DISSERTATION

INSERTION ALGORITHMS FOR MODULI SPACES OF CURVES

Submitted by

Andrew Reimer-Berg

Department of Mathematics

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2026

Doctoral Committee:

Advisor: Maria Gillespie

Sudipto Ghosh
Alexander Hulpke
Mark Shoemaker

ABSTRACT

INSERTION ALGORITHMS FOR MODULI SPACES OF CURVES

Moduli spaces of curves are geometric spaces whose points parameterize some class of geometric object. Two such examples are the Grassmannian $\mathrm{Gr}(k, n)$ of k-planes through the origin and the space $\overline{M}_{0,n+3}$ of stable complex curves with $n + 3$ marked points. Schubert calculus has been hugely successful in using combinatorics to understand the geometry and intersection theory of $\mathrm{Gr}(k, n)$. A more recent trend in enumerative geometry has been to extend the techniques of Schubert calculus to create a similar story for $\overline{M}_{0,n+3}$, using new combinatorial techniques to understand the intersection theory of $\overline{M}_{0,n+3}$. This thesis contains two main parts, that each relate respectively to this story for the two aforementioned spaces.

For our first main result, we give a combinatorial proof of a recent geometric result of Farkas and Lian on linear series on curves with prescribed incidence conditions. The result states that the expected number of degree-d morphisms from a general genus g, n-marked curve C to $\mathbb{P}^r$, sending the marked points on C to specified general points in $\mathbb{P}^r$, is equal to $(r + 1)^g$ for sufficiently large d. This computation may be rephrased as an intersection problem on Grassmannians, which has a natural combinatorial interpretation in terms of Young tableaux by the classical Littlewood-Richardson rule. We give a bijection, generalizing the well-known RSK correspondence, between the tableaux in question and the $(r + 1)$-ary sequences of length g, and we explore our bijection's combinatorial properties.

In our second main result, we resolve a question of Gillespie, Griffin, and Levinson that asks for a combinatorial bijection between two classes of trivalent trees, *tournament trees* and *slide trees*, that both naturally arise in the intersection theory of the moduli space $\overline{M}_{0,n+3}$ of stable genus zero curves with $n + 3$ marked points. Each set of trees enumerates the same intersection product of certain pullbacks of ψ classes under forgetting maps.

We give an explicit combinatorial bijection between these two sets of trees using an insertion algorithm that mimics the RSK algorithm for Young tableaux, providing another step towards the broader goal of developing a combinatorial understanding of the theory of moduli spaces of curves that mirrors the success of Schubert calculus. We also classify the words that appear on the slide trees of caterpillar shape via pattern avoidance conditions.

ACKNOWLEDGEMENTS

I would first like to thank my advisor Maria Gillespie for all of her mentorship, feedback, and support that she has given me over the past six years. I would not be the mathematician I am today without her guidance.

I would like to thank my family for all of their love, support, and encouragement. To all of the friends I've made at various stages of life, thank you for reminding me to slow down and find joy in life.

I also would like to thank the many people whose conversations helped shape the mathematics contained here: Renzo Cavalieri, Jake Levinson, Sean Griffin, Erin Dawson, Carl Lian, Vance Blankers, Matt Larson, Rob Silversmith, and more.

Finally, I would like to thank everyone serving on my committee: Maria Gillespie, Sudipto Ghosh, Alexander Hulpke, and Mark Shoemaker, for taking the time to hear me share what I know.

TABLE OF CONTENTS

Chapter 1

Introduction

This thesis is part of a large and growing body of study in algebraic combinatorics in which geometric problems are solved by way of combinatorial algorithms. Our work is part of a newer trend in intersection theory, in which the aim is to replicate the success of algebraic combinatorics in Schubert calculus (concerned with moduli spaces for linear objects) in the setting of curves.

Our first result concerns maps from marked curves into projective space, which can be reduced to a question involving Schubert calculus. In this case, we utilize the existing theories and combinatorial rules to answer the question posed using combinatorial methods. Our second set of results concerns maps from marked curves into products of projective spaces, in which the questions can be reinterpreted in terms of counting certain classes of trees. Such a combinatorializing of moduli spaces of curves was first suggested by Mumford [21], and much progress has been made in recent years.

More specifically, the goal is to develop combinatorial rules, analogous to the Littlewood–Richardson rule, for the intersection theory of $\overline{M}_{g,n}$ and related spaces. We also draw connections from the new combinatorics developed in pursuit of the above questions with the existing combinatorial theories, such as insertion algorithms, pattern avoidance, and graph theory.

This is a promising angle of study because these geometric spaces, while geometrically rich and interesting in their own right, also possess a lot of combinatorial structure. The combinatorial versions of these questions are then more tractable, yielding themselves more easily to computations. This combinatorialization process frequently creates novel connections between combinatorics and geometry, enriching the body of knowledge of both fields. By combining this with the geometric perspective, we obtain a more complete understanding of the spaces in question.

1.1 Schubert calculus and the Grassmannian

Schubert Calculus is the study of intersection questions of linear spaces. For example, the question "How many lines intersect 4 generic lines in 3-dimensional space?" has the answer 2. The tools of Schubert Calculus allow us to interpret such questions in the algebraic language of intersection products in the cohomology ring of the Grassmannian, and then gives us the tools to answer these questions combinatorially. The **Grassmannian** $\mathrm{Gr}(k, n)$ is the space of all k-dimensional subspaces of the n-dimensional vector space $\mathbb{C}^n$. This is an example of a *moduli space*, a space whose points parameterize some class of geometric objects. In this case, the points of $\mathrm{Gr}(k, n)$ parameterize k-planes in $\mathbb{C}^n$.

Many questions in intersection theory have answers that are either zero (How many lines pass through three points in general position?) or infinity (How many lines pass through a given point and intersect a given line?). It is the remaining cases, where the answer is nonzero and finite, where we will focus our attention.

The development of *Schubert calculus* has yielded many connections between algebraic geometry and algebraic combinatorics. Schubert calculus is the study of the intersections of linear spaces, and is part of a larger branch of algebraic geometry known as intersection theory.

The combinatorializations of these problems given by Schubert Calculus are in terms of *Young tableaux*. A **standard Young tableau** is a bottom-left justified collection of boxes filled with integers that increase left to right across rows and increase bottom to top up columns. The question "How many lines intersect 4 generic lines in 3-dimensional space?" then becomes "How many standard Young tableaux are there of shape $(2, 2)$?" The two such tableaux are shown below.

3	4
1	2

2	4
1	3

In [7], Farkas and Lian answer geometrically an enumerative question in the study of curves. Specifically, suppose C is a complex curve of genus g, and $\mathbb{P}^r$ the (complex) projective space of r dimensions. Additionally, suppose that we have n distinct marked points $x_1, x_2, \ldots, x_n$ on our curve C, and n distinct points $y_1, y_2, \ldots, y_n$ in $\mathbb{P}^r$. Then, let $L_{g,r,d}$ be the number of degree d

morphisms $f : C \to \mathbb{P}^r$, such that $f(x_i) = y_i$ for all $i = 1, 2, \ldots, n$. See Figure 1.1 for a diagram of an example. The question answered by Farkas and Lian is to find explicit formulas for $L_{g,r,d}$ in terms of the parameters g, r, and d.

Remark 1.1.1. Note that we do not list n as a parameter here. This is because this number $L_{g,r,d}$ is both finite and nonzero precisely when $n = \frac{1}{r}(dr + d + r - rg)$.

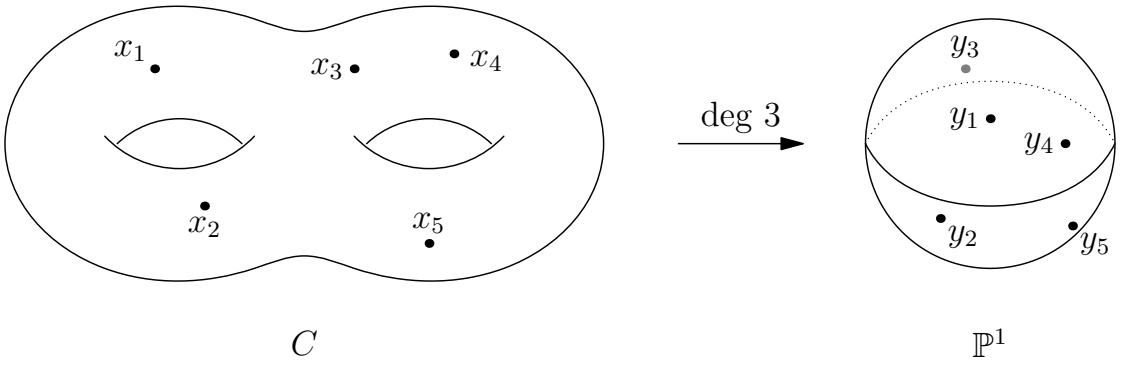


Figure 1.1: Case of $g = 2$, $r = 1$, and $d = 3$. In this case, $n = \frac{1}{1}(3 + 3 + 1 - 2) = 5$.

Farkas and Lian answered this question for three separate cases:

- For $d \geqslant rg + r$
- For $r = 1$
- For $d = r + \frac{rg}{r+1}$

They found that in the first two cases, the enumeration is $L_{g,r,d} = (r + 1)^g$. In the third case, they note that the enumeration matches a formula of Castelnuovo [4]:

$$L_{g,r,d} = g! \cdot \frac{1! \cdot 2! \cdots r!}{s! \cdot (s+1)! \cdot \cdots \cdot (s+r)!},$$

where $s = \frac{g}{r+1}$.

Additionally, they considered a slight variation of this problem. To get this modified problem, first, set $r = 1$. Second, identify some number k of the points y_i. That is, set $y_1 = y_2 = \cdots = y_k$

2	4	1	3	3	3
1	3	4	1	2	2
1	2	3	0	1	1
1	2	3	4	0	0

Figure 1.2: An example of an L-tableau with parameters $g = 4$, $r = 3$, and $d = 9$.

for some integer $1 \leqslant k \leqslant \min\{n, d\}$. Let $L'_{g,d,k}$ be the number of maps $f : C \to \mathbb{P}^1$ such that $f(x_i) = y_i$ for all i. Again, the question is to express $L'_{g,d,k}$ in terms of its parameters g, d, and k. Here, Farkas and Lian answer the question with an intersection formula on the Grassmannian.

Both problems can be interpreted in terms of intersections of Schubert varieties, which as we will see in Chapter 3 can be converted into questions about Young tableaux. In the first problem, Farkas and Lian do this already. They define the structure called an L-tableau with parameters g, r, and d, and show that the number of the above maps equals the number of these tableaux. See Figure 1.2 for an example.

Definition 1.1.2. ([7], reinterpreted in [12].) Define an L-tableau with parameters (g, r, d) to be a way of filling the boxes of an $(r+1) \times (d-r)$ grid with rg "red" integers and $(d-r)(r+1) - rg$ "blue" integers such that:

- The red integers are left-and-bottom justified, and weakly increase up columns and strictly increase across rows. They consist of the numbers $1, 2, \ldots, g$ each occurring exactly r times.
- The blue integers, which are necessarily right-and-top justified, are strictly increasing up columns and weakly increasing across rows. Their values are from $\{0, 1, ..., r\}$.

It is shown in [7] that whenever either $d \geqslant rg + r$, $r = 1$, or $d = r + \frac{rg}{r+1}$, we have

$$L_{g,r,d} = \int_{\mathrm{Gr}(r+1,d+1)} \sigma_{1^r}^g \cdot \left[\sum_{\alpha_0+\cdots+\alpha_r=(r+1)(d-r)-rg} \left(\prod_{i=0}^{r} \sigma_{\alpha_i} \right) \right], \tag{1.1}$$

and that in the first two cases, this formula equals $(r+1)^g$.

Here the notation σ_{1^r} is shorthand for $\sigma_{(1,1,1,\ldots,1)}$ where the tuple $(1,1,\ldots,1)$ has length r, and σ_{α_i} is shorthand for $\sigma_{(\alpha_i)}$. The integral indicates that the sum of products of Schubert cycles in question expands in the Schubert basis as a constant multiple of

$$\sigma_{(d-r)^{r+1}} := \sigma_{(d+1,d+1,\ldots,d+1)},$$

and the integral is defined to be this constant coefficient. The integral in equation (1.1) is equal to the coefficient of $s_{(d-r)^{(r+1)}}$ in the expansion

$$s_{1^r}^g \cdot \left[\sum_{\alpha_0+\cdots+\alpha_r=(r+1)(d-r)-rg} \left(\prod_{i=0}^{r} s_{\alpha_i} \right) \right] \tag{1.2}$$

However, their proof is purely geometric in nature, and they leave finding a combinatorial proof as an open problem. Our first result is a combinatorial proof to a slightly stronger version of the above fact.

Theorem 1.1.3. *The number of L-tableaux with parameters (g,r,d) is $(r+1)^g$ whenever $d \geqslant g+r$.*

Our result is a stronger version of theirs, because our proof only requires $d \geqslant g+r$, as opposed to $d \geqslant rg+r$, and we do not require that d be a multiple of r. We prove this by defining an intermediary object, as well as a bijection between the L-tableaux and these intermediary objects. We then apply the RSK correspondence to form a bijection between this intermediary and $(r+1)$-ary sequences of length g, which are enumerated by $(r+1)^g$. In the case where $r=1$, our first bijection is trivial, and the result follows from the RSK correspondence alone.

These results have since been extended in [19], providing a fuller yet combinatorial interpretation for a wider range of parameters for the maps $f : C \to \mathbb{P}^r$.

The corresponding result from [7] for $L'_{g,d,k}$ is that for $k+g \leqslant 2d+1$ and $2 \leqslant k \leqslant d$, we have

$$L'_{g,d,k} = \int_{\mathrm{Gr}(2,d+1)} \sigma_1^g \sigma_{k-1} \left(\sum_{i+j=2d-g-k-1} \sigma_i \sigma_j \right) - \int_{\mathrm{Gr}(2,d)} \sigma_1^g \sigma_{k-2} \left(\sum_{i+j=2d-g-k-2} \sigma_i \sigma_j \right). \tag{1.3}$$

They leave the enumeration formula for $L'_{g,d,k}$ as the difference of two Schubert class intersection formulas shown in equation (1.3). For our second main result, we first provide a combinatorial interpretation of this expression, and then give an explicit enumeration for this interpretation.

Theorem 1.1.4. *If $d \geqslant g + k$, we have $L'_{g,d,k} = 2^g$.*

Our interpretation makes use of similar structures to the L-tableaux above that we call L'-tableaux (see Definitions 4.3.1 and 4.3.3). Our proof shows that those objects corresponding to this difference can be reduced to the $r = 1$ case of our first result.

1.2 The moduli space of curves and trivalent trees

We also use this recipe for combinatorializing geometric intersection questions on a different class of moduli space. Next, the geometric objects we parameterize are complex curves with marked points.

We consider the moduli space of genus 0 curves with n marked points, denoted $M_{0,n}$. In particular, a curve $C \in M_{0,n}$ is isomorphic to $\mathbb{P}^1$ and consists of the data of n distinct points $p_1, p_2, \ldots, p_n \in \mathbb{P}^1$. One characteristic of this moduli space is that it is not compact, which means we cannot do intersection theory on it. So, geometers prefer to study compactifications of this space. We denote by $\overline{M}_{0,n}$ the **Deligne-Mumford compactification** of $M_{0,n}$, the moduli space of *stable* genus 0 curves with n marked points. Elements of this space now look like trees of $\mathbb{P}^1$s connected at nodes, where the word *stable* means that for each component, the sum of the number of nodes and the number of marked points is at least 3.

Given a stable curve $C \in \overline{M}_{0,n+3}$, we may also consider its **dual tree**. To form the dual tree to a curve, create a vertex for each component of C, as well as one for every marked point. Then, for each marked point, connect its vertex to the vertex corresponding to the component it is contained in. For each node, connect the vertices corresponding to the two components that intersect at that node.

Our results add to a large and growing body of study on the connections between combinatorial algorithms and moduli spaces of curves, aiming to replicate the success of algebraic combinatorics

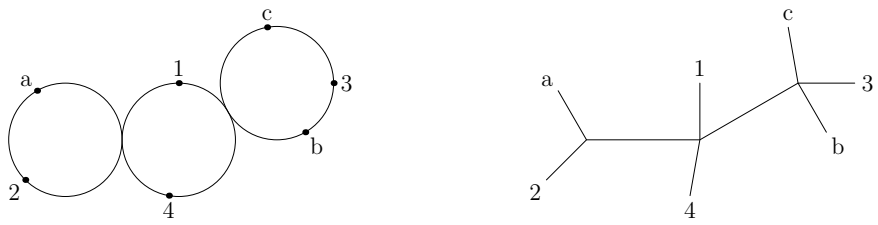


Figure 1.3: An element of $\overline{M}_{0,7}$, along with its dual tree.

in Schubert calculus (concerned with moduli spaces for linear objects) in the setting of curves. Such a combinatorializing of moduli spaces of curves was first suggested by Mumford [21], and much progress has been made in recent years.

Some recent work studying the degrees of projective maps on moduli spaces of curves includes [3, 6, 10, 11, 25]. In particular, [6], [10], and [11] study the Kapranov embedding of $\overline{M}_{0,n+3}$ into $\mathbb{P}^1 \times \cdots \times \mathbb{P}^n$ from a combinatorial perspective using trees and parking functions, [25] uses perfect matchings to study a map into a product of $\mathbb{P}^1$'s given by cross-ratio degrees, and [3] uses a different class of trees to study more general pullbacks of ψ classes. Work on products of ψ classes in tropical $M_{0,n}$ include [13, 17].

Here, we consider the Kapranov embedding and unify several of the combinatorial and geometric understandings of its (multi)degrees. In particular, for a positive integer n, let $\overline{M}_{0,n+3}$ be the Deligne-Mumford moduli space of stable genus 0 curves with $n+3$ marked points labeled by $\{a, b, c, 1, 2, \ldots, n\}$. The ith cotangent line bundle $\mathbb{L}_i$ is the line bundle whose fiber over $C \in \overline{M}_{0,n+3}$ is the cotangent space of C at the marked point i. The **ith psi class** ψ_i is the first Chern class of $\mathbb{L}_i$. In other words, $\psi_i = c_1(\mathbb{L}_i)$. The **$i$th omega class** ω_i is defined as the pullback of ψ_i under the composition of the forgetting maps that forget the marked points $i+1, \ldots, n$. The ω classes give rise to an embedding

$$\Omega_n : \overline{M}_{0,n+3} \hookrightarrow \mathbb{P}^1 \times \mathbb{P}^2 \times \cdots \times \mathbb{P}^n$$

by Kapranov [15].

Let $\underline{k} = (k_1, k_2, \ldots, k_n)$ be a n-tuple of nonnegative integers and assume it is a composition of n, that is, $k_1 + k_2 + \cdots + k_n = n$. Choosing n general hyperplanes, k_i from the ith term $\mathbb{P}^i$ of the product in the Kapranov embedding, and taking their intersection with the image of Ω_n gives the *multidegree* of the embedding, denoted $\deg_{\underline{k}}(\Omega_n)$.

Products in the cohomology ring of $\overline{M}_{0,n+3}$ of the form $\psi^{\underline{k}} := \psi_1^{k_1} \cdots \psi_n^{k_n}$ and of the form $\omega^{\underline{k}} := \omega_1^{k_1} \cdots \omega_n^{k_n}$ were studied in [10, 11] and shown to be computable by way of enumerating certain classes of trees called **slide trees** $\mathrm{Slide}^{\omega}(\underline{k})$ and **tournament trees** $\mathrm{Tour}(\underline{k})$.

Proposition 1.2.1 (From [10] and [11])**.** *Let $\underline{k}$ be a composition of n. Then,*

$$\int_{\overline{M}_{0,n+3}} \psi^{\underline{k}} = \binom{n}{k_1, k_2, \ldots, k_n} = |\mathrm{Slide}^{\psi}(\underline{k})|$$

and

$$\deg_{\underline{k}}(\Omega_n) = \int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ k_1, k_2, \ldots, k_n \end{matrix} \right\rangle = |\mathrm{Tour}(\underline{k})| = |\mathrm{Slide}^{\omega}(\underline{k})|.$$

The coefficients in the third part of the last equality are the **asymmetric multinomial coefficients**, which we define in Section 5.1. Several recent papers [6, 10, 11] have studied these asymmetric multinomial coefficients from both a geometric and combinatorial perspective. This forms part of a growing body of work that has been done using combinatorial objects to study the intersection theory of $\overline{M}_{0,n}$.

The original motivation behind the construction of both the tournament and slide trees is the observation that the total degree of the Kapranov embedding (the sum of the multidegrees) is $(2n-1)!! = (2n-1) \cdot (2n-3) \cdot \cdots \cdot 5 \cdot 3 \cdot 1$. However, $(2n+1)!!$ is also the number of boundary points in $\overline{M}_{0,n+3}$. Since the double factorial arises in both counts, a natural question then is to ask if there is a way to construct disjoint sets of these boundary points by degenerating the hyperplanes in the Kapranov embedding, in such a way that the union of the intersection sets is a natural set of boundary points.

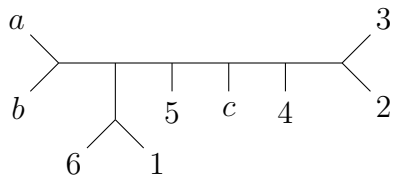


Figure 1.4: An example of a tree $T \in \mathrm{Slide}^{\omega}(0,0,2,1,1,2)$.

In [10, 11], this question was partially resolved in two different ways. One came by combinatorially constructing disjoint sets of boundary points that numerically matched the multi degrees, called *Tournament trees*. The other was by successfully constructing hyperplane degenerations to obtain sets of boundary points called *Slide trees* that geometrically enumerate the multidegrees. The slide trees are also easily obtained by a choice of convention that produces an algorithm for inductively expressing any product of ω classes in terms of boundary divisors, and yet this recursive procedure does not match the recursion given by the asymmetric multinomial recursion coming from the string equation.

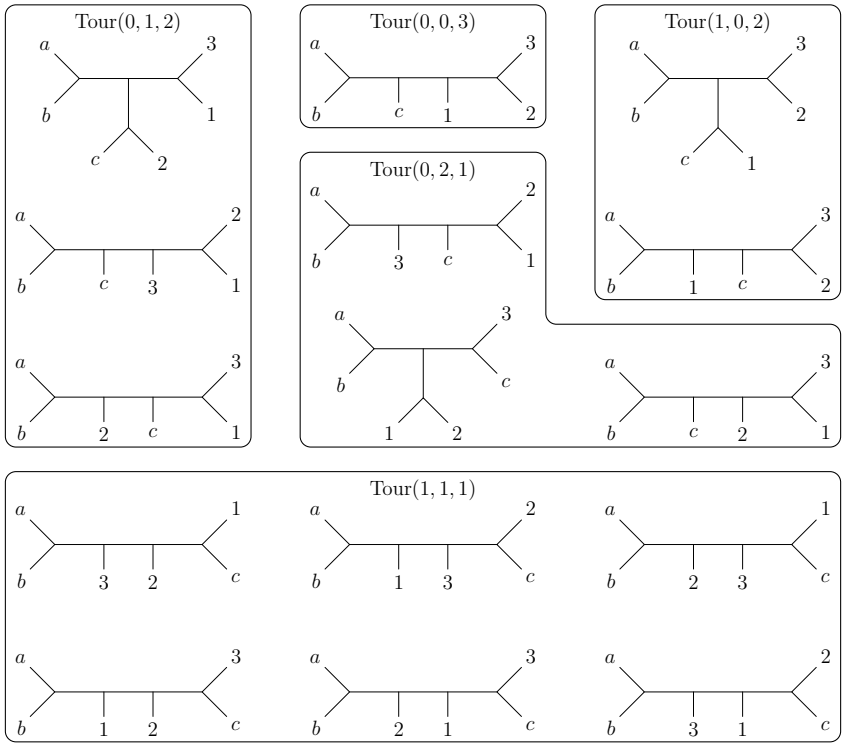


Figure 1.5: The tournament trees for $n = 3$, organized according to which $\mathrm{Tour}(\underline{k})$ they appear in.

In addition, neither the slide sets nor the tournament sets satisfy both of the desired criteria. The slide sets are not disjoint like the tournament trees are, and the tournament sets can not arise from a degeneration of hyperplanes. The degeneration construction makes the intersection products simple to compute, while the disjointness of the tournament trees gives them a clear correspondence to the boundary points of $\overline{M}_{0,n+3}$. So, both slide trees and tournament trees have both combinatorial and geometric advantages and disadvantages. Therefore, it would be advantageous to have a natural bijection between these two sets, allowing us to make use of the advantages of each type of tree. Despite this, finding a combinatorial bijection between these two sets has until now been an open question. (See Problem 6.1 in [10]).

We illustrate the combinatorial differences between $\mathrm{Tour}(\underline{k})$ and $\mathrm{Slide}^{\omega}(\underline{k})$ in the following examples for when $n = 3$.

Example 1.2.2. When $n = 3$, the $(2(3) - 1)!! = 5!! = 15$ tournament trees are shown in Figure 1.5, organized according to which composition $\underline{k}$ they correspond with. As one can see, the sets are disjoint.

On the other hand, Figure 1.6 shows the slide trees for $n = 3$. These sets are heavily overlapping, as while their individual cardinalities still add up to 15, there are only seven distinct trees. In fact, for any n, the unique tree in $\mathrm{Slide}(0, \ldots, 0, n)$ will be in the intersection of all $\mathrm{Slide}^{\omega}(\underline{k})$ for that n.

In this paper, we find a direct connection between the slide and tournament trees, thereby bridging this gap by showing that the slide trees are indeed in natural bijection with disjoint sets of boundary points (namely the already-constructed tournament trees).

The set $\mathrm{CPF}(\underline{k})$ above in Proposition 1.2.1 was the first combinatorial interpretation of this geometric problem, in terms of *parking functions* [6]. A bijection between $\mathrm{Tour}(\underline{k})$ and $\mathrm{CPF}(\underline{k})$ is given in [11]. Both the parking functions and tournaments interpretations were shown to satisfy the asymmetric multinomial recursion (5.1) defined in Section 5.1.

We similarly build our bijection recursively, with the main step being to show that $|\mathrm{Slide}(\underline{k})|$ also satisfies the same recursion. In particular, we build a bijection between $\mathrm{Slide}^{\omega}(\underline{k})$ and a

disjoint union of slide sets $\mathrm{Slide}^{\omega}(\underline{k}^{(j)})$ for compositions $\underline{k}^{(j)}$ of $n-1$, via an insertion algorithm on $\mathrm{Slide}^{\omega}(\underline{k})$. Then, we can unwind the recursive algorithms for each of $\mathrm{Slide}(\underline{k})$ and $\mathrm{Tour}(\underline{k})$ to recover a full bijection $F : \mathrm{Slide}(\underline{k}) \to \mathrm{Tour}(\underline{k})$.

Our insertion algorithm works by inserting a new leaf into our tree. Under certain conditions, we insert it in the position of an existing leaf, and 'bump' the original label to a new position. This in turn may cause a cascade of bumping labels, until we finally bump a label to a new leaf. This is reminiscent of the well-known RSK correspondence, which inserts labels into tableaux, sometimes bumping labels out into higher rows, until a label is put in a new box.

Theorem 1.2.3. *The map F is a combinatorial bijection between the sets* $\mathrm{Tour}(\underline{k})$ *and* $\mathrm{Slide}^{\omega}(\underline{k})$.

Our last main result is on caterpillar trees. We say a trivalent tree is a **caterpillar tree** if its internal edges form a path (see Figure 1.7). We can form a word from a caterpillar tree by reading the slide labels defined in Section 5.2, and obtain the following pattern avoidance condition that generalizes results in [10].

We have a map Tree that, given a word, constructs the slide tree of caterpillar shape whose edge labels, as read off from the root, form the given word, if such a tree exists. Otherwise, it forms

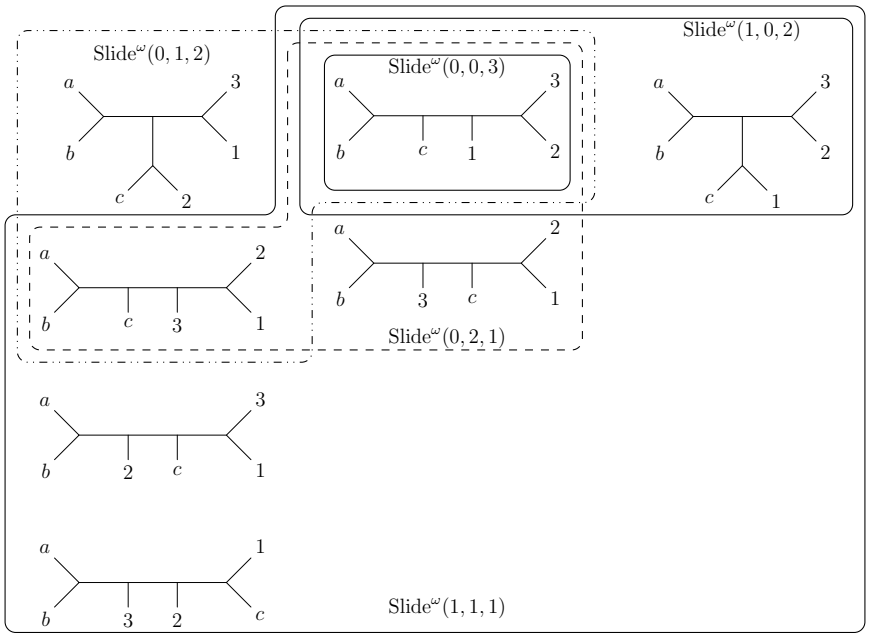


Figure 1.6: The slide trees for $n = 3$, organized according to which $\mathrm{Slide}^{\omega}(\underline{k})$ they appear in.

