilde{F}_{k_1,\ell_1} \cdots \tilde{F}_{k_r,\ell_r} \cdot k \cdot \\ &\cdot \binom{n}{k, k_1, \ldots, k_r} \binom{n-\tilde{m}}{k-\ell, k_1-\ell_1, \ldots, k_r-\ell_r} \\ &+ \frac{1}{\tilde{m}} \sum_{r\geq 0} \frac{1}{r!} \sum_{k+\sum k_i = n-1} \sum_{\ell+\sum \ell_i = \tilde{m}-1} \tilde{C}_{k,\ell} \tilde{F}_{k_1,\ell_1} \cdots \tilde{F}_{k_r,\ell_r} \cdot k \cdot \\ &\cdot \binom{n}{k, k_1, \ldots, k_r} \binom{n-\tilde{m}}{k-\ell, k_1-\ell_1, \ldots, k_r-\ell_r},\end{aligned} \tag{63}$$

with initial values $\tilde{C}_{n,0} = C_n$. When introducing the generating function

$$\tilde{C}(z,u) := \sum_{n\geq 1} \sum_{\tilde{m}\geq 0} \tilde{C}_{n,\tilde{m}} \frac{z^n u^{\tilde{m}}}{n!(n-\tilde{m})!}, \tag{64}$$

recurrence (63) yields the following first order linear partial differential equation for the function $\tilde{C}(z,u)$:

$$\begin{aligned}\tilde{C}_u(z,u) = &z^2 \tilde{C}_z(z,u) \exp(\tilde{F}(z,u)) + z \exp(\tilde{F}(z,u)) \\ &+ z^2 \tilde{F}_z(z,u) \exp(\tilde{F}(z,u)),\end{aligned} \tag{65}$$

with $\tilde{F}(z,u) = \sum_{n,\tilde{m}} \tilde{F}_{n,\tilde{m}} \frac{z^n u^{\tilde{m}}}{n!(n-\tilde{m})!}$ the corresponding generating function for the number of tree parking functions given in Proposition 9.19, and initial condition $\tilde{C}(z,0) = C(z)$, with $C(z) = \sum_{n\geq 1} C_n \frac{z^n}{(n!)^2}$ given by (50). A suitable representation of the solution of the PDE (65) is given in the following proposition.

Proposition 9.20. *The generating function $\tilde{C}(z,u)$ defined in (64) is given as follows:*

$$\tilde{C}(z,u) = \ln\left(\frac{1}{(1-Q)(1-uQ)}\right),$$

where the function $Q = Q(z,u)$ is given implicitly as the solution of the following functional equation:

$$Q = z \cdot e^{Q\cdot(2+u(1-Q))}.$$

Proof. To solve equation (65) we first consider the partial derivatives of the function $Q = Q(z,u)$ occurring in the characterization of the

function $\tilde{F} = \tilde{F}(z,u)$ given in Proposition 9.19. Starting with (56), implicit differentiation yields

$$Q_z = \frac{Q}{z(1-2Q)(1-uQ)} \quad \text{and} \quad Q_u = \frac{Q^2(1-Q)}{(1-2Q)(1-uQ)}. \tag{66}$$

Thus, due to (62), it holds

$$Q_u(z,u) = z^2 Q_z(z,u) e^{\tilde{F}(z,u)},$$

i.e., $Q(z,u)$ solves the reduced PDE corresponding to (65). This suggests the substitution $z = z(Q) := \frac{Q}{e^{Q(2+u(1-Q))}}$ and we introduce

$$\hat{C}(Q,u) := \tilde{C}(z(Q),u) = \tilde{C}\left(\frac{Q}{e^{Q(2+u(1-Q))}}, u\right).$$

After straightforward computations, which are thus omitted, equation (65) reads as

$$\hat{C}_u(Q,u) = \frac{Q}{1-uQ}.$$

Thus, after backsubstituting z and $\tilde{C}(z,u)$, the general solution of this equation is given by

$$\tilde{C}(z,u) = \ln\left(\frac{1}{1-uQ(z,u)}\right) + \tilde{h}(Q(z,u)), \tag{67}$$

with an arbitrary differentiable function $\tilde{h}(x)$. To characterize it, we evaluate (67) at $u = 0$ and use the initial condition $\tilde{C}(z,u) = C(z)$, with $C(z)$ given by (50). Using the abbreviation $\tilde{T} := \frac{T(2z)}{2}$, one easily gets $Q(z,0) = \tilde{T}$ and further

$$\tilde{h}(\tilde{T}) = \tilde{h}(Q(z,u)) = \tilde{C}(z,0) = C(z) = \ln\left(\frac{1}{1-\tilde{T}}\right),$$

which characterizes the function $\tilde{h}(x)$. The proposition follows immediately. □

We are now able to treat the total number $M_{n,m}$ of (n,m)-mapping parking functions. We introduce $\tilde{M}_{n,\tilde{m}} := M_{n,n-\tilde{m}}$ and the generating function

$$\tilde{M}(z,u) := \sum_{n\geq 0}\sum_{\tilde{m}\geq 0} \tilde{M}_{n,\tilde{m}} \frac{z^n u^{\tilde{m}}}{n!(n-\tilde{m})!}. \tag{68}$$

The decomposition of mapping parking functions into parking functions for their connected components immediately gives the relation

$$\tilde{M}(z,u) = \exp\left(\tilde{C}(z,u)\right)$$

for the respective generating functions. According to Proposition 9.20 we obtain the following solution of $M(z,u)$.

Proposition 9.21. *The generating function $\tilde{M}(z,u)$ defined in (68) is given as follows:*

$$\tilde{M}(z,u) = \frac{1}{(1-Q)(1-uQ)},$$

where the function $Q = Q(z,u)$ is given implicitly as the solution of the following functional equation:

$$Q = z \cdot e^{Q\cdot(2+u(1-Q))}.$$

Using the representations of the generating functions $\tilde{F}(z,u)$ and $\tilde{M}(z,u)$ for the number of tree and mapping parking functions given in Proposition 9.19 and 9.21, respectively, it can be shown easily how they are connected with each other. Namely, together with (66), we obtain

$$\begin{aligned} 1 + z\tilde{F}_z(z,u) &= 1 + z\left(\frac{1-2Q}{Q(1-Q)}Q_z - \frac{1}{z}\right) \\ &= \frac{1-2Q}{Q(1-Q)}\frac{Q}{(1-2Q)(1-uQ)} = \frac{1}{(1-Q)(1-uQ)} \\ &= \tilde{M}(z,u). \end{aligned}$$

Thus, at the level of their coefficients, we obtain the following simple relation between the total number of tree and mapping parking functions extending Theorem 9.12.

Theorem 9.22. *For all $n \geq 1$ it holds that the total numbers $F_{n,m}$ and $M_{n,m}$ of (n,m)-tree parking functions and (n,m)-mapping parking functions, respectively, satisfy:*

$$M_{n,m} = n \cdot F_{n,m}.$$

In Section 9.4 we will extend the considerations made in Section 9.3 for the particular case $m = n$ and provide a combinatorial proof of this relation.

Using Proposition 9.21, extracting coefficients leads to the following explicit formulæ for the numbers $M_{n,m}$ and $F_{n,m}$. Note that specializing $m = n$ restates Theorem 9.13 and Corollary 9.14.

Theorem 9.23. *The total number $M_{n,m}$ of (n,m)-mapping parking functions is, for $0 \leq m \leq n$ and $n \geq 1$, given as follows:*

$$M_{n,m} = \frac{(n-1)!m!n^{n-m}}{(n-m)!}\sum_{j=0}^{m}\binom{2m-n-j}{m-j}\frac{(2n)^j(n-j)}{j!}.$$

Corollary 9.24. *The total number $F_{n,m}$ of (n,m)-tree parking functions is, for $0 \leq m \leq n$ and $n \geq 1$, given as follows:*

$$F_{n,m} = \frac{(n-1)!m!n^{n-m-1}}{(n-m)!}\sum_{j=0}^{m}\binom{2m-n-j}{m-j}\frac{(2n)^j(n-j)}{j!}.$$

Proof of Theorem 9.23. In view of the representation of $\tilde{M}(z,u)$ given in Proposition 9.21 containing the function $Q = Q(z,u)$, we make a change of variables in order to extract coefficients. Using the functional equation (56) and the derivative (66) of Q w.r.t. z, an application of Cauchy's integral formula gives

$$\begin{aligned}[z^n]\tilde{M}(z,u) &= \frac{1}{2\pi i}\oint \frac{\tilde{M}(z,u)}{z^{n+1}}dz = \frac{1}{2\pi i}\oint \frac{1}{z^{n+1}}\frac{1}{(1-Q)(1-uQ)}dQ \\ &= \frac{1}{2\pi i}\oint \frac{e^{(uQ(1-Q)+2Q)(n+1)}}{(1-Q)(1-uQ)Q^{n+1}}\frac{(1-2Q)(1-uQ)}{e^{uQ(1-Q)+2Q}}dQ \\ &= [Q^n]\frac{e^{n(uQ(1-Q)+2Q)}(1-2Q)}{1-Q}.\end{aligned}$$

Further, for $0 \leq m \leq n$,

$$\begin{aligned}[z^n u^{n-m}]\tilde{M}(z,u) &= [Q^n u^{n-m}]\frac{e^{unQ(1-Q)}e^{2nQ}(1-2Q)}{1-Q} \\ &= \frac{n^{n-m}}{(n-m)!}[Q^m](1-Q)^{n-m-1}e^{2nQ}(1-2Q).\end{aligned} \tag{69}$$

We get

$$\begin{aligned}M_{n,m} &= n!m![z^n u^{n-m}]\tilde{M}(z,u) \\ &= \frac{n!m!n^{n-m}}{(n-m)!}[Q^m](1-Q)^{n-m-1}e^{2nQ}(1-2Q) \\ &= \frac{n!m!n^{n-m}}{(n-m)!}\sum_{j=0}^{m}\binom{n-m-1}{j}(-1)^j[Q^{m-j}]e^{2nQ}(1-2Q) \\ &= \frac{n!m!n^{n-m}}{(n-m)!}\sum_{j=0}^{m}\binom{n-m-1}{j}(-1)^j\frac{2(n-m+j)(2n)^{m-j-1}}{(m-j)!} \\ &= \frac{(n-1)!m!n^{n-m}}{(n-m)!}\sum_{j=0}^{m}\binom{j+m-n}{j}\frac{(n-m+j)\,(2n)^{m-j}}{(m-j)!} \\ &= \frac{(n-1)!m!n^{n-m}}{(n-m)!}\sum_{j=0}^{m}\binom{2m-n-j}{m-j}\frac{(n-j)\,(2n)^j}{j!}.\end{aligned} \tag{70}$$

□

From Theorem 9.23 we can easily derive exact values for the total number of (n,m)-mapping parking functions for a moderate size of n. However, due to the alternating sign of the summands in the explicit formula of $M_{n,m}$ that is inherent in the binomial coefficient, it is not well suited to deduce the asymptotic behaviour of these numbers and thus to give answers to questions concerning the probability $p_{n,m}$ that a random pair (f,s) of an n-mapping f and a sequence $s \in [n]^m$ of addresses of length m is a parking function, when $n \to \infty$. Starting from (69), such asymptotic considerations will be carried out in Section 9.4 using saddle point methods.

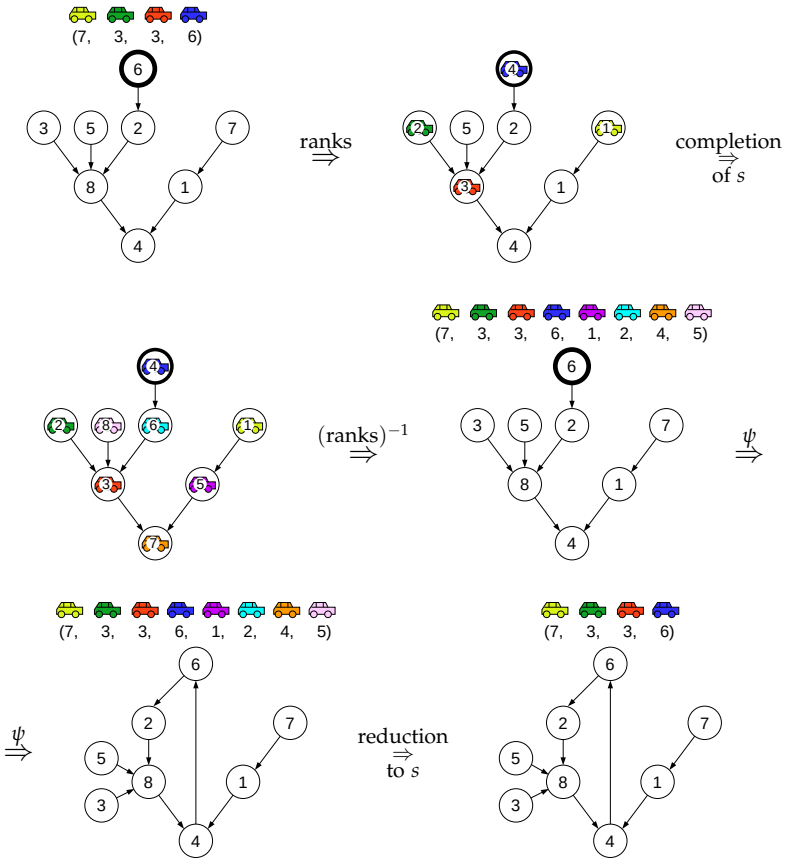


Figure 45: The bijection ψ' described in Theorem 9.25 is applied to the triple (T, s, w) with T the tree depicted in the top left corner, $s = (7, 3, 3, 6)$ a parking function for T and the node $w = 6$. It yields the mapping parking function (f, s) represented in the bottom right corner. The function ψ is the bijection described in Theorem 9.16.

Bijective relation between parking functions for trees and mappings

We will extend the bijection given in Theorem 9.16, such that it also works for the general case and thus gives a bijective proof of Theorem 9.22.

Theorem 9.25. *For $0 \leq m \leq n$ and $n \geq 1$, there exist a bijection φ' from the set of triples (T, s, w), with $T \in \mathcal{T}_n$ a tree of size n, $s \in [n]^m$ a parking function for T with m drivers, and $w \in T$ a node of T, to the set of pairs (f, s) of (n, m)-mapping parking functions, i.e, $f \in \mathcal{M}_n$ an n-mapping and $s \in [n]^m$ a parking function for f with m drivers. Thus*

$$n \cdot F_{n,m} = M_{n,m}, \quad \text{for } n \geq 1.$$

Remark 9.26. It holds that the parking function s remains unchanged under the bijection φ'. Thus, also the general case satisfies the relation

$$n \cdot \hat{F}_n(s) = \hat{M}_n(s), \quad \text{for } n \geq 1,$$

where again $\hat{F}_n(s)$ and $\hat{M}_n(s)$ denote the number of trees $T \in \mathcal{T}_n$ and mappings $f \in \mathcal{M}_n$, respectively, such that a given sequence $s \in [n]^m$ is a parking function for T and f, respectively.

Proof of Theorem 9.25. In order to a establish a bijection ψ' from the set of triples (T, s, w) to pairs (f, s), we will first extend the tree parking function $s \in [n]^m$ to a tree parking function $s' \in [n]^n$ with n drivers. Then we apply the bijection ψ described in the proof of Theorem 9.16 Finally, we reduce s' to the original parking function s. We only need to ensure that the extension from s to s' is done in such a way that the whole procedure can be reversed in a unique way. This can be done as follows.

Starting with a triple (T, s, w), let us denote by V_π the set of nodes which are occupied after the parking procedure, i.e., $V_\pi := \pi([m]) = \{\pi(k) : 1 \leq k \leq m\}$, where $\pi = \pi_{(T,s)}$ is the output-function of (T, s). Let us arrange the $n - m$ free nodes in ascending order w.r.t. their labels: $V \setminus V_\pi = \{x_1, x_2, \ldots, x_{n-m}\}$, with $x_1 < x_2 < \cdots < x_{n-m}$. Then we define the sequence $s' = (s'_1, \ldots, s'_n) \in [n]^n$ as follows:

$$\begin{aligned} s'_i &:= s_i, \quad 1 \leq i \leq m, \\ s'_{m+i} &:= x_i, \quad 1 \leq i \leq n - m. \end{aligned}$$

Of course s' is a parking function for T since s is a parking function for T and every one of the drivers $m+1, \ldots, n$ can park at their preferred parking space. Applying ψ from Theorem 9.16 gives an n-mapping f, such that s' is a parking function for f. Thus the sequence $s = (s_1, \ldots, s_m) = (s'_1, \ldots, s'_m)$, which contains the preferences of the first m drivers of s', is a parking function for f. We define the pair (f, s) to be the outcome of ψ'.

As for the case $m = n$, it holds that the parking paths of the drivers coincide for T and f. In particular, it holds $\pi_{(f,s)} = \pi_{(T,s)}$ for the corresponding output-functions. Thus, ψ' can be reversed easily, since the extension from s to s' can also be constructed when starting with f. □

Example 9.27. Consider the tree T of size 8 depicted in the top left corner of Figure 45. The sequence $s = (7, 3, 3, 6)$ is a parking function for T with 4 drivers. In the top right corner of the figure, the ranks of the nodes in T that are occupied by a car can be found. Completing s to a parking function with 8 drivers leads to the ranks in the middle left and the sequence $s' = 7, 3, 3, 6, 1, 2, 4, 5$ as described in the middle right of the figure. Applying the mapping ψ described in the proof of Theorem 9.16 to (T, s', w) yields the mapping parking function (f, s')

as depicted in the bottom left corner. Reducing s' to the first 4 drivers leads to the original parking function s. The image (f, s) of (T, s, w) under ψ' is depicted in the bottom right. It can be checked very easily that s is indeed a parking function for f. ⊣

Asymptotic considerations

Let us now turn to the asymptotic analysis of the number $M_{n,m}$ of (n, m)-mapping parking functions. Due to Theorem 9.22, our results for parking functions for mappings can automatically be translated to results for parking functions for trees. In this context the following question will be of particular interest to us: How does the probability $p_{n,m} := \frac{M_{n,m}}{n^{n+m}}$ that a randomly chosen sequence of length m on the set $[n]$ is a parking function for a randomly chosen n-mapping swap from being equal to 1 (which is the case for m=1) to being close to 0 (which is the case for $m = n$) when the ratio $\rho := \frac{m}{n}$ increases?

In order to get asymptotic results for $M_{n,m}$ (and so for $F_{n,m}$ and $p_{n,m}$, too) we start with the representation (70), which can be written as

$$M_{n,m} = \frac{n!m!n^{n-m}}{(n-m)!} A_{n,m}, \tag{71}$$

with

$$\begin{aligned} A_{n,m} &= [w^m](1-2w)e^{2nw}(1-w)^{n-m-1} \\ &= \frac{1}{2\pi i} \oint \frac{(1-2w)e^{2nw}(1-w)^{n-m-1}}{w^{m+1}} dw, \end{aligned} \tag{72}$$

where, in the latter expression, we choose as contour a suitable simple positively oriented closed curve around the origin, e.g., for each choice of m and n, we may choose any such curve in the dotted disk $0 < |w| < 1$.

Next we will use the integral representation (72) of $A_{n,m}$ and make use of the saddle point method. We write the integral as follows:

$$A_{n,m} = \frac{1}{2\pi i} \int_{\Gamma} g(w) e^{nh(w)} dw, \tag{73}$$

with Γ a suitable contour and

$$\begin{aligned} g(w) &:= \frac{1-2w}{(1-w)w} \quad \text{and} \\ h(w) = h_{n,m}(w) &:= 2w + \left(1 - \frac{m}{n}\right) \log(1-w) - \frac{m}{n} \log w. \end{aligned} \tag{74}$$

In the terminology of [78] the integral (73) has the form of a "large power integral" and saddle points of the relevant part $e^{nh(w)}$ of the integrand can thus be found as the zeros of the derivative $h'(w)$. The resulting equation

$$h'(w) = 2 - \left(1 - \frac{m}{n}\right) \frac{1}{1-w} - \frac{m}{n} \frac{1}{w} = 0$$

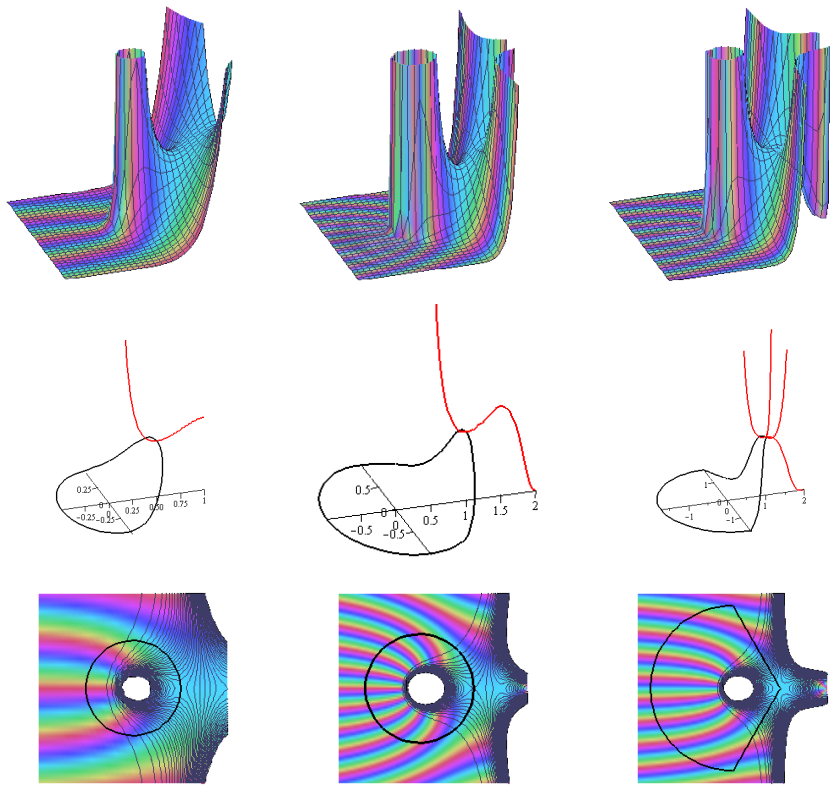


Figure 46: Plots of the modulus of the function $e^{nh(w)}$ near the saddle point for the three different regions; to the left the case $\rho < 1/2$, in the middle $\rho > 1/2$ and to the right $\rho = 1/2$. In the middle row: plots of the integration paths and steepest ascent/descent lines. The functions depicted here correspond to $n = 12$ and $m = 3$, $m = 9$ and $m = 6$ (from left to right).

yields the following two solutions:

$$w_1 = \frac{m}{n} \quad \text{and} \quad w_2 = \frac{1}{2}.$$

The present situation is illustrated in Figure 46. In our asymptotic analysis we will have to distinguish whether $w_1 < w_2$, $w_1 > w_2$ or $w_1 = w_2$. Actually, we will restrict ourselves to the cases (i) $\rho = \frac{m}{n} \leq \frac{1}{2} - \delta$ (with an arbitrary small, but fixed constant $\delta > 0$), (ii) $\rho = \frac{m}{n} \geq \frac{1}{2} + \delta$, and (iii) $\rho = \frac{m}{n} = \frac{1}{2}$. However, the transient behaviour of the sequences $M_{n,m}$, etc. for $m \sim \frac{n}{2}$ could be described via Airy functions as illustrated in [16]. The Airy function is a probability distribution that occurs as a limiting distribution for various combinatorial objects, amongst others for the area below lattice paths [118] and sums of parking functions [115].