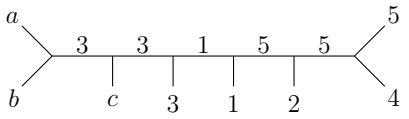


Figure 1.7: An example of a caterpillar slide tree, along with its edge labels

some caterpillar tree that is not a valid slide tree. That is, a word w can be read off of the edges of a caterpillar slide tree if and only if $\mathrm{Tree}(w)$ is a valid slide tree. Our result then characterizes caterpillar trees by way of characterizing these words w.

Theorem 1.2.4. *Let $\underline{k}$ be a right-justified composition of n. Then, $\mathrm{Tree}(w)$ is a valid slide tree if and only if w avoids the patterns $2{-}1{-}2$ and $23{-}\overline{2}{-}1$. (See Section 2.3.)*

Above, a *right-justified* composition is a composition where all non-zero entries appear right of any 0 entries. Otherwise, if $\underline{k}$ is not right-justified, the set of caterpillar trees can not be characterized solely by a pattern avoidance criterion. We state the characterization result below and define the terms precisely in Section 8.

Theorem 1.2.5. *Let $\underline{k}$ be a reverse-Catalan composition of n, and let w be a word of content $\underline{k}$. Then the caterpillar tree $\mathrm{Tree}(w)$ is in $\mathrm{Slide}^{\psi}(\underline{k})$ (resp., $\mathrm{Tree}(w)$ is in $\mathrm{Slide}^{\omega}(\underline{k})$) if and only if w avoids the patterns $2{-}1{-}2$ and $23{-}\overline{2}{-}1$, and the inequality*

$$\mathrm{TotalRep}_w(i) + \ell_w(i) \geqslant \mathrm{z}(i)$$

holds for all i (respectively, $\mathrm{BigRep}_w(i) \geqslant \mathrm{z}(i)$ holds for all i).

Here, $\ell_w(i)$ is the number of is in the rightmost consecutive group of is in w, $\mathrm{TotalRep}_w(i)$ is the total number of repeated letters to the right of the rightmost i in w, and $\mathrm{BigRep}_w(i)$ is the total number of repeated letters larger than i to the right of the rightmost i in w.

The results found in Chapters 6–9 first appeared in [23].

Chapter 2

Combinatorics background

2.1 Young tableaux

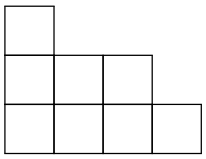
Young tableaux are a well studied structure in algebraic combinatorics. We introduce them by first defining partitions.

Definition 2.1.1. A **partition** of an integer n is a tuple $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_r)$, with the properties that $\lambda_1 + \lambda_2 + \cdots + \lambda_r = n$, and that $\lambda_i \geqslant \lambda_{i+1}$ for all i. We call r the number of *parts* of the partition. We sometimes write $\lambda \vdash n$.

Although tuples are one possible way to denote a partition, it can be useful to have a more visual depiction of partitions. Using Young diagrams is one common choice.

Definition 2.1.2. A **Young diagram** is a drawing of bottom- and left-justified unit squares in the first quadrant. For a particular partition $\lambda \vdash n$, the corresponding Young diagram consists of n boxes, with λ_1 in the first (bottom) row, λ_2 in the second (from bottom) row, and so forth.

Example 2.1.3. Below is an example of a Young diagram. It corresponds to the partition $\lambda = (4, 3, 1)$, and has size $4 + 3 + 1 = 8$.

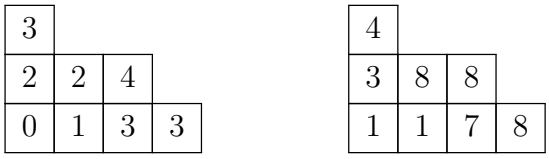


One reason we often denote partitions as collections of boxes is so that we can label the boxes. Such a labeling is called a *tableau*.

Definition 2.1.4. A **semistandard Young tableau** is a Young diagram where each box is filled with a nonnegative integer, where the entry in each box must be strictly greater than the entry in the box underneath it, and weakly greater than the entry in the box to its left.

In general, a Young tableau may be filled using any alphabet, but in this thesis we will only use nonnegative integers. The semistandard condition allows us to repeat entries as we move left to right across rows, but not as we move bottom to top along columns. We will often refer to the Young diagram underlying a Young tableau as the *shape* of that tableau.

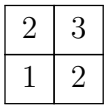
Example 2.1.5. Here are two different semistandard Young tableaux of shape $(4, 3, 1)$.



We refer to the multiset of our choice of labels the *content* of the tableau.

Definition 2.1.6. Given a tuple $\mu = (\mu_1, \mu_2, \ldots, \mu_k)$, we say that a tableau has **content** μ if it has μ_1 boxes labeled 1, μ_2 boxes labeled 2, and so on.

Remark 2.1.7. The content of a tableau is not always a partition. For example, the tableau



has content $(1, 2, 1)$, which is not a partition.

Next, we define the related concept of the *standard* tableau. This is a special case of a semistandard tableau, which is defined by adding additional conditions.

Definition 2.1.8. A **standard** Young tableau is a Young diagram where each box is filled with a nonnegative integer, where the entry in each box must be strictly greater than both the entry to its left and below it, and each integer from $1, \ldots, n$ is used exactly once, where n is the total number of boxes.

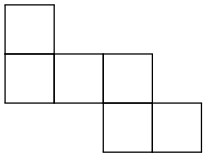
Example 2.1.9. Here is a standard Young tableaux of shape $(4, 3, 1)$. Each integer $1, 2, \ldots, 8$ occurs exactly once.

7			
2	5	8	
1	3	4	6

We will also consider filling shapes other than full partitions. One way we can vary the shape of a filling is by removing the shape of a smaller partition from that of a larger one.

Definition 2.1.10. Let λ and μ be two partitions such that $\lambda_i \geqslant \mu_i$ for all i. Then, a **skew tableau** of shape λ/μ is a semistandard filling of the shape formed by superimposing the shapes of λ and μ and removing the boxes corresponding to the entries of μ.

Example 2.1.11. Let $\lambda = (4, 3, 1)$ and $\mu = (2)$. Then, the skew shape λ/μ is drawn below on the left, and a skew tableau of shape λ/μ is given on the right.



1			
0	2	2	
		1	3

We define a canonical way of reading off the entries of a tableau.

Definition 2.1.12. The **reading word** of a tableau is formed by concatenating the entries of each row from top to bottom.

Thus, we read off the entries of a tableau in the same way as one would read the words on a page of a book.

Example 2.1.13. The reading word of the tableau in Example 2.1.9 is 72581346. The reading word of the skew tableaux in Example 2.1.11 is 102213.

A particular type of tableau that is used for computing intersections of Schubert varieties is that of *Littlewood-Richardson tableaux*. To define them, we first need to define what it means for a reading word to be *Yamanouchi*.

Definition 2.1.14. A word $w = w_1 w_2 \dots w_{n-1} w_n$is **Yamanouchi** if every subword of the form $w_k w_{k+1} \dots w_{n-1} w_n$ has the property that for each i, there are at least as many instances of i as there are of $i + 1$.

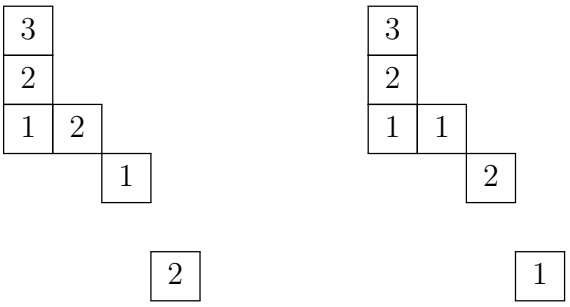
Example 2.1.15. The word 2123121 is Yamanouchi, while the word 1223121 is not. This is because the subword 223121 contains three 2s, but only two 1s.

Combining these last two definitions, we can now define a Littlewood-Richardson tableau.

Definition 2.1.16. A (skew) semistandard tableau is a **Littlewood-Richardson tableau** if its reading word is Yamanouchi.

Remark 2.1.17. The condition that there be at least as many total is as total $(i+1)$s means that $\mu_i \geqslant \mu_{i+1}$, where μ is the content of the tableau. Thus, all Littlewood-Richardson tableaux have partitions as content.

Example 2.1.18. Consider the following two skew tableaux of shape $(4,3,3,2,1,1)/(3,3,2)$.



They have reading words 321212 and 321121, respectively. The second is Yamanouchi, while the first is not. So, only the second one is a Littlewood-Richardson tableau.

2.2 The RSK correspondence

The RSK correspondence is a famous bijection in combinatorics. It is named for the mathematicians Robinson, Schensted, and Knuth. There are several different variations of the correspondence. The most general is between two-line arrays and pairs of semistandard Young tableaux. Here, we will use use a specialization in which one of the two tableaux is standard, and the pairs are in bijection with general sequences of length n. We state this version below, without proof. (See [8] or [26, Ch. 7] for a proof of the correspondence.)

Proposition 2.2.1 (The RSK correspondence for words)**.** *Let $A(n)$ be the set of all sequences of length n and $B(n)$ the set of all pairs of tableaux (P,Q), such that P and Q have the same shape*

of size n, P is semistandard, and Q is standard. Then, there exists a constructive bijection between $A(n)$ and $B(n)$.

If we place some restriction on the content of the sequence in $A(n)$, that imposes the same restriction on the content of P. This observation leads to the following corollary.

Corollary 2.2.2. Let $A(r, n)$ be the set of all sequences of length n from the alphabet $\{0, 1, \ldots, r\}$. Let $B(r, n)$ the set of all pairs of tableaux (P, Q), such that P and Q have the same shape of size n, P is semistandard with content from the set $\{0, 1, \ldots, r\}$, and Q is standard. Then, there exists a constructive bijection between $A(r, n)$ and $B(r, n)$.

We now describe and illustrate the RSK correspondence.

Algorithm 2.2.3. Let $w = w_1 w_2 \ldots w_n$ be an arbitrary sequence of length n. Set P and Q to each be the empty tableau. Then, for each letter w_i in order, we do the following:

1. Set $r := 1$, and attempt to insert w_i into row r of P.

2. To insert w_i into row r, check if either w_i is at least as large as the largest entry of row r, or that row r is empty. If so, add a new box to the end of row r labeled w_i, and go to step 4.

3. Otherwise, find the leftmost entry b in row r strictly larger than w_i, and 'bump' b up one row. That is, replace the label b with w_i, and then insert b into row $r + 1$, going back to step 2.

4. Add a new box to Q in the same position as that of the new box that was added to P, and label it i.

At the end, this will result in two tableaux P and Q with the same shape, and P and Q will be semistandard and standard, respectively.

Example 2.2.4. Start with the sequence $0, 2, 1, 1, 0, 3, 0, 0, 1$. Since P is currently empty, we can freely add the initial 0 to the first row of P, also creating a box in Q labeled 1. At each step, we will draw the tableau P on the left, and the tableau Q on the right.

0

1

The next entry of the sequence is a 2, and since $2 > 0$, we add a box to the right of the 0 and fill it with 2. Since this is the second entry, we do the same in Q.

0	2

1	2

Next, we try to insert a 1 into the first row of P, but since 1 is larger than 2, we instead 'bump' out the 2. That is, we replace the 2 with 1, then insert the 2 in the first position of the second row. In Q, that same box is labeled with a 3.

2	
0	1

3	
1	2

The fourth entry is a 1, and even though this is not larger than the last entry in the first row of P, repeated entries are fine, so we add on a second 1 to the end of the row.

2		
0	1	1

3		
1	2	4

Then, we have another 0, so we bump the first 1 out of the first row. Since this 1 is smaller than the 2 in the second row, it in turn bumps the 2 up into the third row.

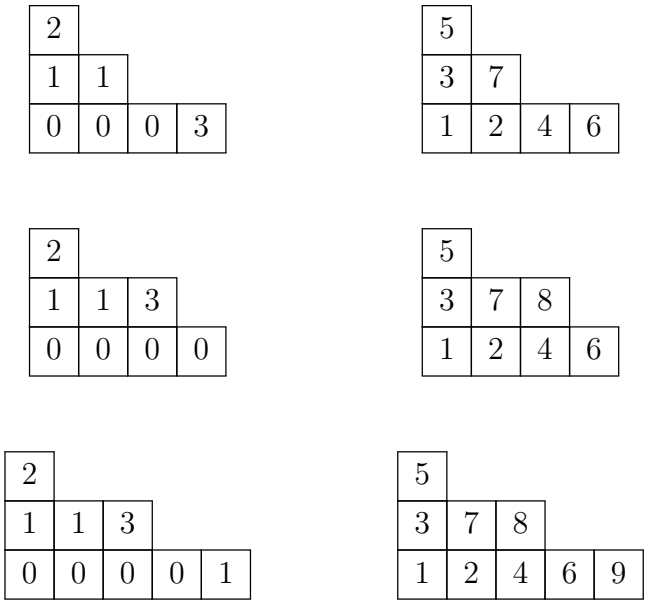
2		
1		
0	0	1

5		
3		
1	2	4

We continue on in the same manner described above:

2			
1			
0	0	1	3

5			
3			
1	2	4	6



Thus, we are left with the above pair (P, Q), which do indeed have the same shape, and have that P is semistandard and Q is standard.

Remark 2.2.5. It is well known (see [8, Ch. 3], for example) that in the RSK bijection, the number of rows corresponds to the length of the longest increasing subsequence, and the number of columns corresponds to the length of the longest nondecreasing subsequence.

2.3 Pattern Avoidance

We start by defining the main ideas of classical pattern avoidance, and then define two particular variants that will show up in this paper. For a more thorough summary of classical pattern avoidance, we refer the reader to [2].

Definition 2.3.1. A permutation σ **contains** another permutation τ if σ contains a subword with the same relative order as τ. Conversely, a permutation σ **avoids** a pattern τ if σ contains no subword with the same relative order as τ.

Another way to formulate this concept is via *reductions*.

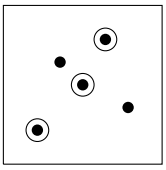
Definition 2.3.2. The **reduction** $\mathrm{red}(w)$ of a permutation or word w is the word obtained by replacing the ith smallest entry of w by i, for all i. If $\mathrm{red}(w) = v$, we say w **reduces** to v.

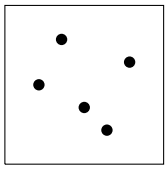
For example, $\mathrm{red}(2574) = 1342$. Then, pattern containment can be rephrased as a matter of a permutation containing a subword that reduces to the desired pattern.

It is common practice to visualize words or patterns by graphing them. This is done by plotting the points (i, σ_i) for $i = 1, 2, \ldots, n$ on an $n \times n$ grid. We illustrate this in the following example.

Example 2.3.3. The word 14352 contains the pattern $\tau = 123$, since the subword 135 has the same relative order as $\tau = 123$. See the circled entries in the diagram below on the left.

On the other hand, the word $\sigma = 35214$ avoids $\tau = 123$, since none of its length 3 subwords are in increasing order. See the diagram below on the right.





Definition 2.3.4. For a pattern τ, we denote the set of permutations of length n that avoid τ by

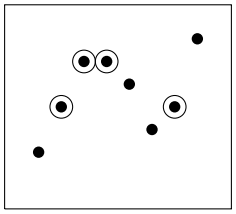
$$\mathrm{Av}_n(\tau) = \{\sigma \in S_n : \sigma \text{ avoids } \tau\}.$$

Similarly, for a collection of patterns $\tau^{(1)}, \ldots, \tau^{(k)}$, we define the set of permutations of length n that avoid each $\tau^{(i)}$ as

$$\mathrm{Av}_n(\tau^{(1)}, \ldots, \tau^{(k)}) = \{\sigma \in S_n : \sigma \in \mathrm{Av}_n(\tau^{(i)})\, \forall i \in [k]\}.$$

Throughout this work, we will primarily be working with words that are not permutations. We can extend all of these ideas to the case where σ and τ are both words instead of permutations.

Example 2.3.5. The word 24665347 contains the pattern 1221, since it contains the subword 4664, which has the same relative order as 1221. See the diagram below.



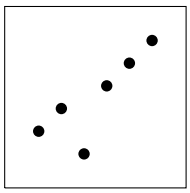
Notation 2.3.6. We denote the set of words with content $\underline{k}$ that avoid a pattern τ by $\mathrm{Av}_{\underline{k}}(\tau)$, and similarly for a collection of patterns. Observe that $\mathrm{Av}_{(1,1,\ldots,1)}(\tau) = \mathrm{Av}_n(\tau)$.

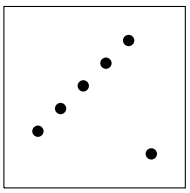
There are two further variants of pattern avoidance that we will define. The first is that of barred patterns. For a more comprehensive study of barred patterns, see [22].

Definition 2.3.7. Let τ be a word where some letters are barred, and the remaining letters are unbarred. We call τ a **barred pattern**. We say that σ **contains** τ if σ has a subword with the same relative order as the non-barred letterns of τ that does *not* extend to a subword with the same relative order as all of τ. Otherwise, σ **avoids** τ.

Example 2.3.8. The word $\sigma = 231456$ contains the barred pattern $\tau = 23\overline{1}$. Although the length-2 subword 23 can extend to the subword 231, the subword 45 (among others) cannot, meaning it is an instance of a $23\overline{1}$ pattern.

On the other hand, the word $\sigma = 234561$ avoids the barred pattern $\tau = 23\overline{1}$, since any subword with relative order 23 can be extended, by adding the letter 1 at the end, to a subword with relative order 231.





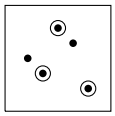
Finally, we can also impose adjacency conditions on some entries of our pattern. Such patterns are called *vincular* patterns. Vincular patterns were first introduced in [1] as *generalized permutations*, and a more detailed history on them can be found in [27].

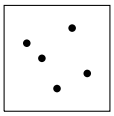
Definition 2.3.9. Let τ be a word of length k with dashes between some entries. We call τ a **vincular pattern**. A word σ **contains** τ if σ contains a subword σ' that has the same relative order as τ, and for each $i \in [k-1]$, if the en entries τ_i, τ_{i+1} in τ are not separated by a dash, then σ'_i and σ'_{i+1} come from adjacent entries σ_j, σ_{j+1} in σ. Otherwise, we say σ **avoids** the vincular pattern τ.

In other words, to contain a vincular pattern, we must have a consecutive subword with the relative order of the pattern, except that the dashes in the pattern indicate entries that need not come from consecutive letters of our word.

Example 2.3.10. The word $\sigma = 32541$ contains the pattern $\tau = 23{-}1$, since it contains the subword 251, where the 2 and 5 are adjacent, and 251 has the relative order 231.

On the other hand, although $\sigma = 43152$ contains the classical pattern $\tau = 231$ (consider the subword 352), σ avoids the vincular pattern $\tau = 23{-}1$, since the 3 and 5 in 352 (as well as the 4 and 5 of 452) are not adjacent.





Chapter 3

Geometry background

3.1 The Grassmannian and Schubert varieties

Definition 3.1.1. The **(complex) Grassmannian**, denoted $\mathrm{Gr}(k, n)$ is the set of all k-dimensional subspaces of $\mathbb{C}^n$.

One example of a Grassmannian is the projective space.

Definition 3.1.2. The n-dimensional (complex) **projective space** $\mathbb{P}^n$ is the quotient of the space $\mathbb{C}^{n+1}\backslash\{(0, 0, \ldots, 0)\}$ by the equivalence relation $(x_0, x_1, \ldots, x_n) \sim (\lambda x_0, \lambda x_1, \ldots, \lambda x_n)$, for any $\lambda \in \mathbb{C}\backslash\{0\}$.

In other words, we can view projective space as a complex space of dimension one higher, where we identify points on the same line through the origin. In this way, $\mathbb{P}^n \cong \mathrm{Gr}(1, n+1)$. Since we do not care about scaling, we can also view projective space as the set of all ratios on $n+1$ coordinates, and will denote elements accordingly by $(x_0 : x_1 : \cdots : x_n)$.

Example 3.1.3. It is common practice to express elements of projective space in the form where the first non-zero entry is 1. Consider the space $\mathbb{P}^2 \cong \mathrm{Gr}(1, 3)$. We can express each element in exactly one of the following three forms: $(1 : a : b)$, $(0 : 1 : c)$, and $(0 : 0 : 1)$, where $a, b, c \in \mathbb{C}$.

We can conversely embed the Grassmannian into a projective space, but to do so we first need to introduce the concept of a variety.

Definition 3.1.4. A **variety** is the set of all common zeros to some collection of polynomials, usually denoted $\mathbb{V}(f_1, f_2, \ldots, f_r)$. In this work, we are interested only in **projective varieties**. That is, for given homogeneous polynomials $f_1, f_2, \ldots, f_r \in \mathbb{P}[x_1, x_2, \ldots, x_n]$,

$$\mathbb{V}(f_1, f_2, \ldots, f_r) = \{(a_0 : a_1 : \cdots : a_n) \in \mathbb{P}^n \mid f_i(a_0 : a_1 : \cdots : a_n) = 0 \,\forall\, i\}.$$

A particular element of the Grassmannian $\mathrm{Gr}(k,n)$ is a k-dimensional subspace of $\mathbb{C}^n$, so we can think of it in terms of being the span of k vectors of length n. We can arrange these into a $k \times n$ matrix. This allows us to view the Grassmannian as a projective variety via what is known as the **Plücker embedding**.

Given an element of $\mathrm{Gr}(k,n)$, we can compute all $\binom{n}{k}$ of the $k \times k$ minors. Given a particular k-element subset $\{i_1, i_2, \ldots i_k\} \subseteq [n]$, the determinant of the matrix given by choosing columns $i_1, i_2, \ldots, i_k$ we call $x_{i_1,i_2,\ldots,i_k}$. Then, for a fixed ordering of these k-element subsets $K_1, K_2, \ldots, K_m$ (with $m = \binom{n}{k}$), we get the **Plücker coordinate** $(x_{K_1} : x_{K_2} : \cdots : x_{K_m}) \in \mathbb{P}^{m-1}$. Since rescaling any row of our original matrix rescales each determinant equally, this embedding is well-defined.

Example 3.1.5. Consider the element of $\mathrm{Gr}(2,4)$ given by

$$\begin{bmatrix} 1 & 0 & 4 & -2 \\ 0 & 1 & 3 & 2 \end{bmatrix}.$$

We then have $x_{12} = 1$, $x_{13} = 3$, $x_{14} = -2$, $x_{23} = -4$, $x_{24} = 2$, and $x_{34} = 2$. Then, this element has Plücker coordinate $(x_{12} : x_{13} : x_{14} : x_{23} : x_{24} : x_{34}) = (1 : 3 : -2 : -4 : 2 : 2)$.

As noted above, each point in the Grassmannian may be represented by a $k \times n$ matrix. However, these representatives are not unique. So, to define a canonical form, we choose by convention the matrix representative that is in reduced row echelon form.

In many contexts, we care less about the exact entries of the matrix corresponding to a point in the Grassmannian, and more about the locations of the pivot columns.

Definition 3.1.6. A **Schubert cell** is the set of all points in the Grassmannian whose matrices have the same row reduced echelon form.

In this way, we can decompose the Grassmannian into the disjoint union of its Schubert cells.

Example 3.1.7. The matrices

$$\begin{bmatrix} 1 & 5 & 7 & 0 & 6 & 7 \\ 2 & 4 & 8 & 8 & 2 & 3 \\ -1 & 0 & -2 & 0 & 1 & 5 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & 0 & 1 & 7 & 0 & 0 \\ 1 & -1 & 0 & 0 & 4 & 8 \\ 1 & 1 & 2 & 3 & 2 & -1 \end{bmatrix}$$

represent distinct points in $\mathrm{Gr}(3, 6)$, but share the same reduced row echelon form

$$\begin{bmatrix} 1 & 0 & * & 0 & * & * \\ 0 & 1 & * & 0 & * & * \\ 0 & 0 & 0 & 1 & * & * \end{bmatrix},$$

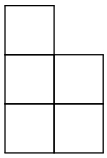
where $*$ represents an entry that can be any complex number. In other words, they both lie in the same Schubert cell.

We wish to classify the different Schubert cells of a particular Grassmannian. To do so, we will relate Schubert cells to partitions. To determine the partition that indexes a particular Schubert cell, we consider the locations of the pivots. In general, the pivots will occur on the main diagonal, with a staircase of zeros underneath. However, these are the leftmost possible positions, as one or more pivots may appear further to the right. For a given row r, the number of extra spaces to the right of this expected position that the pivot occurs will correspond to the length of the $(k - r)$th row of the partition.

Example 3.1.8. The point in $\mathrm{Gr}(4, 8)$ represented by the matrix

$$\begin{bmatrix} 1 & * & 0 & * & 0 & 0 & * & * \\ 0 & 0 & 1 & * & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & 1 & 0 & * & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * & * \end{bmatrix}$$

corresponds to the Young diagram below.



Put another way, if we are given a matrix in the above form with entries that are 0, 1, or $*$, we can form the tableau by deleting all the pivot columns and turning each 0 entry into a box.

Remark 3.1.9. This does allow for an empty tableau, in the case where the only 0s are all in the bottom left staircase.

Remark 3.1.10. The largest possible tableau is a $k \times (n-k)$ box, which is known as the **ambient rectangle**, often called B. Then, it can be shown that a tableau corresponds to a nonempty Schubert cell if and only if it fits inside of the ambient rectangle.

We now revisit our definition of Schubert cells.

Definition 3.1.11. For a partition λ with at most k parts and $\lambda_i \leqslant n-k$ for each i, the **Schubert cell** Ω_λ° is the set of all points in the Grassmannian whose reduced row echelon matrices have forms corresponding to λ.

Next, we generalize this idea back to the concept of varieties.

Definition 3.1.12. The **Schubert variety** Ω_λ is the set

$$\Omega_\lambda = \{V \in \mathrm{Gr}(k,n) \mid \dim(V \cap \langle e_1, e_2, \ldots, e_{n-k+i-\lambda_i}\rangle) \geqslant i\},$$

where the e_i are the standard unit vectors.

Put another way, the Schubert variety Ω_λ is the disjoint union of all Schubert cells whose tableaux fit inside of λ. In this way, the Schubert variety can be seen as the *closure* of its Schubert cell. That is, $\Omega_\lambda = \overline{\Omega_\lambda^\circ}$.

The last related construction we will need is that of the flag.

Definition 3.1.13. A **(complete) flag** is a chain of subspaces

$$F_\bullet : 0 = F_0 \subset F_1 \subset \cdots \subset F_n = \mathbb{C}^n,$$

where each space F_i has dimension i.

This allows us to define Schubert varieties relative to any basis of $\mathbb{C}^n$, not just the standard basis. So, we can generalize the definition of a Schubert variety to be

$$\Omega_\lambda(F_\bullet) = \{V \in \mathrm{Gr}(k,n) \mid \dim(V \cap F_{n-k+i-\lambda_i}) \geqslant i\},$$

for a given flag $F_\bullet$.

3.2 The Littlewood-Richardson rule for intersecting Schubert varieties

Recall our earlier discussion on intersection problems, and how we are in particular interested in when an intersection problem has an answer that is both positive and finite. For the Grassmannian, the answer can be related to intersections of Schubert varieties.

Example 3.2.1. This example is problem 3.8(a) from [9].

In $\mathbb{P}^4$, let 2-planes A and B intersect in a point X, and let P and Q be distinct points different from X. Let S be the set of all 2-planes C that contain both P and Q and intersect A and B each in a line. Express S as an intersection of Schubert varieties in $\mathrm{Gr}(3,5)$, when P is contained in A and Q is contained in B.

First, we let $F_\bullet^{(1)}$ and $F_\bullet^{(2)}$ be sufficiently general flags. Then, we will relate $F_\bullet^{(1)}$ to A and P, and $F_\bullet^{(2)}$ to B and Q. When we projectivize, P becomes 1-dimensional and A becomes 3-dimensional. So, we are looking for points $V \in \mathrm{Gr}(3,5)$ that intersect P in at least 1 dimension and A in at least 2. In other words, $V \cap F_3^{(1)} \geqslant 2$ and $V \cap F_1^{(1)} \geqslant 1$. The first equation gives us $i = 2$ and $5-3+2-\lambda_2 = 3$, so $\lambda_2 = 1$. The second equation gives us $i = 1$ and $5-3+1-\lambda_1 = 1$,

so $\lambda_1 = 2$. Thus, the variety corresponding to A and P is $\Omega_{(2,1)}(F_\bullet^{(1)})$. Similarly, for B and Q we get $\Omega_{(2,1)}(F_\bullet^{(2)})$. To find S, we take the intersection of these two, $\Omega_{(2,1)}(F_\bullet^{(1)}) \cap \Omega_{(2,1)}(F_\bullet^{(2)})$.

For a given property to be true for "sufficiently general flags", it means that there must be a dense open subset of the flag variety for which the given property holds. In the 2-dimensional case above, it is sufficient to choose *transverse* flags. That is, choose $F_\bullet^{(1)}$ and $F_\bullet^{(2)}$ such that $F_i^{(1)}$ and $F_{n-i}^{(2)}$ intersect trivially for all i. We refer the reader to [9] for more details. In this work, we will always assume that our choice of flags is general enough to ensure that these properties always hold.

Now we have seen how these intersection questions relate to Schubert varieties, but when should we expect the answer to be interesting? That is, how can we tell from the varieties when the answer should be finite and positive? Or, put yet another way, when is an intersection of Schubert varieties of dimension zero? An intersection

$$\Omega_{\lambda^{(1)}}(F_\bullet^{(1)}) \cap \Omega_{\lambda^{(2)}}(F_\bullet^{(2)}) \cap \cdots \cap \Omega_{\lambda^{(r)}}(F_\bullet^{(r)})$$

is zero-dimensional when $\sum_{i=1}^{r} |\lambda^{(i)}| = k(n-k)$. Recall that $|B| = k(n-k)$, where B is our ambient rectangle.

We will now digress briefly into the world of algebraic topology. For more details, we refer the reader to [14].

First, we need the construction of a cell complex.

Definition 3.2.2. A **cell complex** is a topological space X constructed as follows.

1. Start with a set of points X^0, called the **0-skeleton**.

2. Given the $(n-1)$-skeleton X^{n-1}, we 'glue' to it some number of n-cells. That is, we take the disjoint union $X^{n-1} \bigsqcup_\alpha D_\alpha^n$, where D_α^n is some collection of n-cells, and then identify the boundary of each n-cell with X^{n-1}. The result is the n**-skeleton** X^n.

3. We either terminate after a finite number of iterations, in which case the cell complex is just $X := X^n$ for some n, or we continue infinitely, in which case $X := \bigcup_n X^n$.

What we will want to do is to build a cell-complex representation of the Grassmannian, and then take its cohomology ring $H^*(\mathrm{Gr}(k,n)) = \bigoplus H^i(\mathrm{Gr}(k,n))$ as the setting where we compute the intersections.

Remark 3.2.3. When constructing a Grassmannian space, we only make use of even-dimension real cells (since each complex dimension is made up of two real dimensions), so the odd cohomology groups are all 0, and the even cohomology groups are solely determined by the number of partitions λ of each size that fit inside the ambient rectangle B.

The way this construction works is that we first take X^0 to be Ω_B°. Then, we make X^2 by attaching the Schubert cell corresponding to removing 1 box from B, X^4 by attaching the cells corresponding to both ways of removing 2 boxes, and so on. The final skeleton $X^{2k(n-k)}$ is formed by attaching $\Omega_\varnothing^\circ$.

Then, the cohomology ring $H^*(\mathrm{Gr}(k,n))$, also sometimes called the *Chow ring*, will have as a basis the elements $\sigma_\lambda \in H^{2|\lambda|}(\mathrm{Gr}(k,n))$, where the σ_λ are indexed by λ that fit inside B.

The intersection of Schubert varieties $\Omega_{\lambda^{(1)}}(F_\bullet^{(1)}) \cap \Omega_{\lambda^{(2)}}(F_\bullet^{(2)}) \cap \cdots \cap \Omega_{\lambda^{(r)}}(F_\bullet^{(r)})$ is equivalent to taking the product $\sigma_{\lambda^{(1)}} \cdot \sigma_{\lambda^{(2)}} \cdot \cdots \cdot \sigma_{\lambda^{(r)}}$ in the cohomology ring. In the desired case where $\sum_{i=1}^r |\lambda_i| = n(n-k)$, this product must live inside $H^{2k(n-k)}$, so it is some multiple of the generator σ_B. In other words,

$$\sigma_{\lambda^{(1)}} \cdot \sigma_{\lambda^{(2)}} \cdot \cdots \cdot \sigma_{\lambda^{(r)}} = c^B_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}} \sigma_B,$$

for some coefficient $c^B_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}}$.

This coefficient $c^B_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}}$ is called a **Littlewood-Richardson coefficient**, and can be interpreted purely combinatorially.

Definition 3.2.4. A **chain** of skew tableaux is a sequence of skew tableaux $T_1, T_2, \ldots, T_n$, where each pair T_i and T_j are disjoint, and $T_1 \cup T_2 \cup \cdots \cup T_i$ is a partition shape for all i.

Proposition 3.2.5 (Littlewood-Richardson Rule)**.** *Let $c^{\nu}_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}}$ denote the number of chains of Littlewood-Richardson tableaux with contents $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(r)}$ and total shape ν. Then,*

$$\sigma_{\lambda^{(1)}} \cdot \sigma_{\lambda^{(2)}} \cdot \cdots \cdot \sigma_{\lambda^{(r)}} = \sum_{\nu} c^{\nu}_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}} \sigma_{\nu},$$

where ν varies over all tableaux that fit inside of the ambient rectangle.

Recall that we are interested in the case where $\sum |\lambda_i| = n(n-k)$. So, in this case we can only have $\nu = B$, which gives the following corollary.

Corollary 3.2.6. Let $c^{B}_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}}$ denote the number of chains of Littlewood-Richardson tableaux with contents $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(r)}$ and total shape B. Then,

$$\sigma_{\lambda^{(1)}} \cdot \sigma_{\lambda^{(2)}} \cdot \cdots \cdot \sigma_{\lambda^{(r)}} = c^{B}_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}} \sigma_{B}.$$

This aligns with our observation from earlier.

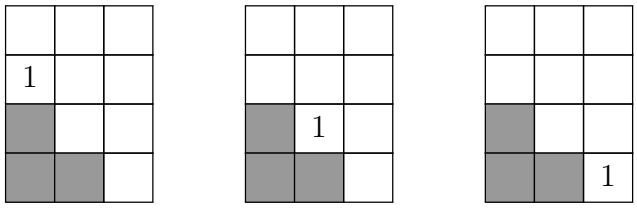
Example 3.2.7. Let $k = 4$ and $n = 7$. If $\lambda^{(1)} = (2,1)$, $\lambda^{(2)} = (1)$, $\lambda^{(3)} = (2,2)$, and $\lambda^{(4)} = (2,1,1)$, then there are three chains of Littlewood-Richardson tableaux with contents $\lambda^{(1)}, \lambda^{(2)}, \lambda^{(3)}, \lambda^{(4)}$ that fill $B = (3,3,3,3)$. That is, $c^{B}_{\lambda^{(1)},\lambda^{(2)},\lambda^{(3)},\lambda^{(r)}} = 3$, so $\sigma_{(2,1)}\sigma_{(1)}\sigma_{(2,2)}\sigma_{(2,1,1)} = 3\sigma_B$.

To show this, we explicitly calculate the number of chains with contents $(2,1)$, (1), $(2,2)$, and $(2,1,1)$.

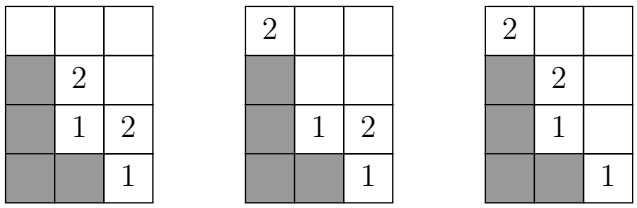
First, there are two semistandard tableau with content $(2,1)$. However, $\begin{array}{|c|c|c|}\hline 1 & 1 & 2 \\ \hline \end{array}$ is not Yamanouchi, so so we fill the bottom of the ambient rectangle with the only valid tableau of content $(2,1)$.

$$\begin{array}{|c|c|c|}\hline & & \\ \hline & & \\ \hline 2 & & \\ \hline 1 & 1 & \\ \hline \end{array}$$

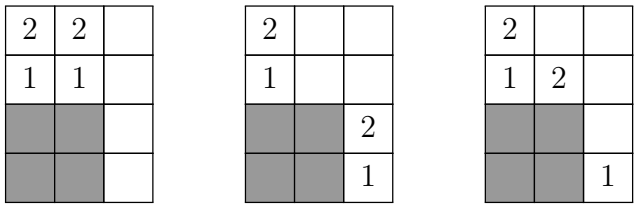
Next, there are three places to place a single box such that the shape is still a chain of skew tableaux.



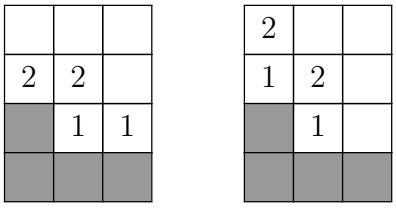
Then, we must fill in two 1s and two 2s. In the first case, there are three Yamanouchi ways to do this



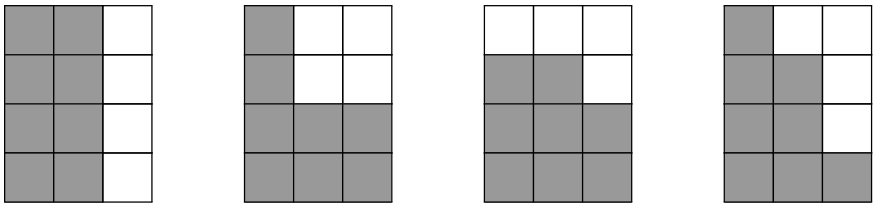
In the second case, there are also three Yamanouchi fillings with shape $(2, 2)$.



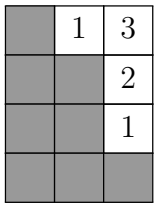
In the third case, there are only two Yamanouchi fillings.



Finally, for the tableaux in the chain, we can distill the above eight cases into four shapes.



Respectively, we have one instance of the first case above, two of the second and third, and three of the fourth. However, we cannot fill the first case with content $(2,1,1)$ at all, since we have two 1s. Additionally, the middle two cases can only be filled by $\begin{array}{|c|c|}\hline 2&3\\\hline 1&1\\\hline\end{array}$ and $\begin{array}{|c|c|c|}\hline 1&2&3\\\hline \multicolumn{2}{c}{}&1\\\cline{3-3}\end{array}$, respectively, neither of which is Yamanouchi. Thus, all valid fillings are of the fourth case, which is filled by:



Thus, there are 3 total valid chains, and $c^{(3,3,3,3)}_{(2,1),(1),(2,2),(2,1,1)} = 3$.

We now introduce integral notation for 0-dimensional products of Schubert classes.

Definition 3.2.8. Let $\sum_{i=1}^{r} |\lambda^{(i)}| = k(n-k)$. Then, we define

$$\int_{\mathrm{Gr}(k,n)} \sigma_{\lambda^{(1)}}\sigma_{\lambda^{(2)}}\cdots\sigma_{\lambda^{(r)}} = C,$$

where C is the coefficient $c^{B}_{\lambda^{(1)},\lambda^{(2)},\ldots,\lambda^{(r)}}$ from Corollary 3.2.6.

Intuitively, what this tells us is that the intersection of the varieties $\Omega_{\lambda^{(i)}}(F_\bullet^{(i)})$ will have C distinct points. This allows us to discuss intersections of Schubert varieties in terms of generators of the cohomology ring, without having to make any particular choice of flags.

Example 3.2.9. In the above problem, rather than saying that the intersection

$$\Omega_{2,1}(F_\bullet^{(1)}) \cap \Omega_1(F_\bullet^{(2)}) \cap \Omega_{2,2}(F_\bullet^{(3)}) \cap \Omega_{2,1,1}(F_\bullet^{(4)})$$

has three points in $\mathrm{Gr}(4,7)$, we can just write

$$\int_{\mathrm{Gr}(4,7)} \sigma_{2,1}\cdot\sigma_1\cdot\sigma_{2,2}\cdot\sigma_{2,1,1} = 3.$$

3.3 Moduli spaces of curves

We will now discus a much broader class of geometric spaces. Specifically, the spaces we will be covering going forward are called *moduli spaces*. Broadly speaking, a moduli space is a geometric space whose points parameterize some class of geometric objects. In fact, the Grassmannian $\mathrm{Gr}(k, n)$ we covered in the previous two sections is an example of a moduli space, as the points in $\mathrm{Gr}(k, n)$ parameterize k-planes in $\mathbb{C}^n$.

Now, however, our geometric objects we will be parameterizing will be complex curves with marked points. The similarities between such spaces and the Grassmannian are extensive: much like the Grassmannian, we will construct its cohomology ring and compute zero-dimensional intersection products in that ring by way of counting a class of combinatorial objects.

Definition 3.3.1. Let $M_{0,n}$ be the moduli space of (complex) genus 0 curves with n marked points. In particular, a curve $C \in M_{0,n}$ is isomorphic to $\mathbb{P}^1$ and consists of the data of n distinct points $p_1, p_2, \ldots, p_n \in \mathbb{P}^1$.

Definition 3.3.2. Then, let $\overline{M}_{0,n}$ be the **Deligne-Mumford compactification** of $M_{0,n}$, the moduli space of *stable* genus 0 curves with n marked points. A boundary curve $C \in \overline{M}_{0,n} \backslash M_{0,n}$ will instead consist of two or more components, each individually isomorphic to $\mathbb{P}^1$, joined together at nodes in a tree structure (to keep the total genus 0), with the n marked points distributed between them. (See Figure 3.1.)

We will often refer to $M_{0,n}$ as the *interior* of $\overline{M}_{0,n}$, and $\overline{M}_{0,n} \backslash M_{0,n}$ as the *boundary*. We use the term **special point** to mean either a marked point or a node. The word *stable* in the above definition means that each component has at least three special points.

Remark 3.3.3. The stability condition above is required because projective transformations on $\mathbb{P}^1$ are determined by three points. That is, for two sets of three distinct points $P = (p_1, p_2, p_3)$ and $Q = (q_1, q_2, q_3)$, there is a unique automorphism $\alpha : \mathbb{P}^1 \to \mathbb{P}^1$ such that $\alpha(p_i) = q_i$ for $i = 1, 2, 3$. Thus, stable curves have no nontrivial automorphisms.

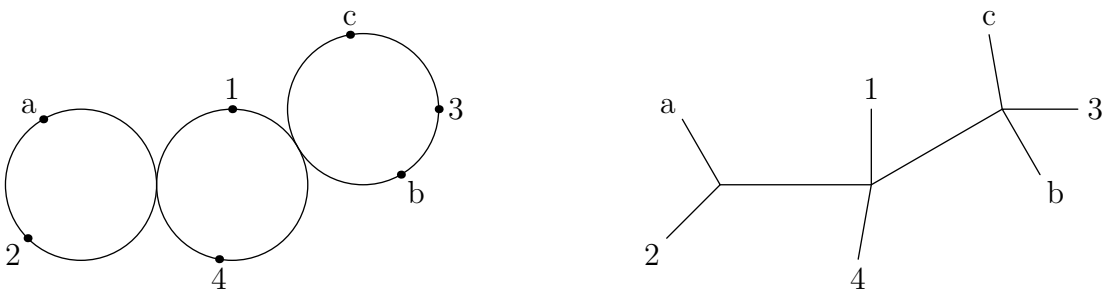


Figure 3.1: An element of $\overline{M}_{0,n+3}$ for $n = 4$, along with its dual tree.

For more details on the construction and properties of $\overline{M}_{0,n}$, we refer the reader to [5] or [18, Ch. 1].

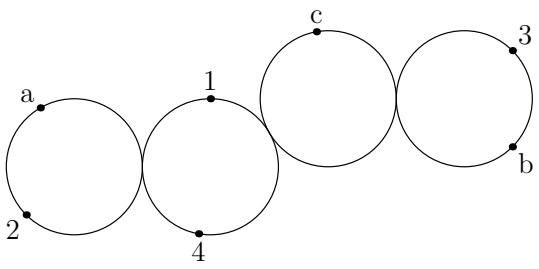
Remark 3.3.4. Throughout this work, as in [10], we label our marked points by the alphabet $\{a, b, c, 1, 2, \ldots, n\}$, and thus will henceforth refer to $\overline{M}_{0,n}$ as $\overline{M}_{0,n+3}$ instead. We will order this alphabet by $a < b < c < 1 < 2 < \cdots < n$.

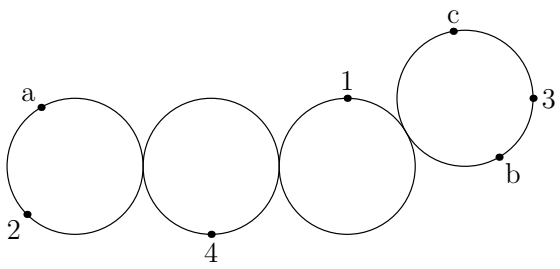
Given a stable curve $C \in \overline{M}_{0,n+3}$, we may also consider its **dual tree**. To form the dual tree to a curve, create a vertex for each component of C, as well as one for every marked point. Then, for each marked point, connect its vertex to the vertex corresponding to the component it is on. For each node, connect the vertices corresponding to the two components that intersect at that node.

Example 3.3.5. On the left of Figure 3.1 is an example of a boundary curve in $\overline{M}_{0,7}$ (so $n = 4$). To its right, we construct its dual tree.

The boundary $\overline{M}_{0,n+3} \backslash M_{0,n+3}$ can be divided into different **boundary strata**. A particular stratum consists of all curves in $\overline{M}_{0,n+3}$ that share the same dual tree. The interior $M_{0,n+3}$ is a single stratum, as all interior curves have the star graph as their dual tree.

Example 3.3.6. We now illustrate collisions between points. Let C be the curve on the left of Figure 3.1. If we allow the marked points b and 3 to collide, they 'bubble off' into a new component, forming the curve shown below on the left. Alternatively, if we collide the marked point 4 with the node to the component with marked points a and 2, the node itself bubbles off, forming a new component containing the point 4. The resulting curve is shown below to the right.



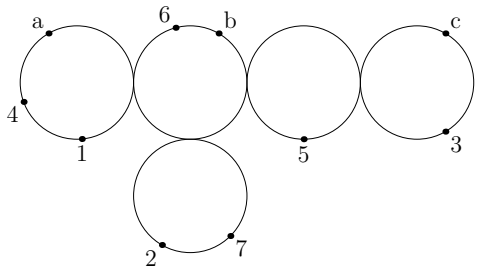


Definition 3.3.7. Define the n**th forgetting map** $\pi_n : \overline{M}_{0,n+3} \to \overline{M}_{0,n+2}$ as the map that forgets the marked point n. If this results in a component having less than three special points, we must *stabilize* the curve. To do this, if forgetting n results in a component with two nodes and no marked points, replace that component with a node. If this results in a component with one node and one marked point, replace it with that marked point.

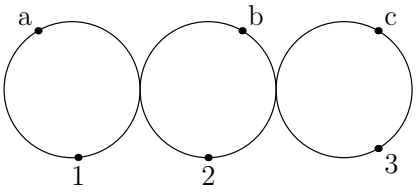
Definition 3.3.8. Define the n**th section map** $\sigma_n : \overline{M}_{0,n+3} \to \overline{M}_{0,n+4}$ as the section map that associates to a curve $C \in \overline{M}_{0,n+3}$ the curve obtained by replacing the marked point n with a node, connected to a new component containing the marked points n and $n+1$.

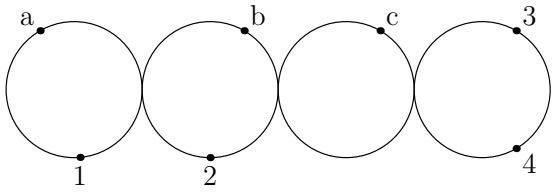
The forgetting and section maps are one-sided inverses: $\pi_{n+1} \circ \sigma_n = \mathrm{id}_{\overline{M}_{0,n+3}}$. We may compose these maps as desired: $\pi_{n-2} \circ \pi_{n-1} \circ \pi_n : \overline{M}_{0,n+3} \to \overline{M}_{0,n}$, and so forth.

Example 3.3.9. Let C be the curve in $\overline{M}_{0,10}$ shown below.



Then, $\pi_4 \circ \pi_5 \circ \pi_6 \circ \pi_7(C)$ is the curve below on the left, as when we forget the marked points $4, 5, 6, 7$, we must stabilize by collapsing the components of C that contained the 5 and 7. Then, $\sigma_3 \circ \pi_4 \circ \pi_5 \circ \pi_6 \circ \pi_7(C)$ is the curve to the right, where we take the previous curve and bubble off the 3 into a new component containing 3 and 4.





3.4 Psi and omega classes and the Kapranov embedding

We next define certain classes in the cohomology of $\overline{M}_{0,n+3}$, as well as the corresponding map to projective space. For full details, see [15], where Kapranov first defined this map.

Definition 3.4.1. First, the ith cotangent line bundle $\mathbb{L}_i$ is the line bundle whose fiber over $C \in \overline{M}_{0,n+3}$ is the cotangent space of C at the marked point i. Then, the i**th psi class** ψ_i is the first Chern class of $\mathbb{L}_i$. In other words, $\psi_i = c_1(\mathbb{L}_i)$.

The corresponding map $|\psi_i| : \overline{M}_{0,n+3} \to \mathbb{P}^n$ (called the Kapranov morphism) is defined as follows. On the interior $M_{0,n+3}$,

$$|\psi_i|(C) = \left[\frac{p_a - p_b}{p_i - p_b} : \frac{p_a - p_c}{p_i - p_c} : \frac{p_a - p_1}{p_i - p_1} : \cdots : \frac{p_a - p_{i-1}}{p_i - p_{i-1}} : \frac{p_a - p_{i+1}}{p_i - p_{i+1}} : \cdots : \frac{p_a - p_n}{p_i - p_n}\right],$$

where p_j is the coordinate of the marked point j. If we set $p_a = 0$ and $p_i = \infty$, this reduces to

$$|\psi_i|(C) = [p_b : p_c : p_1 : \cdots : p_{i-1} : p_{i+1} : \cdots : p_n].$$

Next, for a boundary curve $C \in \overline{M}_{0,n+3} \backslash M_{0,n+3}$, let C' be the component of C containing the point i. Then, consider the coordinates of each special point of C'. For each marked point j on C', we let $q_j = p_j$ be that point's coordinate in C' as normal. However, for $j \notin C'$, we define q_j to be the coordinate of the node connecting the branch of the dual tree containing j to C'. Then,

$$|\psi_i|(C) = [q_b : q_c : q_1 : \cdots : q_{i-1} : q_{i+1} : \cdots : q_n].$$

The second type of cohomology classes we need to define are the *omega classes*, which relate to psi classes.

Definition 3.4.2. The i**th omega class** ω_i is defined as the pullback of ψ_i under the forgetting maps that forget the marked points $i+1, \ldots, n$.

Then, the corresponding map to projective space $|\omega_i| : \overline{M}_{0,n+3} \to \mathbb{P}^i$ is given by $|\omega_i| = |\psi_i| \circ \pi_{i+1} \circ \cdots \circ \pi_n$.

Example 3.4.3. Let C be the curve in $\overline{M}_{0,7}$ seen in Figure 3.1. Suppose that on the central component C', the right node has coordinate s, and the point 4 has coordinate t. Then, $|\psi_1|(C) = [s : s : 0 : s : t]$ and $|\omega_1|(C) = [s : s]$.

We will now combine these Kapranov morphisms together to form two different maps from $\overline{M}_{0,n+3}$ into products of projective spaces.

Definition 3.4.4. The **total Kapranov map** $\Psi_n : \overline{M}_{0,n+3} \to \mathbb{P}^n \times \mathbb{P}^n \times \cdots \times \mathbb{P}^n$ is the map given by

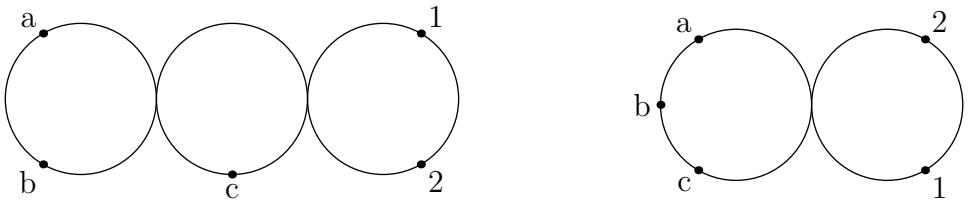
$$\Psi_n(C) = (|\psi_1|(C), |\psi_2|(C), \ldots, |\psi_n|(C)),$$

and the **iterated Kapranov map** $\Omega_n : \overline{M}_{0,n+3} \hookrightarrow \mathbb{P}^1 \times \mathbb{P}^2 \times \cdots \times \mathbb{P}^n$ is the map given by

$$\Omega_n(C) = (|\omega_1|(C), |\omega_2|(C), \ldots, |\omega_n|(C)).$$

Unfortunately, Ψ_n is not an embedding, as it can obscure information about any components of C that contain no marked points. However, Ω_n is an embedding, as paradoxically, strategically forgetting information allows us to see 'more' of the structure of the curve. This allows us to view $\overline{M}_{0,n}$ as a projective variety. We illustrate this difference in Example 3.4.5. The iterated Kapranov map was studied in more detail in [16, 20].

Example 3.4.5. Let C_1 and C_2 be the two curves in $\overline{M}_{0,5}$ shown below, respectively.



We will compute $\Psi_2(C_1)$, $\Psi_2(C_2)$, $\Omega_2(C_1)$, and $\Omega_2(C_2)$.

First, we will do the total Kapranov maps. Since a, b, c all lie across the same node from 1 and 2, we have $\Psi_2(C_1) = ([0:0:1],[0:0:1])$ and $\Psi_2(C_2) = ([0:0:1],[0:0:1])$. As one can see, C_1 and C_2 are mapped to the same point in $\mathbb{P}^2 \times \mathbb{P}^2$, so the total Kapranov map is not an embedding.

On the other hand, when we compute $|\omega_1|$, we must first forget the marked point 2 from both curves, which will collapse the rightmost component of each. Then, we have $\Omega_2(C_1) = ([0:1],[0:0:1])$ and $\Omega_2(C_2) = ([1:s],[0:0:1])$. So, the iterated Kapranov map can distinguish between C_1 and C_2.

When we took intersection products over the Grassmannian, our classes were parameterized by partitions. For the moduli space of curves, we will instead use the closely related objects called compositions.

Definition 3.4.6. Let $\underline{k} = (k_1, k_2, \ldots, k_n)$ be a n-tuple of nonnegative integers. Then $\underline{k}$ is a **composition of** n if $k_1 + k_2 + \cdots + k_n = n$.

Definition 3.4.7. Let $\underline{k}$ be a composition of n. Then, consider the intersection between the image of Ψ_n (resp. Ω_n) with n hyperplanes, where we choose k_i general hyperplanes from the ith component of the product. Then, the **multidegree** of Ψ_n (resp. Ω_n) is the number of points in this intersection. This is denoted as $\deg_{\underline{k}}(\Psi_n)$ (resp. $\deg_{\underline{k}}(\Omega_n)$).

It is known (see [5], for example) that when $\sum k_i = n$,

$$\deg_{\underline{k}}(\Psi_n) = \int_{\overline{M}_{0,n+3}} \psi^{\underline{k}} = \binom{n}{k_1, k_2, \ldots, k_n},$$

where $\psi^{\underline{k}}$ denotes the product $\prod_i \psi_i^{k_i}$.

A similar result for omega classes also exists. It is shown in [6] that when $\sum k_i = n$,

$$\deg_{\underline{k}}(\Omega_n) = \int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ k_1, k_2, \ldots, k_n \end{matrix} \right\rangle.$$

The coefficients in the third part of the above equality are the *asymmetric multinomial coefficients*, which we define in the Section 5.1.

3.5 Hyperplane degenerations and multidegrees

We will now compute products of psi classes explicitly. To do this, we need to make use of one more cohomology class.

Definition 3.5.1. Define the class of a **boundary divisor** $D(A|A^c)$ as the class of the codimension 1 stratum where the marked points A are separated from the points A^c by a node, where $A \subset \{a, b, c, 1, 2, \ldots, n\}$, A^c is the complement of A, and $2 \leqslant |A| \leqslant n+1$.

We can express psi classes in terms of boundary divisors, following the below formula, where j and k are fixed and arbitrary, and $*$ varies over all nonempty subsets of $\{a, b, c, 1, 2, \ldots, n\} \backslash \{i, j, k\}$.

$$\psi_i = \sum_{*} D\big(i, * \big| j, k, *^c\big) \tag{3.1}$$

However, using the above formula to expand products of ψ classes does not in general produce distinct terms.