We can sum up our results in the following theorem.

Theorem 9.28. *The total number $M_{n,m}$ of (n,m)-mapping parking functions is asymptotically, for $n \to \infty$, given as follows (where δ denotes an arbitrary small, but fixed, constant):*

$$M_{n,m} \sim \begin{cases} \frac{n^{n+m+\frac{1}{2}}\sqrt{n-2m}}{n-m}, & \text{for } 1 \le m \le (\frac{1}{2}-\delta)n, \\ \Gamma\left(\frac{2}{3}\right)\sqrt{\frac{2}{\pi}}\,3^{\frac{1}{6}}n^{\frac{3n}{2}-\frac{1}{6}}, & \text{for } m = \frac{n}{2}, \\ \frac{m!}{(n-m)!}\cdot\frac{n^{2n-m+\frac{3}{2}}2^{2m-n+1}}{(2m-n)^{\frac{5}{2}}}, & \text{for } (\frac{1}{2}+\delta)n \le m \le n. \end{cases}$$

Let us fix the ratio $\rho = m/n$. This ratio can be interpreted as a "load factor" – a term used in open addressing hashing. Then the asymptotic behaviour of the probabilities $p_{n,m} = p_{n,\rho n}$ follows immediately.

Corollary 9.29. *The probability $p_{n,m}$ that a randomly chosen pair (f,s), with f an n-mapping and s a sequence in $[n]^m$, represents a parking function is asymptotically, for $n \to \infty$ and $m = \rho n$ with $0 < \rho < 1$ fixed, given as follows:*

$$p_{n,m} \sim \begin{cases} C_<(\rho), & \text{for } 0 < \rho < \frac{1}{2}, \\ C_{1/2}\cdot n^{-1/6}, & \text{for } \rho = 1/2, \\ C_>(\rho)\cdot n^{-1}\cdot (D_>(\rho))^n, & \text{for } 1/2 < \rho < 1, \end{cases}$$

with

$$C_<(\rho) = \frac{\sqrt{1-2\rho}}{1-\rho}, \qquad C_{1/2} = \sqrt{\frac{6}{\pi}}\frac{\Gamma(2/3)}{3^{1/3}} \approx 1.298\ldots,$$

$$C_>(\rho) = 2\cdot\sqrt{\frac{\rho}{(1-\rho)(2\rho-1)^5}}, \quad D_>(\rho) = \left(\frac{4\rho}{e^2}\right)^{\rho}\frac{e}{2(1-\rho)^{1-\rho}}.$$

From Corollary 9.29 it follows that the limiting probability $L(\rho) := \lim_{n\to\infty} p_{n,\rho n}$ that all drivers can park successfully for a load factor ρ is given as follows :

$$L(\rho) = \begin{cases} \frac{\sqrt{1-2\rho}}{1-\rho}, & \text{for } 0 \le \rho \le \frac{1}{2}, \\ 0, & \text{for } \frac{1}{2} \le \rho \le 1. \end{cases}$$

See Figure 47 for an illustration: On the left hand-side the limiting distribution $L(\rho)$ is plotted and on the right hand-side the exact probabilities $p_{n,\rho n}$ for some values of n can be found.

The proof of Theorem 9.28 is given in Sections 9.4-9.4. As we can see from Theorem 9.28 and Corollary 9.29, the most interesting region for us is case (i), i.e., where less than half of the parking spaces are occupied. We will thus provide the calculations for this region in detail, whereas the application of the saddle point method is only sketched for the other two regions. Before we continue with the computations, we want to comment on relations between mapping parking functions and ordered forests of unrooted trees.

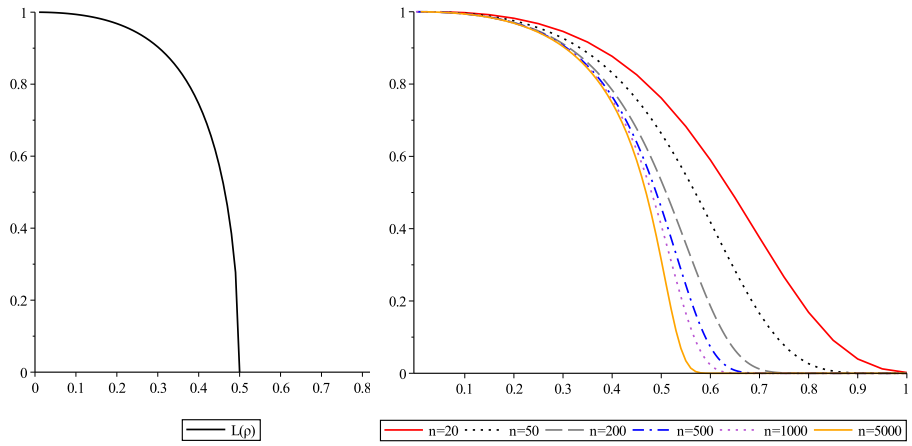


Figure 47: To the left: The limiting probability $L(\rho)$ that all drivers are able to park successfully in a mapping, for a load factor $0 \leq \rho \leq 1$. To the right: The exact probabilities $p_{n,\rho n}$ for $n = 20, 50, 200, 500, 1000, 5000$.

Remark 9.30. Qualitatively, the transient behaviour at $m \sim \frac{n}{2}$ is the same as observed previously in other combinatorial contexts, such as, e.g., in the analysis of random graphs. The study of random graphs $G(n, m)$ with n nodes and m edges was initiated by Erdős and Rényi in the late 1950s [66]. The structure of $G(n, m)$ has three different phases depending on the ratio $\frac{m}{n}$:

- When $m = \mu n$ for some $\mu < \frac{1}{2}$, the graph $G(n, m)$ consists with high probability, i.e., with probability tending to one when $n \to \infty$, only of trees and of unicyclic components. Moreover, the size of the largest component is of order $\mathcal{O}(\log n)$.
- For $m = \frac{1}{2}n + \mathcal{O}(n^{2/3}$, there appear with high probability one or several semi-giant components of size $\mathcal{O}(n^{2/3})$.
- When $m = \mu n$ for some $\mu > \frac{1}{2}$, the graph $G(n, m)$ consists with high probability of a unique giant component of size proportional to n.

For each case refined estimates that are obtained with the help of analytic techniques can be found in [74] and [104].

Such phase transition phenomena have for instance also been observed for random planar maps, see [16].

Remark 9.31. Let $G_{n,m}$ denote the number of ordered forests, i.e., sequences of m unrooted labelled trees and comprised of n nodes in total. The problem of evaluating $G_{n,m}$ asymptotically by using saddle point techniques has been considered in [78, p. 603f.]; as has been mentioned there, it is also relevant to the analysis of random graphs during the phase where a giant component has not yet emerged [74].

Since there are n^{n-2} unrooted labelled trees of size $n \geq 1$, the exponential generating function $U(z) = \sum_{n \geq 1} n^{n-2} \frac{z^n}{n!}$ is given by $U(z) =$

$T(z) - \frac{T^2(z)}{2}$, with $T(z)$ the tree-function. Thus the numbers $G_{n,m}$ can be obtained as follows:

$$G_{n,m} = n![z^n]U(z)^m = n![z^n]\left(T(z) - \frac{T^2(z)}{2}\right)^m$$
$$= n!\frac{1}{2\pi i}\oint \left(T(z) - \frac{T^2(z)}{2}\right)^m \frac{dz}{z^{n+1}},$$

by using a suitable contour around the origin. The substitution $z = \frac{T}{e^T}$ leads to

$$G_{n,m} = n!\frac{1}{2\pi i}\oint \frac{e^{nT}\left(1 - \frac{T}{2}\right)^m (1-T)}{T^{n-m+1}}dT.$$

After substituting $T = 2w$, one obtains

$$G_{n,m} = n!\frac{1}{2\pi i}\oint \frac{e^{2nw}(1-w)^m(1-2w)}{2^{n-m}w^{n-m+1}}dw$$
$$= \frac{n!}{2^{n-m}}[w^{n-m}](1-2w)e^{2nw}(1-w)^m,$$

and so

$$[w^m](1-2w)e^{2nw}(1-w)^{n-m} = \frac{2^m}{n!}G_{n,n-m}. \tag{75}$$

Comparing Equation (75) with Equation (72) for $A_{n,m}$ suggests that the same phase change behaviour occurs for $A_{n,m}$ (and thus also $M_{n,m}$) and for $G_{n,n-m}$ as studied in [78].

Remark 9.32. By slightly adapting the considerations made in the previous remark we can even express the numbers $M_{n,m}$ directly via the number of ordered forests. Namely, let $\tilde{G}_{n,m}$ denote the number of ordered forests made of one rooted labelled tree followed by $m-1$ unrooted labelled trees and comprised of n nodes in total. This yields

$$\tilde{G}_{n,m} = n![z^n]T(z)U(z)^{m-1} = n![z^n]T(z)\left(T(z) - \frac{T^2(z)}{2}\right)^{m-1}$$
$$= n!\frac{1}{2\pi i}\oint T(z)\left(T(z) - \frac{T^2(z)}{2}\right)^{m-1}\frac{dz}{z^{n+1}},$$

and, after the substitutions $z = \frac{T}{e^T}$ and $T = 2w$, we end up with

$$\tilde{G}_{n,m} = \frac{n!}{2^{n-m}}\oint \frac{e^{2nw}(1-w)^{m-1}(1-2w)}{w^{n-m+1}}dw$$
$$= \frac{n!}{2^{n-m}}[w^{n-m}](1-2w)e^{2nw}(1-w)^{m-1}.$$

We get

$$A_{n,m} = \frac{2^m}{n!}\tilde{G}_{n,n-m},$$

and thus we are able to express $M_{n,m}$ by counting a certain number of ordered forests:

$$M_{n,m} = \frac{m!2^m n^{n-m}}{(n-m)!}\tilde{G}_{n,n-m}.$$

The region $\rho \leq \frac{1}{2} - \delta$

The geometry of the modulus of the integrand of (73) as depicted in Figure 46 is easily described: There is a simple dominant saddle point at $w = w_1$, where the surface resembles an ordinary horse saddle or a mountain pass. There are two steepest ascent/steepest ascent lines: one following the real axis and one parallel to the imaginary axis. It is thus natural to adopt an integration contour that lies close to the steepest ascent and steepest descent line perpendicular to the real axis. In Equation (73) we thus choose the contour Γ to be a circle centered at the origin and passing through the dominant saddle point w_1, i.e., it has radius $r = \rho$.

Using the parametrization $\Gamma = \{w = \rho e^{i\phi} : \phi \in [-\pi, \pi]\}$, we obtain from (73) the representation

$$A_{n,m} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \rho e^{i\phi} g(\rho e^{i\phi}) e^{nh(\rho e^{i\phi})} d\phi. \tag{76}$$

Next we want to find a suitable splitting of the integral into the central approximation and the remainder. That is , we need to choose a proper value $\phi_0 = \phi_0(n, m)$ to write the contour as $\Gamma = \Gamma_1 \cup \Gamma_2$, with $\Gamma_1 := \{w = \rho e^{i\phi} : \phi \in [-\phi_0, \phi_0]\}$ and $\Gamma_2 := \{w = \rho e^{i\phi} : \phi \in [-\pi, -\phi_0] \cup [\phi_0, \pi]\}$ yielding the representation $A_{n,m} = I^{(1)}_{n,m} + I^{(2)}_{n,m}$, such that $I^{(2)}_{n,m} = o\left(I^{(1)}_{n,m}\right)$, where

$$I^{(1)}_{n,m} := \frac{1}{2\pi} \int_{\Gamma_1} \rho e^{i\phi} g(\rho e^{i\phi}) e^{nh(\rho e^{i\phi})} d\phi \quad \text{and}$$
$$I^{(2)}_{n,m} := \frac{1}{2\pi} \int_{\Gamma_2} \rho e^{i\phi} g(\rho e^{i\phi}) e^{nh(\rho e^{i\phi})} d\phi.$$

To do this we consider the local expansion of the integral around $\phi = 0$; the following results are obtained by straightforward computations, which are thus omitted:

$$\begin{aligned}
\rho e^{i\phi} g(\rho e^{i\phi}) =& \frac{1 - \frac{2m}{n} e^{i\phi}}{1 - \frac{m}{n} e^{i\phi}} = \frac{1 - \frac{2m}{n}}{1 - \frac{m}{n}} \cdot \left(1 + \mathcal{O}\left(\frac{m\phi}{n}\right)\right), \\
nh(\rho e^{i\phi}) =& n \left(2\frac{m}{n} e^{i\phi} + \left(1 - \frac{m}{n}\right) \log\left(1 - \frac{m}{n} e^{i\phi}\right) - \frac{m}{n} \log\left(\frac{m}{n} e^{i\phi}\right)\right) \\
=& 2m + (n - m) \log\left(1 - \frac{m}{n}\right) - m \log\left(\frac{m}{n}\right) \\
& - \frac{m\phi^2}{2} + \frac{m^2}{2(n-m)} \phi^2 + \mathcal{O}(m\phi^3),
\end{aligned}$$

yielding

$$\begin{aligned}
\rho e^{i\phi} g(\rho e^{i\phi}) e^{nh(\rho e^{i\phi})} = & \left(1 - \frac{2m}{n}\right) \left(\frac{n}{m}\right)^m e^{2m} \left(1 - \frac{m}{n}\right)^{n-m-1} \cdot \\
& \cdot e^{-\left(\frac{m}{2} - \frac{m^2}{2(n-m)}\right)\phi^2} \left(1 + \mathcal{O}\left(m\phi^3\right) + \mathcal{O}\left(\frac{m\phi}{n}\right)\right).
\end{aligned} \tag{77}$$

From the latter expansion we obtain that we shall choose ϕ_0, such that $m\phi_0^2 \to \infty$ (then the central approximation contains the main contributions) and $m\phi_0^3 \to 0$ (then the remainder term is asymptotically negligible). E.g., we may choose $\phi_0 = m^{-\frac{1}{2}+\frac{\epsilon}{3}}$, for a constant $0 < \epsilon < \frac{1}{2}$. With such a choice of ϕ_0 we obtain for the integral $I_{n,m}^{(1)}$ the following asymptotic expansion:

$$\begin{aligned} I_{n,m}^{(1)} =& \frac{1}{2\pi}\left(1-\frac{2m}{n}\right)\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m-1}\cdot \\ & \cdot \int_{-\phi_0}^{\phi_0} e^{-\left(\frac{m}{2}-\frac{m^2}{2(n-m)}\right)\phi^2} d\phi \cdot \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right) \\ =& \frac{1}{2\pi}\left(1-\frac{2m}{n}\right)\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m-1}\frac{1}{\sqrt{m}}\cdot \\ & \cdot \int_{-m^{\frac{\epsilon}{3}}}^{m^{\frac{\epsilon}{3}}} e^{-\frac{1}{2}\left(1-\frac{m}{n-m}\right)t^2} dt \cdot \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right), \end{aligned}$$

where we used the substitution $\phi = \frac{t}{\sqrt{m}}$ for the latter expression.

For the so-called tail completion we use that

$$\int_c^\infty e^{-\alpha t^2} dt = \mathcal{O}\left(e^{-\alpha c^2}\right), \quad \text{for } c > 0 \text{ and } \alpha > 0,$$

which can be shown, e.g., via

$$\int_c^\infty e^{-\alpha t^2} dt \le \sum_{i=0}^\infty e^{-\alpha(c+i)^2} = e^{-\alpha c^2}\cdot \sum_{i=0}^\infty e^{-\alpha i(2c+i)} = \mathcal{O}\left(e^{-\alpha c^2}\right),$$

since $\sum_{i=0}^\infty e^{-\alpha(c+i)^2}$ converges.

Thus we obtain

$$\int_{m^{\frac{\epsilon}{3}}}^\infty e^{-\frac{1}{2}\left(1-\frac{m}{n-m}\right)t^2} dt = \mathcal{O}\left(e^{-\frac{1}{2}\left(1-\frac{m}{n-m}\right)m^{\frac{2\epsilon}{3}}}\right),$$

which yields a subexponentially small and thus negligible error term. Using this we may proceed in the asymptotic evaluation of $I_{n,m}^{(1)}$ and get

$$\begin{aligned} I_{n,m}^{(1)} =& \frac{1}{2\pi}\left(1-\frac{2m}{n}\right)\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m-1}\frac{1}{\sqrt{m}}\cdot \\ & \int_{-\infty}^{\infty} e^{-\frac{1}{2}\left(1-\frac{m}{n-m}\right)t^2} dt \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right). \end{aligned}$$

Using

$$\int_{-\infty}^\infty e^{-\alpha t^2} dt = \frac{\sqrt{\pi}}{\sqrt{\alpha}}, \quad \text{for } \alpha > 0,$$

the Gaussian integral occurring can be evaluated easily, which yields

$$\begin{aligned} I_{n,m}^{(1)} =& \frac{1}{2\pi}\left(1-\frac{2m}{n}\right)\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m-1}\frac{1}{\sqrt{m}}\cdot \\ & \cdot \frac{\sqrt{2\pi}}{\sqrt{1-\frac{m}{n-m}}}\cdot\left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right) \\ =& \frac{(n-m)^{n-m-\frac{1}{2}}\sqrt{n-2m}e^{2m}}{\sqrt{2\pi}m^{m+\frac{1}{2}}n^{n-2m}}\cdot\left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right). \end{aligned}$$

Next we consider the remainder integral

$$\begin{aligned} I_{n,m}^{(2)} =& \frac{1}{2\pi}\left(1-\frac{2m}{n}\right)\cdot \\ & \cdot\int_{\Gamma_2}\frac{1-\frac{2m}{n}e^{i\phi}}{1-\frac{m}{n}e^{i\phi}}e^{n\left(2\frac{m}{n}e^{i\phi}+(1-\frac{m}{n})\log(1-\frac{m}{n}e^{i\phi})-\frac{m}{n}\log(\frac{m}{n})-\frac{m}{n}i\phi\right)}d\phi. \end{aligned}$$

To estimate the integrand we use the obvious bounds

$$\left|1-\frac{2m}{n}e^{i\phi}\right|\leq 1+\frac{2m}{n}\quad\text{and}\quad\frac{1}{\left|1-\frac{m}{n}e^{i\phi}\right|}\leq\frac{1}{1-\frac{m}{n}},$$

as well as the following:

$$\begin{aligned} & \left|e^{n\left(2\frac{m}{n}e^{i\phi}+(1-\frac{m}{n})\log(1-\frac{m}{n}e^{i\phi})-\frac{m}{n}\log(\frac{m}{n})-\frac{m}{n}i\phi\right)}\right| \\ =& \left(\frac{n}{m}\right)^m e^{n\left(2\rho\cos\phi+\frac{1-\rho}{2}\log(1-2\rho\cos\phi+\rho^2)\right)}. \end{aligned}$$

This yields

$$\begin{aligned} \left|I_{n,m}^{(2)}\right| \leq& \frac{1}{2\pi}\frac{(1-\frac{2m}{n})(1+\frac{2m}{n})}{1-\frac{m}{n}}\cdot\left(\frac{n}{m}\right)^m\cdot \\ & \cdot\int_{\Gamma_2}e^{n\left(2\rho\cos\phi+\frac{1-\rho}{2}\log(1-2\rho\cos\phi+\rho^2)\right)}d\phi. \end{aligned}$$

Considering the function

$$\tilde{H}(x) := 2\rho x+\frac{1-\rho}{2}\log(1-2\rho x+\rho^2),$$

it can be shown by applying standard calculus that $\tilde{H}(x)$ is a monotonically increasing function for $x\in[-1,1]$. Setting $x=\cos\phi$ it follows that amongst all points of the contour Γ_2 the integrand reaches its maximum at $\phi=\phi_0$. Thus, we obtain

$$\begin{aligned} \left|I_{n,m}^{(2)}\right| \leq& \frac{(1-\frac{2m}{n})(1+\frac{2m}{n})}{1-\frac{m}{n}}\cdot\left(\frac{n}{m}\right)^m\cdot e^{2m\cos\phi_0+\frac{n-m}{2}\log\left(1-\frac{2m}{n}\cos\phi_0+(\frac{m}{n})^2\right)} \\ \leq& 2\cdot\left(\frac{n}{m}\right)^m\cdot e^{2m\cos\phi_0+\frac{n-m}{2}\log\left(1-\frac{2m}{n}\cos\phi_0+(\frac{m}{n})^2\right)} \\ =& 2\left(\frac{n}{m}\right)^m e^{2m}e^{\frac{n-m}{2}\log\left(1-\frac{2m}{n}+\left(\frac{m}{n}\right)^2\right)}\cdot \\ & \cdot e^{2m(\cos\phi_0-1)+\frac{n-m}{2}\log\left(\frac{1-\frac{2m}{n}\cos\phi_0+\left(\frac{m}{n}\right)^2}{1-\frac{2m}{n}+\left(\frac{m}{n}\right)^2}\right)} \\ =& 2\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m}\cdot e^{2m(\cos\phi_0-1)+\frac{n-m}{2}\log\left(1-\frac{\frac{2m}{n}(\cos\phi_0-1)}{(1-\frac{m}{n})^2}\right)}. \end{aligned}$$

Using the estimates

$$\log(1-x) \le -x, \quad \text{for } x < 1, \quad \text{and} \quad \cos x - 1 \le -\frac{x^2}{6}, \quad \text{for } x \in [-\pi, \pi],$$

we may proceed f as follows:

$$\begin{aligned} |I_{n,m}^{(2)}| &\le 2\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m} \cdot e^{m(\cos\phi_0-1)\cdot\left(2-\frac{1}{1-\frac{m}{n}}\right)} \\ &\le 2\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m} \cdot e^{-\frac{\phi_0^2}{6}m\left(2-\frac{1}{1-\frac{m}{n}}\right)} \\ &= 2\left(\frac{n}{m}\right)^m e^{2m}\left(1-\frac{m}{n}\right)^{n-m} \cdot e^{-\frac{1}{6}\left(2-\frac{1}{1-\frac{m}{n}}\right)m^{\frac{2\epsilon}{3}}}. \end{aligned}$$

Thus we obtain

$$|I_{n,m}^{(2)}| = |I_{n,m}^{(1)}| \cdot \mathcal{O}\left(\sqrt{m}e^{-cm^{\frac{2\epsilon}{3}}}\right), \quad \text{with } c = \frac{1}{6}\left(2-\frac{1}{1-\frac{m}{n}}\right),$$

i.e., $I_{n,m}^{(2)}$ is subexponentially small compared to $I_{n,m}^{(1)}$.