Example 3.5.2. Consider the product $\psi_1\psi_2$. From the previous discussion, we know the answer is

$$\int_{\overline{M}_{0,5}} \psi_1\psi_2 = \binom{2}{1,1} = 2.$$

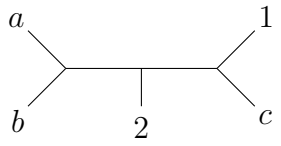
To compute directly, we first use Equation 3.1 to expand ψ_1 using $j = a$ and $k = b$:

$$\begin{aligned}\psi_1\psi_2 &= \big(D(ab|c12) + D(abc|12) + D(ab2|c1)\big)\psi_2 \\ &= D(ab|c12)\psi_2 + D(abc|12)\psi_2 + D(ab2|c1)\psi_2.\end{aligned}$$

Then, we expand the first two copies of ψ_2 using Equation 3.1 and $j = c$ and $k = 1$, and the third copy using $j = a$ and $k = b$:

$$\begin{aligned}\psi_1\psi_2 =& D(ab|c12)\big(D(c1|ab2) + D(ac1|b2) + D(bc1|a2)\big) \\ &+ D(abc|12)\big(D(c1|ab2) + D(ac1|b2) + D(bc1|a2)\big) \\ &+ D(ab2|c1)\big(D(ab|c12) + D(abc|12) + D(ab1|c2)\big) \\ =& \big([X_T] + 0 + 0\big) + (0 + 0 + 0) + \big([X_T] + 0 + 0\big) \\ =& 2\,[X_T]\,,\end{aligned}$$

where $[X_T]$ is the class of the 0-dimensional stratum with dual tree T below.



So, we get twice the class of a point, as desired. However, we obtained the same boundary tree twice.

So in general, using Equation 3.1 for products of ψ classes results in sums with multiplicity. An alternative method of computing ψ products with multiplicity-free formulas is given by [10]. Their method makes use of degenerating hyperplanes in the image of the Kapranov map. We summarize their methods in the ψ class case, and obverse that the process for ω classes is very similar.

First, define a class of hyperplanes $H_i^{\psi}(t)$ by

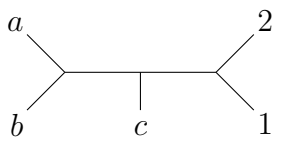
$$H_i^{\psi}(t) = \mathbb{V}(p_b + tp_c + t^2p_1 + \cdots + t^ip_{i-1} + t^{i+1}p_{i+1} + \cdots + t^np_n) \subseteq \mathbb{P}^n.$$

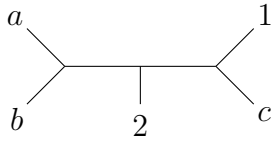
Then, take a collection of hyperplanes where we take k_i from the ith term of the product $(\mathbb{P}^i)^n$, each with their own parameter t. Then, if we take the limit as these parameters t go to 0 in a particular order, we obtain a set of $\binom{n}{\underline{k}}$ distinct points corresponding to boundary trees. We illustrate this process with an example.

Example 3.5.3 (Based on Example 1.7 from [10])**.** We again consider the product $\psi_1\psi_2$ in $\overline{M}_{0,5}$. Recall that Ψ_2 is a map $\overline{M}_{0,5} \to \mathbb{P}^2 \times \mathbb{P}^2$, whose image we coordinatize by $[x_b : x_c : x_2] \times [y_b : y_c : y_1]$. Our two hyperplanes are $x_b + tx_c + t^2x_2 = 0$ and $y_b + sy_c + s^2y_1 = 0$. We first take the limit $t \to 0$, and get the hyperplane $x_b = 0$. Pulling this hyperplane back along Ψ_2 gives us the union of boundary divisors where the marked points a and b are separated from the point 1 by a node. This is the union

$$D(ab|c12) \bigcup D(ab2|c1) \bigcup D(abc|12).$$

We then map this forward via $|\psi_2|$ to the second copy of $\mathbb{P}^2$. The images of these three divisors are given by $y_b = 0$, $y_c = y_1$, and $y_b = y_c = 0$, respectively. Generically, the hyperplane $y_b + sy_c + s^2y_1 = 0$ intersects the first two divisors. One can then show that these two intersection points can be expressed as $[0 : -s : 1]$ and $[-(s^2 + s) : 1 : 1]$, respectively. Finally, when we take the limit as $s \to 0$, we get the points $[0 : 0 : 1]$ and $[0 : 1 : 1]$. Combining this with the ψ_1 part, we get two points in $\mathbb{P}^2 \times \mathbb{P}^2$, with coordinates $[0 : 0 : 1] \times [0 : 0 : 1]$ and $[0 : 1 : 0] \times [0 : 1 : 1]$. So, if T_1 and T_2 are the trees below respectively, then we get that $\psi_1\psi_2 = [X_{T_1}] + [X_{T_2}]$.





More generally, [10] uses this process to prove that in $A^0(\overline{M}_{0,n+3})$,

$$\psi^{\underline{k}} = \sum_{T \in \mathrm{Slide}^{\omega}(\underline{k})} [X_T],$$

where $\mathrm{Slide}^{\omega}(\underline{k})$ is a set of trivalent trees we will introduce in Chapter 5.

Chapter 4

Counting L-tableaux using a generalized RSK

We now return to the question posed by Farkas and Lian mentioned in the introduction: Can we prove Theorem 1.1.3 using combinatorial methods? We will use the tools built up in the previous two chapters to do this. We will show that the L-tableaux with parameters (g, r, d) enumerate the integrals $L_{g,r,d}$ in the Grassmannian, starting from equation (1.1). We will then show that the L-tableaux are enumerated by $(r+1)^g$.

The results in this Chapter first appeared in [12].

4.1 The L-tableaux

Proposition 3.2.5, combined with the fact that the integral in equation (1.1) is the coefficient of $s_{(d-r)^{r+1}}$ in the corresponding product (1.2), shows that

$$L_{g,r,d} = \sum_{\alpha_0+\cdots+\alpha_r+rg=(r+1)(d-r)} c^{(d-r)^{r+1}}_{(1^r),(1^r),\ldots,(1^r),(\alpha_0),\ldots,(\alpha_r)}$$

where the subscripts on the coefficient contain g copies of (1^r). This summation is therefore the number of ways to form a transposed semistandard Young tableau using each of the numbers $1, 2, \ldots, g$ exactly r times, and then extend it to fill the rest of the $(r+1) \times (d-r)$ grid with a semistandard Young tableau using the numbers $0, 1, \ldots, r$ in some varying amounts $\alpha_0, \ldots, \alpha_r$ each.

We restate Definition 1.1.2, this in terms of our new notation.

Definition 4.1.1. An **L-tableau with parameters** (g, r, d) is a way of filling an $(r+1) \times (d-r)$ rectangular grid with:

- (The 'red' tableau.) A transposed semistandard Young tableau having exactly r copies of each of the numbers $1, 2, \ldots, g$. That is, its content is $(r^g) = (r, r, r, \ldots, r)$.

- (The 'blue' tableau.) A semistandard Young tableau on the remaining skew shape of boxes, with values from $\{0, 1, \ldots, r\}$.

Example 4.1.2. The following is an L-tableau with parameters $(4, 3, 9)$. We write the "red" numbers as black font with a red shaded background for clarity.

2	4	1	3	3	3
1	3	4	1	2	2
1	2	3	0	1	1
1	2	3	4	0	0

Our discussion thus far, starting from Equation (1.1), has shown:

Proposition 4.1.3. *The number of L-tableau with parameters (g, r, d) is equal to $L_{g,r,d}$ whenever either $d \geqslant rg + r$, $r = 1$, or $d = r + \frac{rg}{r+1}$, .*

We now show that we can "truncate" by removing some of the right-hand columns of the grid to reduce to a simpler case.

Lemma 4.1.4 (Truncation)**.** *For any g, r, d with $d \geqslant g + r$, the number of L-tableaux with parameters (g, r, d) is equal to the number of L-tableaux with parameters $(g, r, g + r)$.*

Proof. Suppose $d \geqslant g + r$. Notice that, since it is transposed semistandard, the red tableau has width at most g, since its bottom row is strictly increasing from left to right and uses only the numbers $1, 2, \ldots, g$. Therefore, any column to the right of the g-th column is filled entirely with blue numbers, which strictly increase up the columns using the numbers $0, 1, 2, \ldots, r$, necessarily exactly once since the columns have height $r + 1$.

It follows that there is only one way to fill each of the columns to the right of column g, and these columns therefore do not contribute to the enumeration. We therefore may remove the last $d - r - g$ columns and find that the number of L-tableaux with parameters (g, r, d) equals the number with parameters $(g, r, g + r)$. □

Lemma 4.1.4 tells us that in order to understand $L_{g,r,d}$ for $d \geqslant g + r$, it suffices to study the case $d = g + r$. We will restrict to this case throughout the remainder of this chapter.

Remark 4.1.5. When $d = g + r$, the rectangle containing the L-tableaux is size $(r+1)g = rg + g$. The red tableau has size rg and so the blue tableau has size g.

4.2 Enumeration by $(r+1)^g$

In this section we prove Theorem 1.1.3. We first define the following sets of tableaux.

Definition 4.2.1. Let $\mathrm{TrSSYT}(g, r)$ be the set of all transposed semistandard Young tableaux of content $(r^g) = (r, r, \ldots, r)$ and height $\leqslant r + 1$.

Note that $\mathrm{TrSSYT}(g, r)$ is the set of all possible 'red' tableaux in Definition 4.1.1. We will refer to them as **red tableaux** throughout this work.

Definition 4.2.2. Define a 180°**-rotated SYT** to be the result of rotating a standard Young tableaux 180° in the plane. We write $\mathrm{SYT}^{180^\circ}(g, r)$ for the set of all 180°-rotated SYT of size g and height at most $r + 1$.

We informally call such tableaux **purple tableaux**, as they will be used as an intermediate object relating the red and blue tableaux of Definition 4.1.1.

Example 4.2.3. Below is an example of a purple tableau in $\mathrm{SYT}^{180^\circ}(7, 3)$.

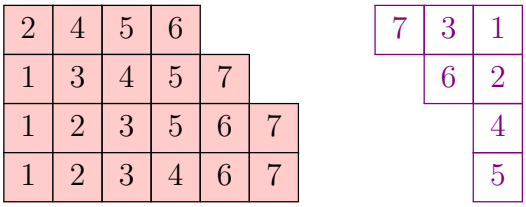
7	3	1
	6	2
		4
		5

Given a red tableau, note that each number $1, \ldots, g$ occurs once in every row except one. Relatedly, a purple tableau in the position of the blue tableau will have each number $1, \ldots, g$ in exactly one row. This leads us to define a bijection between the two as follows.

Definition 4.2.4 (Red to purple bijection)**.** Let $R \in \mathrm{TrSSYT}(g, r)$ be a red tableau. We define a 180°-rotated tableau $\varphi(R)$ in the upper right corner of a rectangle by the following iterative

process. We add boxes labeled $1, 2, \ldots, g$ in order, where on the ith step we place a box labeled i as far to the right as possible in the unique row that does **not** contain an i in R.

Example 4.2.5. If R is the tableau at left below, $\varphi(R)$ is shown at right below.



We now show φ is a bijection. We note that a generalized version of this map was shown to be a bijection in [24], but we include a direct proof here for the special case that we are considering, for the reader's convenience.

Lemma 4.2.6. *The map φ is a bijection from $\mathrm{TrSSYT}(g, r)$ to $\mathrm{SYT}^{180^\circ}(g, r)$ for all g, r. Moreover, for any $R \in \mathrm{TrSSYT}(g, r)$, the shapes of R and $\varphi(R)$ are complementary in an $(r+1) \times g$ rectangle.*

Proof. We show both statements by induction on g. For $g = 1$, the tableau R must be a column of 1's of height r, and $\varphi(R)$ is a single 1 in the top row. They are clearly complementary in an $(r+1) \times 1$ rectangle (column).

For $g > 1$, assume the statements hold for $g-1$ and let $R \in \mathrm{TrSSYT}(g, r)$. Consider the tableau R' formed by removing the vertical strip of g's from R. Let $T' = \varphi(R')$. Then T' and R' are complementary in an $(r+1) \times (g-1)$ rectangle by the inductive hypothesis.

By shifting T' one unit to the right, we form an empty vertical strip V of size $r+1$ between the two tableaux. Then all r of the g's in R must lie in this strip, and in fact the one remaining square x must be at the top of some column of V. Then, the square x is precisely the one that we label g to form $T = \varphi(R)$ starting from T', by Definition 4.2.4. Since the entries immediately above and to the right of x will have entries smaller than g (or x is on the right hand or top edge of the rectangle), this construction forms a 180°-rotated SYT T, so φ is well-defined and the resulting pair $(R, \varphi(R))$ is complementary.

Finally, note that by the induction hypothesis, φ is a bijection for $g-1$, and the possible squares we can add to T' to form g are precisely in bijection with the possible sub-strips of g's of the vertical strip V that we may add to R' to form R. Thus φ is a bijection for size g as well. $\square$

We now make precise the notion of a "blue tableau" (see Definition 4.1.1).

Definition 4.2.7. Define a 180°**-rotated semistandard tableau**, or **blue tableau** (with parameters r, g), to be a filling of a 180°-rotated Young diagram of size g with numbers $0, 1, 2, \ldots, r$ such that the rows are weakly increasing from left to right and columns are strictly increasing from bottom to top.

Example 4.2.8. Below is an example of a blue tableau. It has the same shape as the purple tableau above in Example 4.2.5.

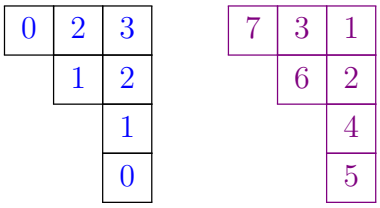
0	2	3
	1	2
		1
		0

Lemma 4.2.9. *The pairs of blue and purple tableaux of the same shape correspond to $(r+1)$-ary sequences of length g bijectively, via inverting the entries of the blue tableau (that is, replacing each entry i by $r-i$), rotating both 180°, and applying RSK.*

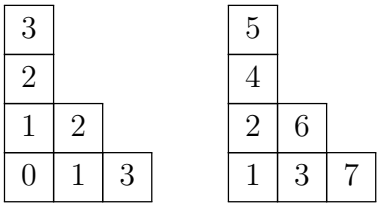
Proof. Given a blue tableau S and purple tableau T of the same shape, rotate both by 180 degrees to form S^{180} and T^{180}. Then, T^{180} is a standard tableau, which we call Q. We also form a semistandard tableau P out of S^{180} by inverting its entries; that is, we replace each i in S^{180} with $r-i$ in P.

Then, (P, Q) is a pair in $B(r, g)$ (see Corollary 2.2.2), so by the RSK bijection on words, we have a bijection between these pairs (P, Q) and $A(r, g)$, which is precisely the set of $(r+1)$-ary sequences of length g. $\square$

Example 4.2.10. Consider the pair of blue and purple tableaux below.



The corresponding pair (P, Q) is as follows.



Then, via RSK, this pair corresponds to the $(r+1)$-ary sequence $3, 2, 2, 1, 0, 1, 3$ where $r = 3$.

We now have the tools to produce a bijection between L-tableaux and $(r+1)$-ary sequences.

Proposition 4.2.11. *The L-tableaux with parameters $(g, r, g+r)$ are in bijection with the $(r+1)$-ary sequences of length g (with letters from the alphabet $\{0, 1, 2, \ldots, r\}$).*

Proof. Each such L-tableau consists of a red tableau and a blue tableau. The bijection follows from combining Lemma 4.2.6 with Lemma 4.2.9, which provide bijections between red tableaux with purple tableaux, and between pairs of blue and purple tableaux with $(r+1)$-ary sequences of length g, respectively. □

Example 4.2.12. Below is an L-tableau with parameters $(7, 3, 10)$. From our previous examples, we see that it corresponds to the $(r+1)$-ary sequence $3, 2, 2, 1, 0, 1, 3$.

2	4	5	6	0	2	3
1	3	4	5	7	1	2
1	2	3	5	6	7	1
1	2	3	4	6	7	0

There are $(r+1)^g$ sequences of length g in the alphabet $0, 1, 2, \ldots, r$. Combining Proposition 4.2.11 with truncation (Lemma 4.1.4), we get as a corollary Theorem 1.1.3.

Theorem 1.1.3. The number of L-tableaux with parameters (g, r, d) is $(r+1)^g$ for all $d \geqslant r + g$.

4.3 L'-tableaux and enumeration by 2^g

In this section, we give a tableau interpretation of $L'_{g,d,k}$, starting from equation (1.3), and show that these tableaux are enumerated by 2^g to prove Theorem 1.1.4.

Recall that equation (1.3) states that if $k + g \leqslant 2d + 1$ and $2 \leqslant k \leqslant d$,

$$L'_{g,d,k} = \int_{\mathrm{Gr}(2,d+1)} \sigma_1^g \sigma_{k-1} \left(\sum_{i+j=2d-g-k-1} \sigma_i \sigma_j \right) - \int_{\mathrm{Gr}(2,d)} \sigma_1^g \sigma_{k-2} \left(\sum_{i+j=2d-g-k-2} \sigma_i \sigma_j \right).$$

We first give an interpretation of the left hand integral in the equation above.

Definition 4.3.1. A **positive L'-tableau with parameters** (g, d, k) is a filling of a $2 \times (d-1)$ grid with:

- A standard Young tableau of size g in the lower left corner (shaded red),
- A shading of the $k-1$ rightmost boxes in the top row (gray),
- A skew semistandard Young tableau of the letters $\{0, 1\}$ on the remaining squares (blue).

By rearranging so that we think of the σ_{k-1} as last in each product, and applying Proposition 3.2.5, we see that the positive term in equation (1.3) is equal to the number of positive L'-tableaux.

Example 4.3.2. Here is an example of a positive L'-tableau with parameters $(3, 7, 4)$.

3	0	1			
1	2	0	0	1	1

The second term in (1.3), which we are subtracting, is similarly given by a set of smaller tableaux that we call *negative* tableaux.

Definition 4.3.3. A **negative L'-tableau with parameters** (g, d, k) is a filling of a $2 \times (d-2)$ grid with:

- A standard Young tableau of size g in the lower left corner (shaded red),
- A shading of the $k-2$ rightmost boxes in the top row (gray),
- A skew semistandard Young tableau of the letters $\{0,1\}$ on the remaining squares (blue).

Example 4.3.4. Here is an example of a negative L'-tableau with parameters $(3,7,4)$.

3	0	1		
1	2	0	0	1

Notice that there exist positive L'-tableaux if and only if $(k-1)+g \leqslant 2(d-1)$, which is slightly stronger than the given condition $k+g \leqslant 2d+1$. That is, if $k+g=2d$ or $k+g=2d+1$ we have $L'_{g,d,k}=0$, so we restrict our attention to the case that $k+g \leqslant 2d-1$.

Definition 4.3.5. For fixed g,d,k, write L'_+ and L'_- for the set of positive and negative L' tableaux respectively of type (g,d,k). Also write $\eta : L'_- \to L'_+$ for the map that takes a negative tableau T and creates a positive tableau by adding a blue 1 to the end of the bottom row of T and a gray box to the end of the top row of T.

Our above analysis shows that

$$L'_{g,d,k} = |L'_+| - |L'_-|,$$

and we analyze this difference combinatorially. The definitions above directly show that η is a well-defined injective map, and so

$$L'_{g,d,k} = |L'_+ \backslash \eta(L'_-)|. \tag{4.1}$$

The following proposition characterizes the image $\eta(L'_-)$.

Proposition 4.3.6. *A positive tableau T is equal to $\eta(S)$ for some negative tableaux S if and only if the bottom row of T contains a blue* 1.

Proof. By definition of η, any tableau in $\eta(L'\_)$ has a blue 1 on the bottom right. Conversely, if the bottom row of T contains a blue 1, then by semistandardness of the blue tableau, the bottom-rightmost entry is a blue 1 as well, and removing the last column of T yields a negative tableau S for which $\eta(S) = T$. □

As a consequence of this and equation (4.1), we obtain the following combinatorial interpretation of $L'_{g,d,k}$.

Corollary 4.3.7. The quantity $L'_{g,d,k}$ is equal to the number of positive L'-tableaux with parameters (g, d, k) for which the bottom row contains no blue 1 (and hence the only blue numbers in the bottom row are 0's).

For sufficiently large d, we can simplify this characterization even further, proving Theorem 1.1.4.

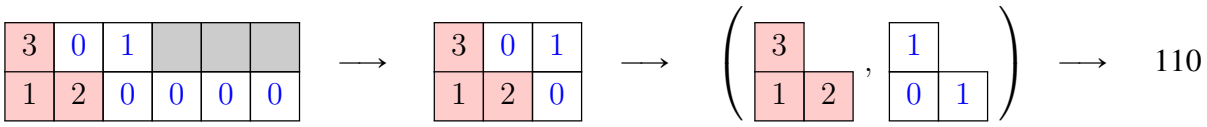
Theorem 1.1.4. If $d \geqslant g + k$, we have $L'_{g,d,k} = 2^g$.

Proof. Suppose $d \geqslant g + k$. Then $d - 1 \geqslant g + (k - 1)$, so the red and gray boxes of any positive L'-tableau with parameters (g, d, k) cannot share a column. In particular, for any positive tableau T that has no blue 1 in the bottom row, there are all blue 0's under the gray squares, and moreover any remaining columns to the right of the red tableau are uniquely determined (having one 0 and one 1) as well. Thus the data determining T is entirely contained in its first g columns, which consists of a red standard tableau Q, and a binary tableau P of the same shape as Q, where P is obtained by rotating the blue numbers in these columns 180° and replacing all 0's with 1's and 1's with 0's.

By the RSK correspondence, these pairs (P, Q) are precisely in bijection with the binary sequences of length g, and so we have that $L'_{g,d,k} = 2^g$ as desired. □

Example 4.3.8. The tableau below at left is a positive L'-tableau with parameters $(3, 7, 4)$ that is not the image of η. Since $g = 3$, we restrict our attention to the first three columns (second image below), then consider the associated pair of tableaux of the same shape by rotating the blue

tableaux and inverting the labels. Finally, this pair corresponds under RSK to a unique length 3 binary sequence.



Chapter 5

Combinatorics of asymmetric multinomial coefficients and slide trees

5.1 Asymmetric multinomial coefficients

The asymmetric multinomial coefficients were originally defined in [6], whose definition we restate here, using modified notation that matches the indexing used in [11].

Definition 5.1.1. Let $\underline{k}$ be a composition of n. Let k_i be the rightmost 0 in $\underline{k}$ (we set $i = 0$ if there are no zeroes in $\underline{k}$), and let $j > i$ be a positive integer. Define $\underline{k}^{(j)}$ to be the composition of $n-1$ formed by decreasing k_j by 1 and then removing the rightmost 0 (which is either in position j or i) from the resulting tuple. Then, the **asymmetric multinomial coefficients** $\left\langle {n \atop \underline{k}} \right\rangle$ are defined by $\left\langle {1 \atop 1} \right\rangle = 1$ and the recursion

$$\left\langle {n \atop \underline{k}} \right\rangle = \sum_{j=i+1}^{n} \left\langle {n-1 \atop \underline{k}^{(j)}} \right\rangle. \tag{5.1}$$

Notably, if $\underline{k} = (1, 1, \ldots, 1)$, then $\left\langle {n \atop \underline{k}} \right\rangle = n!$, and if $\underline{k} = (0, \ldots, 0, n)$, then $\left\langle {n \atop \underline{k}} \right\rangle = 1$.

Example 5.1.2. We compute the coefficient $\left\langle {4 \atop 1,0,2,1} \right\rangle$:

$$\begin{aligned} \left\langle {4 \atop 1,0,2,1} \right\rangle &= \left\langle {3 \atop 1,1,1} \right\rangle + \left\langle {3 \atop 1,0,2} \right\rangle \\ &= 3! + \left\langle {2 \atop 1,1} \right\rangle \\ &= 3! + 2! = 8. \end{aligned}$$

Example 5.1.3. We compute the coefficient $\left\langle \begin{smallmatrix} 4 \\ 0,1,2,1 \end{smallmatrix} \right\rangle$:

$$\begin{aligned}\left\langle \begin{matrix} 4 \\ 0,1,2,1 \end{matrix} \right\rangle &= \left\langle \begin{matrix} 3 \\ 0,2,1 \end{matrix} \right\rangle + \left\langle \begin{matrix} 3 \\ 1,1,1 \end{matrix} \right\rangle + \left\langle \begin{matrix} 3 \\ 0,1,2 \end{matrix} \right\rangle \\ &= \left\langle \begin{matrix} 2 \\ 1,1 \end{matrix} \right\rangle + \left\langle \begin{matrix} 2 \\ 0,2 \end{matrix} \right\rangle + 3! + \left\langle \begin{matrix} 2 \\ 0,2 \end{matrix} \right\rangle + \left\langle \begin{matrix} 2 \\ 1,1 \end{matrix} \right\rangle \\ &= 2! + 1 + 3! + 1 + 2! = 12.\end{aligned}$$

Note that in this case, the asymmetric multinomial $\left\langle \begin{smallmatrix} 4 \\ 0,1,2,1 \end{smallmatrix} \right\rangle$ is equal to the corresponding symmetric multinomial coefficient $\binom{n}{0,1,2,1}$. (In fact, this will happen any time that $\underline{k}$ is *right-justified*, that is, where all entries that are zero occur before all non-zero entries.)

Definition 5.1.4. Let $\underline{k} = (k_1, k_2, \ldots, k_n)$ be a composition of n. We say that $\underline{k}$ is **reverse-Catalan** if for all i, $k_{n-i+1} + \cdots + k_{n-1} + k_n \geqslant i$.

Proposition 5.1.5 (Corollary 4.14 from [6])**.** *The coefficient* $\left\langle \begin{smallmatrix} n \\ \underline{k} \end{smallmatrix} \right\rangle \neq 0$ *if and only if* $\underline{k}$ *is reverse Catalan.*

Several different combinatorial objects are counted by the asymmetric multinomial coefficients. The first one that we discuss are a variant of parking functions called *column-restricted parking functions*. First, we define parking functions.

Definition 5.1.6. A **Dyck path** is a lattice path from $(0, n)$ to $(n, 0)$ consisting of n unit-length right steps and n unit-length down steps, that stays weakly above the diagonal $y = n - x$. A **parking function** is a labeling of the down steps of a Dyck path with the numbers $1, 2, \ldots, n$, such that in each column, the labels increase from bottom to top. We call n the **semilength** of a parking function, and denote the set of semilength n parking functions by $\mathrm{PF}(n)$.

We label the columns of a parking function P from left to right. We now restate the definitions of *dominance* and of *column-restricted parking functions* found in [6] and [11].

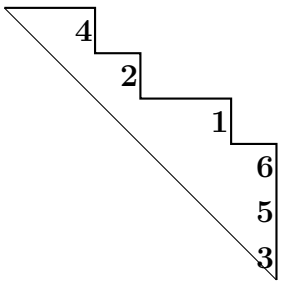
Definition 5.1.7. Let $P \in \mathrm{PF}(n)$ be a parking function. For a label $x \in [n]$ in column j of P, we say x **dominates** a column $i > j$ if x is larger than every entry of i. (This includes empty

columns.) We define the **dominance index** $d_P(x)$ of x to be the number of columns to the right of x dominated by x.

Definition 5.1.8. Let P be a parking function where $d_P(x) < x$ for all x. Then P is a **column-restricted parking function**. We denote the set of such parking functions by $\mathrm{CPF}(n) \subseteq \mathrm{PF}(n)$.

We may also write $\mathrm{PF}(\underline{k})$ or $\mathrm{CPF}(\underline{k})$ to denote the set of parking functions or column-restricted parking functions with k_i down-steps in column i, respectively.

Example 5.1.9. Below is an example of a parking function $P \in \mathrm{PF}(0,1,1,0,1,3)$. The dominance indices of P are $d_P(1) = 0$, $d_P(2) = 2$, $d_P(3) = 0$, $d_P(4) = 3$, $d_P(5) = 0$, and $d_P(6) = 0$. Since $d_P(2) \geqslant 2$, P is not a column-restricted parking function.



Secondly, [11] defines a set of trivalent trees $\mathrm{Tour}(\underline{k})$, called **tournament trees**, which are defined via a process called *lazy tournaments* and are also counted by the asymmetric multinomial coefficients.

Definition 5.1.10 (Definitions 1.2 and 1.4 from [11])**.** Let T be a leaf-labeled trivalent tree. The lazy tournament of T is a labeling of the edges of T computed as follows. Start by labeling each leaf edge (that is, an edge adjacent to a leaf vertex) by the value on the corresponding leaf. Then iterate the following process:

1. **Identify which pair 'face off'.** Among all pairs of labeled edges (i, j) (ordered so that $i < j$) that share a vertex and have a third unlabeled edge E attached to that vertex, choose the pair with the largest value of i.

2. **Determine the winner.** The larger number j is the winner, and the smaller number i is the loser of the match.

3. **Determine which of i or j advances to the next round.** Label E by either i or j as follows:

   (a) If E is adjacent to a labeled edge $u \neq j$ with $u > i$, then label E by i. (We say i *advances*.)

   (b) Otherwise, label E by j. (We say j *advances*.)

We then repeat steps 1–3 until all edges of the tree are labeled.