Combining these results we get

$$A_{n,m} = \frac{(n-m)^{n-m-\frac{1}{2}}\sqrt{n-2m}e^{2m}}{\sqrt{2\pi}m^{m+\frac{1}{2}}n^{n-2m}} \cdot \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right)$$

and, by using (71) and applying Stirling's approximation formula for the factorials,

$$\begin{aligned} M_{n,m} &= \frac{n!m!n^{n-m}(n-m)^{n-m-\frac{1}{2}}\sqrt{n-2m}e^{2m}}{\sqrt{2\pi}(n-m)!m^{m+\frac{1}{2}}n^{n-2m}} \cdot \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right) \\ &= \frac{n^{n+m}\sqrt{1-\frac{2m}{n}}}{1-\frac{m}{n}} \cdot \left(1+\mathcal{O}\left(m^{-\frac{1}{2}+\epsilon}\right)\right). \end{aligned} \tag{78}$$

Note that according to the remainder term in (78) we have only shown the required result for $m \to \infty$. However, again starting with (76), we can easily show a refined bound on the error term for small m. Namely, we may write the integral as follows:

$$A_{n,m} = \frac{1}{2\pi}\left(\frac{n}{m}\right)^m \cdot \int_{-\pi}^{\pi} \frac{e^{2me^{i\phi}}(1-\frac{m}{n}e^{i\phi})^{n-m-1}(1-\frac{2m}{n}e^{i\phi})}{e^{im\phi}} d\phi,$$

and use for $m = o(\sqrt{n})$ the expansions

$$\begin{aligned} \left(1-\frac{m}{n}e^{i\phi}\right)^{n-m-1} &= e^{-me^{i\phi}} \cdot \left(1+\mathcal{O}\left(\frac{m^2}{n}\right)\right) \text{ and} \\ 1-\frac{2m}{n}e^{i\phi} &= 1+\mathcal{O}\left(\frac{m}{n}\right), \end{aligned}$$

which gives

$$A_{n,m} = \frac{1}{2\pi}\left(\frac{n}{m}\right)^m \cdot \int_{-\pi}^{\pi} \frac{e^{me^{i\phi}}}{e^{im\phi}} d\phi \cdot \left(1+\mathcal{O}\left(\frac{m^2}{n}\right)\right).$$

Using the substitution $z = e^{i\phi}$ this yields for $m = o(\sqrt{n})$

$$A_{n,m} = \frac{1}{2\pi i}\left(\frac{n}{m}\right)^m \cdot \oint \frac{e^{mz}}{z^{m+1}} dz \cdot \left(1 + \mathcal{O}\left(\frac{m^2}{n}\right)\right) = \frac{n^m}{m!} \cdot \left(1 + \mathcal{O}\left(\frac{m^2}{n}\right)\right)$$

and, again by using (71) and applying Stirling's approximation formula for the factorials, furthermore

$$\begin{aligned} M_{n,m} &= \frac{n! n^n}{(n-m)!} \cdot \left(1 + \mathcal{O}\left(\frac{m^2}{n}\right)\right) = n^{n+m} \cdot \left(1 + \mathcal{O}\left(\frac{m^2}{n}\right)\right) \\ &= \frac{n^{n+m}\sqrt{1 - \frac{2m}{n}}}{1 - \frac{m}{n}} \cdot \left(1 + \mathcal{O}\left(\frac{m^2}{n}\right)\right). \end{aligned}$$

The region $\rho \geq \frac{1}{2} + \delta$

For this region we choose in (73) the contour Γ to be a circle centred at the origin and passing through the dominant saddle point w_2, i.e., it has radius $r = \frac{1}{2}$. Using the parametrization $\Gamma = \{w = \frac{1}{2}e^{i\phi} : \phi \in [-\pi, \pi]\}$, we obtain from (73) the representation

$$A_{n,m} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{1}{2} e^{i\phi} g(\frac{1}{2}e^{i\phi}) e^{nh(\frac{1}{2}e^{i\phi})} d\phi, \tag{79}$$

with functions $g(w)$ and $h(w)$ defined in (74).

As in the previous region we expand the integrand in (79) around $\phi = 0$ to find a suitable choice for ϕ_0 to split the integral. However, due to cancellations, we require a more refined expansion which can again be obtained by straightforward computations. Namely, we obtain

$$\begin{aligned} \frac{1}{2}e^{i\phi} g\left(\frac{1}{2}e^{i\phi}\right) =& -2i\phi + 3\phi^2 + \mathcal{O}(\phi^3), \\ nh\left(\frac{1}{2}e^{i\phi}\right) =& n + (2m - n)\log 2 - \left(m - \frac{n}{2}\right)\phi^2 \\ &+ i\left(\frac{5n}{6} - m\right)\phi^3 + \mathcal{O}(n\phi^4), \end{aligned}$$

which gives

$$\begin{aligned} \frac{1}{2}e^{i\phi} g\left(\frac{1}{2}e^{i\phi}\right) e^{nh\left(\frac{1}{2}e^{i\phi}\right)} =& e^n 2^{2m-n} e^{-(m-\frac{n}{2})\phi^2} \cdot \\ & \cdot \left(-2i\phi + 3\phi^2 + \left(\frac{5n - 6m}{3}\right)\phi^4 + \right. \\ & \left. + \mathcal{O}(\phi^3) + \mathcal{O}(n\phi^5) + \mathcal{O}(n^2\phi^7)\right). \end{aligned}$$

Thus we may choose $\phi_0 = n^{-\frac{1}{2}+\epsilon}$, with $0 < \epsilon < \frac{1}{6}$ to split the contour $\Gamma = \Gamma_1 \cup \Gamma_2$, with $\Gamma_1 = \{w = \frac{1}{2}e^{i\phi} : \phi \in [-\phi_0, \phi_0]\}$ and $\Gamma_2 = \{w = \frac{1}{2}e^{i\phi} : \phi \in [-\pi, -\phi_0] \cup [\phi_0, \pi]\}$. Let us again denote by $I_{n,m}^{(1)}$ and $I_{n,m}^{(2)}$

the contribution of the integral in the representation (79) over Γ_1 and Γ_2, respectively.

For $I^{(1)}_{n,m}$ we use the above expansion for the integrand and obtain after simple manipulations

$$I^{(1)}_{n,m} = c_{n,m} \int_{-\phi_0}^{\phi_0} e^{-(m-\frac{n}{2})\phi^2} \cdot \cdot \left(-2i\phi + 3\phi^2 + \left(\frac{5n-6m}{3} \right) \phi^4 + \mathcal{O}(n^{-\frac{3}{2}+7\epsilon}) \right) d\phi,$$

where the multiplicative factor $c_{n,m}$ is equal to $e^n 2^{2m-n} \frac{1}{2\pi}$. Again it holds that completing the tails only gives a subexponentially small error term and we obtain

$$\begin{aligned} I^{(1)}_{n,m} &= c_{n,m} \int_{-\infty}^{\infty} e^{-(m-\frac{n}{2})\phi^2} \cdot \\ &\qquad \cdot \left(-2i\phi + 3\phi^2 + \left(\frac{5n-6m}{3} \right) \phi^4 + \mathcal{O}(n^{-\frac{3}{2}+7\epsilon}) \right) d\phi \\ &= \frac{c_{n,m}}{\sqrt{n}} \int_{-\infty}^{\infty} e^{-(\frac{m}{n}-\frac{1}{2})t^2} \cdot \\ &\qquad \cdot \left(-2i \frac{t}{\sqrt{n}} + \frac{3t^2}{n} + \frac{5n-6m}{3} \frac{t^4}{n^2} + \mathcal{O}\left(n^{-\frac{3}{2}+7\epsilon} \right) \right) dt, \end{aligned}$$

where we used the substitution $\phi = \frac{t}{\sqrt{n}}$ to get the latter expression.

Using the integral evaluations (with $\alpha > 0$):

$$\int_{-\infty}^{\infty} t e^{-\alpha t^2} dt = 0, \quad \int_{-\infty}^{\infty} t^2 e^{-\alpha t^2} dt = \frac{\sqrt{\pi}}{2\alpha^{\frac{3}{2}}}, \quad \int_{-\infty}^{\infty} t^4 e^{-\alpha t^2} dt = \frac{3\sqrt{\pi}}{4\alpha^{\frac{5}{2}}},$$

we obtain

$$\begin{aligned} I^{(1)}_{n,m} &= \frac{c_{n,m}}{n^{\frac{3}{2}}} \left(\frac{3\sqrt{\pi}}{2\left(\frac{m}{n}-\frac{1}{2}\right)^{\frac{3}{2}}} + \frac{\left(5-\frac{6m}{n}\right)\sqrt{\pi}}{4\left(\frac{m}{n}-\frac{1}{2}\right)^{\frac{5}{2}}} \right) \cdot \left(1 + \mathcal{O}\left(n^{-\frac{1}{2}+7\epsilon} \right) \right) \\ &= \frac{e^n 2^{2m-n+1}}{\sqrt{2\pi} n^{\frac{3}{2}} \left(\frac{2m}{n}-1\right)^{\frac{5}{2}}} \cdot \left(1 + \mathcal{O}\left(n^{-\frac{1}{2}+7\epsilon} \right) \right). \end{aligned}$$

Again, it can be shown that the main contribution of $A_{n,m}$ comes from $I^{(1)}_{n,m}$, i.e., that it holds

$$I^{(2)}_{n,m} = \frac{1}{2\pi} \int_{\Gamma_2} \frac{1}{2} e^{i\phi} g\left(\frac{1}{2} e^{i\phi} \right) e^{nh\left(\frac{1}{2}e^{i\phi}\right)} d\phi = o\left(I^{(1)}_{n,m} \right),$$

but here we omit these computations. Thus we obtain

$$A_{n,m} = \frac{e^n 2^{2m-n+1}}{\sqrt{2\pi} n^{\frac{3}{2}} \left(\frac{2m}{n}-1\right)^{\frac{5}{2}}} \cdot \left(1 + \mathcal{O}\left(n^{-\frac{1}{2}+7\epsilon} \right) \right),$$

and

$$M_{n,m} = \frac{m!}{(n-m)!} \cdot \frac{n^{2n-m-1} 2^{2m-n+1}}{\left(\frac{2m}{n}-1\right)^{\frac{5}{2}}} \cdot \left(1 + \mathcal{O}\left(n^{-\frac{1}{2}+7\epsilon} \right) \right).$$

The monkey saddle for $\rho = 1/2$

For $\rho = \frac{m}{n} = \frac{1}{2}$, the situation is slightly different to the previous regions since the two otherwise distinct saddle points coalesce to a unique double saddle point. The difference in the geometry of the surface, i.e., of the modulus of the large power $e^{n \cdot h(w)}$ in (73), is that there are now three steepest descent lines and three steepest ascents lines departing from the saddle point (in contrast to two steepest descent lines and two steepest ascents lines for the case of a simple saddle point). This explains why such saddle points are also referred to as "monkey saddles": they do not only offer space for two legs but also for a tail. In this particular case the three steepest descent and steepest ascent lines departing from the saddle point at $w = w_1 = w_2 = \frac{1}{2}$ have angles $0, 2\pi/3$ and $-2\pi/3$ as can be seen in the middle right of Figure 46. This also follows from a local expansion of $h(w)$ as defined in (74) around $w = \frac{1}{2}$:

$$h(w) = 1 - \frac{8}{3}\left(w - \frac{1}{2}\right)^3 + \mathcal{O}\left(\left(w - \frac{1}{2}\right)^4\right).$$

Thus, we may choose as integration contour two line segments joining the point $w = \frac{1}{2}$ with the imaginary axis at an angle of $-2\pi/3$ and $2\pi/3$, respectively, as well as a half circle centred at the origin and joining the two line segments. See the bottom right of Figure 46. This yields $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$ and $A_{n,m} = I^{(1)}_{n,m} + I^{(2)}_{n,m} + I^{(3)}_{n,m}$ for the corresponding integrals, where we use the parametrizations $\Gamma_1 := \left\{\frac{1}{2} - e^{-\frac{2\pi i}{3}} t : t \in [-1, 0]\right\}$, $\Gamma_2 := \left\{\frac{1}{2} + e^{\frac{2\pi i}{3}} t : t \in [0, 1]\right\}$, and $\Gamma_3 := \left\{\frac{\sqrt{3}}{2} e^{it} : t \in [\frac{\pi}{2}, \frac{3\pi}{2}]\right\}$.

We first treat

$$\begin{aligned} I^{(1)}_{n,m} &= \frac{1}{2\pi i} \int_{-1}^{0} \left(-e^{-\frac{2\pi i}{3}}\right) g\left(\frac{1}{2} - e^{-\frac{2\pi i}{3}} t\right) e^{nh\left(\frac{1}{2} - e^{-\frac{2\pi i}{3}} t\right)} dt \\ &= \frac{1}{2\pi i} \int_{0}^{1} \left(-e^{-\frac{2\pi i}{3}}\right) g\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right) e^{nh\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right)} dt. \end{aligned}$$

In order to find a suitable choice t_0 for splitting the integral for the central approximation and the remainder we consider the expansion of the integrand around $t = 0$, which can be obtained easily:

$$\begin{aligned} &\frac{-e^{-\frac{2\pi i}{3}}}{2\pi i} g\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right) e^{nh\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right)} \\ =&\frac{4e^n e^{-\frac{4\pi i}{3}}}{\pi i} t e^{-\frac{8}{3} n t^3} \cdot \left(1 + \mathcal{O}(t^2) + \mathcal{O}(nt^5)\right). \end{aligned} \tag{80}$$

Thus we obtain the restrictions $nt_0^3 \to \infty$ and $nt_0^5 \to 0$ which are, e.g., satisfied when choosing $t_0 = n^{-\frac{1}{4}}$. This splitting yields $I^{(1)}_{n,m} = I^{(1,1)}_{n,m} + I^{(1,2)}_{n,m}$, for the integration paths $t \in [0, t_0]$ and $t \in [t_0, 1]$, respectively.

Using the local expansion of the integrand (80) as well as the beforementioned choice for t_0, the central approximation $I_{n,m}^{(1,1)}$ gives

$$\begin{aligned} I_{n,m}^{(1,1)} =& \frac{4e^n e^{-\frac{4\pi i}{3}}}{\pi i} \int_0^{t_0} t e^{-\frac{8}{3}nt^3} dt \cdot \left(1 + \mathcal{O}(n^{-\frac{1}{4}})\right) \\ =& \frac{4e^n e^{-\frac{4\pi i}{3}}}{\pi i} \int_0^{\infty} t e^{-\frac{8}{3}nt^3} dt \cdot \left(1 + \mathcal{O}(n^{-\frac{1}{4}})\right), \end{aligned}$$

since one can show easily that completing the integral only yields a subexponentially small error term. Moreover, also the remainder

$$I_{n,m}^{(1,2)} = \frac{1}{2\pi i} \int_{t_0}^{1} \left(-e^{-\frac{2\pi i}{3}}\right) g\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right) e^{nh\left(\frac{1}{2} + e^{-\frac{2\pi i}{3}} t\right)} dt$$

only yields a subexponentially small error term compared to $I_{n,m}^{(1,1)}$. Thus, we get the contribution

$$I_{n,m}^{(1)} = \frac{4e^n e^{-\frac{4\pi i}{3}}}{\pi i} \int_0^{\infty} t e^{-\frac{8}{3}nt^3} dt \cdot \left(1 + \mathcal{O}(n^{-\frac{1}{4}})\right).$$

The integral

$$I_{n,m}^{(2)} = \frac{1}{2\pi i} \int_0^{1} \left(e^{\frac{2\pi i}{3}}\right) g\left(\frac{1}{2} + e^{\frac{2\pi i}{3}} t\right) e^{nh\left(\frac{1}{2} + e^{\frac{2\pi i}{3}} t\right)} dt,$$

can be treated in an analogous manner which gives the contribution

$$I_{n,m}^{(2)} = -\frac{4e^n e^{\frac{4\pi i}{3}}}{\pi i} \int_0^{\infty} t e^{-\frac{8}{3}nt^3} dt \cdot \left(1 + \mathcal{O}(n^{-\frac{1}{4}})\right).$$

Moreover, one can show that the contribution of

$$I_{n,m}^{(3)} = \frac{1}{2\pi i} \int_{\frac{\pi}{2}}^{\frac{3\pi}{2}} \frac{\sqrt{3}}{2} i e^{it} g\left(\frac{\sqrt{3}}{2} e^{it}\right) e^{nh\left(\frac{\sqrt{3}}{2} e^{it}\right)} dt$$

is asymptotically negligible compared to $I_{n,m}^{(1)}$ and $I_{n,m}^{(2)}$.

Collecting the contributions and evaluating the integral yields

$$\begin{aligned} A_{n,m} \sim& \frac{4e^n}{\pi i} \left(e^{-\frac{4\pi i}{3}} - e^{\frac{4\pi i}{3}}\right) \cdot \int_0^{\infty} t e^{-\frac{8}{3}nt^3} dt = \frac{4\sqrt{3}e^n}{\pi} \int_0^{\infty} t e^{-\frac{8}{3}nt^3} dt \\ =& \frac{3^{\frac{1}{6}} e^n n^{-\frac{2}{3}}}{\pi} \Gamma\left(\frac{2}{3}\right), \end{aligned}$$

and thus by using (71):

$$M_{n,m} \sim \frac{\sqrt{2}\, 3^{\frac{1}{6}} \Gamma(\frac{2}{3}) n^{\frac{3n}{2}}}{\sqrt{\pi}\, n^{\frac{1}{6}}}.$$

9.5 SUMMARY OF THE RESULTS

This chapter constitutes the first treatment of parking functions for trees and mappings. Let us summarize our contributions:

- We have provided several different characterizations of parking functions on trees and mappings, showing that results for ordinary parking functions may be extended. Also, we were able to characterize the extremal cases for the number of parking functions for trees, i.e., the types of trees which allow for a maximal or a minimal number of parking functions.

- The main task of this chapter was to count the number of pairs (T, s) where T is a tree of size n and s a parking function for s as well as the number of pairs (f, s) that are mapping parking functions. This was first done for the case that the number of drivers m coincides with the number n of parking spaces and later on for the general case $m = n$. The generating functions approach also led to the surprising result that $n \cdot F_{n,m} = M_{n,m}$. We were able to provide a bijective proof for this result, thus providing a combinatorial explanation of this interesting fact.

- The numbers $F_{n,m}$ and $M_{n,m}$ of tree and mapping parking functions were also analysed from an asymptotic point of view. Using the saddle point method, we could show that these numbers exhibit a phase transition behaviour for $m = n/2$, i.e., when half of the parking spaces are occupied. Similar phase transition phenomena have for instance been observed in the analysis of random graphs during the phase where a giant component has not yet emerged.

Part III

PREFERENCES AND ELECTIONS

This part deals with patterns in elections which are referred to as so-called *configurations* and occur within the concept of *domain restrictions*. We establish a connection to permutation patterns and study the likelihood of the single-peaked domain restriction under several probability models.

10

ON THE LIKELIHOOD OF SINGLE-PEAKED ELECTIONS

This chapter is based on joint work with Martin Lackner. A preliminary version of the results presented here has been published in the Proceedings of the 8th Multidisciplinary Workshop on Advances in Preference Handling (MPref 2014) [43].

The single-peaked restriction [22] is an extensively studied preference domain in social choice theory. A collection of preferences, i.e., total orders on a set of candidates, is single-peaked if the candidates can be ordered linearly – on a so-called axis – so that each preference is either strictly increasing along this ordering, or strictly decreasing, or first increasing and then decreasing. See Figure 48 for examples. Intuitively, the axis reflects the society's ordering of the candidates and voters always prefer candidates that are closer to their ideal candidate over those farther away. In political elections, for example, this axis could reflect the left-right spectrum of the candidates or a natural ordering of political issues such as the maximum income tax.

Single-peaked preferences have several nice properties. First, they guarantee that a Condorcet winner exists and further that the pairwise majority relation is transitive [100]. Let us explain this briefly: The pairwise majority relation $\prec_M$ is obtained by considering all pairs of candidates (c_i, c_j). If a majority of voters ranks c_i above c_j, then we set $c_j \prec_M c_i$. The relation obtained in this way is not necessarily transitive since cycles can occur. However, for single-peaked elections cycles are not possible and $\prec_M$ is indeed transitive which implies that it can serve as output for a social welfare function. The highest ranked candidate with respect to $\prec_M$ is called the *Condorcet winner* and is the candidate that wins most pairwise comparisons. Thus single-peaked preferences are a way to escape Arrow's paradox, Theorem 2.19. Second, non-manipulable voting rules exist for single-peaked preferences (Moulin 1980) and hence the single-peaked restriction also offers a way to circumvent the Gibbard-Satterthwaite paradox [89, 137]. By adopting an algorithmic viewpoint, a third advantage becomes apparent. Restricting the input to single-peaked preferences often allows for faster algorithms for computationally hard voting problems [20, 37, 71, 154].

In this chapter we perform the first combinatorial analysis of the single-peaked domain. Our aim is to establish results on the likelihood that a collection of preferences – which we call an election – is single-peaked for some axis. To be more precise, we allow the axis to be chosen depending on the preferences and do not assume that it is

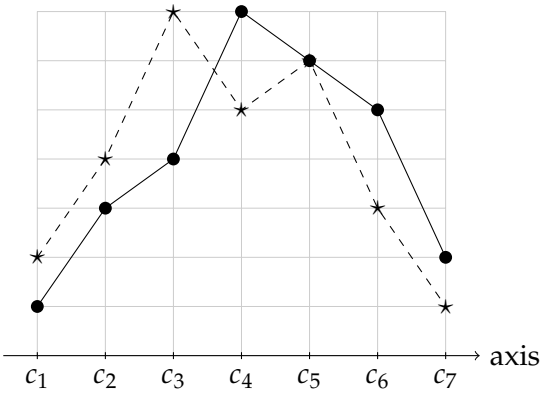


Figure 48: The vote $V_1 : c_4 > c_5 > c_6 > c_3 > c_2 > c_7 > c_1$, shown as a solid line, is single-peaked with respect to the axis $c_1 < c_2 < c_3 < c_4 < c_5 < c_6 < c_7$. The vote $V_2 : c_3 > c_5 > c_4 > c_2 > c_6 > c_1 > c_7$, depicted as a dashed line, is not single-peaked with respect to this axis since both c_3 and c_5 form a peak. However, note that both votes are single-peaked with respect to the axis $c_1 < c_2 < c_3 < c_5 < c_4 < c_6 < c_7$.

given together with the election. We consider three probability distributions for elections: the Impartial Culture (IC) assumption in which all total orders are equally likely and are chosen independently, the Pólya urn model which assumes a certain homogeneity among voters and Mallows model in which the probability of a vote depends on its Kendall-tau distance to a given reference vote.

This chapter is organized as follows: We start by presenting some related work as well as the studied probability models and give a formal definition of single-peaked elections. The results on configuration definable restrictions can be found in Section 10.1, results on counting single-peaked elections in Section 10.2, results on the Pólya urn model in Section 10.3 and results on the Mallows model in Section 10.4. In Section 10.5 we provide numerical evaluations of our results and discuss their implications. We summarize our results in Section 10.6.

RELATED WORK. Computing the likelihood of properties related to voting has been the focus of a large body of research. The most fundamental question in this line of research is the choice of appropriate probability distributions, see [54] for a survey. We would like to mention two particular properties of elections that have been studied from a probability theoretic point of view: the likelihood of manipulability and the likelihood of having a Condorcet winner.

An election is manipulable if a voter or a coalition of voters is better off by not voting sincerely but by misrepresenting their true preferences. The Gibbard-Satterthwaite paradox [89, 137] states that every reasonable voting system for more than two candidates is susceptible

to manipulation. However, the Gibbard-Satterthwaite does not offer insight into how likely it is that manipulation is possible. Determining this likelihood both for single manipulators and coalitions of manipulators has been the focus of intensive research. Results have been obtained under a variety of probability distributions: for example under the Independent Culture assumption [82, 101, 141, 142], the Pólya urn model [116], the Independent Anonymous Culture [72, 143].

The likelihood that an election has a Condorcet winner or, its converse, the likelihood of the Condorcet paradox has been the focus of many publications (see [86] for a survey and [87, 88] for more recent research). In particular, we would like to mention a result of Gehrlein [85] who determined that the likelihood of an election with three candidates having a Condorcet winner under the Impartial Culture assumption is $15(n+3)^2/[16(n+2)(n+4)]$. We will comment on the relation between this result and our results in Chapter 10.6.

PROBABILITY DISTRIBUTIONS OVER ELECTIONS. In this chapter we consider three probability distributions. The first and simplest is the Impartial Culture (IC) which assumes that in an election all votes, i.e., total orders of candidates, are equally likely and are chosen independently. Thus, the IC assumption can be seen as the uniform distribution over all elections of a given size. Our results concerning IC do not state probabilities but rather count the number of elections. If, e.g., the number of single-peaked (n,m)-elections is $a(n,m,SP)$, then the probability under the IC assumption that an (n,m)-election is single-peaked is $\frac{a(n,m,SP)}{(m!)^n}$. It is important to note that elections contain an ordered list of votes. Thus, we distinguish elections that consists of the same votes but appearing in a different order. This is in contrast to the Impartial Anonymous Culture assumption, in which elections contain a multiset of votes and thus elections are not ordered. We do not consider the Impartial Anonymous Culture assumption here.

In addition to the IC assumption, we consider the Pólya urn model and the Mallows model. Both distributions are generalizations of the IC assumption and generate more structured elections. We are going to define the Pólya urn and the Mallows model in Section 10.3 and 10.4, respectively.

SINGLE-PEAKED PREFERENCES The single-peaked restriction assumes that the candidates can be ordered linearly on a so-called axis and voters prefer candidates close to their ideal point to candidates that are further away.

Definition 10.1. *Let $(C,\mathcal{P})$ be an election and A a total order of C. A vote V on C* contains a valley with respect to A on the candidates $c_1,c_2,c_3 \in C$ *if $A : c_1\,c_2\,c_3$ and V ranks c_2 below c_1 and c_3. The election $(C,\mathcal{P})$ is* single-peaked with respect to A *if for every $V \in \mathcal{P}$ and for all candidates $c_1,c_2,c_3 \in C$, V does not contain a valley with respect to*

A on c_1, c_2, c_3. We then call the total order A the axis. The election $(C, \mathcal{P})$ is single-peaked *if there exists a total order A of C such that $(C, \mathcal{P})$ is single-peaked with respect to A.*

Remark 10.2. Given an axis on m candidates, there are 2^{m-1} votes that are single-peaked with respect to this axis. This has been shown in [68] and can be seen as follows: The last ranked candidate has to be one of the two outermost candidates on the axis and hence there are two possibilities. Once we have picked this last candidate, we can iterate the argument for the next lowest ranked candidate, where we again have two possibilities. Thus, for all positions in the total order (except for the top ranked candidate), there are two candidates to choose from – which yields 2^{m-1} possibilities in total.

10.1 A GENERAL RESULT BASED ON PERMUTATION PATTERNS

Before we study the single-peaked domain in detail, we prove a general result that is applicable to a large class of domain restrictions including the single-peaked domain. To precisely define this class of domain restrictions, we require the notion of configuration definability.

Configuration definable domain restrictions

Single-peaked elections may also be defined in the following way:

Theorem 10.3 (15)**.** *An (n, m)-election $(C, \mathcal{P})$ is single-peaked if and only if there do not exist candidates $a, b, c, d \in C$ and indices $i, j \in [n]$ such that*

- *$V_i : abc$, $V_i : db$,*
- *$V_j : cba$ and $V_j : db$ holds*

and there do not exist candidates $a, b, c \in C$ and indices $i, j, k \in [n]$ such that

- *$V_i : ba, ca$ (i.e., a is ranked below b and c),*
- *$V_j : ab, cb$ and*
- *$V_k : ac, bc$ holds.*

Note that this theorem defines single-peakedness without referring to an axis. Indeed, single-peakedness is now defined as a local property in the sense that certain *configurations* must not be contained in the election. Similar definitions have also been found for the single-crossing [39] and group-separable [15] domain. Our aim is to prove a theorem that is applicable to arbitrary domain restrictions that can be defined in such a way. Thus, we give a precise definition of what it means for a domain to be *configuration definable*.

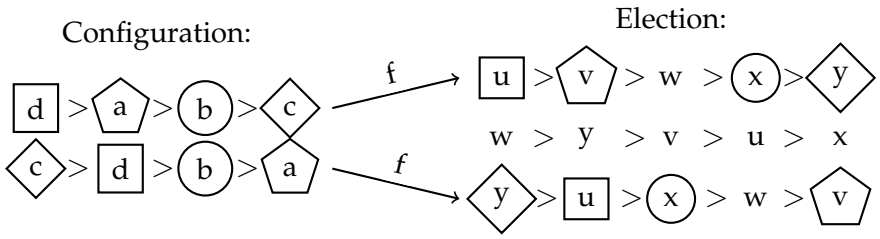


Figure 49: The configuration on the left-hand side is contained in the election on the right-hand side as witnessed by $f : \{1 \mapsto 1, 2 \mapsto 3\}$ and $g : \{a \mapsto v, b \mapsto x, c \mapsto y, d \mapsto u\}$.

Definition 10.4. *An* (l,k)-configuration $(S,\mathcal{T})$ *consists of a finite set* S *of cardinality* k *and a tuple* $\mathcal{T} = (T_1,\dots,T_l)$*, where* $T_1,\dots,T_l$ *are total orders on* S*. An election* $(C,\mathcal{P})$ contains *configuration* $\mathcal{C}$ *if there exist an injective function* f *from* $[l]$ *into* $[n]$ *and an injective function* g *from* S *into* C *such that, for any* $x,y \in S$ *and* $i \in [l]$*, it holds that* $T_i : xy$ *implies* $V_{f(i)} : g(x)g(y)$*.*

We use $(S,\mathcal{T}) \sqsubseteq (C,\mathcal{P})$ as a shorthand notation to denote that the election $(C,\mathcal{P})$ contains the configuration $(S,\mathcal{T})$. An election $(C,\mathcal{P})$ *avoids* a configuration $(S,\mathcal{T})$ if $(C,\mathcal{P})$ does not contain $(S,\mathcal{T})$. In such a case we say that $(C,\mathcal{P})$ is $(S,\mathcal{T})$*-restricted*. If the set S is clear from the context, we omit it and just use $\mathcal{T}$ to describe a configuration.

Example 10.5. Let us consider an election $(C,\mathcal{P})$ with $C = \{u,v,w,x,y\}$ and $\mathcal{P} = (uvwxy, wyvux, yuxwv)$ and a configuration $(S,\mathcal{T})$ with $S = \{a,b,c,d\}$ and $\mathcal{T} = (dabc, cdba)$. Election $(C,\mathcal{P})$ contains the configuration $(S,\mathcal{T})$ as witnessed by the functions $f : \{1 \mapsto 1, 2 \mapsto 3\}$ and $g : \{a \mapsto v, b \mapsto x, c \mapsto y, d \mapsto u\}$. In Figure 49, the functions f and g are depicted graphically.

By considering all linearizations of the partial orders appearing in Theorem 10.3 we can now restate it as follows.

Theorem 10.6 (15)**.** *An election is single-peaked if and only if it avoids*

- *the following* $(2,4)$*-configurations:*
  $(dabc, dcba)$, $(adbc, dcba)$, $(dabc, cdba)$ *and* $(adbc, cdba)$
- *as well as the following* $(3,3)$*-configurations:*
  (bca, acb, abc), (cba, acb, abc), (bca, cab, abc), (cba, cab, abc),
  (bca, acb, bac), (cba, acb, bac), (bca, cab, bac), (cba, cab, bac).

The first four configurations correspond to the first condition in Theorem 10.3, the remaining eight correspond to the second condition.

Definition 10.7. *Let* Γ *be a set of configurations. A set of elections* Π *is* defined by Γ *if* Π *consists exactly of those elections that avoid all configurations in* Γ*. We call* Π configuration definable *if there exists a set of configurations* Γ *which defines* Π*. If* Π *is definable by a finite set of configurations, it is called* finitely configuration definable*.*

By Theorem 10.6 we know that the set of all single-peaked elections is finitely configuration definable. This is also true for the set of group-separable elections [15] and for the set of single-crossing elections [39].

We are now going to characterize which sets of elections are configuration definable. In the following definition, for two elections $(C, \mathcal{P})$ and $(C', \mathcal{P}')$, we write $(C', \mathcal{P}') \sqsubseteq (C, \mathcal{P})$ if $(C', \mathcal{P}')$, considered as a configuration, is contained in $(C, \mathcal{P})$. Since every election can be seen as a configuration, the configuration containment relation immediately translates to election containment.

Definition 10.8. *A set of elections* Π *is* hereditary *if for every election* $(C', \mathcal{P}')$ *it holds that if there exists an election* $(C, \mathcal{P}) \in \Pi$ *with* $(C', \mathcal{P}') \sqsubseteq (C, \mathcal{P})$*, then* $(C', \mathcal{P}') \in \Pi$*.*

Proposition 10.9. *A set of elections is configuration definable if and only if it is hereditary.*

Proof. Let a set of elections Π be defined by a set of configurations Γ and $(C, \mathcal{P}) \in \Pi$. Let $(C', \mathcal{P}') \sqsubseteq (C, \mathcal{P})$. Since $(C, \mathcal{P}) \in \Pi$, $(C, \mathcal{P})$ avoids all configurations in Γ. Due to $(C', \mathcal{P}') \sqsubseteq (C, \mathcal{P})$, also $(C', \mathcal{P}')$ avoids all configurations in Γ and is therefore contained in Π. For the other direction, let Π^c denote the set of all elections that are not contained in Π. It is easy to observe that Π^c is a (possibly infinite) set of configurations that defines Π. □

As a consequence of Proposition 10.9, we know that 2D single-peaked elections [17] and 1D Euclidean [50, 110] are configuration definable. However, Proposition 10.9 does not help to answer whether these restrictions are *finitely* configuration definable. Finite configuration definability has been crucial for establishing algorithmic results [38, 64].

A natural example of a meaningful restriction that is not configuration definable is the set of all elections that have a Condorcet winner. The property of having a Condorcet winner is not hereditary and thus cannot be defined by configurations. There are also sets of elections that are configuration definable but not finitely configuration definable. The proof of this statement builds upon a connection between configuration avoiding elections and permutation patterns, which we establish now.

The connection to permutation patterns

In this section, we establish a strong link between the concept of configuration containment and the concept of pattern containment in permutations. We are going to prove two lemmas. The first lemma (Lemma 10.10) states that every permutation pattern matching query can naturally be translated in a configuration containment query. The second lemma (Lemma 10.12) states that for $(2,k)$-configurations, a configuration containment query can naturally be translated in a permutation pattern query.

Lemma 10.10. *Let π be a k-permutation and τ an m-permutation. We define the corresponding configuration and election as follows: Let $(C,\mathcal{P})$ be a $(3,m)$-election with $C = \{c_1,\dots,c_m\}$, $\mathcal{P} = (V_1,V_2,V_3)$ and*

$$V_1 : c_1c_2\cdots c_m \qquad V_2 : c_1c_2\cdots c_m \qquad V_3 : c_{\tau(1)}c_{\tau(2)}\cdots c_{\tau(m)}.$$

Furthermore, let $(S,\mathcal{T})$ be a $(3,k)$-configuration with $S = \{x_1,\dots,x_k\}$, $\mathcal{T} = (T_1,T_2,T_3)$ and

$$T_1 : x_1x_2\cdots x_m \qquad T_2 : x_1x_2\cdots x_m \qquad T_3 : x_{\pi(1)}x_{\pi(2)}\cdots x_{\pi(k)}.$$

Then $(C,\mathcal{P})$ contains $(S,\mathcal{T})$ if and only if τ contains π.

Proof. Assume that we have a matching μ from π into τ. We have to find an injective function f from $\{1,2,3\}$ into $\{1,2,3\}$ and an injective function g from S into C such that, for any $x,y \in S$ and $i \in \{1,2,3\}$, it holds that $T_i : xy$ implies $V_{f(i)} : g(x)g(y)$. Let f be the function $\{1 \mapsto 1, 2 \mapsto 2, 3 \mapsto 3\}$ and $g = \mu$. It holds for $x_i, x_j \in S$ that $T_1 : x_i\,x_j$ if and only if $V_1 : c_{\mu(i)}\,c_{\mu(j)}$ since μ is monotone. The same holds for T_2 and V_2. For T_3 and V_3 observe that $T_3 : x_i\,x_j$ implies $V_3 : c_{\mu(i)}\,c_{\mu(j)}$ since μ is a matching. Thus, the election fulfils $(S,\mathcal{T}) \sqsubseteq (C,\mathcal{P})$.

For the other direction, assume that $(C,\mathcal{P})$ contains $(S,\mathcal{T})$. Consequently, there exists an injective function f from $\{1,2,3\}$ into $\{1,2,3\}$ and an injective function g from S into C such that, for any $x,y \in S$ and $i \in \{1,2,3\}$, it holds that $T_i : xy$ implies $V_{f(i)} : g(x)g(y)$. First, we claim that $f(3) = 3$. Observe that f has to map T_1 and T_2 to identical total orders. Thus, unless $V_1 = V_2 = V_3$, $f(3) = 3$. In the case that $V_1 = V_2 = V_3$, we can assume without loss of generality that $f(3) = 3$. We will construct a function μ and show that μ is a matching from π into τ. Let us define $\mu(i) = j$ if $g(x_i) = c_j$. Observe that μ is strictly increasing since for $i < j$, $V_1 : c_{g(i)}\,c_{g(j)}$ and $V_1 : c_1c_2\cdots c_m$. In addition, $\mu(\pi) = \big(\mu(\pi(1)),\mu(\pi(2)),\dots,\mu(\pi(k))\big)$ is a subsequence of τ since, by definition of T_3 and V_3 and the fact that $f(3) = 3$, $\big(g(x_{\pi(1)}),g(x_{\pi(2)}),\dots,g(x_{\pi(k)})\big)$ is a subsequence of $\big(c_{\tau(1)},c_{\tau(2)},\dots,c_{\tau(m)}\big)$. □

We can now state our first result that builds upon the connection between configuration avoiding elections and permutation patterns.

Proposition 10.11. *There are sets of elections that are configuration definable but not finitely configuration definable.*

Proof. Take an infinite set of permutations that pairwise avoid each other [25]. Then use the construction in Lemma 10.10 to generate corresponding configurations. There are no two configurations that pairwise contain each other. Thus the set of elections defined by avoiding these configurations is not definable by a finite set of configurations. □

Next, we will prove the second lemma, which is essential for Theorem 10.13. As of now, we shall denote by $S_m(\pi_1, \ldots, \pi_l)$ the cardinality of the set of m-permutations that avoid the patterns $\pi_1, \ldots, \pi_l$.

Lemma 10.12. *Let $(S, \mathcal{T})$ be a $(2,k)$-configuration with $\mathcal{T} = (T_1, T_2)$. Furthermore, let V_1 be a total order on the candidate set $C = \{c_1, \ldots, c_m\}$. Then the number of total orders V_2 such that the election $(C, \mathcal{P})$ with $\mathcal{P} = (V_1, V_2)$ avoids $(S, \mathcal{T})$ is equal to $S_m(\pi, \pi^{-1})$, where $\pi = p(C_1, C_2)$.*

Proof. Let us start by proving the following statement: The configuration $(S, \mathcal{T})$ is contained in an election $(C, \mathcal{P})$ with $\mathcal{P} = (V_1, V_2)$ if and only if the permutation π or the permutation π^{-1} is contained in $p(V_1, V_2)$. In order to alleviate notation, we will assume in the following that $C = \{1, 2, \ldots m\}$ and $S = \{1, 2, \ldots k\}$.

"$\Leftarrow$" We can assume without loss of generality that $T_1 : 12 \ldots k$ and $V_1 : 12 \ldots m$. If π is contained in $p(V_1, V_2)$ as witnessed by a matching μ, then the functions $f = \{1 \mapsto 1, 2 \mapsto 2\}$ and $g = \mu$ show that $(S, \mathcal{T}) \sqsubseteq (C, \mathcal{P})$ (cf. Definition 10.4). If π^{-1} is contained in $p(V_1, V_2)$ as witnessed by a matching μ, then the functions $f = \{1 \mapsto 2, 2 \mapsto 1\}$ and $g = \mu$ show that $(S, \mathcal{T}) \sqsubseteq (C, \mathcal{P})$.

"$\Rightarrow$" Let $(S, \mathcal{T}) \sqsubseteq (C, \mathcal{P})$. Without loss of generality we assume that $T_1 : 12 \ldots k$. Note that renaming C does not change whether $(S, \mathcal{T}) \sqsubseteq (C, \mathcal{P})$. Thus, it is safe to rename the candidates according to the f function: If $f = \{1 \mapsto 1, 2 \mapsto 2\}$, let $V_1 : 12 \ldots n$. Since $f(1) = 1$, g is monotonic. It is easy to verify that g is a matching from π into $p(V_1, V_2)$. If $f = \{1 \mapsto 2, 2 \mapsto 1\}$, let $V_2 : 12 \ldots n$. Now, g is a matching from π into $p(V_2, V_1) = (p(V_1, V_2))^{-1}$. This is equivalent to g being a matching from π^{-1} into $p(V_1, V_2)$.

It follows that $(C, \mathcal{P})$ avoids the configuration $(S, \mathcal{T})$ if and only if the permutation $p(V_1, V_2)$ avoids both the patterns π and π^{-1}. Moreover, for the fixed total order V_1 and a fixed m-permutation τ, there is a single total order V_2 such that $p(V_1, V_2) = \tau$. Thus the number of votes V_2 such that $p(V_1, V_2)$ avoids π and π^{-1} (and equivalently the number of votes V_2 such that $(C, \mathcal{P})$ avoids $(S, \mathcal{T})$) is equal to $S_m(\pi, \pi^{-1})$, the number of m-permutations avoiding π and π^{-1}. □

From this lemma follows the main theorem of this section that is applicable to any set of configurations that contains at least one configuration of cardinality two.