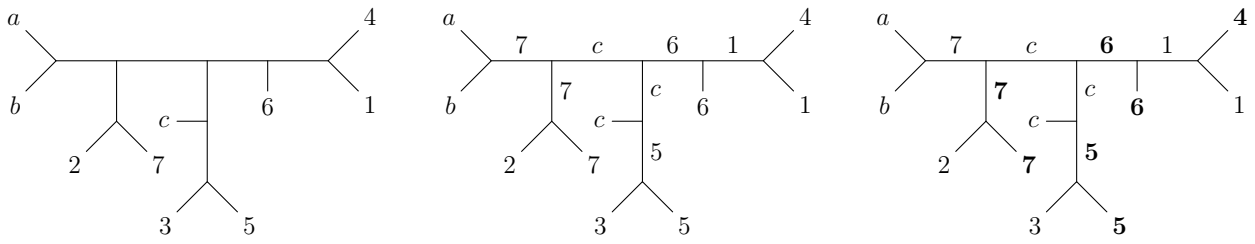
Finally, for any weak composition $\underline{k} = (k_1, \ldots, k_n)$ of n, define $\mathrm{Tour}(\underline{k})$ be the set of trivalent trees whose leaves are labeled by $\{a, b, c, 1, \ldots, n\}$, in which the leaf edges a and b share a vertex, and each label $i \geqslant 1$ wins exactly k_i times in the tournament.

We refer to case 3.(a) above as the *laziness rule*.

Example 5.1.11. Consider the trivalent tree T below on the left. Initially, the three pairs available to face off in the lazy tournament are $(2, 7)$, $(3, 5)$, and $(1, 4)$. Of these, $(3, 5)$ is the one with the largest smaller value. So, 5 is the winner of the first match. In this case, the next label is c, which is smaller than both 3 and 5. So, 5 advances and labels the next edge. Second, 2 and 7 face off. Since there are no labeled edges adjacent to the next unlabeled edge E, 7 advances. Then, 1 and 4 face off, and 6 wins. In this case, since the next label 6 is greater than 1, the laziness rule applies, and 1 advances instead of 4.

In a similar fashion, 1 and 6 face off with 6 advancing, then c and 6 face off and c advances by the laziness rule. Finally, c and 7 face off and 7 advances. The results of this tournament are shown below in the center tree.

Finally, we bold the labels that won each match-up below in the right image. Thus, we see that $T \in \mathrm{Tour}(0, 0, 0, 1, 2, 2, 2)$.



Finally, there is a second class of trivalent trees that are also counted by the asymmetric multinomial coefficients. These are called **slide trees**, denoted $\mathrm{Slide}^{\omega}(\underline{k})$. These were originally defined in [10], and we describe them in detail in the next subsection. Putting this all together, we have the following proposition, where the last three equalities come from [6], [11], and [10], respectively.

Proposition 5.1.12. *Let $\underline{k}$ be a composition of n. Then,*

$$\deg_{\underline{k}}(\Omega_n) = \left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle = |\mathrm{CPF}(\underline{k})| = |\mathrm{Tour}(\underline{k})| = |\mathrm{Slide}^{\omega}(\underline{k})|.$$

The third and fourth objects above are two different sets of trivalent trees. This leads naturally to the following combinatorial question:

Question 5.1.13 (Problem 6.1 from [10])**.** Find a combinatorial bijection between the sets $\mathrm{Tour}(\underline{k})$ and $\mathrm{Slide}^{\omega}(\underline{k})$.

5.2 Slide trees

We now recall the definitions of $\mathrm{Slide}^{\psi}(\underline{k})$ and $\mathrm{Slide}^{\omega}(\underline{k})$ given in [10].

Definition 5.2.1. A tree T is **at least trivalent** or **stable** if every non-leaf vertex has degree at least 3. A tree T is **trivalent** if every non-leaf vertex has degree exactly 3.

Comparing this to our definition of stable curves, we see that a curve C is stable precisely when its dual tree is stable. Let T be a stable tree with leaves labeled by $a < b < c < 1 < 2 < \ldots < n$. For $i \in [n]$, let v_i be the vertex adjacent to the leaf i.

Definition 5.2.2. For a fixed $i \in [n]$, let Br_a be the branch at v_i containing a, and e_a the edge connecting Br_a to v_i. Let m be the minimal leaf label of $T\backslash(\mathrm{Br}_a \cup \{i\})$, and Br_m be the branch at i containing m.

Definition 5.2.3. We define an **i-slide** on a stable tree T as follows. Add a vertex v' in the middle of edge e_a. Reattach the branch Br_m to be rooted at v'. Leave Br_a and the branch that is just i as is. For all other branches of T at v_i, either leave them rooted at v_i, or reattach them to be rooted at v'. Denote the set of trees obtained this way as $\mathrm{slide}_i(T)$.

For an example of i-slides, see Example 3.5 from [10]. Note that in order for the result of an i-slide to be stable, at least one branch (besides i) must remain at v_i. In particular, if $\deg(v_i) = 3$, then $\mathrm{slide}_i(T) = \varnothing$.

Definition 5.2.4 (ψ slide rule)**.** Define $\mathrm{Slide}^{\psi}(\underline{k})$ as the set of all stable trees obtained by the following process.

1. For step $i = 0$, start with $\divideontimes$.
2. For steps $i = 1, \ldots, n$, perform k_i i-slides to all trees obtained in step $i - 1$ in all possible ways.

Definition 5.2.5 (ω slide rule)**.** Define $\mathrm{Slide}^{\omega}(\underline{k})$ as the set of all stable trees obtained by the following process.

1. For step $i = 0$, start with $\curlyvee$.
2. For steps $i = 1, \ldots, n$:
   (a) Consider the set of all trees formed by attaching the leaf i to a non-leaf vertex of a tree obtained in step $i - 1$ in all possible ways.
   (b) Perform k_i i-slides to all trees obtained in (a) in all possible ways.

This definition is motivated by the work in [10] involving hyperplane arrangements that we discussed in Section 3.5.

Alternatively, we define a procedure known as the $\underline{k}$-slide labeling algorithm, which is a method for determining whether a given tree is in a given slide set. Theorem 3.14 from [10] states that a tree T is in $\mathrm{Slide}^{\omega}(\underline{k})$ (resp. $\mathrm{Slide}^{\psi}(\underline{k})$) precisely if it admits an ω (resp. ψ) $\underline{k}$-slide labeling. So, instead of computing an entire slide set using Definition 5.2.4 or 5.2.5, we can instead use Definition 5.2.6 to test whether a particular tree is in a particular slide set.

Definition 5.2.6. Define the ω (resp. ψ) $\underline{k}$-slide labeling of a stable tree T as the result of the following procedure, if it finishes. (Otherwise, the $\underline{k}$-slide labeling does not exist.)

0. Start with $\ell = n$.

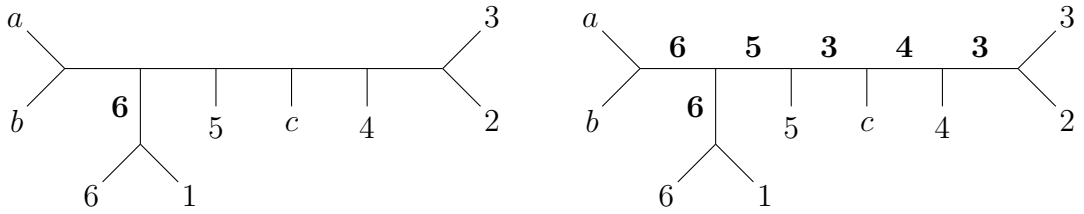
1. **Contract labeled edges.** Let T' be the tree formed from T by contracting all internal labeled edges.

2. **Identify the next edge to label:** Let e be the first unlabeled internal edge on the path in T' from the leaf ℓ to a. (If no such edge exists, then the process terminates and no $\underline{k}$-slide labeling exists.) Let v_ℓ be the vertex adjacent to ℓ, and v_a be the other vertex of e.

3. **Verify that label is valid.** Let m_ℓ be the smallest leaf label among all branches of v_ℓ not containing a or ℓ, and m_a as the smallest leaf label among all branches of v_a not containing a or ℓ. In the ω case (resp. ψ case), if $\ell \geqslant m_\ell \geqslant m_a$ (resp. $m_\ell \geqslant m_a$), then label e with ℓ. Otherwise, the process terminates.

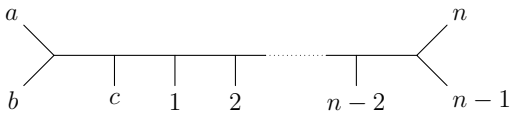
4. **Iterate.** If ℓ has labeled fewer than k_ℓ edges, repeat this process with the same ℓ. Otherwise, decrement ℓ. If $\ell = 0$, then we have successfully constructed the $\underline{k}$-slide labeling of T.

Example 5.2.7. Consider the tree below on the left. As we perform the first round of the ω $(0, 0, 2, 1, 1, 2)$-slide algorithm, we have $\ell = 6$, and try to label the edge above the leaf 6. We have $m_\ell = 1$ and $m_a = c$, so since $c < 1 < 6$, we label this edge $\mathbf{6}$, as shown below on the left. Continuing this process, we end up with the labeling on the right.



As a consequence of Proposition 5.1.5, we known that $\mathrm{Slide}^{\omega}(\underline{k})$ is nonempty if and only if $\underline{k}$ is reverse Catalan. This can easily be shown combinatorially, as there is a tree T that is in every nonempty slide set for a particular n, which we illustrate in the next example.

Example 5.2.8. The tree below lies in $\mathrm{Slide}^{\omega}(\underline{k})$ for all reverse-Catalan $\underline{k}$. This is because the reverse-Catalan condition ensures that the edges are labeled in order from right to left.



Another known fact about caterpillar trees is the characterization of the their edge labels in the case where $\underline{k} = (1, 1, \ldots, 1)$, which we restate below. This result is Proposition 6.2 from [10], which we restate here with modified notation. Recall Definition 2.3.9 for an explanation of the notation $23{-}1$.

Proposition 5.2.9 (Proposition 6.2 from [10])**.** *Let* $\mathrm{Cat}^{\omega}(1, 1, \ldots, 1) \subseteq \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ *be the subset of trivalent trees that correspond to caterpillar curves. For each tree* $T \in \mathrm{Cat}^{\omega}(1, 1, \ldots, 1)$*, define the word* $w(T)$ *by reading the labels in the slide labeling of* T *from left to right. The set of words* $\{w(T) : T \in \mathrm{Cat}^{\omega}(1, 1, \ldots, 1)\}$ *are precisely the* $23{-}1$*-avoiding permutations of length* n*, and in fact the words* $w(T)$ *are all distinct.*

We generalize this result to when $\underline{k}$ is not the all 1's composition in Chapter 8.

5.3 Combinatorics of slide trees

In this section we define several useful characteristics of slide trees and summarize notions that we will use throughout the rest of this work.

Definition 5.3.1. For $\underline{k}$ a composition of n, define $\mathrm{maxzero}(\underline{k})$ to be the largest $z \in [n]$ such that $k_z = 0$. If $\underline{k} = (1, 1, \ldots, 1)$, set $\mathrm{maxzero}(1, 1, \ldots, 1) = c$ or $\mathrm{maxzero}(1, 1, \ldots, 1) = 0$ as appropriate in the given context.

Note that in [6], $\mathrm{maxzero}(\underline{k})$ was called $i_{\mathbf{k}}$.

Definition 5.3.2. For a reverse-Catalan composition $\underline{k}$, define $\mathrm{z}(i) := \#\{j > i \mid k_j = 0\}$.

Notation 5.3.3. In order to maintain clarity between edge and leaf labels, we use bolded labels for the edges of a tree and nonbolded labels for the leaves of a tree, such as $\mathbf{x}$ vs x.

Recall that in a trivalent tree, every vertex either has degree 1 (a leaf) or degree 3. We call an edge an **internal edge** if it connects two non-leaf vertices. A leaf of a slide tree will always be adjacent to exactly one internal vertex. It is useful, however, to consider when two leaves are as close to each other in a tree as is possible. For a leaf i, call its unique neighbor v_i. We say two leaves i and j are **adjacent** if $v_i = v_j$. That is, i and j are adjacent (in the traditional sense) to the same internal vertex. For a leaf i in a trivalent tree, exactly one of the following will occur:

- The leaf i is adjacent to another leaf j, and v_i has one internal edge.
- The leaf i is adjacent to no other leaves and v_i has two internal edges.

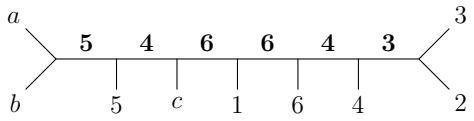
In a slide tree, the leaves a and b are always adjacent. We treat the collection of a, b, $v_a = v_b$, and the two edges between them as the **root** of a tree.

Notation 5.3.4. We standardize our drawings of trees by always drawing them with the root on the left, with the internal edge coming out of v_a pointing to the right. In this way, when we say something is *left* of a leaf/edge/etc, we mean 'towards the root', and when we say *right*, we mean 'away from the root'.

Definition 5.3.5. Let T be a trivalent tree whose set of internal edges of T form a path.We call T a **caterpillar tree**. We define the subset of slide trees that are also caterpillar trees by $\mathrm{Cat}^{\omega}(\underline{k}) \subseteq \mathrm{Slide}^{\omega}(\underline{k})$ (and equivalently for $\mathrm{Cat}^{\psi}(\underline{k})$).

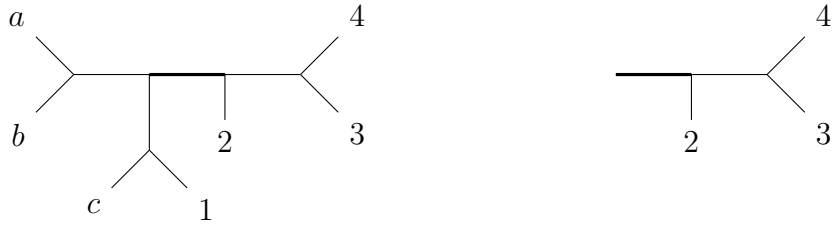
With the root on the left, all the internal edges of a caterpillar tree are drawn horizontally. It is then natural to treat the edge labels of a caterpillar tree as a word, read off from left to right.

Example 5.3.6. The tree below is a tree in $\mathrm{Cat}^{\omega}(0,0,1,2,1,2)$, drawn along with its edge labels from the slide algorithm. With the root drawn on the left, it is natural to read off the word 546643 from the edge labels.



We say 'the branch starting at edge $\mathbf{e}$' to refer to the collection of all edges and vertices (including $\mathbf{e}$ itself) on the opposite side of $\mathbf{e}$ as the root.

Example 5.3.7. Consider the tree T below on the left, and let $\mathbf{e}$ be the bolded edge. Then, the branch starting at $\mathbf{e}$ is the branch below on the right.



Similarly, we can consider the maximal branch that contains some leaf j but not some other leaf i. For example, in Example 5.3.7 above, the branch B on the right is the maximal branch of T that contains 3, but not 1, since adding any additional edges to B would necessarily add the edge to the left of $\mathbf{e}$, which would also add the leaves c and 1.

Definition 5.3.8. For a branch B of a slide tree, let $\min(B)$ denote the minimal leaf label in B.

Next, we prove a number of basic facts about slide trees. We state these in terms of ψ-slide trees, but the statements are all also true for ω-slide trees, as a result of the following lemma.

Lemma 5.3.9. *Let $\underline{k}$ be a composition of n. Then,* $\mathrm{Slide}^{\omega}(\underline{k}) \subseteq \mathrm{Slide}^{\psi}(\underline{k})$.

Proof. This is an immediate consequence of the definition of the ψ and ω $\underline{k}$-slides. (See Definition 5.2.6.) □

Although a slide tree, as originally defined, only has labels on its leaves, there is a unique edge label for each internal edge as given by the $\underline{k}$-slide algorithm. Thus, throughout this work we will view a slide tree as consisting of the data of the underlying graph, along with both leaf and edge labels.

Lemma 5.3.10. *Let i be a leaf of a tree $T \in \mathrm{Slide}^{\psi}(\underline{k})$. Then, all edges labeled* $\mathbf{i}$ *must lie on the path from i to a.*

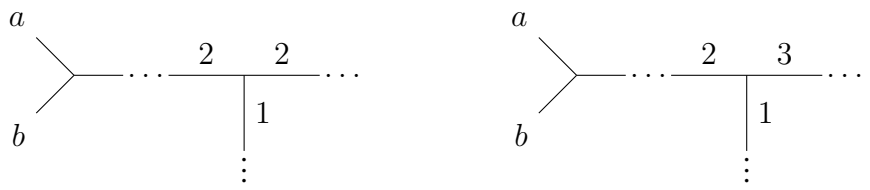
Proof. This is immediate from Definition 5.2.6. □

Lemma 5.3.11. *Let v be a vertex of $T \in \mathrm{Slide}^{\psi}(\underline{k})$ that is adjacent to three internal vertices* $\mathbf{x}$*,* $\mathbf{y}$*, and* $\mathbf{z}$*. Let* $\mathbf{x}$ *be the edge towards a, and suppose (without loss of generality) that* $\mathbf{y} \geqslant \mathbf{z}$*. Then, the slide labeling algorithm will always label the three edges in the order* $\mathbf{y}$*, then* $\mathbf{x}$*, then* $\mathbf{z}$*.*

Proof. By Lemma 5.3.10, we cannot have $\mathbf{y} = \mathbf{z}$, so we must have that $\mathbf{y} > \mathbf{z}$. Thus, since the slide labeling algorithm uses labels in decreasing order, $\mathbf{y}$ is labeled before $\mathbf{z}$. For the rest, it suffices to show that $\mathbf{x}$ is labeled second.

We can never label $\mathbf{x}$ first, since with neither $\mathbf{y}$ and $\mathbf{z}$ contracted, there is no leaf for the label x to come from. Suppose instead that $\mathbf{x}$ were labeled last. Let B and C be the branches starting at $\mathbf{y}$ and $\mathbf{z}$, respectively. When $\mathbf{y}$ slides, since $\mathbf{x}$ is still not contracted, the algorithm will compare $\min(B)$ against $\min(C)$, so we must have $\min(B) > \min(C)$. Similarly, if $\mathbf{x}$ is still not contracted when $\mathbf{z}$ slides, we will also require that $\min(C) > \min(B)$. These cannot possibly both be true at once, so we have a contradiction. Thus, $\mathbf{x}$ also cannot be labeled last. □

So, regardless of the exact labels involved, there is only one order we can label the three edges around an internal vertex. If we additionally consider the relative order of the three labels, we see that $\mathbf{z}$ must always be a unique label, while $\mathbf{y}$ can either be equal to or larger than $\mathbf{x}$. The two options are shown in the figure below. As a consequence, we have the following corollary, which says that these are the only two types of internal vertices.



Corollary 5.3.12. Let v be a vertex of $T \in \mathrm{Slide}^{\psi}(\underline{k})$ that is adjacent to three internal vertices $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$. Let $\mathbf{x}$ be the edge towards a, and suppose (without loss of generality) that $\mathbf{y} \geqslant \mathbf{z}$. Then, either $\mathbf{x} = \mathbf{y} > \mathbf{z}$, or $\mathbf{y} > \mathbf{x} > \mathbf{z}$.

Intuitively, these two types of internal vertices correspond to $2{-}1{-}2$ patterns and $23{-}1$ patterns, respectively, appearing in the words corresponding to a tree.

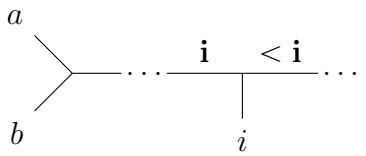
Lemma 5.3.13. *Let v be a vertex of $T \in \mathrm{Slide}^{\psi}(\underline{k})$ that is adjacent to three internal vertices $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$. Let $\mathbf{x}$ the edge towards a, and $\mathbf{z}$ the unique smallest edge given by Corollary 5.3.12. Let B be the branch starting at $\mathbf{x}$, and B' the branch starting at $\mathbf{z}$. Then, $\min(B) \in B'$.*

Proof. By Lemma 5.3.11, $\mathbf{y}$ slides first out of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$. Let C be the branch starting at $\mathbf{y}$. When $\mathbf{y}$ slides, it will compare then minimal element of C to that of B'. So, we must have $\min(B') < \min(C)$, which must mean that $\min(B) = \min(B')$. □

Remark 5.3.14. Given the edge labels of a slide tree, we can find the location of the leaf c. Starting at the root, walk down the internal edges until the labels weakly increase or we reach a branch. If we reach an ascent in the edge labels, the leaf between them is c, or else the larger edge (or the right one in the case of a tie) cannot slide. If we reach a branching vertex first, then by Lemma 5.3.13 it must be down the branch with the smaller edge. Then, we can repeat this process with that branch until we reach a weak ascent across a leaf, or the end of a branch.

Our last result for this section is only for the case of ω-slide trees.

Lemma 5.3.15. *Let i be a leaf of a tree $T \in \mathrm{Slide}^{\omega}(\underline{k})$ such that v_i has two internal edges. Call these edges $\mathbf{x}$ and $\mathbf{y}$, with $\mathbf{x}$ the edge towards a. If $i = \mathbf{x}$, then $\mathbf{x} > \mathbf{y}$.*



Proof. We can not have $\mathbf{y} = \mathbf{x} = i$, since that would contradict Lemma 5.3.10. So, we must show that if $\mathbf{x} = i$, we can not have $\mathbf{y} > \mathbf{x}$. So, let $\mathbf{x} = i$ and suppose instead that $\mathbf{y}$ is larger than $\mathbf{x}$. Then, $\mathbf{y}$ slides before $\mathbf{x}$ in the slide labeling. Let B be the branch starting at $\mathbf{y}$. When $\mathbf{y}$ slides, since $\mathbf{x}$ has not yet slid, $\mathbf{y}$ will compare the minimal leaf of B to i. For $\mathbf{y}$ to be a valid slide, we must have $\min(B) > i$. However, then when i later slides down edge $\mathbf{x}$, it will need to compare $\min(B)$ with something further down the tree. Since $\min(B) > i$, this is an invalid slide, since it violates the definition of an ω-slide in Definition 5.2.6. Thus, for T to be a valid ω-slide tree, we must have $\mathbf{y} < \mathbf{x}$. □

Chapter 6

The case $\mathbf{k} = (1, 1, \ldots, 1)$

We will now demonstrate our iterative bijection in the case $\underline{k} = (1, 1, 1, \ldots, 1)$, we can express the iterative bijection more directly. Recall Proposition 5.2.9, in which [10] define a bijection between $\mathrm{Cat}(1, 1, \ldots, 1)$ and the set of $23{-}1$-avoiding permutations. Our work in this section will extend this map to a full bijection between $\mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ and S_n.

Definition 6.0.1. Let $\tau \in S_n \backslash \mathrm{Av}_n(23{-}1)$. We call x, y, z the **earliest** instance of $23{-}1$ if $\tau = wxyuzv$, where w, u, and v are (possibly empty) words, x, y, and z form a $23{-}1$ pattern, and τ has no instances of $23{-}1$ that either use letters from w, or use x, y, and a letter from u.

Next, we define the map ϕ that tells us how to read a word (a permutation, in fact) off from the edge labels of a $(1, 1, \ldots, 1)$-slide tree.

Definition 6.0.2. We define the map $\phi : \mathrm{Slide}^{\omega}(1, 1, \ldots, 1) \to S_n$ as follows. Consider a slide tree $T \in \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ along with its edge labels given by the slide labeling algorithm. Then, let T' be the tree obtained by deleting the leaves and leaf edges of T, rooted at the vertex adjacent to a in T. (In other words, we record the slide labels and then delete all leaves and non-internal edges.) Note that T' is an at-most-trivalent tree.

Next, starting at the root of T', read off and record the edge labels of T' in order from left to right. When a degree 3 vertex is reached, recursively apply this same process to both branches B and B', where B is the branch with larger minimal label. Then, take the words obtained from B and B' and concatenate them in that order to the word obtained thus far for T'. Since the content of the edge labels is $(1, 1, \ldots, 1)$, and this process reads off each edge exactly once, the resulting word is an element of S_n.

Definition 6.0.3. Define EL_n to be the set of rooted **edge-labeled trivalent trees** with n internal edges labeled by $[n]$ (and unlabeled leaves).

We define the map $\nu : \mathrm{Av}_n(23{-}1) \to \mathrm{Cat}^{\omega}(1,1,\ldots,1)$ to be the *leaf labeling algorithm* defined in Definition 6.3 of [10]. Each tree in its image is a valid caterpillar slide tree. Note that this map coincides with the the map $\mathrm{Tree}()$ from which we define later in Section 8, when restricted to permutations that avoid the pattern $23{-}1$.

Definition 6.0.4 ([10], Definition 6.3)**.** Let w be a $23{-}1$-avoiding permutation. Define the tree T_w to be the tree constructed as follows: First label the internal edges of a caterpillar tree by $w_1, \ldots, w_n$ from left to right, and label the leftmost two leaves a, b. Then label the remaining leaves $n, n-1, n-2, \ldots, 1, c$ in descending order via the following rule:

At step $n-i$, let j be the edge label just to the right of edge $n-i$ (if such an edge j exists).

Case 1: If $j < n-i$, then label the leaf just to the right of $n-i$ by $n-i$.

Case 2: If $j > n-i$ or j does not exist, label the rightmost unlabeled leaf to the right of $n-i$ by $n-i$.

Finally, label the remaining unlabeled leaf by c.

We wish to extend the map ν to a map $\rho : S_n \to \mathrm{Slide}^{\omega}(1,1,\ldots,1)$. We first define a map $\hat{\rho} : S_n \to EL_n$, and then show that each tree in the image $\hat{\rho}$ has a unique leaf labeling that obtains its edge labels under the slide labeling algorithm, and thus $\hat{\rho}$ induces the map ρ.

Definition 6.0.5. Define $\hat{\rho} : S_n \to EL_n$ as follows. If $\tau \in \mathrm{Av}_n(23{-}1)$, we let $\hat{\rho}(\tau) = \nu(\tau)$ (that is, the caterpillar tree whose edge labels read from left to right as τ). Otherwise, let x, y, z be the *earliest* $23{-}1$ pattern in τ, and w, u, v be words such that $\tau = wxyuzv$. Then we construct $\hat{\rho}(\tau)$ as follows. Start with the caterpillar subtree $\hat{\rho}(wx)$. Then, to the internal vertex furthest from the root, we replace the two external edges with the two subtrees $\hat{\rho}(yu)$ and $\hat{\rho}(zv)$.

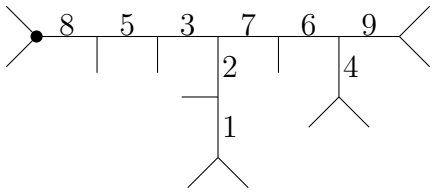
Notation 6.0.6. Throughout this section, we use the abuse of notation $\hat{\rho}(w)$, where w is a subword of a permutation, to denote the tree formed by applying $\hat{\rho}$ to the reduction $\mathrm{red}(w)$, then relabeling the result to have the same labels as the entries of w.

Definition 6.0.7. Given $\tau \in S_n$, define $\rho(\tau)$ to be the unique element of $\mathrm{Slide}^{\omega}(1,1,\ldots,1)$ whose $(1,1,\ldots,1)$-slide labeling admits the same edge labels as $\hat{\rho}(\tau)$.

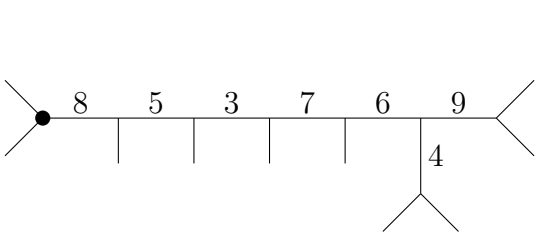
Theorem 6.0.8. *For any $\tau \in S_n$, there is exactly one tree of* $\mathrm{Slide}^{\omega}(1,1,\ldots,1)$ *that admits the edge labeling $\hat{\rho}(\tau)$. In other words, $\rho : S_n \to \mathrm{Slide}^{\omega}(1,1,\ldots,1)$ is well-defined.*

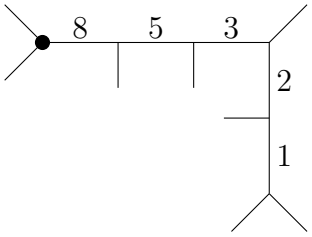
Proof. We prove this result in Section 9.1. □

Example 6.0.9. Let $\tau = 853769421$. Then, $\hat{\rho}(\tau)$ is the tree

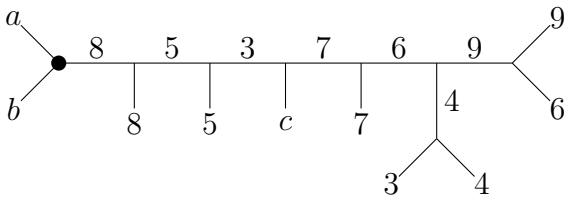


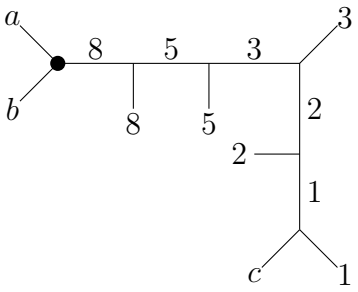
The earliest instance of $23{-}1$ in τ is the subword 372, so $w = 85$, $x = 3$, $y = 7$, $u = 694$, $z = 2$, and $v = 1$. Then, the trees $A = \hat{\rho}(txyu)$ and $B = \hat{\rho}(txzv)$ are the trees drawn below, respectively.



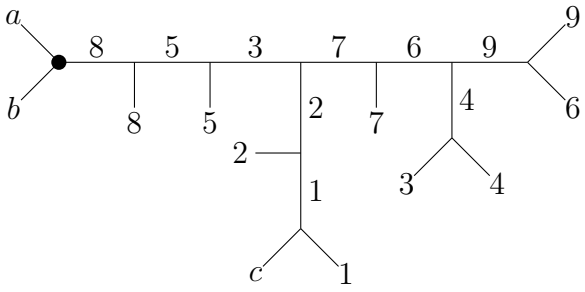


Then, inductively we know that A and B have unique leaf labels that give these edge labelings.





Then, we can piece the leaf labels back together.



Thus, $\rho(\tau) \in \mathrm{Slide}^{\omega}(1,1,1,1,1,1,1,1,1)$.

Theorem 6.0.10. *The maps ϕ and ρ are inverse bijections.*

Proof. Since S_n and $\mathrm{Slide}^{\omega}(1,1,\ldots,1)$ are finite sets of the same cardinality, it suffices to show that $\phi(\rho(\tau)) = \tau$ for all $\tau \in S_n$.

We do induction on the number of internal vertices of $\rho(\tau)$. If $\tau \in \mathrm{Av}_n(23{-}1)$, then $\phi(\tau)$ is a caterpillar tree with edge labels in the same order as the letters in τ. So, ϕ starts at the root of $\rho(\tau)$ and reads off the edges in order, recovering τ.

Otherwise, let $\tau = wxyuzv$, such that x, y, z is the earliest instance of $23{-}1$. Then, $\rho(\tau)$ can be partitioned into the subtrees $\rho(wx)$, $\rho(yu)$, and $\rho(zv)$, where $\rho(wx)$ is caterpillar and $\rho(yu)$ and $\rho(zv)$ are rooted at the end of $\rho(wx)$. Since x is the minimal leaf of $\rho(yu)$ and $z < x$, we will read $\rho(yu)$ before $\rho(zv)$ when we apply ϕ to $\rho(\tau)$. Since $\rho(yu)$ and $\rho(zv)$ have strictly fewer internal vertices, by are inductive hypothesis $\phi(\rho(yu)) = yu$ and $\phi(\rho(zv)) = zv$. So,

$$\phi(\rho(\tau)) = wx\phi(\rho(yu))\phi(\rho(zv)) = wxyuzv = \tau$$

as desired. □

Chapter 7

The main bijection

In this section, we answer Question 5.1.13 by generalizing what we did in Chapter 6 to all $\underline{k}$.

Recall the *asymmetric multinomial coefficients* $\left\langle \begin{smallmatrix} n \\ \underline{k} \end{smallmatrix} \right\rangle$. They satisfy the recurrence relation given in Definition 5.1.1, which is discussed in more detail in [6]. It was already shown in [11] that $\mathrm{Tour}(\underline{k})$ satisfies Equation 5.1.

Definition 7.0.1 (Definition 3.11 from [11])**.** Define $\pi_{\text{lazy}} : \mathrm{Tour}(\underline{k}) \to \coprod_{j>i} \mathrm{Tour}(\underline{k}^{(j)})$ as follows. Let $T \in \mathrm{Tour}(\underline{k})$ and consider the pair (i, j) in the tournament of T that faces off first, with $j > i$. Let v be the vertex adjacent to i and j.

- If $k_j > 1$, then define $\pi_{\text{lazy}}(T)$ to be the tree formed by labeling v by j, removing the leaves and leaf edges of i and j, and decreasing all of the labels $i+1, i+2, \ldots, n$ by 1.
- If $k_j = 1$, then define $\pi_{\text{lazy}}(T)$ to be the tree formed by labeling v by i, removing the leaves and leaf edges of i and j, and decreasing all of the labels $j+1, j+2, \ldots, n$ by 1.

The two cases above correspond to whether or not the laziness rule applies to the first round of the tournament. They show in [11] that this map is well-defined, and is in fact a bijection. This then proves that the sets $\mathrm{Tour}(\underline{k})$ satisfies Equation 5.1. We now wish to show combinatorially that the sets $\mathrm{Slide}^{\omega}(\underline{k})$ satisfy the same recursion.

We will show this as follows. In [6], it was shown that the coefficient $\left\langle \begin{smallmatrix} n \\ \underline{k} \end{smallmatrix} \right\rangle$ is equal to a sum of terms $\left\langle \begin{smallmatrix} n-1 \\ \underline{k}^{(j)} \end{smallmatrix} \right\rangle$ for compositions $\underline{k}^{(j)}$ of $n-1$. We will define maps $\hat{\sigma}_{i,j}$ and $\hat{\sigma}_i$ from $\mathrm{Slide}^{\omega}(\underline{k}^{(j)})$ to $\mathrm{Slide}^{\omega}(\underline{k})$, along with inverses $\hat{\pi}_{i,j}$ and $\hat{\pi}_i$. Then, we will show that these maps are injective and none of their images overlap, so the collection of these maps together defines a bijection between $\mathrm{Slide}^{\omega}(\underline{k})$ and the union of the sets $\mathrm{Slide}^{\omega}(\underline{k}^{(j)})$. Then, we can find the word corresponding to a tree T recursively as follows. Knowing that T is in the image of exactly one map $\hat{\sigma}_\bullet$, one can find which edge of T was added by $\hat{\sigma}_\bullet$, and find its label under the ω $\underline{k}$-slide labeling algorithm. Then, the word for T is that edge label appended to the end of the word for $\hat{\pi}_\bullet(T)$.

7.1 Preliminaries

This technique relies on having a means of determining which image of a map $\hat{\sigma}_\bullet$ a given tree T is in. This is done by the map $\mathrm{last}(T)$, which we now define.

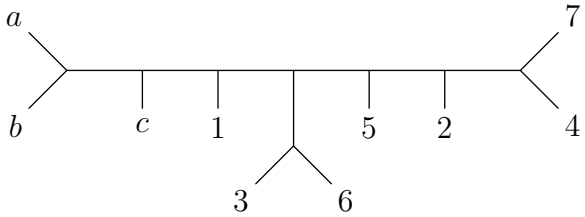
Definition 7.1.1. Let $T \in \mathrm{Slide}^\omega(\underline{k})$, and B be a branch of T. Let i and j be the smallest and second smallest leaves of B, respectively. Then, define $\min_2(B)$ to be the largest branch of B containing j but not i.

Definition 7.1.2. Define the map $\mathrm{last} : \mathrm{Slide}^\omega(\underline{k}) \to [n]$ as follows.

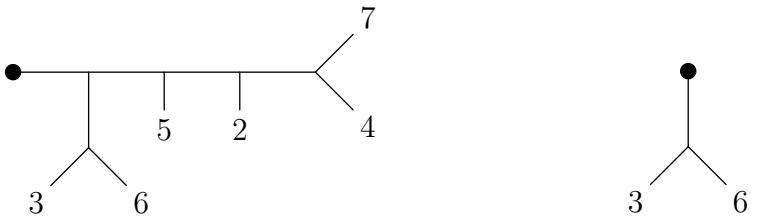
1. For a given $T \in \mathrm{Slide}^\omega(\underline{k})$, let B be the largest branch of T that has $\mathrm{maxzero}(\underline{k})$ as its smallest leaf.
2. If B has at least two leaves, replace B with $\min_2(B)$.
3. Repeat Step 2 until B has a single leaf.
4. Define $\mathrm{last}(T)$ to be the single leaf of B.

Remark 7.1.3. Note that since T is a finite tree, this process terminates after a finite number of steps. Secondly, since the new B must always contain the leaf j, every step gives a nonempty branch of T. Thus, we will always end up with a branch containing a single leaf, so the map $\mathrm{last}(T)$ is well-defined.

Example 7.1.4. Consider the following tree $T \in \mathrm{Slide}^\omega(0, 0, 1, 1, 2, 2)$.



We will compute $\mathrm{last}(T)$. First, $\mathrm{maxzero}(0, 0, 1, 1, 2, 2) = 2$, so our initial B is the branch below on the left, since making it any larger would add the leaf $1 < 2$.



Then, we apply $\min_2$ to B once to get the largest branch containing 3 but not 2, shown above on the right. Finally, applying $\min_2$ a second time yields just the leaf 6, and so $\mathrm{last}(T) = 6$.

Lemma 7.1.5. *For any* $T \in \mathrm{Slide}^{\omega}(\underline{k})$, $\mathrm{last}(T) > \mathrm{maxzero}(\underline{k})$.

Proof. Let $z = \mathrm{maxzero}(\underline{k})$. It is clear that $\mathrm{last}(T) \geqslant z$. We show that $\mathrm{last}(T) \neq z$. To do so, it is enough to demonstrate that the initial branch B contains at least two leaves.

If z is adjacent to another leaf, then since z does not slide, the other leaf x must label the adjacent edge, and for x to slide we must have $z < x$. So, at a minimum, B contains both of the leaves z and x. Alternatively, suppose z is between two edges $\mathbf{x}$ and $\mathbf{y}$, with $\mathbf{x}$ on the path towards a. Since z does not slide, $\mathbf{x} \leqslant \mathbf{y}$, so $\mathbf{y}$ slides before $\mathbf{x}$. For this to be a valid slide, z must be smaller than everything on the branch starting at $\mathbf{y}$, and so B will include all of this branch.

Thus, the initial branch B will contain at least two leaves. So, $\min_2$ will be applied at least once, and thus $\mathrm{last}(T) > z$. □

Lemma 7.1.6. *The map* $\mathrm{last}(T)$ *returns a leaf* j *adjacent to another leaf* i *with* $j > i$.

Proof. We first show that last cannot return a leaf that is between two edges. Suppose instead that $\mathrm{last}(T) = j$ is a leaf between two edges $\mathbf{x}$ and $\mathbf{y}$, where $\mathbf{x}$ is the edge towards a. As shown in the previous proof, the initial branch B has more than one leaf, so $\min_2$ is applied at least once. Consider the state of B just before the last time $\min_2$ is applied. Since B is a branch, and B contains j and at least one other leaf, B must include the whole branch starting at $\mathbf{x}$. Then, since $\min_2(B)$ is just j, j must be the second smallest leaf of B, and the smallest leaf l must be in the branch starting at $\mathbf{y}$. Since $\mathbf{y}$ slides, we have $\mathbf{y} > l$. By Lemma 5.3.15, if $j = \mathbf{x}$, then $j = \mathbf{x} > \mathbf{y} > l$, which contradicts that j is the second smallest leaf of B. So, $j \neq \mathbf{x}$. Then, $\mathbf{x} \leqslant \mathbf{y}$, so $\mathbf{y}$ slides before $\mathbf{x}$. However, $\mathbf{y}$ cannot slide from l to j since $l < j$. Thus, we have a

contradiction, so $\mathrm{last}(T) = j$ does not lie between two edges, and instead is adjacent to another leaf i.

Finally, we must show that the leaf i that is adjacent to $\mathrm{last}(T) = j$ is smaller than j. This follows immediately from the fact that the only way for $\min_2$ to separate j from i is if for some branch, i is the smallest leaf and j is the second smallest leaf. Thus, $j > i$. □

7.2 The map $\hat{\sigma}_{i,j}$

In the next two subsections, we define the two maps $\hat{\sigma}_{i,j}$ and $\hat{\sigma}_j$, which we give this name to be reminiscent of the section maps defined in Section 3.3. These maps take a slide tree T and add an additional edge j to create a larger slide tree. The map $\hat{\sigma}_j$ corresponds to the case in Definition 5.1.1 where we delete a 1 in position j of $\underline{k}$, and $\hat{\sigma}_{i,j}$ corresponds to the case where we subtract 1 from an entry greater than 1 in position j, and delete the rightmost zero in position $i < j$. We will also show that $\mathrm{last}(\hat{\sigma}_\bullet(T))$ will always return the label of the edge added to T by $\hat{\sigma}_\bullet$.

Definition 7.2.1. Let $\underline{k}$ be a reverse-Catalan composition of n, j, and i be integers such that $\mathrm{maxzero}(\underline{k}) < i < j \leqslant n+1$, and $\underline{k}'$ be the composition of $n+1$ obtained from $\underline{k}$ by inserting a zero between k_{i-1} and k_i, and then increasing the jth entry of the result by 1. Note that using the notation from Definition 5.1.1, $\underline{k} = \underline{k}'^{(j)}$. We define the map $\hat{\sigma}_{i,j} : \mathrm{Slide}^\omega(\underline{k}) \to \mathrm{Slide}^\omega(\underline{k}')$ as follows.

1. Given a tree $T \in \mathrm{Slide}^\omega(\underline{k})$, add 1 to all leaf (and edge) labels greater than or equal to i.

2. On the path from a to j, consider the maximal length decreasing sequences of edge labels.

3. Let $B_1, B_2, \ldots, B_l$ be the branches of T off of this path that lie between these maximal length decreasing sequences, and B_l the branch immediately next to leaf j. (See Figure 7.1 for an example.) Note that some (or all) of these branches may consist of a single leaf, and there may be additional branches that connect to this path in the middle of a decreasing sequence.

4. For $r \in [l]$, let $m_r := \min(B_r)$.

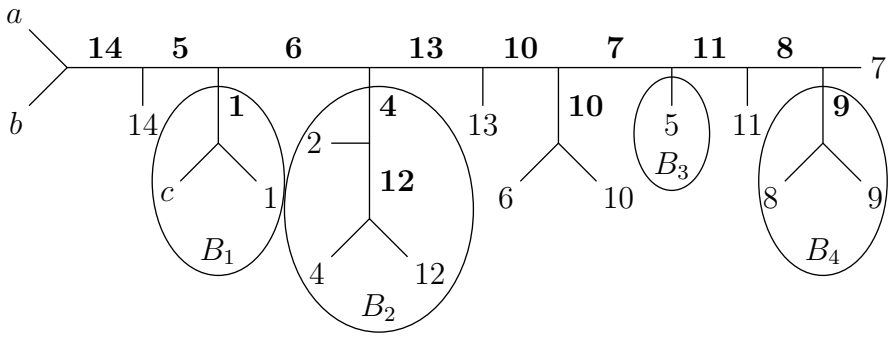


Figure 7.1: A slide tree in $\mathrm{Slide}^{\omega}(1,0,1,1,1,1,1,1,2,1,1,1)$ with the labels $\geqslant 3$ incremented by 1. In this case, $j = 7$.

5. As we will show in Lemma 7.2.4, we need only consider the following three cases for the ordering of $m_1, \ldots, m_l$, i, and j. For each case, we say how to get $\hat{\sigma}_{i,j}(T)$:

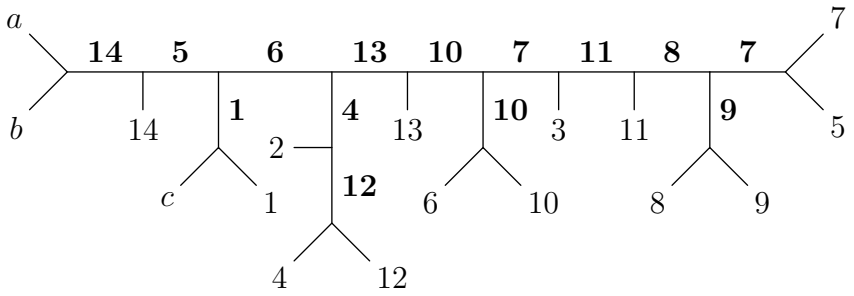
   - $m_l < i < j$ *or* $m_{l-1} < i < j < m_l$: Replace the leaf j by an edge labeled **j** with leaves j and i.
   - $m_1 < \cdots < m_{d-1} < i < m_d < \cdots < m_l < j$: Replace the leaf m_d by i, m_{d+1} by m_d, and so on, replace m_l by m_{l-1}, and replace leaf j by an edge **j** with leaves j and m_l.
   - $m_1 < \cdots < m_{d-1} < i < m_d < \cdots < j < m_l$: Replace the leaf m_d by i, m_{d+1} by m_d, and so on up to replacing m_{l-1} with m_{l-2}, then replace leaf j by an edge **j** with leaves j and m_{l-1}.

   Note that the first case is actually subsumed by the second two cases, but we write it out separately for clarity and to make the proofs clearer.

Remark 7.2.2. In this definition, and as necessary throughout this section, we use the abuse of notation of using the same label to refer to a tree $T \in \mathrm{Slide}^{\omega}(\underline{k})$ as to the tree after having some of its leaf and edge labels changed in Step 1 of Definition 7.2.1.

Example 7.2.3. Consider the tree $T \in \mathrm{Slide}^{\omega}(1,0,1,1,1,1,1,1,2,1,1,1)$ depicted in Figure 7.1. We will demonstrate how we construct the tree $\hat{\sigma}_{3,7}(T)$. The figure depicts T after we have already incremented all the leaf and edge labels greater than or equal to 3 by 1. Then, the maximal decreasing sequences of edge labels from a to 7 are $(14,5)$, (6), $(13,10,7)$, and $(11,8)$. Then, $l = 4$

and the branches B_1, B_2, B_3, B_4 are as depicted in the figure. Their minimal leaves are $m_1 = c$, $m_2 = 2$, $m_3 = 5$, and $m_4 = 8$. We have $c < 2 < i = 3 < 5 < i = 7 < 8$, so $d = 3$ and we are in the third case of Step 5. Thus, we replace 5 with 3 and 7 with an edge $\mathbf{7}$ with leaves 7 and 5 to form $\hat{\sigma}_{3,7}(T) \in \mathrm{Slide}^{\omega}(1,0,0,1,1,1,2,1,1,2,1,1,1)$:



To show that our map $\hat{\sigma}_{i,j}$ is well defined, we first must demonstrate a few facts about the leaves $m_1, \ldots, m_l$, which we do in the next two lemmas.

Lemma 7.2.4. *Let j be a leaf of $T \in \mathrm{Slide}^{\omega}(\underline{k})$, and let $B_1, \ldots, B_l$ and $m_1, \ldots, m_l$ be as in Definition 7.2.1. Then,*

(1) $m_1 = c$,

(2) $m_1 < m_2 < \cdots < m_l$, and

(3) $m_{l-1} < j$.

Proof. For part (1), consider the path from a to B_1. Since the sequence of edge labels decreases, any leaves here must be the same as the adjacent edge label. For any internal vertex v between a and B_1, the edge going towards B_1 must be the smallest edge adjacent to v, so c cannot be down any branches that split off here by Remark 5.3.14. Then, where B_1 splits off, if B_1 has any edges, the edge at the base of B_1 must be smaller than the two edges on the path from a to j by Lemma 5.3.11, so it must contain c by Remark 5.3.14. If B_1 is just a leaf, then it must be c anyway, or else the edge to its right (which slides before the one to the left) is unable to slide. This proves Lemma 7.2.4(1).

For claim (2), suppose instead that there is some d for which $m_d > m_{d+1}$. Then, the edge on the path from a to j immediately to the right of B_d cannot slide, as it would be comparing

something to its right that is at most m_{d+1} to m_d, but we assumed that $m_d > m_{d+1}$. Thus, we must instead have that $m_1 < \cdots < m_l$.

Finally, for (3), the proof follows identically to that of (2): if $m_{l-1} > j$, then the edge to the right of B_{l-1} cannot make a valid slide, so we do not have a valid slide tree, and we must have that $m_{l-1} < j$. □

Corollary 7.2.5. As a consequence, we must have either $m_l < i < j$, $m_{l-1} < i < j < m_l$, $m_1 < \cdots < m_{d-1} < i < m_d < \cdots < m_l < j$, or $m_1 < \cdots < m_{d-1} < i < m_d < \cdots < j < m_l$. Thus, the three cases in Step 5 of Definition 7.2.1 are the only three cases to consider.

Lemma 7.2.6. *Let j be a leaf of $T \in \mathrm{Slide}^{\omega}(\underline{k})$, let $B_1, \ldots, B_l$ and $m_1, \ldots, m_l$ be as in Definition 7.2.1, and let $s \in \{2, 3, \ldots, l\}$. If $s < l$, or $s = l$ and $j > m_l$, then m_s is the smallest leaf on the maximal branch containing m_s but not m_{s-1}.*

Proof. We prove this Lemma in Section 9.2. □

Theorem 7.2.7. *The map $\hat{\sigma}_{i,j}$ is well-defined.*

Proof. We prove this result in Section 9.2. □

Lemma 7.2.8. *For $\underline{k}$ a composition of n and $T \in \mathrm{Slide}^{\omega}(\underline{k})$, $\mathrm{last}(\hat{\sigma}_{i,j}(T)) = j$.*

Proof. If $T' = \hat{\sigma}_{i,j}(T)$, then $\mathrm{maxzero}(\underline{k}') = i$. Let l, d, $B_1, \ldots, B_l$, and $m_1, \ldots, m_l$ be as in Definition 7.2.1. We first show that the maximal branch of T' containing i as its smallest leaf (i.e. the branch B in Step 1 of Definition 7.1.2) is the maximal branch not containing B_{d-1}. Clearly it can be no larger than this, as $i > m_{d-1}$. By Lemma 7.2.6, m_d is the smallest leaf in the corresponding branch in T. So, since $i < m_d$, i is the smallest leaf on this branch in T'.

Next, we iteratively apply $\min_2$ to B. Since m_d is smaller than every leaf of B except i, $\min_2(B)$ is the largest branch of T' containing m_d but not i. Then, since for $s \geqslant d$, the leaf m_s in T' is labeled m_{s+1} in T, the same process will repeat, until we are left with the maximal branch with minimal leaf either m_l (in Case 2 of Step 5 above) or m_{l-1} (in Case 3 of Step 5 above). If we are in Case 2, then j and m_l are the only two leaves left in B, so applying $\min_2$ one last time

leaves just the leaf j. In Case 3, applying $\min_2$ one more time leaves us with just j, since $j < m_l$. Note that we do not include Case 1 here, since it is covered by Cases 2 and 3. In any case, we get that $\text{last}(\hat{\sigma}_{i,j}(T)) = j$. □

Next, we show that $\hat{\sigma}_{i,j}$ is injective by building an inverse map $\hat{\pi}_{i,j}$ that undoes $\hat{\sigma}_{i,j}$. From the definition of $\hat{\sigma}_{i,j}$, in $\hat{\sigma}_{i,j}(T)$ the leaf j is adjacent to some other leaf v and an edge labeled **j**. Secondly, i is the largest leaf label of $\hat{\sigma}_{i,j}(T)$ that does not slide. Bearing that in mind, we now define the inverse map $\hat{\pi}_{i,j}$.

Definition 7.2.9. Define

$$\hat{\pi}_{i,j} : \hat{\sigma}_{i,j}(\text{Slide}^{\omega}(\underline{k})) \to \text{Slide}^{\omega}(\underline{k})$$

as follows. For $T \in \hat{\sigma}_{i,j}(\text{Slide}^{\omega}(\underline{k}))$, $\text{last}(\hat{\sigma}_{i,j}(T)) = j$ by Lemma 7.2.8. Let v be the leaf adjacent to leaf j.

If $v = i$, then replace the branch consisting of the edge **j** and leaves j and i with the leaf j. Then, subtract 1 from all labels in T greater than j. Define $\hat{\pi}_{i,j}(T)$ to be the result.

Otherwise, $v \neq i$, so define branches $B_1, \ldots, B_l$ and leaves $m_1, \ldots, m_l$ as in Definition 7.2.1. (Note: Since $\hat{\sigma}_{i,j}$ does not change any edges, but does shuffle the leaf labels, the branches $B_1, \ldots, B_l$ are the same in T and $\hat{\sigma}_{i,j}^{-1}(T)$, but the leaves $m_1, \ldots, m_l$ may be different.) By definition of $\hat{\sigma}_{i,j}$, there is some leaf m_d such that $m_d = i$. If $m_l < j$, replace leaf m_l with v and m_{l-1} with m_l. Otherwise, replace m_{l-1} with v. Then, regardless, replace m_{l-2} with m_{l-1}, and so on, until we replace $m_d = i$ with m_{d+1}. Next, replace the branch consisting of the edge **j** and leaves j and v with a leaf j. Then, subtract 1 from all labels in T greater than i. Define $\hat{\pi}_{i,j}(T)$ to be the result.

Lemma 7.2.10. *For* $T \in \text{Slide}^{\omega}(\underline{k})$, $\hat{\pi}_{i,j}(\hat{\sigma}_{i,j}(T)) = T$.

Proof. It is clear from the construction of $\hat{\sigma}_{i,j}$ and $\hat{\pi}_{i,j}$ that $\hat{\pi}_{i,j}$ undoes the changes made by $\hat{\sigma}_{i,j}$ to T. □

As an immediate consequence, we get the following corollary.

Corollary 7.2.11. The map $\hat{\sigma}_{i,j}$ is injective.

7.3 The map $\hat{\sigma}_j$

In this subsection, we define the other map from $\mathrm{Slide}^{\omega}(\underline{k})$ into $\mathrm{Slide}^{\omega}(\underline{k}')$, along with its inverse. This one corresponds to the case in the asymmetric multinomial recursion where we decrement a 1 in position j, thus removing the 0 that then appears in that position. Hence, this map uses only one subscript.

Definition 7.3.1. Let $\underline{k}$ be a reverse-Catalan composition of n, j an integer such that $\mathrm{maxzero}(\underline{k}) < j \leqslant n+1$, and $\underline{k}'$ be the composition of $n+1$ obtained from $\underline{k}$ by inserting a 1 between k_{j-1} and k_j. We define the map $\hat{\sigma}_j : \mathrm{Slide}^{\omega}(\underline{k}) \to \mathrm{Slide}^{\omega}(\underline{k}')$ as follows.

1. Given a tree $T \in \mathrm{Slide}^{\omega}(\underline{k})$, add 1 to all leaf (and edge) labels greater than or equal to j.

2. Consider the leaf $v = \mathrm{last}(T)$. By Lemma 7.1.6, it is at the end of an edge along with some other leaf $i < v$. There are three cases to consider:

   - $v < j$: Replace the leaf v by an edge **j** with leaves j and v. The result is $\hat{\sigma}_j(T)$.
   - $i < j < v$: Replace the leaf i by an edge **j** with leaves j and i. The result is $\hat{\sigma}_j(T)$.
   - $j < i$: Continue on to Step 3.

3. On the path from a to v, consider the maximal length decreasing sequences of edge labels.

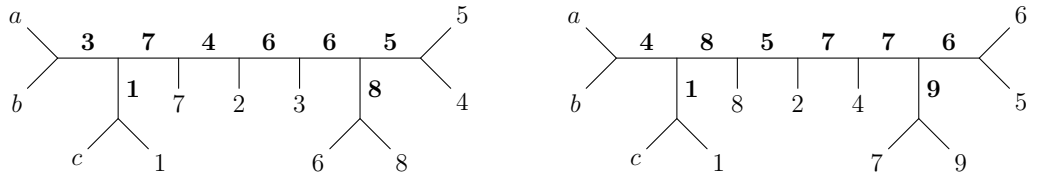
4. Let $B_1, B_2, \ldots, B_l$ be the branches away from this path between the maximal decreasing sequences, with B_l the branch immediately next to leaf v. (See Example 7.3.2.)

5. For $s \in [l]$, let $m_s := \min(B_s)$.

6. As already shown in Lemma 7.2.4, $c = m_1 < m_2 < \cdots < m_l$. So, there is some d such that $m_d < j < m_{d+1}$.

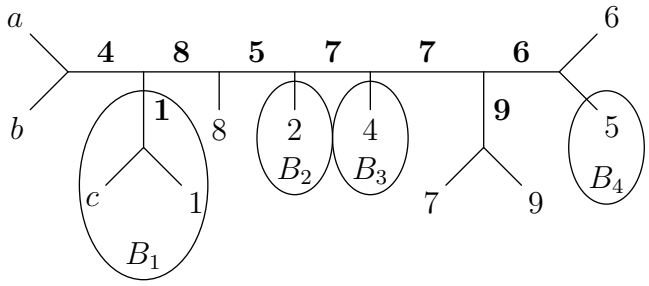
7. Replace the leaf m_d by an edge **j** with leaves j and m_d. The result is $\hat{\sigma}_j(T)$.

Example 7.3.2. Consider the tree $T \in \mathrm{Slide}^{\omega}(1,0,1,1,1,2,1,1)$ below on the left. We will compute $\hat{\sigma}_3(T)$.

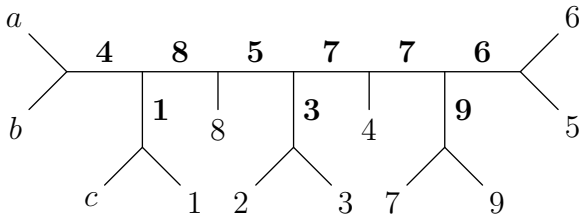


The first step is to increment every label greater than or equal to 3, which gives us the tree above on the right.