Elections that avoid a $(2,k)$-configuration

With the help of Lemma 10.12, we are able to establish the following result.

Theorem 10.13. *Let $a(n,m,\Gamma)$ be the number of (n,m)-elections avoiding a set of configurations Γ. Let $k \geq 2$. If a set of configurations Γ contains a $(2,k)$-configuration, then it holds for all $n,m \in \mathbb{N}$ that*

$$a(n,m,\Gamma) \leq m! \cdot c_k^{(n-1)m},$$

where c_k is a constant depending only on k.

This result shows that forbidding any $(2,k)$-configuration is a very strong restriction. Indeed, $m! \cdot c_k^{(n-1)m}$ is very small compared to the total number of (n,m)-elections which is $(m!)^n$. This result allows us to bound the number of single-peaked and group-separable elections. However, let us prove this result first before we explore its consequences.

In order to prove this result we make use of the link between configuration avoiding elections and pattern avoiding permutations established in Lemma 10.12 and profit from a very strong result within the theory of pattern avoidance in permutations, the Marcus-Tardos theorem (former Stanley-Wilf conjecture).

Proof. We will provide an upper bound on the number of (n,m)-elections avoiding a $(2,k)$-configuration $(S,\mathcal{T})$ with $\mathcal{T} = (T_1,T_2)$. Let us start by choosing the first vote V_1 of the election at random. For this there are $m!$ possibilities. When choosing the remaining $(n-1)$ votes $V_2,\dots,V_n$, we have to make sure that no selection of two votes contains the forbidden configuration $(S,\mathcal{T})$. We relax this condition and only demand that for none of the pairs (V_1,V_i) with $i \neq 1$, the election $(C,(V_1,V_i))$ contains the forbidden configuration. Hereby we obtain an upper bound for $a(n,m,\{\mathcal{C}\})$. Now Lemma 10.12 tells us that there are – under this relaxed condition – $S_m(\pi,\pi^{-1})$ choices for every V_i where $\pi = p(C_1,C_2)$. Thus we have the following upper bound:

$$a(n,m,\{\mathcal{C}\}) \leq m! S_m(\pi,\pi^{-1})^{n-1} \leq m! S_m(\pi)^{n-1}, \tag{81}$$

where the second inequality follows since all permutations avoiding both π and π^{-1} clearly avoid π.

Now we apply the famous Marcus-Tardos theorem [124]: For every permutation π of length k there exists a constant c_k such that for all positive integers m we have $S_m(\pi) \leq {c_k}^m$. Putting this together with Equation (81) and noting that $a(n,m,\{\mathcal{C}\})$ is an upper bound for $a(n,m,\Gamma)$, we obtain the desired upper bound. □

The proof of the Marcus-Tardos theorem provides an explicit exponential formula for the constants c_k. Indeed, it holds that

$$S_m(\pi) \leq \left(15^{2k^4\binom{k^2}{k}}\right)^m.$$

These constants are however far from being optimal and there is an ongoing effort to find minimal values for c_k with fixed k. In particular it has been shown that $c_2 = 1$, $c_3 = 4$ [140] and $c_4 \leq 13.738$ [28].

Let us discuss the implications of this theorem. It is applicable to all (not necessarily finite) configuration definable domain restrictions that contain a configuration of cardinality two. In particular, we obtain the following upper bounds for single-peaked and group-separable elections.

Corollary 10.14. *Let $a(n, m, \Gamma_{sp})$ denote the number of single-peaked (n, m)-elections. For $n, m \geq 2$ it holds that*

$$a(n, m, \Gamma_{sp}) \leq m! \cdot 4^{(m-1)(n-1)}.$$

Proof. We know from Theorem 10.3 that the single-peaked domain avoids the $(2, 4)$-configurations $(dabc, dcba)$, $(adbc, dcba)$, $(dabc, cdba)$ and $(adbc, cdba)$. We can use Equation (81) in the proof of Theorem 10.13 to bound $a(n, m, \Gamma_{sp})$.

For this, we have to compute the permutations and their inverses corresponding to the four configurations. We obtain the permutations $\pi_1 = p(dabc, dcba) = 1432$, $\pi_2 = p(adbc, dcba) = 4132$, $\pi_3 = p(dabc, cdba) = 2431$ and $\pi_4 = p(adbc, cdba) = 4231$. Their inverses are $\pi_1^{-1} = \pi_1$, $\pi_2^{-1} = \pi_3$, $\pi_3^{-1} = \pi_2$ and $\pi_4^{-1} = \pi_4$. Hence it holds that the number of (n, m)-elections that avoid these four configurations is bounded by $m! \cdot S_m(\pi_1, \pi_2, \pi_3, \pi_4)^{n-1}$. As shown in Theorem 5.1, $S_m(\pi_1, \pi_2, \pi_3, \pi_4) = \binom{2m-2}{m-1}$, which, in turn, is bounded by 4^{m-1}. □

In the next section, we will see that the growth rate of $a(n, m, \Gamma_{sp})$ is indeed of the form $m! \cdot c^{(m-1)(n-1)}$ for some constant c. However, the constant found in Corollary 10.14 is not optimal as we will see by providing a better bound for the single-peaked restriction that is even asymptotically optimal.

As another corollary of Theorem 10.13, we prove a bound on the number of group-separable elections. An election is group separable if for every subset of candidates C' there exists a partition C_1, C_2 of C' such that in every vote either all candidates in C_1 are preferred to all candidates in C_2 or vice versa. In [15] it is shown that the group-separable domain is finitely configuration definable. In particular, this domain avoids the configuration $(abcd, bdac)$. Therefore, Theorem 10.13 is applicable.

Corollary 10.15. *Let $a(n, m, \Gamma_{gs})$ denote the number of group-separable (n, m)-elections. For $n, m \geq 2$ it holds that*

$$a(n, m, \Gamma_{gs}) \leq m! \cdot (3 + 2\sqrt{2})^{m(n-1)}.$$

Proof. The proof is similar to the one of Corollary 10.14. We use Equation (81) in the proof of Theorem 10.13 to bound $a(n, m, \Gamma_{gs})$, i.e., $a(n, m, \Gamma_{gs}) \leq m! \cdot S_m(\pi, \pi^{-1})^{n-1}$, where $\pi = p(abcd, bdac) = 2413$ and $\pi^{-1} = 3142$. Permutations avoiding these two patterns are known under the name of *separable permutations*. It is known that separable permutations are counted by the large Schröder numbers (OEIS A006318) and that $S_m(\pi, \pi^{-1}) \leq (3 + 2\sqrt{2})^m$ [156]. □

10.2 COUNTING RESULTS AND THE IMPARTIAL CULTURE ASSUMPTION

As in the previous section, let $a(n, m, \Gamma_{\text{sp}})$ denote the number of single-peaked elections. In this section, we prove a lower and upper bound on $a(n, m, \Gamma_{\text{sp}})$. These two bounds are asymptotically optimal, i.e., the lower bound converges to the upper bound for every fixed m and $n \to \infty$. In addition, we prove exact enumeration results for $a(2, m, \Gamma_{\text{sp}})$, $a(n, 3, \Gamma_{\text{sp}})$ and $a(n, 4, \Gamma_{\text{sp}})$.

Our results immediately imply derive bounds on the probability that an (n, m)-election is single-peaked assuming that elections are drawn uniformly at random, i.e., according to the Impartial Culture assumption. The probability is simply $a(n, m, \Gamma_{\text{sp}})/(m!)^n$.

Theorem 10.16. *It holds that*

$$\frac{m!}{2} \cdot 2^{(m-1)\cdot n} \cdot (1 - \epsilon(n, m)) \leq a(n, m, \Gamma_{sp}) \leq \frac{m!}{2} \cdot 2^{(m-1)\cdot n},$$

where $\epsilon(n, m) \to 0$ for every fixed m and $n \to \infty$.

Proof. First observe that an election is single-peaked with respect to an axis if and only if it is single-peaked with respect to its reverse, i.e., the axis read from right to left. Thus the total number of axes on m candidates that need to be considered is $m!/2$. Second, recall that the number of votes that are single-peaked with respect to a given axis is 2^{m-1} (cf. Remark 10.2).

Now we have gathered all facts necessary for the upper bound. For every one of the $m!/2$ axes considered, select an ordered set of votes from the 2^{m-1} votes that are single-peaked with respect to this axis. There are exactly $2^{(m-1)\cdot n}$ such possibilities, which yields the upper bound. Since an election may be single-peaked with respect to more than two axes, this number is only an upper bound for $a(n, m, \Gamma_{\text{sp}})$.

Let us turn to the lower bound. Given a vote V, there are only two axes with respect to which both V and its reverse $\bar{V}$ are single-peaked,

namely the total orders V and $\bar{V}$ themselves. Thus the presence of the votes V and $\bar{V}$ in an election forces the axis to be equal to either V or $\bar{V}$. If we fix a vote V, the number of single-peaked elections containing both V and $\bar{V}$ can thus be determined exactly. Multiplying this by the number of possible choices for V leads to:

$$\frac{m!}{2} \cdot \sum_{\substack{1 \le i,j \\ i+j \le n}} \binom{n}{i} \cdot \binom{n-i}{j} \left(2^{m-1}-2\right)^{n-i-j}, \tag{82}$$

where i denotes the number of times the vote V appears in the elections and j the number of times the vote $\bar{V}$ appears. The other votes may be any of the 2^{m-1} votes that are single-peaked with respect to the axis V respectively $\bar{V}$ – except V and $\bar{V}$ themselves – and $n-i-j$ of them must be chosen.

Setting $k = i + j$ in equation (82) leads to:

$$\begin{aligned}
&\sum_{k=2}^{n} \sum_{i=1}^{k-1} \binom{n}{i} \cdot \binom{n-i}{k-i} \left(2^{m-1}-2\right)^{n-k} \\
&= \sum_{k=2}^{n} \left(2^{m-1}-2\right)^{n-k} \binom{n}{k} \sum_{i=1}^{k-1} \binom{k}{i} \\
&= \sum_{k=2}^{n} \left(2^{m-1}-2\right)^{n-k} \binom{n}{k} \left(2^{k}-2\right) \\
&= \left(2^{m-1}\right)^{n} - 2 \cdot n \left(2^{m-1}-2\right)^{n-1} - \left(2^{m-1}-2\right)^{n} \\
&\quad - 2 \cdot \left(\left(2^{m-1}-1\right)^{n} - n \cdot \left(2^{m-1}-2\right)^{n-1} - \left(2^{m-1}-2\right)^{n} \right) \\
&= 2^{(m-1)\cdot n} + \left(2^{m-1}-2\right)^{n} - 2 \cdot \left(2^{m-1}-1\right)^{n} \\
&= 2^{(m-1)\cdot n} \cdot \left(1 - \epsilon(n,m)\right), \\
&\qquad \text{where } \epsilon(n,m) = \frac{2 \cdot (2^{m-1}-1)^{n} - \left(2^{m-1}-2\right)^{n}}{2^{(m-1)n}}.
\end{aligned}$$

Since $\epsilon(n,m) \le 2 \cdot \left(\frac{2^{m-1}-1}{2^{m-1}}\right)^{n}$, $\epsilon(n,m)$ tends to 0 for every fixed m and $n \to \infty$. Clearly, not all single-peaked elections contain a pair of votes where one is the reverse of the other. Thus this number is indeed only a lower bound. □

In the next theorem we prove exact enumeration formulæ for the number of (n,m)-single-peaked elections for $n = 2$, $m = 3$ and $m = 4$. Note that for $m \le 2$ and for $n = 1$ all (n,m)-elections are single-peaked. For $n > 2$ and for $m > 4$ we have not been able to find exact enumeration formulæ.

Theorem 10.17. *It holds that*

(i.) $a(2,m,\Gamma_{sp}) = m! \cdot \binom{2m-2}{m-1}$ *for* $m \ge 1$,

(ii.) $a(n, 3, \Gamma_{sp}) = 6 \cdot 2^{n-1}\,(2^n - 1)$ *and*

(iii.) $a(n, 4, \Gamma_{sp}) = 24 \cdot 4^{n-1} \cdot \left(2^{n+1} - 3\right)$.

Proof. (i.) $a(2, m, \Gamma_{\text{sp}}) = m! \cdot \binom{2m-2}{m-1}$: This follows from Lemma 10.12. We choose the first vote arbitrarily ($m!$ possibilities). The second vote has to be chosen in such a way that all configurations that characterize single-peakedness are avoided. Since we consider only elections with two votes, the relevant configurations are $(dabc, dcba)$, $(adbc, dcba)$, $(dabc, cdba)$ and $(adbc, cdba)$ (Theorem 10.6). We obtain the permutations $\pi_1 = p(dabc, dcba) = 1432$, $\pi_2 = p(adbc, dcba) = 4132$, $\pi_3 = p(dabc, cdba) = 2431$ and $\pi_4 = p(adbc, cdba) = 4231$. Their inverses are $\pi_1^{-1} = \pi_1$, $\pi_2^{-1} = \pi_3$, $\pi_3^{-1} = \pi_2$ and $\pi_4^{-1} = \pi_4$. Thus, the number of $a(2, m, \Gamma_{\text{sp}}) = S_m(\pi_1, \pi_2, \pi_3, \pi_4)$. As shown in Theorem 5.1, $S_m(\pi_1, \pi_2, \pi_3, \pi_4) = \binom{2m-2}{m-1}$.

(ii.) $a(n, 3, \Gamma_{\text{sp}}) = m! \cdot 2^{n-1}\,(2^n - 1)$: We consider all elections with three candidates. There are $m!$ many possibilities for the first vote V_1. Without loss of generality, let us consider only the vote $V_1 : abc$. Since we have only three candidates, single-peakedness boils down to having at most two last ranked candidates (cf. Theorem 10.3). Due to our assumption that $V_1 : abc$, we distinguish three cases: elections in which the votes rank either a or c last, elections in which the votes rank either b or c last and elections in which all votes rank c last. The number of elections in which the votes rank either a or c last can be determined as follows: every vote can either be abc, bac, cba or bca. Hence, there are 4^{n-1} possibilities for elections in which the votes rank either a or c last and where $V_1 : abc$ holds. By the same argument, the number of elections in which the votes rank either b or c last is 4^{n-1} as well. The number of elections where c is always ranked last is 2^{n-1}. We obtain a total number of single-peaked elections with a fixed first vote of $4^{n-1} + 4^{n-1} - 2^{n-1} = 2^{n-1} \cdot (2 \cdot 2^{n-1} - 1)$. Given that 6 options for the first vote exist, we obtain the stated enumeration result.

(iii.) $a(n, 4, \Gamma_{\text{sp}}) = m! \cdot 4^{n-1} \cdot \left(2^{n+1} - 3\right)$: As in the previous proof, we fix $V_1 : abcd$. This vote V_1 already rules out some possible axes. Indeed, only eight axes are single-peaked axes for V_1, namely $A_1 : abcd$, $A_2 : bacd$, $A_3 : cabd$, $A_4 : cbad$, and their reverses. Since the reverse of an axis permits the same single-peaked votes, we have to consider only A_1, A_2, A_3, A_4. For $1 \leq i \leq 4$, let W_i denote the set of four-candidate votes that are single-peaked with respect to axis A_i.

We count the number of single-peaked elections with four candidates by using the inclusion-exclusion principle, i.e.,

$$\begin{aligned} a(n,4,\Gamma_{\text{sp}}) =& m! \cdot (|W_1| + |W_2| + |W_3| + |W_4| \\ & - |W_1 \cap W_2| - |W_1 \cap W_3| - |W_1 \cap W_4| - \\ & - |W_2 \cap W_3| - |W_2 \cap W_4| - |W_3 \cap W_4| \\ & + |W_1 \cap W_2 \cap W_3| + |W_1 \cap W_2 \cap W_4| + |W_2 \cap W_3 \cap W_4| \\ & - |W_1 \cap W_2 \cap W_3 \cap W_4|) \end{aligned}$$

It is easy to verify that $W_1 \cap W_2 = W_1 \cap W_4 = W_2 \cap W_3 = \{abcd, bacd\}$. Consequently, all intersections of three or four sets consist also of these two votes. The remaining intersections look as follows:

$$\begin{aligned} W_1 \cap W_3 &= \{abcd, bacd, cbad, bcad\}, \\ W_2 \cap W_4 &= \{abcd, bacd, cabd, acbd\}, \\ W_3 \cap W_4 &= \{abcd, bacd, badc, abdc\}. \end{aligned}$$

The number of votes single-peaked with respect to one axis is 2^{m-1} (see Remark 10.2), i.e., in our case 8. We obtain

$$\begin{aligned} a(n,4,\Gamma_{\text{sp}}) &= 4! \cdot (4 \cdot 8^{n-1} - 3 \cdot 2^{n-1} - 3 \cdot 4^{n-1} + 4 \cdot 2^{n-1} - 2^{n-1}) = \\ &= 24 \cdot (4 \cdot 8^{n-1} - 3 \cdot 4^{n-1}). \end{aligned}$$

□

10.3 THE PÓLYA URN MODEL

The Pólya urn model (also refereed to as the Pólya-Eggenberger urn model) [19, 105, 121] is an approach to sample elections with a variable degree of *social homogeneity*, i.e., where preferences are not independent but voters tend to have the same preferences as other voters. In the following the parameter a, a non-negative integer, describes the degree of social homogeneity. As we will see in a moment, the case $a = 0$ corresponds to the Impartial Culture assumption, i.e., a population with no homogeneity.

The setting of the Pólya urn model for an election with n votes and m candidates can be described as follows. Consider a large urn containing $m!$ balls. Every ball represents one of the $m!$ possible votes on the candidate set and has a different colour. An election is then created by subsequently pulling n balls out of the urn according to the following rule. The first ball is pulled at random and constitutes the first vote of the election. Then the pulled ball is returned to the urn and a other balls of the same colour are added to the urn. This procedure is repeated n times until an election consisting of n votes is created.

At a first glance, it might seem that the probability assigned to a certain election within the Pólya urn model depends on the order

of the votes. However this is not the case: Any election that can be obtained by rearranging a given election $(C, \mathcal{P})$, i.e, by changing the order of the votes, has exactly the same probability of occurring as the election $(C, \mathcal{P})$ itself. First, when the i-th ball is drawn from the urn, i.e., when the i-th vote is chosen, there are always $m! + (i-1) \cdot a$ balls present in the urn. Second, for any vote V the number of balls corresponding to V, i.e., the number of favourable cases, only depends on how often the vote V has already been pulled out of the urn and is equal to $(1 + k \cdot a)$ where k is the number of times V has already been pulled.

It is now easy to give a concise characterization of this discrete distribution. In order to alleviate notation, let us use the so called *Pochhammer k-symbol* as introduced in [58].

Definition 10.18. *The Pochhammer k-symbol is defined as*

$$(x)_{n,k} = \prod_{i=1}^{n} (x + (i-1) \cdot k)$$

where in our context $x \in \mathbb{R}$ and n, k are non-negative integers. Note that $(x)_{n,1} = x(x+1)(x+2)\dots(x+n-1)$ is the ordinary Pochhammer symbol (also known as rising factorial) and $(1)_{n,1} = n!$.

We can now define the probability of a given (n, m)-election with ℓ distinct votes. Let $n_i, i \in [m!]$ be non-negative integers with $\sum_{i=1}^{m!} n_i = n$ such that, for all $i \in [\ell]$ vote V_i appears n_i times. The probability of such an election is given by:

$$\binom{n}{n_1, \dots, n_\ell} \cdot \frac{\prod_{i=1}^{\ell} (1)_{n_i,a}}{(m!)_{n,a}} = \frac{n!}{\prod_{i=1}^{\ell} n_i!} \cdot \frac{\prod_{i=1}^{\ell} (1)_{n_i,a}}{(m!)_{n,a}}. \tag{83}$$

Note that setting $a = 0$ corresponds to the case where every one of the n votes is drawn from exactly the same urn, namely the urn containing every one of the $m!$ balls exactly once. Thus, the votes are chosen independently and every vote has the same probability of occurring; this corresponds to the Impartial Culture assumption.

Theorem 10.19. *Let $p_P(n, m, a)$ denote the probability of an (n, m)-election being single-peaked if it is created according to the Pólya urn model with homogeneity a. It holds that:*

$$p_P(n, m, a) \geq \frac{m!(n-1)!}{a\left(\frac{m!}{a}\right)_{n,1}} \cdot \left[1 + \frac{2}{a}\binom{2m-2}{m-1} H_{n-1} + \right.$$
$$\left. + \frac{n}{a} \sum_{l=2}^{n-1} \frac{(2^{m-1}-2)_{n-l,a}}{a^{n-l}} \cdot \frac{H_{l-1}}{l(n-l)!}\right],$$

where H_k denotes the k-th harmonic number $\sum_{i=1}^{k} \frac{1}{i}$.

Proof. Before we start with the actual proof, let us collect a few useful observations. In the following we will use that

$$(m!)_{n,a} = a^n \cdot \prod_{i=1}^{n} \left(\frac{m!}{a} + (i-1) \right) = a^n \left(\frac{m!}{a} \right)_{n,1}. \tag{84}$$

Moreover, we will use the following bound:

$$(1)_{k,a} = \prod_{i=1}^{k} (1 + (i-1) \cdot a) \geq \prod_{i=2}^{k} (i-1) \cdot a = (k-1)! \cdot a^{k-1}. \tag{85}$$

Since it holds that

$$\frac{1}{i \cdot (l-i)} = \frac{1}{l} \cdot \left(\frac{l-i+i}{i \cdot (l-i)} \right) = \frac{1}{l} \cdot \left(\frac{1}{i} + \frac{1}{l-i} \right)$$

the following sum can be expressed with the help of the harmonic numbers:

$$\sum_{i=1}^{l-1} \frac{1}{i \cdot (l-i)} = \frac{1}{l} \cdot \left(\sum_{i=1}^{l-1} \frac{1}{i} + \sum_{i=1}^{l-1} \frac{1}{l-i} \right) = \frac{2}{l} \cdot \sum_{i=1}^{l-1} \frac{1}{i} = \frac{2}{l} H_{l-1}. \tag{86}$$

The proof of the theorem is now split in three parts: First, we consider elections with only one distinct vote. Then, we determine a lower bound on the probability of single-peaked elections that consist of exactly two distinct votes. Third, we give a lower bound on the number of single-peaked elections that contain at least three distinct votes.