Then, $\text{last}(T) = 5$, so on the relabeled tree $v = 6$ and $i = 5$. Since $3 < 5 < 6$, we are in the third case, so we consider the maximal length decreasing subsequences from a to 6, which are (4), $(8, 5)$, (7), and $(7, 6)$. So, the branches $B_1, \ldots, B_4$ are as shown below.



So, $m_1 = c$, $m_2 = 2$, $m_3 = 4$, and $m_4 = 5$. Since $2 < 3 < 4$, we replace the leaf 2. Thus, we get the tree $\hat{\sigma}_3(T)$ below.



Theorem 7.3.3. *The map $\hat{\sigma}_j$ is well-defined.*

Proof. To show that $\hat{\sigma}_j$ is well-defined, we need to show that the result T' is an element of $\text{Slide}^{\omega}(\underline{k}')$. We consider the three separate cases of Step 2 in Definition 7.3.1. For all three cases, the only change done to T is replacing one leaf i with an edge **j** and leaves j and i, where $j > i$. So, this does not affect the ability for any other label to slide, and we need only check that **j** is a valid slide.

In the first case, $j > l > i$, so j can successfully slide from l to i. In the second case, l slides from i to some $d < i$ in T. As $j < l$ slides after l, j will compare i to something that is at most d, so j is a valid slide.

Finally, in the third case, $\text{maxzero}(\underline{k}) < j < m_{d+1}$, so when we found $\text{last}(T)$, the branch B in Step 1 included all of $B_d, \ldots, B_l$. So, m_{d+1} must be smaller than every leaf in B_d except m_d, or else $\text{last}(T)$ would be a leaf in B_d. Thus, since $j < m_{d+1}$ (by Step 6 of Definition 7.3.1), every edge in B_d slides before j, so j will compare m_d to some leaf further down the tree, that can be at most m_{d-1}. Therefore, j is a valid slide, and in all cases $\hat{\sigma}_j(T)$ is a valid slide tree. □

Lemma 7.3.4. *For $\underline{k}$ a composition of n and $T \in \text{Slide}^{\omega}(\underline{k})$, $\text{last}(\hat{\sigma}_j(T)) = j$.*

Proof. Let i and v be as in Definition 7.3.1. We first consider the first two cases in Step 2. We are given that $\text{last}(T) = v$. So, there is some branch B with smallest leaf i and second smallest leaf v that is obtained at some step in computing $\text{last}(T)$. Adding a leaf $j > i$ to this branch (as we do in these two cases) does not change any steps up to this point in computing $\text{last}(\hat{\sigma}_j(T))$. So, in the first case, applying $\min_2$ to this branch B gives the branch with only the leaves j and v, since $j > v$. Then, applying $\min_2$ a second time leaves us with just j. Alternatively, in the second case, $i < j < v$, so applying $\min_2$ to B separates j from i, and gives just the leaf j.

Next, we consider the third case. As in the proof of the previous lemma, the initial branch B in Step 1 of computing $\text{last}(T)$ includes all of $B_d, \ldots, B_l$. So, at some point during that process we have a B with smallest leaf m_d and second smallest leaf m_{d+1}. Adding an edge and leaf j to m_d does not change any of the steps up to this point when computing $\text{last}(\hat{\sigma}_j(T))$, but $j < m_{d+1}$, so j is the new second smallest leaf of this branch. So, applying $\min_2$ to this B gives just the leaf j. Thus, in each case $\text{last}(\hat{\sigma}_j(T)) = j$. □

We will now show that $\hat{\sigma}_j$ is injective by building an inverse map $\hat{\pi}_j$ that undoes $\hat{\sigma}_j$. Notice that in all three cases of Definition 7.3.1, all we do to T is replace some leaf i by an edge **j** with leaves j and i, where j occurs as exactly one edge label in $\hat{\sigma}_j(T)$.

Definition 7.3.5. Define

$$\hat{\pi}_j : \hat{\sigma}_j(\mathrm{Slide}^\omega(\underline{k})) \to \mathrm{Slide}^\omega(\underline{k})$$

as follows. For $T \in \hat{\sigma}_j(\mathrm{Slide}^\omega(\underline{k}))$, $\mathrm{last}(\hat{\sigma}_j(T)) = j$ by Lemma 7.3.4. There is exactly one edge in T with label **j**, and it has leaves j and l. Replace the branch consisting of this edge and these two leaves by the leaf l. Then, subtract 1 from all labels in T greater than j. Let the resulting tree be $\hat{\pi}_j(T)$.

Lemma 7.3.6. *For* $T \in \mathrm{Slide}^\omega(\underline{k})$, $\hat{\pi}_j(\hat{\sigma}_j(T)) = T$.

Proof. It is clear from the construction of $\hat{\sigma}_j$ and $\hat{\pi}_j$ that $\hat{\pi}_j$ undoes the changes made to T by $\hat{\sigma}_j$. □

Corollary 7.3.7. The map $\hat{\sigma}_j$ is injective.

Proof. This follows immediately from 7.3.6. □

7.4 Constructing the full bijection

So far, we have defined two classes of maps $\hat{\sigma}_{i,j}$ and $\hat{\sigma}_j$, and shown that for any tree in their image, the function $\mathrm{last}()$ returns the value of the added edge. This is summarized by the following theorem.

Theorem 7.4.1. *Let* $T \in \mathrm{Slide}^\omega(\underline{k})$. *If* $T \in \hat{\sigma}_{i,j}(\mathrm{Slide}^\omega(\underline{k}'))$ *or* $T \in \hat{\sigma}_j(\mathrm{Slide}^\omega(\underline{k}'))$ *for some composition* $\underline{k}'$, *then* $\mathrm{last}(T) = j$.

Proof. This follows immediately from Lemmas 7.2.8 and 7.3.4. □

Our ultimate goal is to combine these maps into a bijection from a disjoint union of slide sets for compositions of $n-1$ to $\mathrm{Slide}^\omega(\underline{k})$. We will next define this desired disjoint union, recalling Definition 5.1.1.

Definition 7.4.2. Let $i = \mathrm{maxzero}(\underline{k})$. Then, define

$$D^\omega(\underline{k}) := \bigsqcup_{j=i+1}^{n} \mathrm{Slide}^\omega(\underline{k}^j).$$

Then, we can piece together our maps $\hat{\sigma}_{i,j}$ and $\hat{\sigma}_j$ into one large map from $D^\omega(\underline{k})$ to $\mathrm{Slide}^\omega(\underline{k})$:

Definition 7.4.3. Let $i = \mathrm{maxzero}(\underline{k})$. Define $\Sigma_{\underline{k}} : D^\omega(\underline{k}) \to \mathrm{Slide}^\omega(\underline{k})$ by

$$\Sigma_{\underline{k}}(T) := \begin{cases} \hat{\sigma}_{i,j}(T) & \text{if } T \in \mathrm{Slide}^\omega(\underline{k}^j) \text{ for } k_j > 1 \\ \hat{\sigma}_j(T) & \text{if } T \in \mathrm{Slide}^\omega(\underline{k}^j) \text{ for } k_j = 1 \end{cases}.$$

We now prove Theorem 1.2.3, which we restate below.

Theorem 1.2.3. The map $\Sigma_{\underline{k}}$ is a bijection.

Proof. By the definition of the asymmetric multinomial coefficients (see [6] or Definition 5.1.1), $|D^\omega(\underline{k})| = |\mathrm{Slide}^\omega(\underline{k})|$. So, $\Sigma_{\underline{k}}$ is a bijection if and only if it is injective.

Let $T \in \mathrm{Slide}^\omega(\underline{k})$. Define $j = \mathrm{last}(T)$. By Lemma 7.1.5, $j > \mathrm{maxzero}(\underline{k})$, so j is a leaf that slides in T. By Theorem 7.4.1, if T is in the image of either $\hat{\sigma}_j$ or $\hat{\sigma}_{i,j}$ for some i, then **j** is the edge added by the map in question. In the latter case, i must be $\mathrm{maxzero}(\underline{k})$. In particular, T cannot be in the image of $\hat{\sigma}_{j'}$ or $\hat{\sigma}_{i,j'}$ for any $j' \neq j$.

Note that if T is in the image of $\hat{\sigma}_j$ then j slides only once, and if it is in the image of some $\hat{\sigma}_{i,j}$ then j slides more than once. So we can determine which case T lies in by counting the number of edges labeled **j**. By Corollaries 7.2.11 and 7.3.7, both $\hat{\sigma}_i$ and $\hat{\sigma}_{i,j}$ are injective, so T has at most one preimage under that particular $\hat{\sigma}_j$ or $\hat{\sigma}_{i,j}$, and none under any of the others. Thus, $\Sigma_{\underline{k}}$ also is an injection, and so is a bijection. □

Finally, we define the inverse map $\Pi_{\underline{k}}$ to $\Sigma_{\underline{k}}$ where $\underline{k}$ is a composition of n.

Definition 7.4.4. Let $i = \mathrm{maxzero}(\underline{k})$. Define $\Pi_{\underline{k}} : \mathrm{Slide}^\omega(\underline{k}) \to D^\omega(\underline{k})$ by

$$\Pi_{\underline{k}}(T) := \begin{cases} \hat{\pi}_{i,\mathrm{last}(T)}(T) & \text{if } k_{\mathrm{last}(T)} > 1 \\ \hat{\pi}_{\mathrm{last}(T)}(T) & \text{if } k_{\mathrm{last}(T)} = 1 \end{cases}.$$

It follows from the definitions of $\hat{\sigma}_j$, $\hat{\pi}_j$, $\hat{\sigma}_{i,j}$, and $\hat{\pi}_{i,j}$ that $\Sigma_{\underline{k}}$ and $\Pi_{\underline{k}}$ are inverses.

7.5 An explicit example of the bijection

So far, we have shown that $\mathrm{Slide}^{\omega}(\underline{k})$ satisfies the the asymmetric multinomial recursion (5.1), meaning we can define a bijection between $\mathrm{Tour}(\underline{k})$ and $\mathrm{Slide}^{\omega}(\underline{k})$ by unwinding both recurrences in parallel. However, it would be more satisfying to have a way of going between $\mathrm{Tour}(\underline{k})$ and $\mathrm{Slide}^{\omega}(\underline{k})$ directly. We will illustrate what this bijection looks like through a running example we will use throughout this subsection. This bijection will run through column-restricted parking functions and words as intermediate steps. We now describe how to read off a word from a slide tree.

Definition 7.5.1. First, for a word w, define $\bar{\sigma}_{i,j}(w)$ as the word obtained from w by adding one to every entry i or greater, then appending j to the end, and $\bar{\sigma}_j(w)$ as the word obtained from w by adding one to every entry j or greater, then appending j to the end. Then, we can define a map

$$\mathrm{word} : \mathrm{Slide}^{\omega}(\underline{k}) \to \text{words of content } \underline{k}$$

as follows.

$$\mathrm{word}(T) = \begin{cases} \varnothing & \text{if } T = T_0 \\ \bar{\sigma}_{i,j}(\mathrm{word}(T')) & \text{if } T = \hat{\sigma}_{i,j}(T') \\ \bar{\sigma}_j(\mathrm{word}(T')) & \text{if } T = \hat{\sigma}_j(T') \end{cases},$$

where T_0 denotes the unique slide tree in $\mathrm{Slide}^{\omega}(\underline{k})$ for $\underline{k}$ the empty composition.

In other words, the word for a tree T is the slide labels of the internal edges, read off in the order the edges were added to the tree under the $\Sigma_{\underline{k}}$ recursion.

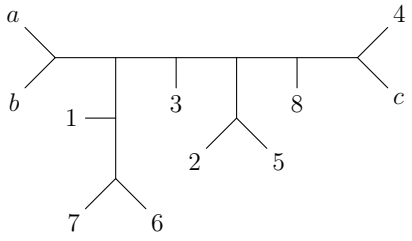
Meanwhile, for parking functions, a natural way to read off a word w from a parking function P is to let the value of w_i be the column of P that the number i occurs in. As it turns out, there is a nice relationship between the words that can be obtained from slide trees and the words obtained from column-restricted parking functions.

Proposition 7.5.2. *Let* $\mathrm{rev}(w) = w_n w_{n-1} \cdots w_2 w_1$ *denote the reverse of a word* w*. Then,* w *comes from an* ω *slide tree if and only if* $\mathrm{rev}(w)$ *comes from a column-restricted parking function.*

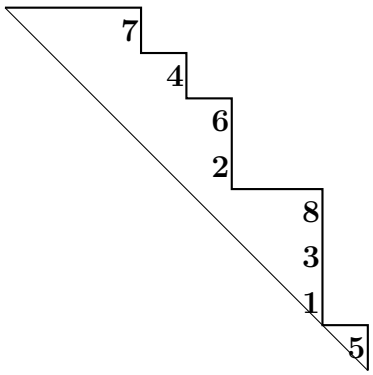
Proof. If a word w comes from a tree $T \in \mathrm{Slide}^{\omega}(\underline{k})$, the last letter j of w must be greater than $\mathrm{maxzero}(\underline{k})$, since T is in the image of some $\hat{\sigma}_{i,j}$ or $\hat{\sigma}_j$. Similarly, the dominance condition on column-restricted parking functions forces the 1 to be left of all empty columns, which in turn means the first entry in the corresponding CPF-word must be greater than $\mathrm{maxzero}(\underline{k})$. From here, it is clear from the recursions $\Sigma_{\underline{k}}$ above in Definition 7.4.3 and the map φ from Section 5 of [6] result in words that are reverses of each other. □

We now illustrate an example of the full sequence of maps from $\mathrm{Tour}(\underline{k})$ to $\mathrm{Slide}^{\omega}(\underline{k})$.

Example 7.5.3. Let $T \in \mathrm{Tour}(0,0,1,1,2,0,3,1)$ be the tournament tree below.



Using the map $\tau : \mathrm{Tour}(\underline{k}) \to \mathrm{CPF}(\underline{k})$ from [11], this corresponds to the following column-restricted parking function.

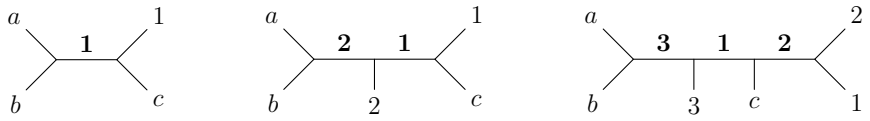


Next, reading off from this parking function, we get the word 75748537. Taking the reverse of this results in the word 73584757, which is what we will use to construct our slide tree. To do this,

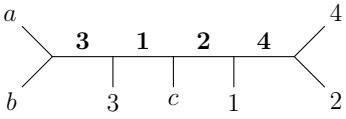
we have

$$\begin{aligned}
\text{word}^{-1}(73584757) &= \hat{\sigma}_{6,7}(\text{word}^{-1}(6357465)) = \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\text{word}^{-1}(524635))) \\
&= \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\text{word}^{-1}(41352)))) = \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\hat{\sigma}_2(\text{word}^{-1}(3124))))) \\
&= \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\hat{\sigma}_2(\hat{\sigma}_4(\text{word}^{-1}(312)))))) \\
&= \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\hat{\sigma}_2(\hat{\sigma}_4(\hat{\sigma}_2(\text{word}^{-1}(21))))))) \\
&= \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\hat{\sigma}_2(\hat{\sigma}_4(\hat{\sigma}_2(\hat{\sigma}_1(\text{word}^{-1}(1)))))))) \\
&= \hat{\sigma}_{6,7}(\hat{\sigma}_{2,5}(\hat{\sigma}_{1,5}(\hat{\sigma}_2(\hat{\sigma}_4(\hat{\sigma}_2(\hat{\sigma}_1(\hat{\sigma}_1(T_0)))))))).
\end{aligned}$$

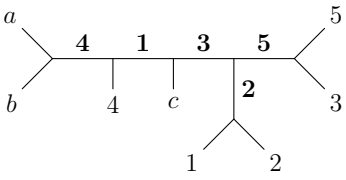
So, we start with adding the edge $\mathbf{1}$ to the empty tree. Then, the next two steps, $\hat{\sigma}_1$ and $\hat{\sigma}_2$, do not require us to move where we add an edge, so we add the edges $\mathbf{1}$ and $\mathbf{2}$ to the right of the previous steps, respectively. These first three steps are shown below.



Similarly, since $4 > 2$, $\hat{\sigma}_4$ can again add 4 to the end with no complications.

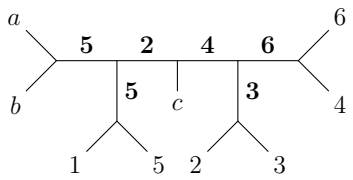


However since $2 < 3 < 5$, $\hat{\sigma}_2$ is in the third case of Step 2 of Definition 7.3.1. So, we consider the decreasing sequences $(3, 1)$, (2), and (4) from a to 4, and conclude that we add the edge $\mathbf{2}$ off of the leaf 1.

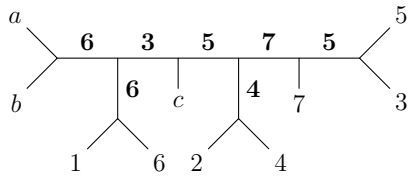


Next, $\hat{\sigma}_{1,5}$ adds an edge $\mathbf{5}$ and leaf 1. Since the leaf 4 only has one internal edge between it and a, there is only one branch B_1 to consider, which has minimal leaf c. Thus, we are in Case 1 of

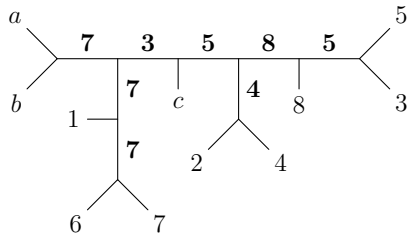
Step 5 of Definition 7.2.1. Thus, we add the new edge and leaf right where the old leaf 5 (relabeled from 4) was.



For $\hat{\sigma}_{2,5}$, we have decreasing sequences $(6, 3)$, (5), and (7) (after relabeling). So, since $c < 2 < 3 < 7$, we are in Case 3, so replace the leaf 3 with the leaf 2, and move the three to the end of the new edge $\mathbf{5}$. Note that the relabeling of the leaves makes it look look like the leaf 2 does not move, but the new 3 is actually the relabeled 2.



Finally, we apply $\hat{\sigma}_{6,7}$. Since 6 is larger than c and 1, we are again in Case 1. So, we add the leaf 6 to the end of the new edge $\mathbf{7}$.



Thus, the above tree $T' \in \mathrm{Slide}^{\omega}(0, 0, 1, 1, 2, 0, 3, 1)$ is the slide tree in bijection with the tournament tree T we started with.

Chapter 8

Caterpillar trees and pattern avoidance

In this section, we discuss specific case of caterpillar trees. Recall the definition of caterpillar trees from Definition 5.3.5.

As restated in Proposition 5.2.9, it was found in [10] that in the case where $\underline{k} = (1, 1, \ldots, 1)$, the words obtained from caterpillar trees are precisely the set of $23{-}1$ avoiding permutations. In other words, they found a bijection $\mathrm{Cat}^{\psi}(1, 1, \ldots, 1) \leftrightarrow \mathrm{Av}_n(23{-}1)$. We wish to describe a similar correspondence in terms of pattern avoidance for any composition $\underline{k}$.

If we consider what the map $\mathrm{word}()$ in Definition 7.5.1 does to caterpillars, we see that it will always read off the slide labels from left to right, in order. This coincides with the notion of reading edge labels off of a tree from the root to the right that we discussed in Section 5.

We first show that every word comes from at most one caterpillar slide-tree, and define a map from words to trivalent, leaf-labeled trees that outputs this unique tree for a particular word, if such a tree exists. Then, we give such a correspondence for the case where $\underline{k}$ is right-justified, as that is precisely when the caterpillar words can be described solely by a pattern avoidance condition. After that, we give the more general characterizations for all cateprillar slide trees, first for $\mathrm{Cat}^{\psi}(\underline{k})$, and then for $\mathrm{Cat}^{\omega}(\underline{k})$.

8.1 Preliminaries on caterpillar slide trees

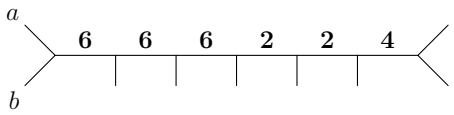
Before we describe how to go between caterpillars and words, we first show that such a correspondence is well-defined. That is, we first show that each word w can come from at most one caterpillar tree, and then describe which words do arise as edge words of caterpillar trees. We define an algorithm that takes a word w and constructs the only caterpillar tree $\mathrm{Tree}(w)$ that could have w as its word of edge labels. That is, $\mathrm{Tree}(w)$ is always a trivalent caterpillar tree with labeled leaves, and if $\mathrm{Tree}(w)$ is a slide tree, then w is its edge word.

Definition 8.1.1. Given a word w of content $\underline{k}$ of length n, we construct the caterpillar tree $\mathrm{Tree}(w)$ as follows.

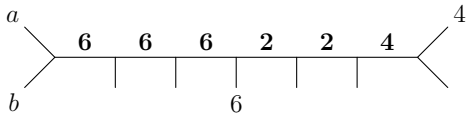
1. Construct a path graph of length n, and add leaf edges to it to make it trivalent.
2. By convention, we treat the left end as the root and label the two leaves on the left end by a and b.
3. Label the internal edges from left to right using the letters of w in order.
4. For each pair of adjacent internal edges, let $\mathbf{x}$ be the label of the left edge and $\mathbf{y}$ the label of the right edge (that is, $\mathbf{x}$ is closer to the root). We call the leaf between them a **descent** leaf if $\mathbf{x} > \mathbf{y}$, and a **nondescent** leaf if $\mathbf{x} \leqslant \mathbf{y}$. For the two leaves on the right end, we define one to be a descent leaf and the other a nondescent leaf.
5. Label each descent leaf with the label of the edge immediately to its left.
6. Create a list of the remaining unused labels among $c, 1, \ldots, n$.
7. From left to right, label each nondescent leaf by the smallest remaining unused label, unless that label labels the edge immediately to the right of that leaf. In that case, instead label the leaf with the second smallest unused label.
8. Forget the edge labels.

This forms a leaf-labeled trivalent caterpillar tree $\mathrm{Tree}(w)$.

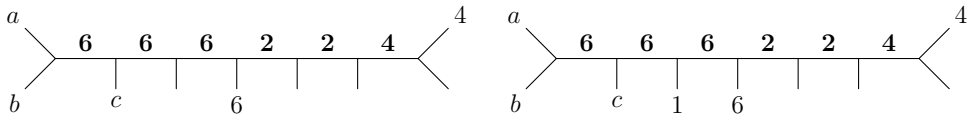
Example 8.1.2. Consider when $w = 666224$. After Step 3, our tree looks like this:



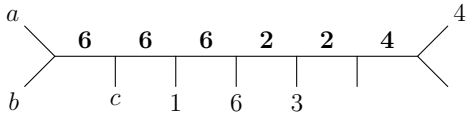
Then, we label only the descent vertices. The only descent is after the last 6, and there is also a descent leaf on the end, so we label the 6 and the 4:



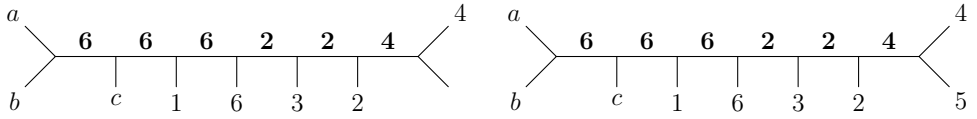
Then, we work from left to right, labeling the nondescent leaves:



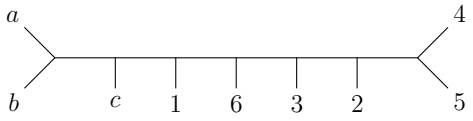
Up until now, we have been able to use the smallest unused label at each step. Now however, the smallest unused label is 2, but since 2 also labels the edge to the right of the next leaf, we skip 2 and instead use 3, the second smallest remaining label:



Now we are free to use the 2 for the following leaf, and can finish off the leaves:



Finally, we remove the edge labels to form $\mathrm{Tree}(w)$:



Note that in this case, $\mathrm{Tree}(666224)$ lies in $\mathrm{Slide}^{\psi}(0,2,0,1,0,3)$, but not $\mathrm{Slide}^{\omega}(0,2,0,1,0,3)$.

We wish to characterize when $\mathrm{Tree}(w)$ is a slide tree. One necessary condition for being a slide tree is that $\mathrm{Tree}(w)$ must use each leaf label $a, b, c, 1, 2, \ldots, n$ exactly once. This happens precisely when w avoids $2{-}1{-}2$, which we show in the following lemma.

Lemma 8.1.3. *When $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2)$, Definition 8.1.1 produces a tree that uses each leaf label $a, b, c, 1, 2, \ldots, n$ exactly once.*

Proof. If w contains no $2{-}1{-}2$ pattern, then no label $\mathbf{i}$ can be used to label two different descent leaves in Step 5, since all edges between two edges with label $\mathbf{i}$ must be weakly larger than $\mathbf{i}$, meaning only the rightmost $\mathbf{i}$ can possibly be left of a descent leaf. So, Step 5 uses each label at most once. Then, Step 7 clearly never uses any label for a second time. There are $n+3$ leaves and $n+3$ labels total, and the last nondescent leaf on the end of the tree can always use the smallest (and only) remaining label, as there is no edge to its right. So, we use every leaf label $a, b, c, 1, \ldots, n$ exactly once. □

In other words, $\mathrm{Tree}(w)$ is well-defined as a map from $\mathrm{Av}_{\underline{k}}(2{-}1{-}2)$ to the set of leaf-labeled trivalent caterpillar trees using the labels $a, b, c, 1, 2, \ldots, n$.

Given a particular caterpillar slide tree, one can also read off the word formed along its edges by the slide-labeling algorithm.

Theorem 8.1.4 (Uniqueness of slide trees)**.** *Each word comes from at most one caterpillar ψ-slide tree.*

Proof. We show that given any word w, if there is a corresponding slide tree, it must be $\mathrm{Tree}(w)$.

First, we observe that if a tree $\mathrm{Tree}(w)$ is a slide tree, the descent leaves are the positions where the edge to the left is contracted before the edge to the right in the $\underline{k}$-slide labeling algorithm, while the nondescent leaves are the positions where the right edge is contracted before the left edge.

Given a word w, any corresponding slide tree must have its descent leaves labeled as in Definition 8.1.1. Otherwise, if $\mathbf{x}$ is the edge label on the left of a descent leaf not labeled this way, then the leaf label x will not label that edge $\mathbf{x}$.

Finally, for each nondescent leaf v, when the edge on its right is labeled by some label $\mathbf{i}$, the $\underline{k}$-slide algorithm compares the label of v against the set of all leaf labels to the right of v except for i. So, the nondescent labels must be labeled as in Step 7, so that it is possible that each nondescent leaf is smaller than everything to its right, expect possibly for the label doing the slide. □

Corollary 8.1.5. Due to Lemma 5.3.9, this also gives us uniqueness for ω-slide trees.

Lemma 8.1.6. *Let v be a nondescent leaf of a caterpillar slide tree $T \in \mathrm{Slide}^{\psi}(\underline{k})$, and $\mathbf{e}$ the edge on its right. If $v > \mathbf{e}$, there is no edge labeled v.*

Proof. Let $\mathbf{d}$ be the edge to the left of v, and assume that $v > \mathbf{e}$. Then, since v is a nondescent leaf, $\mathbf{d} \leqslant \mathbf{e}$, so $\mathbf{d} < v$. Suppose that v labels some edge during the slide algorithm. Since $v > \mathbf{d}$, v would label edges before $\mathbf{d}$, and so the edge on the left of v would be labeled by something at least as large as v, and thus would already be contracted when $\mathbf{d}$ did its labeling, meaning $\mathbf{d}$ could not label it. This contradicts our assumption that $\mathbf{d}$ labels the edge on the left, so v must label no edges. □

Recall Definitions 2.3.7 and 2.3.9. For a word w to avoid the pattern $23{-}\overline{2}{-}1$, then whenever there are indices i, j such that $i + 1 < j$ and $w_j < w_i < w_{i+1}$, there must be some $i + 1 < k < j$ such that $w_k = w_i$. Intuitively, this means that every $23{-}1$ pattern must extend to a $23{-}2{-}1$ pattern.

Example 8.1.7. The word $w = 35432$ avoids the pattern $23{-}\overline{2}{-}1$. Even though the subword 352 forms a $23{-}1$ pattern, it extends to the subword 3532, so this is not an instance of a $23{-}\overline{2}{-}1$ pattern.

Lemma 8.1.8. *Let T be a tree in $\mathrm{Cat}^{\psi}(\underline{k})$. Then, the word w formed by reading off the edges starting from the root avoids the patterns $2{-}1{-}2$ and $23{-}\overline{2}{-}1$.*

Proof. Clearly, w can not contain a $2{-}1{-}2$ pattern, since in each iteration in the $\underline{k}$-slide algorithm, after contracting the already labeled edges, all labels of the same value must be consecutive.