Let us now start with the first part of the lower bound. Clearly, an election in which all votes are identical is single-peaked. Let us denote the probability of this event by p_1 – in the following we fix m, n, a an omit them in the notation. According to the discrete probability distribution of the Pólya urn model (83), the probability of this event is:

$$p_1 = m! \cdot \frac{(1)_{n,a}}{(m!)_{n,a}} \geq \frac{m!(n-1)!a^{n-1}}{a^n \left(\frac{m!}{a} \right)_{n,1}} = \frac{m!(n-1)!}{a \left(\frac{m!}{a} \right)_{n,1}},$$

where we used Equations (84) and (85).

Next, we want to determine the probability that an election is sampled that is single-peaked and consists of exactly two distinct votes. Let us denote the probability of this event by p_2. From the third statement of Theorem 10.17 we know that there are exactly $m! \cdot \binom{2m-2}{m-1}$ elections with two voters and m candidates that are single-peaked. That is, if we pick a first vote V_1 at random, there are $\binom{2m-2}{m-1}$ votes V_2 that form a single-peaked election together with V_1. We thus have the following (again according to (83)):

$$p_2 = m! \binom{2m-2}{m-1} \cdot \sum_{i=1}^{n-1} \underbrace{\mathbb{P}(i \text{ votes equal to } V_1; n-i \text{ votes equal to } V_2)}_{=p'(i)}$$

According to (83) the probability $p'(i)$ is equal to

$$p'(i) = \binom{n}{i} \frac{(1)_{i,a} \cdot (1)_{n-i,a}}{(m!)_{n,a}}.$$

Using the bound in Equation (85) and the equality in Equation (84), $p'(i)$ can be bounded from below as follows:

$$\begin{aligned} p'(i) &\geq \binom{n}{i} \frac{(i-1)! \cdot a^{i-1} \cdot (n-i-1)! \cdot a^{n-i-1}}{(m!)_{n,a}}. \\ &= \frac{n! \cdot a^{n-2}}{a^n \left(\frac{m!}{a}\right)_{n,1} \cdot i(n-i)}. \end{aligned}$$

For p_2 we thus obtain

$$\begin{aligned} p_2 &\geq m! \cdot \binom{2m-2}{m-1} \frac{n!}{a^2 \left(\frac{m!}{a}\right)_{n,1}} \sum_{i=1}^{n-1} \frac{1}{i(n-i)} \\ &= \frac{m!(n-1)!}{\left(\frac{m!}{a}\right)_{n,1}} \cdot \binom{2m-2}{m-1} \frac{2}{a^2} H_{n-1}, \end{aligned}$$

where the transformation from the first to the second line is done with the identity in Equation (86).

Finally, for single-peaked elections that have more than two distinct votes, we only consider elections that contain a vote V and also its reverse $\bar{V}$. Let us denote the probability of this event by p_3. As in the proof of the bounds under the IC assumption this idea is based on the following fact about single-peakedness: If a vote V and its reverse vote $\bar{V}$ are both present within an election, then there are at most two axes with respect to which this election can be single-peaked, namely the axes V and $\bar{V}$. Thus, if a single-peaked election contains both the vote V and $\bar{V}$, all the other votes must be among the $2^{m-1} - 2$ votes that are also single-peaked with respect to the axis V (respectively $\bar{V}$). Let us denote this set of votes that are not equal to V or $\bar{V}$ and that are single-peaked with respect to the axis V by S_V.

The probability p_3 is then given as follows:

$$\begin{aligned} p_3 =& \frac{m!}{2} \sum_{l=2}^{n-1} \sum_{i=1}^{l-1} \mathbb{P}(i \text{ votes equal to } V, l-i \text{ votes equal to } \bar{V}, \text{ rest in } S_V) \\ =& \frac{m!}{2} \sum_{l=2}^{n-1} \sum_{i=1}^{l-1} \binom{n}{i} \binom{n-i}{l-i} \frac{(1)_{i,a} \cdot (1)_{l-i,a} \cdot (2^{m-1}-2)_{n-l,a}}{(m!)_{n,a}}, \end{aligned}$$

where $m!/2$ stands for the number of possible choices for the vote V, i is the number of times the vote V appears and $l - i$ is the number of times the vote $\bar{V}$ appears.

By using the bound in Equation (85) as well as the identity in Equation (86), we obtain the following:

$$
\begin{aligned}
p_3 &\geq \frac{m!}{2(m!)_{n,a}} \sum_{l=2}^{n-1} (2^{m-1}-2)_{n-l,a} \sum_{i=1}^{l-1} \binom{n}{i} \binom{n-i}{l-i} (i-1)!(l-i-1)! a^{l-2} \\
&= \frac{m!}{2(m!)_{n,a}} \sum_{l=2}^{n-1} (2^{m-1}-2)_{n-l,a} \cdot a^{l-2} \sum_{i=1}^{l-1} \frac{n!(n-i)!(i-1)!(l-i-1)!}{i!(n-i)!(l-i)!(n-l)!} \\
&= \frac{m!n!}{2(m!)_{n,a}} \sum_{l=2}^{n-1} (2^{m-1}-2)_{n-l,a} \cdot \frac{a^{l-2}}{(n-l)!} \sum_{i=1}^{l-1} \frac{1}{i(l-i)} \\
&= \frac{m!n!}{\left(\frac{m!}{a}\right)_{n,1}} \sum_{l=2}^{n-1} \frac{(2^{m-1}-2)_{n-l,a}}{a^{n+2-l}} \cdot \frac{1}{l(n-l)!} H_{l-1}.
\end{aligned}
$$

Since $p_1 + p_2 + p_3 \leq p_P(n,m,a)$, we obtain the desired lower bound. □

Corollary 10.20. *If $a = m!$ we have:*

$$
p_P(n,m,a) \geq \frac{1}{n} \cdot \left(1 + 2\frac{\ln(n-1)}{m!} \cdot \frac{(2m-2)!}{((m-1)!)^2}\right).
$$

10.4 MALLOWS MODEL

The Mallows model [123] assumes that there is a *reference vote* and votes are more likely to appear in an election if they are close to this reference vote. Closeness is measured by the Kendall tau rank distance, defined as follows.

Definition 10.21. *Given two votes V and W contained in an election $(C, \mathcal{P})$, the* Kendall tau rank distance *$\kappa(V,W)$ is a metric that counts the number of pairwise disagreements between V and W. To be more precise:*

$$
\kappa(V,W) = |\{\{c,c'\} \subseteq C : (V : cc' \wedge W : c'c) \vee (V : c'c \wedge W : cc')\}|.
$$

Note that $\kappa(V,W)$ is also the minimum number of transpositions, i.e., swaps, of adjacent elements, needed to transform V into W or vice versa. We can now define the Mallows model.

Definition 10.22. *Let C be a set of candidates with $|C| = m$ and let $T(C)$ be the set of all total orders on C. Given a* reference vote *V and a real number $\phi \in (0,1]$, the so-called* dispersion parameter*, the* Mallows model *is defined as follows. Every vote W of an (n,m)-election is determined independently from the others according to the following probability distribution:*

$$
\mathbb{P}_{V,\phi}(W) = \frac{1}{Z} \cdot \phi^{\kappa(V,W)}, \tag{87}
$$

where the normalization constant $Z = \sum_{W \in T(C)} \phi^{\kappa(V,W)}$ fulfils $Z = 1 \cdot (1+\phi) \cdot (1+\phi+\phi^2) \cdots (1+\ldots+\phi^{m-1})$.

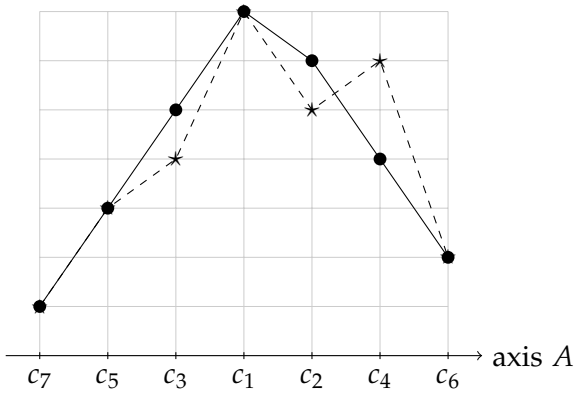


Figure 50: The axis A and the reference vote $V : c_1c_2c_3c_4c_5c_6c_7$, shown as a solid line. The dashed line represents the vote $W : c_1c_4c_2c_3c_6c_7$ and is not single-peaked with respect to A. The Kendall tau distance of V and W is 2. Note that all votes with a Kendall tau distance to V of 1 are single-peaked with respect to A.

Note that choosing $\phi = 1$ corresponds the Impartial Culture assumption and as $\phi \to \infty$ one obtains a distribution that concentrates all mass on V.

Theorem 10.23. *Let $p_M(n, m, \phi)$ denote the probability of an (n, m)-election being single-peaked if it is created according to the Mallows model with dispersion parameter ϕ. Then the following lower bound holds:*

$$p_M(n, m, \phi) \geq \left(\frac{1 + \phi \cdot (m-1) + \phi^2 \cdot (m-2)(m-3)/2}{Z} \right)^n .$$

Proof. Without loss of generality, we can assume that the reference vote is $V : c_1c_2 \dots c_m$. We define a total order A which will serve as axis as follows: $A : \dots c_6c_4c_2c_1c_3c_5 \dots$. Clearly, V is single-peaked with respect to A and it will turn out that "many" other votes that are close to V with respect to the Kendall tau distance are also single-peaked with respect to this axis. See Figure 50 for a representation of the axis A and the reference vote V for the case of seven candidates. In the following, we will write "W is SP" as a shortform of "the total order W on C is single-peaked with respect to axis A".

The idea of this proof is to bound the probability $p_M(n, m, \phi)$ from below as follows:

$$p_M(n, m, \phi) \geq \left(\mathbb{P}_{V,\phi}(W : W \text{ is SP }) \right)^n .$$

Moreover, we use the following bound:

$$\mathbb{P}_{V,\phi}(W : W \text{ is SP}) \geq \mathbb{P}_{V,\phi}(V) + \sum_{\substack{W \text{ is SP and} \\ \kappa(V,W)=1}} \mathbb{P}_{V,\phi}(W) + \sum_{\substack{W \text{ is SP and} \\ \kappa(V,W)=2}} \mathbb{P}_{V,\phi}(W)$$

First, it is clear that $\mathbb{P}(V) = 1/Z$.

Second, we need to compute the number of votes W that are single-peaked with respect to A and that fulfil $\kappa(V,W) = 1$. Votes W with $\kappa(V,W) = 1$ are votes in which the order of exactly one pair of candidates (c_i, c_{i+1}) has been changed in V. Since there are $(m-1)$ pairs of adjacent candidates in V, there are exactly $(m-1)$ votes W with $\kappa(V,W) = 1$. All these votes are single-peaked with respect to the axis A since:

- If c_1 and c_2 are interchanged, the position of the peak on the axis A is changed, but clearly no new peaks arise.
- If two other candidates c_i and c_{i+1} are interchanged, one of these two candidates lies to the left of the peak on A and the other one to the right of the peak. Thus, interchanging only these two candidates does not create a new peak either.

Therefore we have the following:

$$\sum_{\substack{W \text{ is SP and} \\ \kappa(V,W)=1}} \mathbb{P}_{V,\phi}(W) = (m-1) \cdot \frac{\phi}{Z}.$$

Third, we need to compute the number of votes W that are single-peaked with respect to A and that fulfil $\kappa(V,W) = 2$. Here we have a different situation: Not all votes that can be obtained by exactly two swaps of adjacent candidates in V are single-peaked with respect to A. For instance, first swapping the candidates (c_3, c_4) and then swapping (c_2, c_4) in V, does not lead to a vote that is single-peaked with respect to A. For this example, see the vote shown as a dashed line in Figure 50. The problem here is that the swapping of these two pairs changes the order of c_2 and c_4, two elements that both lie on the same side of the peak on A, and thus a valley is created by the elements c_1, c_2 and c_4. In general, a pair of swaps (c_i, c_{i+1}) and (c_j, c_{j+1}) is always allowed if the two pairs of candidates do not have any elements in common. Note that the order of the two (disjoint) swaps is of no importance and without loss of generality we can assume that $i+1 < j$.

Knowing this, we can bound the number of votes W with $\kappa(V,W) = 2$ that are single-peaked with respect to A as follows: If the first swap is (c_i, c_{i+1}) for some $i \in [1, m-3]$ and the second swap (c_j, c_{j+1}) is disjoint from the first one, j has to fulfil $j \in [i+2, m-1]$ and thus there are $(m-i-2)$ possibilities for (c_j, c_{j+1}). Summing over all possible i we obtain that there are at least

$$\sum_{i=1}^{m-3} m - i - 2 = \frac{(m-2)(m-3)}{2}$$

many votes W with $\kappa(V,W) = 2$ that are single-peaked with respect to A. Thus we have

$$\sum_{\substack{W \text{ is SP and} \\ \kappa(V,W)=2}} \mathbb{P}_{V,\phi}(W) \geq \frac{\phi^2}{Z} \cdot \frac{(m-2)(m-3)}{2}.$$

Putting the results for Kendall tau distance equal to 0, 1 and 2 together we obtain the desired lower bound. □

Corollary 10.24. *Assuming $\phi = \frac{1}{m}$, it holds that*

$$p_M(n,m,\phi) \geq \left(1.5\left(\frac{1-\frac{1}{m}}{1-\left(\frac{1}{m}\right)^m}\right)^{m-1}\right)^n > \left(1-\frac{1}{m}\right)^{(m-1)n}.$$

Proof. Inserting $\phi = \frac{1}{m}$ in the lower bound of Theorem 10.23 yields

$$p_M(n,m,\phi) \geq \left(\frac{1+\frac{m-1}{m}+\frac{(m-2)(m-3)}{2m^2}}{\left(1+\frac{1}{m}\right)\cdot\left(1+\frac{1}{m}+\frac{1}{m^2}\right)\cdots\left(1+\frac{1}{m}+\cdots+\frac{1}{m^{m-1}}\right)}\right)^n.$$

As can be checked easily, the numerator is larger than 1.5. Moreover the finite geometric sums in the denominator are all bounded from above by

$$\frac{1-\left(\frac{1}{m}\right)^m}{1-\frac{1}{m}}.$$

We thus obtain

$$p_M(n,m,\phi) \geq \left(1.5\left(\frac{1-\frac{1}{m}}{1-\left(\frac{1}{m}\right)^m}\right)^{m-1}\right)^n.$$

Replacing the numerator by 1 and every one of the finite geometric sums in the denominator by an infinite geometric row leads the second, much rougher lower bound. Note that this is simply a bound on the probability of sampling an (n,m)-election in which all votes are equal to the reference vote. □

Comparing this bound with the asymptotic tight bound for the probability of an (n,m)-election being single peaked under the IC assumption that is given in Theorem 10.16 and is of the form

$$\frac{m!}{2}\cdot\left(\frac{2^{m-1}}{m!}\right)^n,$$

we see that the likelihood of single-peakedness is far larger in the Mallows model with dispersion parameter $\phi = \frac{1}{m}$.

10.5 NUMERICAL EVALUATIONS

In this section we provide numerical evaluations of our probability results from the previous sections. In Table 51, we list exact probabilities that an (n,m)-election is single-peaked assuming the Impartial Culture assumptions for small values of m and bounds for these probabilities for a larger number of candidates. These probabilities illustrate how unlikely it is that an election drawn uniformly at random

(n,m)	exact probability
$(2,3)$	1
$(5,3)$	0.38
$(10,3)$	0.05
$(25,3)$	$1.19 \cdot 10^{-4}$
$(50,3)$	$4.70 \cdot 10^{-9}$
$(2,4)$	0.83
$(5,4)$	0.05
$(10,4)$	$2.03 \cdot 10^{-4}$
$(25,4)$	$1.42 \cdot 10^{-11}$
$(50,4)$	$1.67 \cdot 10^{-23}$

(n,m)	lower bound	upper bound
$(2,5)$	0.58 (via Thm 10.17)	
$(5,5)$	$1.6 \cdot 10^{-4}$	$2.6 \cdot 10^{-3}$
$(10,5)$	$2.2 \cdot 10^{-8}$	$1.1 \cdot 10^{-7}$
$(25,5)$	$5.0 \cdot 10^{-21}$	$8.0 \cdot 10^{-21}$
$(50,5)$	$9.7 \cdot 10^{-43}$	$1.1 \cdot 10^{-42}$
$(2,10)$	$1.3 \cdot 10^{-2}$ (via Thm 10.17)	
$(5,10)$	$7.6 \cdot 10^{-18}$	$1.1 \cdot 10^{-13}$
$(10,10)$	$1.9 \cdot 10^{-36}$	$5.7 \cdot 10^{-33}$
$(25,10)$	$2.3 \cdot 10^{-93}$	$1.0 \cdot 10^{-90}$
$(50,10)$	$4.6 \cdot 10^{-189}$	$5.5 \cdot 10^{-187}$

Figure 51: The likelihood that an (n,m)-election is single-peaked when drawn uniformly at random (Impartial Culture assumption). The probabilities in the left table are obtained via Theorem 10.17; those in the right table are obtained via Theorem 10.16 (except for $n = 2$).

(n,m)	$a = 10$	$a = m!/2$	$a = m!$
$(10,5)$	$1.6 \cdot 10^{-4}$	0.13	0.43
$(25,5)$	$8.4 \cdot 10^{-8}$	$3.0 \cdot 10^{-2}$	0.21
$(50,5)$	$1.5 \cdot 10^{-10}$	$9.1 \cdot 10^{-3}$	0.12
$(10,10)$	$3.6 \cdot 10^{-36}$	$2.0 \cdot 10^{-2}$	0.10
$(25,10)$	$2.3 \cdot 10^{-91}$	$3.6 \cdot 10^{-3}$	$4.4 \cdot 10^{-2}$
$(50,10)$	$2.6 \cdot 10^{-181}$	$9.7 \cdot 10^{-4}$	$2.2 \cdot 10^{-2}$

Figure 52: Lower bounds obtained from Theorem 10.19 on the likelihood that an (n,m)-election is single-peaked when sampled according to the Pólya urn model with homogeneity a

(n,m)	$\phi = 0.2$	$\phi = 0.1$	$\phi = 0.05$	$\phi = 0.01$
$(10,5)$	0.15	0.59	0.86	0.99
$(25,5)$	$8.7 \cdot 10^{-3}$	0.26	0.70	0.98
$(50,5)$	$7.6 \cdot 10^{-5}$	$7.2 \cdot 10^{-2}$	0.49	0.97
$(10,10)$	$2.7 \cdot 10^{-3}$	0.20	0.66	0.98
$(25,10)$	$3.7 \cdot 10^{-7}$	$1.9 \cdot 10^{-2}$	0.36	0.96
$(50,10)$	$1.4 \cdot 10^{-13}$	$3.7 \cdot 10^{-4}$	0.13	0.92

Figure 53: Lower bounds via Theorem 10.23 on the likelihood that an (n,m)-election is single-peaked when sampled according to the Mallows model with dispersion parameter ϕ

is single-peaked – even for small n and m. The probability that an (n, m)-election is single-peaked is considerably higher if we assume that votes are drawn subject to the Pólya urn and Mallows models. Table 52 shows lower bounds for the Pólya urn model and Table 53 shows lower bounds for the Mallows model. Note that Table 53 also includes results for $\phi = \frac{1}{m}$: for $m = 5$ these can be found in the column $\phi = 0.2$ and for $m = 10$ in the column $\phi = 0.1$. Our results indicate that the Mallows model with small ϕ is a very restrictive probability distribution, since it generates single-peaked preferences with rather high probability.

10.6 SUMMARY OF THE RESULTS

In this chapter we performed the first combinatorial study of the likelihood of single-peakedness in random elections. Our main results are the following:

- *Configuration definable restrictions:* Many domain restrictions can be characterized by forbidden configurations, in particular the single-peaked domain. We proved a close connection between configurations and permutations patterns. This novel connection allowed us to obtain a very general result (Theorem 10.13), showing that many domain restrictions characterized by forbidden configurations are very unlikely to appear in a random election chosen according to the Impartial Culture assumption.

- *Counting single-peaked elections:* We performed a detailed combinatorial analysis of the single-peaked domain by counting the number of single-peaked elections. We established an upper bound for single-peaked elections which asymptotically matches our lower bound result (Theorem 10.16). In addition, we showed exact enumeration results for elections with two voters or up to four candidates (Theorem 10.17). Our results rigorously show that the single-peaked restriction is highly unlikely to appear in random elections chosen according to the IC assumption. This holds even for elections with few votes and candidates (cf. Section 10.5).

- *Pólya urn model:* In contrast to the IC assumption, single-peaked elections are considerably more likely if elections are chosen according to the Pólya urn model. We provided a lower bound on the corresponding likelihood (Theorem 10.19) and showed that, if a sufficiently strong homogeneity is assumed, the probability of an election with n votes being single-peaked is larger than $1/n$ (Corollary 10.20).

- *Mallows model:* We encountered the most likely occurrence of single-peaked elections under Mallows model. As for the Pólya

urn model we established a lower bound result on the likelihood (Theorem 10.23). If the dispersion parameter ϕ is sufficiently small, we were able to show that single-peaked elections are likely to appear (Corollary 10.24 and Table 53).

11

FURTHER RESEARCH

Let us conclude this thesis by pointing out some directions for further research. These suggestions are based on the Future research-sections in the respective publications, see page xi.

EFFICIENT PERMUTATION PATTERN MATCHING

In Chapter 3 we performed a parameterized complexity analysis of the PPM problem and constructed an fpt-algorithm for the parameter $\mathsf{run}(\tau)$, the number of alternating runs in the text permutation τ (Theorem 3.1). This fpt-algorithm performs particularly well for PPM instances with few alternating runs.

An immediate consequence of this theorem is that any PPM instance can be reduced by polynomial time preprocessing to an equivalent instance – a kernel – of size depending solely on $\mathsf{run}(\tau)$. This raises the question whether even a polynomial-sized kernel exists. Such kernels, and in particular polynomial kernels, have been the focus of intensive research in algorithmics [95]. In [24], it has been shown that no such polynomial-sized kernel exists for the parameter $k = |\pi|$. Thus a positive result for the parameter $\mathsf{run}(\tau)$ would be of great interest.

Another research direction is the study of further parameters such as the permutation statistics listed in the Appendix A of [108].

At this point, several algorithms exist that solve PPM without imposing restrictions on π and τ. The algorithms by Guillemot and Marx [93], Albert et al. [3] and Ahal and Rabinovich [1] seem to be particularly well-suited for small patterns. In contrast, the runtime of our algorithm does not depend that critically on $|\pi|$. Thus, it may be expected that our algorithm is preferable for large patterns. However, only implementations and benchmarks could allow to settle this question and compare these algorithms.

GENERALIZED PERMUTATION PATTERN MATCHING

In Chapter 4 we analysed the PPM problem for several different types of permutation patterns. We strengthened the previously known NP-hardness result for PPM and proved NP-completeness for its generalizations. We also provided simple polynomial time algorithms for boxed mesh and consecutive PPM. Furthermore, we performed a parameterized complexity analysis, which shows that for vincular, bivincular and mesh PPM a fixed-parameter tractable algorithm is unlikely to exist.

In Section 4.2 we have listed several special cases for which PPM is in P. This list, however, is certainly far from being complete. In particular, polynomial time fragments of vincular, bivincular and mesh permutation pattern matching are not known at all.

In Section 4.3 we studied the influence of the length of the pattern on the complexity of the different types of PERMUTATION PATTERN MATCHING problems. For the cases where W[1]-hardness was shown as well as for PPM which is solvable by an FPT algorithm, it is of interest to find out whether other parameters of the input instances lead to fixed parameter tractability results. For future work any permutation statistic could be taken into account for a parameterized complexity analysis of all versions of PPM. An analysis of PPM with respect to several different parameters would then allow to draw a more detailed picture of the computational landscape of permutation pattern matching.

Another type of patterns was introduced by West [155]: *barred patterns*. A barred pattern $\overline{\pi}$ is a permutation where some letters are barred. One denotes by π the underlying permutation without any bars and by π' the permutation obtained by removing all barred letters. A permutation τ then avoids the barred pattern $\overline{\pi}$ if every occurrence of π' in τ is part of an occurrence of π in τ. For details and results on barred patterns, see Chapter 7 in [108]. Brändén and Claesson [36] showed that mesh patterns can easily be used to write any barred pattern with only one barred letter.

There exist further generalizations of mesh patterns: *marked mesh* and *decorated patterns*. These two types of patterns were introduced by Úlfarsson in [151] and [152], respectively. They allow a finer control over whether certain regions in a permutation may contain elements. Mesh patterns permit to specify regions in a permutation that may not contain elements in a matching. Marked mesh patterns allow more, namely to specify how many elements may be contained in certain regions (exactly, at most or at least). Decorated patterns go even further: they allow to detail in which order these elements may lie by describing forbidden patterns. These forbidden patterns may again be decorated patterns. Since both marked mesh and decorated patterns are generalizations of mesh patterns, the W[1]-hardness result given in Theorem 4.17 can be extended to these two types of patterns.

We considered patterns in *permutations*. However, the concept of pattern avoidance respectively containment can easily be extended to patterns in words over ordered alphabets (or permutations on multisets). In a matching of a word W into another word V, copies of the same letter have to be mapped to copies of some letter in the text. The topic of patterns in words has received quite some attention in the last years, see e.g. Heubach and Mansour's monograph *Combinatorics of compositions and words* [96]. The corresponding pattern matching problems have not yet been studied.

A PERMUTATION CLASS ENUMERATED BY THE CENTRAL BINOMIAL COEFFICIENTS

Chapter 5 was devoted to the study of a special permutation class, namely the class of permutations avoiding the patterns $2431, 4231$, 1432 and 4132 simultaneously. We could show that this class is enumerated by the central binomial coefficients.

The proof presented in Chapter 5 first constructs a bijection to a certain set of words and then counts these words by establishing a correspondence of certain segments of these words to lattice paths. However, it remains open to construct a direct bijection to some combinatorial object that is also counted by the central binomial coefficients. For instance, it would be nice to have a direct bijection to granny walks which are direct routes from the pint $(0, 0)$ to the point (n, n) in an integer grid.

The permutation class studied in this chapter is not the only one known to be enumerated by the central binomial coefficients. Indeed, in [5] the authors describe a general scheme to enumerate permutation classes. As an example, they give the permutation class avoiding the patterns $3124, 4123, 3142$ and 4132 that is also enumerated by the central binomial coefficients.

Preliminary studies show that there are in total nine permutation classes that are defined by four forbidden patterns of length four and that are enumerated by the central binomial coefficients. It would be desirable to develop a general framework that can be used to prove that all these classes are indeed counted by the central binomial coefficients and that no other such classes exist.

LOG-CONCAVITY, LONGEST INCREASING SUBSEQUENCES AND INVOLUTIONS

Chapter 6 was devoted to the sequence $\ell_{n,k}$ counting permutations of length n and with a longest increasing subsequence of length k. We conjectured that this sequence is log-concave for every fixed n and characterized several sets of permutations for which our conjecture holds. One main tool was to first prove our results for involutions and then to transfer them to arbitrary permutations. A next step in this line of research and towards proving our conjecture in general would be to find larger sets of permutations for which this method can be applied. Also, it would be interesting to find applications of this technique to other permutation statistics than the length of the longest increasing subsequence.

ASCENDING RUNS IN MAPPINGS

In Chapter 8 we generalized the patterns "ascents" and "ascending runs" from permutations to Cayley trees and to mappings. We provided exact enumeration formulæ that are linked to the Stirling numbers of the second kind and showed that the random variables counting ascents and ascending runs in random mappings is asymptotically normal distributed.

The combinatorial decomposition of Cayley trees and mappings employed there can also be used to study other label patterns. Indeed, our method can also be applied to count the number of local minima or maxima, thus generalizing the notion of valleys and peaks in permutations. However, the results contained in Okoth's thesis [129] are more general and thus the study of local minima and maxima was not included in Chapter 8. Moreover, preliminary studies show that our method can also be applied to what we call *up-ups*. For an n-mapping f, these are elements i in $[n]$ for which it holds that $i < f(i) < f^2(i)$. The notion of up-ups thus generalizes consecutive $\underline{123}$-patterns in permutations. Permutations avoiding this pattern have been studied in [62].

All label patterns mentioned so far were consecutive patterns, i.e., they concerned the images or preimages of nodes in the mappings graph. A probably more involved task would be to study non-consecutive label patterns in mappings. For instance, the non-consecutive pattern 21 corresponds to an inversion in the mapping. Of course, one could also study longer non-consecutive patterns in mappings or Cayley trees.

PARKING IN TREES AND MAPPINGS

In Chapter 9 we generalized the concept of parking functions to Cayley trees and mappings. As this is the first study of such parking functions, several possible directions for further research arise:

Given a tree T or a mapping f, we obtained general, but simple bounds for the number of tree and mapping parking functions $S(T, m)$ and $S(f, m)$. It is possible to obtain explicit formula for some simple classes of trees (or mappings), e.g., for "chain-like" trees with only few branchings. However, the following question remains open: Is it possible in general to give some "simple characterization" of the numbers $S(T, m)$ and $S(f, m)$?

With the approach presented, one can also study the total number of parking functions for other important tree families such as labelled binary trees or labelled ordered trees. We already performed some preliminary work for these tree families and according to our considerations, the results are considerably more involved than for labelled unordered trees and mappings. However, they do not seem to lead to

new phase change phenomena. Thus we did not comment on these studies here.

In contrast to the previous comment, the problem of determining the total number of parking functions seems to be interesting for so-called increasing (or decreasing) tree families (see, e.g., [60, 131]). That is, the labels along all leaf-to-root-paths form an increasing (or, decreasing) sequence. For so-called recursive trees (see, e.g., [83, 144]), i.e., unordered increasing trees, the approach presented could be applied, but the differential equations occurring do not seem to yield "tractable" solutions. For such tree families quantities such as the "sums of parking functions" as studied in [115] could be worthwhile treating as well.

As for ordinary parking functions one could analyse important quantities for tree and mapping parking functions. E.g., the so-called "total displacement" (which is of particular interest in problems related to hashing algorithms, see [77, 102]), i.e., the total driving distance of the drivers, or "individual displacements" (the driving distance of the k-th driver, see [103, 153]) seem to lead to interesting questions.

A refinement of parking functions can be obtained by studying what has been called "defective parking functions" in [45], or "overflow" in [92], i.e., pairs (T, s) or (f, s), such that exactly k drivers are unsuccessful. Preliminary studies indicate that the approach presented is suitable to obtain results in this direction as well.

Again, as for ordinary parking functions, one could consider enumeration problems for some restricted parking functions for trees (or mappings). E.g., we call (T, s), with T a size-n tree and $s \in [n]^n$ a parking function for T, an ordered tree parking function, if $\pi^{-1}(v) < \pi^{-1}(w)$, whenever $v \prec w$ (i.e., if the k-th driver parks at parking space w, all predecessors of w are already occupied by earlier drivers). Then it is easy to show that for any size-n tree T there are exactly $n!$ ordered tree parking functions.

Let us denote by X_n the random variable measuring the number of parking functions s with n drivers for a randomly chosen labelled unordered tree T of size n. Then, due to our previous results, we get the expected value of X_n via

$$\mathbb{E}(X_n) = \frac{F_n}{T_n} \sim \frac{\sqrt{2}\sqrt{\pi}2^{n+1}n^{n-\frac{1}{2}}}{e^n}.$$

However, with the approach presented here, it seems that we are not able to obtain higher moments or other results on the distribution of X_n.

The rather simple exact enumeration formulæ for perking functions for Cayley trees, connected mappings or arbitrary mappings for the case that the number of drivers coincides with the number of parking spaces also allow for other combinatorial interpretations.

For instance, recall that the number of such parking functions for connected mappings was given by (Lemma 9.11):

$$C_n = n!(n-1)! \sum_{j=0}^{n-1} \frac{(2n)^j}{j!}.$$

Compare this with the number of connected mappings $\tilde{C}_n$ that can be derived easily using the symbolic method and Lagrange inversion:

$$\tilde{C}_n = (n-1)! \cdot \sum_{k=0}^{n-1} \frac{n^k}{k!}.$$

In this formula, k corresponds to the number of elements not lying on the cycle. Thus $C_n/n!$ can also be interpreted as the number of connected n-mappings in which the non-cyclic elements are coloured either blue or red. It would thus be interesting to find a bijection between parking functions for connected mappings and pairs (π, C) where π is a permutation of length n and C is connected mapping with coloured non-cyclic elements. Similar interpretations can be given for the numbers F_n of tree-parking functions and M_n of mappings parking functions and could also allow for bijective correspondences.

As mentioned in the Introduction of this thesis on page 10, ordinary parking functions can bijectively be identified with allowable input-output pairs of priority queues. It would be interesting to find a generalization of this result to parking functions for trees and mappings, thus providing an interpretation of these in terms of forbidden patterns. Such a generalization might be possible by replacing the natural order on the numbers $1, \ldots, n$ by the order induced by the tree or mapping.

ON THE LIKELIHOOD OF SINGLE-PEAKED ELECTIONS

In Chapter 10 we studied the likelihood of single-peakedness in random elections. This was the first combinatorial study of single-peaked elections and we see various directions for future research.

Theorem 10.13 requires that the domain restriction avoids a $(2, k)$-configuration and thus is not applicable to domain restrictions such as the single-crossing restriction [39, 136]. It remains open whether this result can be extended to such domain restrictions as well and how the corresponding bound would look like. It would also be interesting to complement Theorem 10.13 with a corresponding lower bound result.

In general, the likelihood of other domain restrictions such as the single-crossing [136] or the 2D single-peaked restriction [17] has yet to be studied.

In Section 10.2 we obtained counting results for the number of single-peaked elections. In particular, it follows from Theorem 10.17

that under the IC assumption the likelihood that an $(n,3)$-election is single-peaked is $(2^{n-1}-1)/3^{n-1}$. The likelihood that such an election has a Condorcet winner is $15(n+3)2/[16(n+2)(n+4)]$ [85]. Note that the former probability is significantly smaller than the latter and, in particular, the former converges to 0 whereas the latter converges to $15/16$ for $n \to \infty$. Recently, the top monotonicity restriction has been proposed [18] which is a generalization of the single-peaked and single-crossing domain but still guarantees a Condorcet winner. It would be highly interesting to know the likelihood of top monotonicity restricted preferences and whether this probability is non-zero for $n \to \infty$.

Another direction is to consider other probability distributions such as the Plackett-Luce model [119, 134] or Mallows mixture models where more than one reference vote is considered [126]. One could also analyse the probability distribution that arises when assuming that all elections are single-peaked and that all elections of the same size are equally likely. This would allow to ask questions such as "How likely is it that a single-peaked election is also single-crossing?". Finally, a recent research direction is to consider elections that are nearly single-peaked, i.e., elections that have a small distance to being single-peaked according to some notion of distance [38, 52, 63, 65, 70]. The likelihood that elections are nearly single-peaked remains a worthwhile direction for future research.

$[n]$	the set of integers $\{1, \ldots, n\}$
$[m, n]$	for $m \leq n$, the interval of integers $\{m, m+1, \ldots, n\}$
$n!$	factorial of a nonnegative integer n
	$0! = 1$, $n! = n \cdot (n-1)! = n(n-1) \cdot 1$
$\Gamma(z)$	Gamma function: $\Gamma(z) = \int_o^\infty t^{z-1} e^{-t} dt$
	$\Gamma(n) = (n-1)$ for a nonnegative integer n
$\alpha^{\underline{k}}$	falling factorial of α for a real α and a nonnegative k:
	$\alpha^{\underline{0}} = 1$, $\alpha^{\underline{k}} = \alpha \cdot (\alpha-1)^{\underline{k-1}} = \alpha(\alpha-1) \cdot (\alpha-k+1)$
$(x)_{n,k}$	*Pochhammer k-symbol:* $(x)_{n,k} = \prod_{i=1}^{n} (x + (i-1) \cdot k)$
	where $x \in \mathbb{R}$ and $n, k \in \mathbb{N}$
$\binom{\alpha}{k}$	binomial coefficient for real α and nonnegative k:
	$\binom{\alpha}{k} = \frac{\alpha^{\underline{k}}}{k!}$, for $n \in \mathbb{N}$: $\binom{n}{k} = \frac{n!}{(n-k)!k!}$
$\left\{ {n \atop m} \right\}$	Stirling numbers of the second kind
$\mathbb{E}(X)$	expected value of X
$\mathbb{V}(X)$	variance of X
π, σ, τ	permutations
$\lvert T \rvert$	size of the combinatorial object T
iff	if and only if
w.r.t	with respect to
$\square$	end of proof
$\dashv$	end of example

BIBLIOGRAPHY

[1] S. Ahal and Y. Rabinovich. On complexity of the subpattern problem. *SIAM Journal on Discrete Mathematics*, 22(2):629–649, 2008.

[2] M. Aigner and G. M. Ziegler. *Proofs from the Book*. Springer, 2014.

[3] M. Albert, R. Aldred, M. Atkinson, and D. Holton. Algorithms for pattern involvement in permutations. In P. Eades and T. Takaoka, editors, *Algorithms and Computation*, volume 2223 of *Lecture Notes in Computer Science*, pages 355–367. Springer, 2001.

[4] M. H. Albert, R. E. L. Aldred, M. D. Atkinson, H. P. van Ditmarsch, B. D. Handley, C. C. Handley, and J. Opatrny. Longest subsequences in permutations. *Australasian Journal of Combinatorics*, 28:225–238, 2003.

[5] M. H. Albert, S. Linton, and N. Ruškuc. The insertion encoding of permutations. *The Electronic Journal of Combinatorics*, 12(1):31, 2005.

[6] D. Aldous and P. Diaconis. Longest increasing subsequences: from patience sorting to the Baik-Deift-Johansson theorem. *Bull. Amer. Math. Soc. (N.S.)*, 36(4):413–432, 1999.

[7] D. André. Étude sur les maxima, minima et séquences des permutations. *Annales scientifiques de l'École normale supérieure*, 3(1):121–135, 1884.

[8] J. Arney and E. Bender. Random mappings with constraints on coalescence and number of origins. *Pacific Journal of Mathematics*, 103(2):269–294, 1982.

[9] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[10] K. J. Arrow. A difficulty in the concept of social welfare. *The Journal of Political Economy*, 58(4):328–346, 1950.

[11] M. D. Atkinson. Permutations which are the union of an increasing and a decreasing subsequence. *The Electronic Journal of Combinatorics*, 5:87–100, 1998.

[12] S. Avgustinovich, S. Kitaev, and A. Valyuzhenich. Avoidance of boxed mesh patterns on permutations. *Discrete Applied Mathematics*, 161(1-2):43–51, 2013.

[13] E. Babson and E. Steingrímsson. Generalized permutation patterns and a classification of the mahonian statistics. *Séminaire Lotharingien de Combinatoire*, 44:117, 2000.

[14] J. Baik, P. Deift, and K. Johansson. On the distribution of the length of the longest increasing subsequence of random permutations. *J. Amer. Math. Soc.*, 12(4):1119 – 1178, 1999.

[15] M. A. Ballester and G. Haeringer. A characterization of the single-peaked domain. *Social Choice and Welfare*, 36(2):305–322, 2011.

[16] C. Banderier, P. Flajolet, G. Schaeffer, and M. Soria. Random maps, coalescing saddles, singularity analysis, and Airy phenomena. *Random Structures & Algorithms*, 19:194–246, 2001.

[17] S. Barberà, F. Gul, and E. Stacchetti. Generalized median voter schemes and committees. *Journal of Economic Theory*, 61(2):262–289, 1993.

[18] S. Barberà and B. Moreno. Top monotonicity: A common root for single peakedness, single crossing and the median voter result. *Games and Economic Behavior*, 73(2):345–359, Nov. 2011.

[19] S. Berg. Paradox of voting under an urn model: The effect of homogeneity. *Public Choice*, 47(2):377–387, 1985.

[20] N. Betzler, A. Slinko, and J. Uhlmann. On the computation of fully proportional representation. *Journal of Artificial Intelligence Research (JAIR)*, 47:475–519, 2013.

[21] P. Billingsley. *Probability and Measure*. John Wiley & Sons, second edition edition, 1984.

[22] D. Black. On the rationale of group decision making. *Journal of Political Economy*, 56(1):23–34, 1948.

[23] I. F. Blake and A. G. Konheim. Big buckets are (are not) better! *Journal of the Association for Computing Machinery*, 24:591–606, 1977.

[24] I. Bliznets, M. Cygan, P. Komosa, and L. Mach. Kernelization lower bound for permutation pattern matching. *Information Processing Letters*, 2015.

[25] M. Bóna. *Combinatorics of permutations*. Discrete Mathematics and Its Applications. Chapman & Hall/CRC, 2004.

[26] M. Bóna. A combinatorial proof of the log-concavity of a famous sequence counting permutations. *The Electronic Journal of Combinatorics*, 11, 2005.

[27] M. Bóna. *A walk through combinatorics: an introduction to enumeration and graph theory*. World Scientific, 2011.

[28] M. Bóna. A new record for 1324-avoiding permutations. *European Journal of Mathematics*, pages 1–9, 2014.

[29] M. Bóna, editor. *Handbook of Enumerative Combinatorics*. CRC Press – Chapman Hall, 2015.

[30] M. Bóna and M.-L. Bruner. Log-concavity, the Ulam distance and involutions. *arXiv preprint*, arXiv:1502.05438, 2015.

[31] P. Bose, J. F. Buss, and A. Lubiw. Pattern matching for permutations. *Information Processing Letters*, 65(5):277 – 283, 1998.

[32] M. Bousquet-Mélou. Four classes of pattern-avoiding permutations under one roof: Generating trees with two labels. *The Electronic Journal of Combinatorics*, 9(2), 2003.

[33] M. Bousquet-Mélou, A. Claesson, M. Dukes, and S. Kitaev. (2+2)-free posets, ascent sequences and pattern avoiding permutations. *Journal of Combinatorial Theory Series A*, 117(7):884–909, 2010.

[34] M. Bouvel and D. Rossin. The longest common pattern problem for two permutations. *Pure Mathematics and Applications*, 17(1-2):55–69, 2006.

[35] M. Bouvel, D. Rossin, and S. Vialette. Longest common separable pattern among permutations. In B. Ma and K. Zhang, editors, *Combinatorial Pattern Matching*, Lecture Notes in Computer Science, pages 316–327. Springer, 2007.

[36] P. Brändén and A. Claesson. Mesh patterns and the expansion of permutation statistics as sums of permutation patterns. *The Electronic Journal of Combinatorics*, 18(2):P5, 2011.

[37] F. Brandt, M. Brill, E. Hemaspaandra, and L. A. Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010)*, pages 715–722, 2010.

[38] R. Bredereck, J. Chen, and G. J. Woeginger. Are there any nicely structured preference profiles nearby? In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, pages 62–68, 2013.

[39] R. Bredereck, J. Chen, and G. J. Woeginger. A characterization of the single-crossing domain. *Social Choice and Welfare*, 41(4):989–998, 2013.

[40] F. Brenti. Log-concave and unimodal sequences in algebra, combinatorics, and geometry: an update. *Contemporary Mathematics*, 178:71–89, 1994.

[41] M.-L. Bruner and M. Lackner. A fast algorithm for permutation pattern matching based on alternating runs. In F. V. Fomin and P. Kaski, editors, *SWAT*, volume 7357 of *Lecture Notes in Computer Science*, pages 261–270. Springer, 2012.

[42] M.-L. Bruner and M. Lackner. The computational landscape of permutation patterns. *Pure Mathematics and Applications*, 24(2)(2):83–101, 2013.

[43] M.-L. Bruner and M. Lackner. The likelihood of structure in preference profiles. In *Proceedings of the 8th Multidisciplinary Workshop on Advances in Preference Handling (MPref 2014)*, 2014.

[44] M.-L. Bruner and A. Panholzer. Parking functions for trees and mappings. *arXiv preprint*, arXiv:1504.04972, 2015.

[45] P. J. Cameron, D. Johannsen, T. Prellberg, and P. Schweitzer. Counting defective parking functions. *The Electronic Journal of Combinatorics*, 15:R92, 2008.

[46] L. Carlitz, D. Kurtz, R. Scoville, and O. Stackelberg. Asymptotic properties of eulerian numbers. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 23(1):47–54, 1972.

[47] A. Cayley. A theorem on trees. *Quart. J. Math*, 23(376-378):69, 1889.

[48] M.-S. Chang and F.-H. Wang. Efficient algorithms for the maximum weight clique and maximum weight independent set problems on permutation graphs. *Information Processing Letters*, 43(6):293–295, 1992.

[49] L. Clark. Ascents and descents in random trees. *Journal of Discrete Mathematical Sciences and Cryptography*, 11(4):483–492, 2008.

[50] C. H. Coombs. *A Theory of Data*. John Wiley & Sons, 1964.

[51] R. M. Corless, G. H. Gonnet, D. E. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5(1):329–359, 1996.

[52] D. Cornaz, L. Galand, and O. Spanjaard. Bounded single-peaked width and proportional representation. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, volume 242, pages 270–275. IOS Press, 2012.

[53] B. Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

[54] D. E. Critchlow, M. A. Fligner, and J. S. Verducci. Probability models on rankings. *Journal of Mathematical Psychology*, 35(3):294–318, Sept. 1991.

[55] M. Crochemore, C. S. Iliopoulos, T. Kociumaka, M. Kubica, A. Langiu, S. P. Pissis, J. Radoszewski, W. Rytter, and T. Walen. Order-preserving incomplete suffix trees and order-preserving indexes. In O. Kurland, M. Lewenstein, and E. Porat, editors, *String Processing and Information Retrieval*, volume 8214 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2013.

[56] F. N. David and D. E. Barton. *Combinatorial chance*. Griffin, 1962.

[57] N. de Bruijn. *Asymptotic methods in analysis*. North-Holland Publishing Co., 1958.

[58] R. Díaz and E. Pariguan. On hypergeometric functions and pochhammer k-symbol. *Divulgaciones Matemáticas*, 15(2):179–192, 2007.

[59] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[60] M. Drmota. *Random trees*. Springer, 2009.

[61] M. Drmota and M. Soria. Images and preimages in random mappings. *SIAM Journal on Discrete Mathematics*, 10(2):246–269, 1997.

[62] S. Elizalde and M. Noy. Consecutive patterns in permutations. *Advances in Applied Mathematics*, 30(1):110–125, 2003.

[63] E. Elkind, P. Faliszewski, and A. M. Slinko. Clone structures in voters' preferences. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC 2012)*, pages 496–513. ACM, 2012.

[64] E. Elkind and M. Lackner. On detecting nearly structured preference profiles. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI 2014)*, 2014.

[65] G. Erdélyi, M. Lackner, and A. Pfandler. Computational aspects of nearly single-peaked electorates. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI 2013)*. AAAI Press, 2013.

[66] P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[67] P. Erdös and G. Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

[68] B. Escoffier, J. Lang, and M. Öztürk. Single-peaked consistency and its complexity. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, volume 178 of *FAIA*, pages 366–370. IOS Press, 2008.

[69] L. C. Evans. Partial differential equations. *American Mathematical Society*, 2010.

[70] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. The complexity of manipulative attacks in nearly single-peaked electorates. *Artificial Intelligence*, 207(0):69 – 99, 2014.

[71] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. The shield that never was: Societies with single-peaked preferences are more open to manipulation and control. *Information and Computation*, 209(2):89 – 107, 2011.

[72] P. Favardin, D. Lepelley, and J. Serais. Borda rule, copeland method and strategic manipulation. *Review of Economic Design*, 7(2):213–228, 2002.

[73] G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette. *Combinatorics of Genome Rearrangments*. MIT Press, 2009.

[74] P. Flajolet, D. E. Knuth, and B. Pittel. The first cycles in an evolving graph. *Discrete Mathematics*, 75:167–215, 1989.

[75] P. Flajolet and A. M. Odlyzko. Random mapping statistics. In *Advances in cryptology-EUROCRYPT'89*, pages 329–354. Springer, 1990.

[76] P. Flajolet and A. M. Odlyzko. Singularity analysis of generating functions. *SIAM Journal on Discrete Mathematics*, 3:216–240, 1990.

[77] P. Flajolet, P. Poblete, and A. Viola. On the analysis of linear probing hashing. *Algorithmica*, 22:490–515, 1998.

[78] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. Cambridge University press, 2009.

[79] J. Flum and M. Grohe. Model-checking problems as a basis for parameterized intractability. *Logical Methods in Computer Science*, 1(1), 2005.

[80] J. Flum and M. Grohe. *Parameterized complexity theory*. Springer, 2006.

[81] F. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2010.

[82] E. Friedgut, G. Kalai, N. Keller, and N. Nisan. A quantitative version of the gibbard-satterthwaite theorem for three alternatives. *SIAM Journal of Computation*, 40:934–952, 2011.

[83] M. Fuchs, H.-K. Hwang, and R. Neininger. Profiles of random trees: limit theorems for random recursive trees and binary search trees. *Algorithmica*, 46:367–407, 2006.

[84] P. Gawrychowski and P. Uznanski. Order-preserving pattern matching with k mismatches. *arXiv preprint*, arXiv:1309.6453, 2013.

[85] W. V. Gehrlein. The expected probability of Condorcet's paradox. *Economics Letters*, 7(1):33–37, 1981.

[86] W. V. Gehrlein. *Condorcet's Paradox*. Springer, 2006.

[87] W. V. Gehrlein, D. Lepelley, and I. Moyouwou. Voters' preference diversity, concepts of agreement and condorcet's paradox. *Quality & Quantity*, pages 1–24, 2014.

[88] W. V. Gehrlein, I. Moyouwou, and D. Lepelley. The impact of voters' preference diversity on the probability of some electoral outcomes. *Mathematical Social Sciences*, 66(3):352–365, 2013.

[89] A. Gibbard. Manipulation of voting schemes. *Econometrica*, 41(4):587–601, 1973.

[90] J. D. Gilbey and L. H. Kalikow. Parking functions, valet functions and priority queues. *Discrete mathematics*, 197:351–373, 1999.

[91] O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[92] G. H. Gonnet and J. I. Munro. The analysis of linear probing sort by the use of a new mathematical transform. *Journal of Algorithms*, 5:451–470, 1984.

[93] S. Guillemot and D. Marx. Finding small patterns in permutations in linear time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, SODA'14, pages 82–101. SIAM, 2014.

[94] S. Guillemot and S. Vialette. Pattern matching for 321-avoiding permutations. In Y. Dong, D.-Z. Du, and O. Ibarra, editors, *Algorithms and Computation*, volume 5878 of *Lecture Notes in Computer Science*, pages 1064–1073. Springer, 2009.

[95] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, Mar. 2007.

[96] S. Heubach and T. Mansour. *Combinatorics of compositions and words*. Chapman & Hall/CRC, 2009.

[97] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[98] H.-K. Hwang. On convergence rates in the central limit theorems for combinatorial structures. *European Journal of Combinatorics*, 19(3):329–343, 1998.

[99] L. Ibarra. Finding pattern matchings for permutations. *Information Processing Letters*, 61(6):293–295, 1997.

[100] K.-i. Inada. The simple majority decision rule. *Econometrica*, 37(3):490–506, 1969.

[101] M. Isaksson, G. Kindler, and E. Mossel. The geometry of manipulation—a quantitative proof of the Gibbard-Satterthwaite theorem. *Combinatorica*, 32(2):221–250, 2012.

[102] S. Janson. Asymptotic distribution for the cost of linear probing hashing. *Random Structures & Algorithms*, 19:438–471, 2001.

[103] S. Janson. Individual displacements for linear probing hashing with different insertion policies. *ACM Transactions on Algorithms*, 1:177–213, 2005.

[104] S. Janson, D. E. Knuth, T. Łuczak, and B. Pittel. The birth of the giant component. *Random Structures & Algorithms*, 4(3):233–358, 1993.

[105] N. L. Johnson and S. Kotz. *Urn models and their application*. Wiley, 1977.

[106] R. M. Karp. Reducibility among combinatorial problems. In R. Miller, J. Thatcher, and J. Bohlinger, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Springer, 1972.

[107] J. Kim, P. Eades, R. Fleischer, S.-H. Hong, C. S. Iliopoulos, K. Park, S. J. Puglisi, and T. Tokuyama. Order preserving matching. *arxiv preprint*, arxiv:1302.4064, 2013.

[108] S. Kitaev. *Patterns in Permutations and Words*. Springer, 2011.

[109] A. Klenke. *Probability Theory: A Comprehensive Course*. Springer, 2008.

[110] V. Knoblauch. Recognizing one-dimensional Euclidean preference profiles. *Journal of Mathematical Economics*, 46(1):1 – 5, 2010.

[111] D. E. Knuth. *The Art of Computer Programming, Volume I: Fundamental Algorithms*. Addison-Wesley, 1968.

[112] V. F. Kolchin. *Random mappings*. Optimization Software, 1986.

[113] A. G. Konheim and B. Weiss. An occupancy discipline and applications. *SIAM Journal on Applied Mathematics*, 14:1266–1274, 1966.

[114] M. Kubica, T. Kulczyński, J. Radoszewski, W. Rytter, and T. Waleń. A linear time algorithm for consecutive permutation pattern matching. *Information Processing Letters*, 113(12):430 – 433, 2013.

[115] J. P. Kung and C. Yan. Exact formulas for moments of sums of classical parking functions. *Advances in Applied Mathematics*, 31(1):215–241, 2003.

[116] D. Lepelley and F. Valognes. Voting rules, manipulability and social homogeneity. *Public Choice*, 116(1-2):165–184, 2003.

[117] H. Levene and J. Wolfowitz. The covariance matrix of runs up and down. *The Annals of Mathematical Statistics*, 15(1):58–69, 1944.

[118] G. Louchard. Kac's formula, levy's local time and brownian excursion. *Journal of Applied Probability*, 21(3):pp. 479–499, 1984.

[119] R. D. Luce. *Individual choice behavior*. Wiley, 1959.

[120] P. A. MacMahon. *Combinatory Analysis*. Cambridge University Press, 1915–16.

[121] H. Mahmoud. *Pólya urn models*. Texts in Statistical Science. Chapman & Hall/CRC, 2008.

[122] E. Mäkinen. On the longest upsequence problem for permutations. *International Journal of Computer Mathematics*, 77(1):45–53, 2001.

[123] C. L. Mallows. Non-null ranking models. I. *Biometrika*, 44(1/2):114–130, 1957.

[124] A. Marcus and G. Tardos. Excluded permutation matrices and the stanley–wilf conjecture. *Journal of Combinatorial Theory, Series A*, 107(1):153–160, 2004.

[125] S. G. Mohanty. *Lattice path counting and applications*. Academic Press, 1979.

[126] T. B. Murphy and D. Martin. Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41(3–4):645 – 655, 2003.

[127] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics And Its Applications. Oxford University Press, 2006.

[128] The On-Line Encyclopedia of Integer Sequences. Published electronically at http://oeis.org, 2015.

[129] I. O. Okoth. *Combinatorics of oriented trees and tree-like structures*. PhD thesis, University of Stellenbosch, South Africa, 2015.

[130] A. Panholzer. Alternating mapping functions. *Journal of Combinatorial Theory, Series A*, 120(7):1835–1850, 2013.

[131] A. Panholzer and H. Prodinger. Level of nodes in increasing trees revisited. *Random Structures & Algorithms*, 31:203–226, 2007.

[132] C. H. Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[133] R. Pemantle and M. C. Wilson. *Analytic Combinatorics in Several Variables*. Cambridge University Press, 2013.

[134] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(2):193–202, 1975.

[135] A. Postnikov and B. Shapiro. Trees, parking functions, syzygies, and deformations of monomial ideals. *Transactions of the American Mathematical Society*, 356:3109–3142, 2004.

[136] K. W. Roberts. Voting over income tax schedules. *Journal of Public Economics*, 8(3):329–340, 1977.

[137] M. Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, 1975.

[138] S. Saxena and V. Yugandhar. Parallel algorithms for separable permutations. *Discrete Applied Mathematics*, 146(3):343–364, 2005.

[139] C. Schensted. Longest increasing and decreasing subsequences. *Classic Papers in Combinatorics*, pages 299–311, 1987.

[140] R. Simion and F. W. Schmidt. Restricted permutations. *European Journal of Combinatorics*, 6:383–406, 1985.

[141] A. Slinko. On asymptotic strategy-proofness of classical social choice rules. *Theory and Decision*, 52(4):389–398, June 2002.

[142] A. Slinko. On asymptotic strategy-proofness of the plurality and the run-off rules. *Social Choice and Welfare*, 19(2):313–324, 2002.

[143] A. Slinko. How the size of a coalition affects its chances to influence an election. *Social Choice and Welfare*, 26(1):143–153, Dec. 2005.

[144] R. T. Smythe and H. M. Mahmoud. A survey of recursive trees. *Theory of Probability and Mathematical Statistics*, 51:1–27, 1996.

[145] Z. E. Stankova. Forbidden subsequences. *Discrete Mathematics*, 132(1):291–316, 1994.

[146] R. Stanley. *Enumerative combinatorics*, volume I & II. Cambridge University press, 1997 & 1999.

[147] R. P. Stanley. Log-concave and unimodal sequences in algebra, combinatorics, and geometrya. *Annals of the New York Academy of Sciences*, 576(1):500–535, 1989.

[148] R. P. Stanley and J. Pitman. A polytope related to empirical distributions, plane trees, parking functions, and the associahedron. *Discrete & Computational Geometry*, 27:603–634, 2002.

[149] E. Steingrímsson. Generalized permutation patterns - a short survey. *Permutation Patterns*, 376:137–152, 2010.

[150] S. Ulam. Some ideas and prospects in biomathematics. *Annual Review of Biophysics and Bioengineering*, 1(1):277–292, 1972.

[151] H. Úlfarsson. A unification of permutation patterns related to Schubert varieties. In *Formal power series and algebraic combinatorics (FPSAC 2010)*, DMTCS Proceedings, pages 1057–1068, 2010.

[152] H. Úlfarsson. Describing west-3-stack-sortable permutations with permutation patterns. *Séminaire Lotharingien de Combinatoire*, 67:20, 2012.

[153] A. Viola. Exact distribution of individual displacements in linear probing hashing. *ACM Transactions on Algorithms*, 1:214–242, 2005.

[154] T. Walsh. Uncertainty in preference elicitation and aggregation. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI 2007)*, pages 3–8. AAAI Press, 2007.

[155] J. West. *Permutations with forbidden subsequences, and, stack-sortable permutations*. PhD thesis, Massachusetts Institute of Technology, 1990.

[156] J. West. Generating trees and the Catalan and Schröder numbers. *Discrete Mathematics*, 146(1-3):247–262, 1995.

[157] J. West. Generating trees and forbidden subsequences. *Discrete Mathematics*, 157(1):363–374, 1996.

[158] H. S. Wilf. *generatingfunctionology*. A K Peters/CRC Press, third edition, 2006.

[159] C. H. Yan. Generalized parking functions, tree inversions, and multicolored graphs. *Advances in Applied Mathematics*, 27:641–670, 2001.

CURRICULUM VITÆ – MARIE-LOUISE BRUNER

PERSONAL INFORMATION

Address	Josefstädterstraße 43-35/2/10, 1080 Vienna, Austria
Email	marie-louise.bruner@tuwien.ac.at
Web	http://dmg.tuwien.ac.at/mbruner/
Born	July 27th 1987 in Vienna
Citizenship	Austrian

EDUCATION

Oct 2011–Jun 2015	**PHD studies,** *Vienna University of Technology*, under the supervision of Alois Panholzer
Jun 2012	**"Dr. Maria Schaumayer"-prize,** *awarded for the diploma thesis*
Oct 2005–Jun 2011	**Studies of "Technische Mathematik",** *Vienna University of Technology*
Jun 2011	**Master's degree (Dipl.-Ing.),** *Vienna University of Technology,* Master thesis "Restricted Permutations on Multisets" supervised by Alois Panholzer, passed with distinction
Aug 2008–Dec 2008	**Erasmus exchange term,** *Royal Institute of Technology in Stockholm (KTH), Sweden*
Aug 1993–Jun 2005	**Lycée Français de Vienne,** *Ecole primaire, collège, lycée,* Section S (scientifique/sciences)
Jun 2005	**French Baccalauréat,** *Lycée Français de Vienne,* passed with distinction
Jun 2005	**Austrian Matura,** *Lycée Français de Vienne,* passed with distinction

ACADEMIC CAREER

All positions so far at the Institute for Discrete Mathematics and Geometry at the Vienna University of Technology.

Feb 2013–present **Research fellow,** *FWF-project P25337-N23 "Restricted labelled combinatorial objects: new enumerative, statistical, asymptotic and complexity theoretical aspects"*, Project director: Prof. Alois Panholzer

Mar 2012–Jul 2013 **Graduate teaching and research assistant,** Under the responsibility of Prof. Michael Drmota and Prof. Monika Ludwig

Jun 2011–Feb 2012 **Research fellow,** *FWF-project S9608 "Combinatoric analysis of data structures and tree-like structures"*, part of the national research network (NFN) "Analytic Combinatorics and Probabilistic Number Theory", Project director: Prof. Alois Panholzer

Oct 2010–Jun 2011 **Undergraduate teaching assistant,** Under the responsibility of Prof. Monika Ludwig

Mar 2009–Jun 2010 **Tutor,** *Exercises for mechanical engineering and civil engineering students*, Under the responsibility of Prof. Peter M. Gruber

RESEARCH AREAS TO DATE

- Combinatorics: Enumeration of labelled combinatorial objects - exact and asymptotic results; generating functions and bijective proofs
- Complexity analysis of combinatorial problems: classical and parametrized complexity theory; design of (FPT)-algorithms
- Computational Social Choice: combinatorial and complexity theoretic properties of domain restrictions

SCIENTIFIC ARTICLES

2015 *Parking functions for trees and mappings*, with Alois Panholzer, submitted.

2015 *Log-concavity, the Ulam distance and involutions*, with Miklós Bóna, submitted.

2015 *A Fast Algorithm for Permutation Pattern Matching Based on Alternating Runs*, with Martin Lackner, accepted for publication in Algorithmica

2014 *The Likelihood of Structure in Preference Profiles,* with Martin Lackner, in Proceedings of the 8th Multidisciplinary Workshop on Advances in Preference Handling (MPref 2014)

2013 *On restricted permutations on regular multisets,* in Permutation Patterns 2012 Proceedings, special issue of Pure Mathematics and Applications

2013 *The computational landscape of permutation patterns,* with Martin Lackner, in Permutation Patterns 2012 Proceedings, special issue of Pure Mathematics and Applications.

2012 *From Peaks to Valleys, Running Up and Down: Fast Permutation Pattern Matching,* with Martin Lackner, Tiny Transactions on Computer Science

2012 *A Fast Algorithm for Permutation Pattern Matching Based on Alternating Runs,* with Martin Lackner, Algorithm Theory – SWAT 2012

TALKS

2015 *The likelihood of single-peaked preference profiles: a combinatorial approach*
Combinatorics Seminar, University of Florida, Gainesville, USA.

2015 *A combinatorial approach to structure in preference profiles*
Seminar of the "Arbeitsgemeinschaft Diskrete Mathematik", Vienna University of Technology, Austria.

2014 *The Likelihood of Structure in Preference Profiles*
Workshop on Challenges in Algorithmic Social Choice, Bad Belzig, Germany.

2014 *Permutation Pattern Matching: From separable permutations to fixed-parameter algorithms*
Combinatorics Seminar, University of Florida, Gainesville, USA.

2013 *The computational landscape of permutation patterns*
11th Permutation Patterns conference, University Paris Diderot, Paris, France.

2013 *Parking in trees*
4th biennial Canadian Discrete and Algorithmic Mathematics Conference, CanaDAM 2013, Memorial University of Newfoundland, St. John's, Canada.

2013 *Label patterns in mappings*
24th International Meeting on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms, AofA 2013, Cala Galdana, Menorca, Spain.

2013 *Parking arborescent*
Journées Aléa 2013, CIRM, Marseille, France.

2012 *A Fast Algorithm for Permutation Pattern Matching Based on Alternating Runs*
Seminar of the "Arbeitsgemeinschaft Diskrete Mathematik", Vienna University of Technology, Austria.

2012 *A Fast Algorithm for Permutation Pattern Matching Based on Alternating Runs*
10th Permutation Patterns conference, University of Strathclyde, Glasgow, Scotland.

2011 *Counting multiset-permutations avoiding the pattern 122 and another pattern of length three*
30th Colloquium on Combinatorics, Magdeburg, Germany.

2011 *Enumerative formulae for multiset-permutations avoiding the pattern* 122 *and another pattern of length three*
Joint Mathematical Conference of the Austrian, Catalan, Czech, Slovak and Slovenian Mathematical Societies, Krems, Austria.

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/