For $23{-}\overline{2}{-}1$, suppose instead that $w_i = x$, $w_{i+1} = y$, and $w_j = z$ form a $23{-}\overline{2}{-}1$ pattern. Then, w_i is the rightmost instance of x in w, since this assumes there is no x between w_i and w_j, and we already showed that w avoids $2{-}1{-}2$, so there is no x to the right of w_j. Next we have $x < y$, so the leaf labeled x is not a descent leaf, and is instead a nondescent leaf. Since x does slide at some point in T, the leaf x must be smaller than the edge to its right by Lemma 8.1.6. Then, since that edge on the right of x is a valid slide, x must be smaller than every leaf to its right.

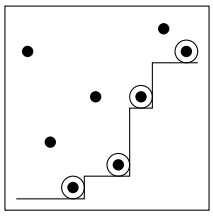
However, the leaf x must be to the left of $\mathbf{w_j}$ in order for it to label $\mathbf{w_i}$, and the leaf z must be to the right of $\mathbf{w_j}$. This is a contradiction, so w avoids $23{-}\overline{2}{-}1$. □

Lemma 8.1.9. *Let $\underline{k}$ be a reverse-Catalan composition of n with m zeros, such that*

$$w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2,\ 23{-}\overline{2}{-}1).$$

Let $i_1 < \ldots < i_l$ be the indices of the entries in w that are strictly smaller than all entries to their right. Then, in $\mathrm{Tree}(w)$, each edge $\mathbf{w_{i_j}}$ has a nondescent leaf immediately to its right, and there are exactly m additional nondescent leaves.

Example 8.1.10. Let $w = 73152587$. Then, $l = 4$, $i_1 = 3$, $i_2 = 5$, $i_3 = 6$, $i_4 = 8$, and $w_{i_1} = 1$, $w_{i_2} = 2$, $w_{i_3} = 5$, and $w_{i_4} = 7$.



Proof (of Lemma 8.1.9). First, note that there are exactly m repeats in w (that is, a position where the letter also occurs again later in w). We show that the nondescent leaves of $\mathrm{Tree}(w)$ correspond precisely to the position of these repeats along with the positions $w_{i_1}, \ldots, w_{i_{l-1}}$.

Consider a pair of consecutive entries $x = w_j$ and $y = w_{j+1}$ of w. If x and y form a descent, then the leaf between them in $\mathrm{Tree}(w)$ is a descent leaf. Then, j is not some i_s and x is not a repeat, since otherwise x and y would form the first two entries of a $2{-}1{-}2$ pattern.

Otherwise, if x and y form a nondescent, then the leaf between them in $\mathrm{Tree}(w)$ is a nondescent leaf. If $x = y$, then clearly x is a repeat and is not some w_{i_s}. If x and y form an ascent, there are three cases: $j \in \{i_1, \ldots, i_l\}$, $x = w_{i_s}$ for some $i_s > j$, or $x > w_{i_s}$ for some $i_s > j$. In the second case, clearly x is a repeat of w_{i_s}. In the third case, x must also be a repeat, or else x, y, and w_{i_s} form a $23{-}\overline{2}{-}1$ pattern.

Finally, there is a nondescent leaf to the right of w_{i_l} by construction. Thus, we have shown that the nondescent leaves are on the right of each edge corresponding to some w_{i_j} and each repeated entry of w. □

Corollary 8.1.11. Let $\underline{k}$, w, and $i_1, \ldots, i_l$ be as in Lemma 8.1.9. Then for each j, i_j has at most $m + j - 1$ nondescent leaves to its left.

8.2 Right-justified compositions

Now, we consider the case of right-justified compositions. These are the only case for which the caterpillar trees are characterized solely by a pattern avoidance criterion.

Definition 8.2.1. A composition $\underline{k}$ of n is **right-justified** if all zeros occur before all non-zero entries.

Theorem 1.2.4. Let $\underline{k}$ be right-justified with m zeros, and let $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2,\ 23{-}\overline{2}{-}1)$. Then, $\mathrm{Tree}(w)$ is a valid slide tree.

Proof. We prove this result in Section 9.3. □

Then, as an immediate consequence of Theorem 1.2.4 and Lemma 8.1.8, we get our desired result.

Corollary 8.2.2. Let $\underline{k}$ be a right-justified composition. Then, the map

$$\mathrm{Tree} : \mathrm{Av}_{\underline{k}}(2{-}1{-}2,\ 23{-}\overline{2}{-}1) \to \mathrm{Cat}^{\psi}(\underline{k})$$

is a bijection.

8.3 Pattern avoidance conditions for $\mathrm{Cat}^{\psi}(\mathbf{k})$

In this subsection, we characterize all the caterpillar trees in $\mathrm{Slide}^{\psi}(\underline{k})$, including when $\underline{k}$ is not right-justified.

Definition 8.3.1. For a word w with content $\underline{k}$, let $k(i,j)$ denote the number of j's in the subword of w right of the rightmost i. Then, define the total number of repeats right of the rightmost i by

$$\mathrm{TotalRep}_w(i) = \sum_{j \neq i} \max((k(i,j)-1), 0).$$

Definition 8.3.2. For a word w, let $\ell_w(i)$ denote the total number of consecutive i's in the rightmost consecutive sequence of i's in w. Let $\mathbf{i_l}$ and $\mathbf{i_r}$ denote the leftmost and rightmost such entries, respectively.

Example 8.3.3. Let $w = 313321$. Then, $\ell_w(1) = 1$, $\ell_w(2) = 1$, and $\ell_w(3) = 2$. Additionally, $\mathbf{3_l}$ is in position 3 and $\mathbf{3_r}$ is in position 4.

Now, we state the ψ version of Theorem 1.2.5.

Theorem 8.3.4. *Let $\underline{k}$ be a reverse-Catalan composition of n, and let w be a word of composition $\underline{k}$. Then, $\mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$ if and only if $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$ and*

$$\mathrm{TotalRep}_w(i) + \ell_w(i) \geqslant \mathrm{z}(i) \tag{8.1}$$

for all i such that $k_i > 0$.

Proof. We prove this result in Section 9.4. □

8.4 Pattern avoidance conditions for $\mathrm{Cat}^{\omega}(\mathbf{k})$

In this last subsection, we us our characterization of ψ-caterpillars to characterize the ω-caterpillar trees in $\mathrm{Slide}^{\omega}(\underline{k})$, including when $\underline{k}$ is not right-justified.

Definition 8.4.1. For a word w with content $\underline{k}$, let $k(i,j)$ denote the number of j's in the subword of w right of the rightmost i. Define the number of repeats larger than i right of the rightmost i by

$$\mathrm{BigRep}_w(i) = \sum_{j=i+1}^{n} \max((k(i,j)-1), 0).$$

Now, we state the ω version of Theorem 1.2.5.

Theorem 8.4.2. *Let $\underline{k}$ be a reverse-Catalan composition, and let w be a word of composition $\underline{k}$. Then, $\mathrm{Tree}(w) \in \mathrm{Cat}^{\omega}(\underline{k})$ if and only if $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$ and*

$$\mathrm{BigRep}_w(i) \geqslant \mathrm{z}(i) \tag{8.2}$$

for all i such that $k_i > 0$.

Proof. We prove this result in Section 9.5. □

Chapter 9

Proofs of main results

In this chapter, we give proofs of our main results.

9.1 Proof of Theorem 6.0.8

Lemma 9.1.1. *Let B be a branch of a slide tree $T \in \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$. Then, all leaves of B except $\min(B)$ slide within B.*

Proof. We first show that $\min(B)$ does not slide within B. Suppose instead that $\min(B)$ did slide along some edge $\mathbf{e}$. Then, let $B_{\mathbf{e}}$ be the branch starting at $\mathbf{e}$, and i be the number of leaves of $B_{\mathbf{e}}$. Then, $B_{\mathbf{e}}$ has $i-1$ internal edges, including $\mathbf{e}$. For $\min(B)$ to slide along $\mathbf{e}$, all $i-1$ larger leaf labels must slide before $\min(B)$. However, there are only $i-2$ other edges in $B_{\mathbf{e}}$ available for them to slide on in order to leave $\mathbf{e}$ unlabeled. This is an impossibility, and so $\min(B)$ cannot slide within B.

Second, we show that all leaves other than $\min(B)$ slide in B. Each internal edge of B must be labeled by some leaf in B, each leaf can can be used for at most one edge (since $\underline{k} = (1, 1, \ldots, 1)$), and we just showed that $\min(B)$ does not slide in B. So, since there is one more leaf than internal edge in B, all leaves other than $\min(B)$ must slide within B. □

Lemma 9.1.2. *Let $T \in \mathrm{Slide}^{\omega}(\underline{k})$, where $\underline{k}$ is the all 1's composition of n. Let $\mathbf{e}$ be an internal edge of T, B the branch of T starting at $\mathbf{e}$, and $i+1$ be the number of leaves of B. Construct T' to be the tree resulting from T by replacing B with a single leaf $\min(B)$. Next, relabel the leaves of T' so that every label $a, b, c, 1, 2, \ldots, n-i$ is used once, while keeping the ordering of the leaves the same. Then, $T' \in \mathrm{Slide}^{\omega}(\underline{k}')$, where $\underline{k}'$ is the all 1's composition of $n-i$.*

Proof. We will show that the slide labeling algorithm completes successfully for T'. We do this by showing that each edge is a valid slide.

First, any internal edge of T in B does not exist in T', so there is no corresponding slide to check in T'. Otherwise, consider some edge $\mathbf{e}$ in $T \backslash B$ that is not labeled $\min(B)$. In T, the comparison made in the slide algorithm at step $\mathbf{e}$ either does not consider any of B, or considers a branch C containing B. The corresponding branch C' in T' still contains $\min(B)$, so $\min(C) = \min(C')$. Thus, the slide labeling $\mathbf{e}$ is valid in T'.

Finally, we consider the label $\min(B)$. If $\min(B) = c$, then there is no edge $\min(B)$, and we are done. Otherwise, $\min(B)$ slides in T after all other leaves of B. So, it slides after the internal edges of B are all contracted. So in T', since $\min(B)$ is a leaf where B was removed, it will label the same edge and compare the same labels as it did in T. Thus, every edge in T' is a valid slide, so $T' \in \mathrm{Slide}^{\omega}(\underline{k}')$. □

Lemma 9.1.3. *If $T \in \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ is not a caterpillar, then $\phi(T)$ contains the pattern $23{-}1$.*

Proof. Let v be a vertex in T adjacent to three internal edges $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$, where $\mathbf{x}$ is the edge towards a. Without loss of generality, we assume that $\mathbf{z} < \mathbf{y}$. By Corollary 5.3.12, we have $\mathbf{z} < \mathbf{x} < \mathbf{y}$. Let U be the branch starting at the edge $\mathbf{y}$, and V the branch starting at $\mathbf{z}$.

Next, since $\mathbf{x} > \mathbf{z}$, the label $\mathbf{x}$ must come from U. Since by Lemma 9.1.1 every leaf in U slides within U except for $\min(U)$, and the edge $\mathbf{x}$ is not in U, $\mathbf{x} = \min(U)$. Since $\mathbf{x}$ is a valid slide, $\min(V) < \min(U)$, so the map ϕ will read off the edges of U before V. Thus, $\phi(T) = \ldots xy \ldots z \ldots$, and x, y, and z will form a $23{-}1$ pattern. □

Finally, we now restate and prove Theorem 6.0.8.

Theorem 6.0.8. For any $\tau \in S_n$, there is exactly one tree of $\mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ that admits the edge labeling $\hat{\rho}(\tau)$. In other words, $\rho : S_n \to \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ is well-defined.

Proof. We proceed by strong induction on the number of branching vertices of $\hat{\rho}(\tau)$ (that is, vertices that are not adjacent to any leaves). The base case is when $\hat{\rho}(\tau)$ has no internal vertices. That is, when $\hat{\rho}(\tau)$ is a caterpillar tree. By Proposition 5.2.9, there is a unique leaf labeling that gives this slide labeling, so, $\rho(\tau)$ is well-defined in this case.

Otherwise, suppose $\hat{\rho}(\tau)$ has m branching vertices. For the inductive hypothesis, we assume that $\rho(\tau')$ is well-defined whenever $\hat{\rho}(\tau')$ has $m-1$ or fewer branching vertices. Since $\hat{\rho}(\tau)$ has a branching vertex, τ contains the pattern $23{-}1$. So, let $\tau = wxyuzv$, where x, y, z is the earliest $23{-}1$ pattern in τ. We will show that $T = \hat{\rho}(\tau)$ is a slide tree.

In particular, let $A = \hat{\rho}(wxyu)$ and $B = \hat{\rho}(wxzv)$. Let U and V be the subtrees formed under $\hat{\rho}$ by yu and zv, respectively. Since x, y, z was the earliest $23{-}1$ pattern in τ, there is no $23{-}1$ pattern in $wxyu$ that involves both x and y, and since $z < x$, there is no $23{-}1$ pattern in $wxzv$ that involves both x and z. This means that all $23{-}1$ patterns in $wxyu$ or $wxzv$ occur entirely within yu or zv respectively. So, neither A nor B have a branching vertex at the end of $\mathbf{x}$, and thus A (respectively B) have the shape of T with V (respectively U) removed. By our inductive hypothesis, both A and B are valid slide trees, so we consider them with their associated leaf labels.

Let d be the leaf of A in the position of V in T, and e the leaf of B in the position of U in T. Next, we apply Lemma 9.1.2 to A and replace the branch U with $\min(U)$ to create a new slide tree T_A. Similarly, we apply Lemma 9.1.2 to B and replace V with $\min(V)$ to form T_B. As a consequence of Lemma 9.1.1 and Lemma 9.1.2, T_A and T_B are caterpillar trees with the same edge labels. Thus, their leaf labels are also the same by the inductive hypothesis. In particular, consider the leaves at the right ends of T_A and T_B. For T_A, we already had d, and then replaced U with $\min(U)$. Similarly for T_B, we have e and $\min(V)$. So as sets, $\{d, \min(U)\} = \{e, \min(V)\}$.

Next, we show that $e = \min(U)$ and $d = \min(V)$. Since $\mathbf{x} > \mathbf{z}$, $\mathbf{x}$ slides before $\mathbf{z}$ in B, so $\mathbf{x} = e$ in order for e to slide along edge $\mathbf{x}$. Since $\mathbf{y}$ is a valid slide in A and $\mathbf{y} > \mathbf{x}$, $\min(U) > d$, and so $\min(U)$ will slide before d in A. From Lemma 9.1.1, $\min(U)$ cannot slide within U, so it must slide across the edge $\mathbf{x}$. Thus, $\min(U) = \mathbf{x} = e$. By process of elimination, $d = \min(V)$.

Thus far, we have shown that A and B each have leaf labelings that make them valid slide trees, these labelings agree on T_A and T_B, $e = \min(U)$, and $d = \min(V)$. We will now use these facts to build a valid leaf labeling for T. For any leaf in U or V, we label that leaf by the same label that leaf has in A or B, respectively. The remaining leaves all lie between a and $\mathbf{x}$, so we give them the corresponding labels from $T_A = T_B$. This uses all the labels $c, 1, 2, \ldots, n$ exactly once.

Next, we show that such a leaf labeling gives a slide labeling matching T. For any edge labeled neither $d = \min(V)$ nor $e = \min(U)$, and not in the branch V, the comparison made at that step by the slide-labeling algorithm will at most only look at a branch containing V, and will make the same comparison it made in A, where all of V was replaced by d. So, the validity of this slide carries over from A. Similarly, any slide other than d or e that is outside U is valid because it makes the same comparison as in B. Lastly, if there is an edge labeled d (resp. e), then d will slide only after the other leaves of V (resp. U) have already done so. So, it will slide the same way as it did in A (resp. B), when it started where V (resp. U) connected to T. These cases cover every edge of T, so T is a valid slide tree.

Finally, we must demonstrate uniqueness. Suppose instead that there were two slide trees $T_1, T_2 \in \mathrm{Slide}^{\omega}(1, 1, \ldots, 1)$ that both had the slide labeling $\hat{\rho}(\tau)$. Since the edge labels are the same for both T_1 and T_2, let A_i, B_i, U_i, V_i be defined as above for $i = 1, 2$. Then, applying Lemma 9.1.2 to U_1 and U_2, by our inductive hypothesis the results are the same tree, so $\min(U_1) = \min(U_2)$, and all other leaves of A_1 and A_2 must be the same as well. However, we can do the same for $\min(V_1)$, $\min(V_2)$, B_1, and B_2. Thus, all leaves of T_1 and T_2 are the same, so in fact $T_1 = T_2$. Therefore, the tree T found above is unique, and ρ is well-defined. □

9.2 Proofs of Lemma 7.2.6 and Theorem 7.2.7

In this Section, we also prove Lemma 7.2.6.

Lemma 7.2.6. Let j be a leaf of $T \in \mathrm{Slide}^{\omega}(\underline{k})$, let $B_1, \ldots, B_l$ and $m_1, \ldots, m_l$ be as in Definition 7.2.1, and let $s \in \{2, 3, \ldots, l\}$. If $s < l$, or $s = l$ and $j > m_l$, then m_s is the smallest leaf on the maximal branch containing m_s but not m_{s-1}.

Proof. Let $s \in \{2, 3, \ldots, l\}$ as above, and let x be the smallest leaf right of B_s. We first show that any leaf off of the path from B_{s-1} to B_s is larger than $\min(m_s, x)$. Let $\mathbf{e_1} > \cdots > \mathbf{e_{r+1}}$ be the edges from B_{s-1} to B_s, and $E_1, \ldots, E_r$ the branches between them. If E_r is a leaf, then that leaf must be labeled e_r since $\mathbf{e_r} > \mathbf{e_{r+1}}$. Otherwise, consider the edge $\mathbf{f}$ of E_r adjacent to $\mathbf{e_r}$ and $\mathbf{e_{r+1}}$. Since $\mathbf{e_r} > \mathbf{e_{r+1}}$, $\mathbf{f} \geqslant \mathbf{e_r}$ by Lemma 5.3.11. So, $\mathbf{f}$ slides before $\mathbf{e_r}$, and $\min(E_r)$ must be larger

than the smallest leaf to the right of E_r. Similarly, we can show that

$$\min(E_{r-1}) > \min(m_s, x, \min(E_r)) = \min(m_s, x)$$

by the same argument. So, continuing this process, we have that every leaf off of the path between B_{s-1} and B_s is larger than $\min(m_s, x)$.

Next, we proceed by downwards induction on s. We have two base cases to consider: either $s = l$ when $j > m_l$, or $s = l - 1$ when $j < m_l$. In the first case, the only leaf right of B_l is j, and $j > m_l$, so m_l is the smallest leaf in or right of B_s. So, by the previous argument, m_l is the smallest leaf on the maximal branch containing m_l but not m_{l-1}. For the second case, by the above argument everything between B_{l-1} and B_l is larger than $j = \min(m_l, j)$. By Lemma 7.2.4(3), $m_{l-1} < j$, so m_{l-1} is the smallest leaf in or right of B_{l-1}, so by the previous paragraph m_{l-1} is the smallest leaf in the maximal branch containing m_{l-1} but not m_{l-2}.

By induction, we assume this is true for m_{s+1}. Then, by an identical argument to m_{l-1}, using the fact that $m_1 < m_2 < \cdots < m_l$ by Lemma 7.2.4(2) we can show the same is true for m_s. □

Theorem 7.2.7. The map $\hat{\sigma}_{i,j}$ is well-defined.

Proof. The algorithm described in Definition 7.2.1 clearly gives a unique output, so showing well-definedness amounts to showing that the result is an element of $\mathrm{Slide}^{\omega}(\underline{k}')$. We do this by first showing that the leaves in $T' = \hat{\sigma}_{i,j}(T)$ will label the same edges as in T (except for the new edge that was created), and then show that every slide is valid.

The only leaves in different positions between T and T' (other than i, which does not slide at all) are $m_d, \ldots, m_l$ (or $m_d, \ldots, m_{l-1}$ when $j < m_l$). However, by Lemma 7.2.6, each such m_s is the smallest leaf in a branch containing both m_s and the position m_s is moved to by $\hat{\sigma}_{i,j}$. So, every edge in that branch is labeled by something larger than m_s, and moving m_s within that branch does not change which edge it will end up labeling. Thus, assuming that every slide is valid, the edges of T' will be labeled in the same order (and thus get the same labels) as in T.

Next, we show that every edge $\mathbf{e}$ in T' is a valid slide, given that T is a valid slide tree. We do this by case of where $\mathbf{e}$ is in T in relation to P, where P is the path from a to j.

Case 1: Suppose $\mathbf{e}$ is on the path P. In T, $\mathbf{e}$ slides from m_{r+1} to m_r for some $r < l$, by Lemma 7.2.6 and the fact that the edges on P decrease from B_r to B_{r+1}. Let v_1 and v_2 be the leaves labeled by m_r and m_{r+1} in T, respectively. Although $\hat{\sigma}_{i,j}$ might relabel one or both of v_1 and v_2, in all cases of Definition 7.2.1, v_1 will still have a smaller label than v_2, and, so long as $r \neq l-1$ or $j > m_l$, both leaves will still have the smallest labels in their respective branches mentioned in Lemma 7.2.6. So in T', $\mathbf{e}$ will slide from v_2 to v_1, which is a valid slide.

We also must consider the case where $\mathbf{e}$ is to the right of B_{l-1} and $j < m_l$. By Lemma 5.3.15, j is strictly smaller than the first edge to the left of B_l on P. So, $\mathbf{e}$ slides from j to m_{l-1} in T. As $m_{l-2} < m_{l-1} < j < m_l$, $\mathbf{e}$ will slide from m_{l-1} to m_{l-2} in T', and this is a valid slide.

Case 2: Suppose $\mathbf{e}$ slides to P, and we are not in the case where $\mathbf{e}$ is in B_l and $j < m_l$. Let s be the largest index such that B_s is to the left of $\mathbf{e}$. Then $\mathbf{e}$ is either on B_{s+1} or on a branch between B_s and B_{s+1}.

Subcase 2(a): Consider when $\mathbf{e}$ is on a branch B' between B_s and B_{s+1}. Then, the larger minimal leaf used in the slide labeling algorithm is on B', and so is some not m_r. So, the smaller leaf used in the comparison of the slide algorithm is either unchanged or made smaller, and the larger leaf is unchanged. So, since $\mathbf{e}$ is a valid slide in T, $\mathbf{e}$ is a valid slide in T'.

Subcase 2(b): Consider when $\mathbf{e}$ is on B_{s+1}. For any $\mathbf{f}$ on P between B_s and B_{s+1}, $\mathbf{f} > \mathbf{e}$, since the edges of P decrease from B_s to B_{s+1}. So, $\mathbf{e}$ slides in T to some m_r with $r \leqslant s$. If $\mathbf{e}$ slides in B_{s+1} from a leaf other than m_{s+1}, then the larger label in the slide algorithm comparison made by $\mathbf{e}$ is unchanged by $\hat{\sigma}_{i,j}$. The smaller label, m_r, is either unchanged or is replaced by something smaller, so $\mathbf{e}$ is still a valid slide. Otherwise, $\mathbf{e}$ slides from m_{s+1}. If $\hat{\sigma}_{i,j}$ does not change m_{s+1}, then $\mathbf{e}$ is still valid by the previous argument. If m_{s+1} is changed, it is replaced with either i or m_s. If m_{s+1} is replaced by i, then $\mathbf{e}$ will slide from i to m_r. Since $i > m_r$, $\mathbf{e}$ is still a valid slide. If m_{s+1} is replaced by m_s, then either m_r in unchanged and is smaller than m_s, or m_r is replaced by

something smaller than itself. In either case, $\mathbf{e}$ can slide from m_s or i to the label in the position of m_r in T. Thus, $\mathbf{e}$ is a valid slide.

Case 3: Suppose $\mathbf{e}$ slides to P, is in B_l, and $j < m_l$. Then, since B_l is unchanged by $\hat{\sigma}_{i,j}$ in this case, the larger leaf used by $\mathbf{e}$ in the slide algorithm is unchanged. If the smaller leaf label is changed (such as changing from j to the new leaf m_{l-1} added next to j), it is changed to something smaller. So, the comparison is still valid, and $\mathbf{e}$ is a valid slide.

Case 4: Suppose $\mathbf{e}$ does not slide to P. Let B be the branch off of P containing $\mathbf{e}$. Then in T, $\mathbf{e}$ slides from some v_1 within B to some v_2 also within B. If $\hat{\sigma}_{i,j}$ changes v_2, it will replace v_2 with a smaller leaf label. Meanwhile, $\hat{\sigma}_{i,j}$ does not change v_1. So, in T' $\mathbf{e}$ will slide from v_1 to either v_2 or something smaller, so $\mathbf{e}$ is still a valid slide.

Finally, we consider the new edge $\mathbf{j}$. In Case 1 of Step 5, j slides from i to either m_l or m_{l-1}. In Case 2, j slides from m_l to m_{l-1}, and in Case 3, j slides from m_{i-1} to m_{i-2}. In all three cases, the leaf slid from is larger than the leaf slide to, so the edge $\mathbf{j}$ is a valid slide. Thus, every edge of T' is a valid slide, and so T' is an element of $\mathrm{Slide}^{\omega}(\underline{k}')$, meaning that $\hat{\sigma}_{i,j}$ is well-defined. □

9.3 Proof of Theorem 1.2.4

Remark 9.3.1. When performing the ψ-slide algorithm on a caterpillar tree $\mathrm{Tree}(w)$, in every comparison, the left element compared will be a nondescent leaf. This is because every descent leaf has its left edge labeled before its right and the left edge is only labeled if there is a smaller leaf to its left. Thus, a descent leaf can not be compared in this way.

Theorem 1.2.4. Let $\underline{k}$ be right-justified with m zeros, and let $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2,\ 23{-}\overline{2}{-}1)$. Then, $\mathrm{Tree}(w)$ is a valid slide tree.

Proof. Let $i_1, \ldots, i_l$ be as before. Since $\underline{k}$ is right-justified, the smallest letter present in w is $w_{i_1} = m + 1$, so in particular, $c, 1, \ldots, m$ are not used as descent leaf labels, and thus are used as nondescent labels. We first prove the claim that during Step 7 of Definition 8.1.1, we will always use the smallest remaining label for each nondescent leaf. In particular, this results in the nondescent leaf labels being in increasing order, as read from left to right.

We proceed by strong induction from left to right in the tree. For the base case, clearly we can always use c first, since there is never an edge labeled c. For the induction step, assume this is true for all nondescent leaves to the left of a nondescent leaf v.

We first consider the case where v is just to the right of an edge $\mathbf{w_{i_j}}$. Let $\mathbf{y}$ be the edge on the right of v. Note that $\mathbf{w_{i_j}} < \mathbf{y}$ by definition of w_{i_j}. We wish to show that $v < \mathbf{y}$. In fact, we will show that in the step of Definition 8.1.1 where we label v, $w_{i_{j-1}}$ (or m in the case where $j = 1$) has not yet been used as a leaf label. By Corollary 8.1.11, v has at most $m + j - 1$ nondescent leaves strictly to its left. We also know that $c, 1, 2, \ldots, m, w_{i_1}, \ldots, w_{i_{j-1}}$ are all used as nondescent labels. This means that there are at least $m+j$ nondescent labels smaller than w_{i_j}. By the inductive hypothesis, all nondescent labels thus far have been used in increasing order, so in particular, the largest such label, $w_{i_{j-1}}$ (or m when $j = 1$), has not yet been used. Since additionally $w_{i_{j-1}} \neq y$ ($m \neq y$), $w_{i_{j-1}}$ (m) can be used to label v. This completes the inductive step for the case where v is just to the right of some w_{i_j}.

Otherwise, v is a nondescent leaf not on the right side of an edge labeled w_{i_j}. Let w_{i_j} be the rightmost among the w_{i_r}'s to the left of v (if such a w_{i_j} exists). Again, the edge $\mathbf{y}$ to the right of v is larger than w_{i_j} by the definition of w_{i_j}. Notice that since $w_{i_{j+1}}$ (which exists because $i_l = n$) cannot have more than $m+j$ nondescent leaves to its left, v has no more than $m+j-1$ nondescent leaves to its left. The argument then follows identically to the previous case. This completes the inductive step. Thus, we have shown that during Step 7 of Definition 8.1.1, we will always use the smallest remaining label for each nondescent leaf.

Next, we show that for any leaf in the tree, the smallest leaf to its right is a nondescent leaf. To do this, it suffices to show that each descent leaf has a smaller nondescent leaf to its right. So, let d be some descent leaf. The edge on its right is smaller than d, so d is not any of the w_{i_j}. Let j be the smallest index such that $\mathbf{w_{i_j}}$ is to the right of d, and e the leaf on the right of $\mathbf{w_{i_j}}$. If $d < w_{i_j}$, then since d is not a w_{i_r}, there must be another one between d and w_{i_j}, which contradicts the assumption that w_{i_j} is the closest one to the left. So, $d > \mathbf{w_{i_j}} \geqslant e$, where e is a nondescent leaf label by Lemma 8.1.9. Thus, for any internal edge, the smallest leaf label to its right is a nondescent leaf.

Thus far, we have shown that for any internal edge, the smallest leaf label to its right is a nondescent leaf, and the nondescent leaves are in increasing order from left to right. So, during the slide algorithm, each comparison compares a nondescent leaf somewhere to the right against a nondescent leaf to the left, which must be smaller it. So, assuming that the leaf label is in the correct position to slide, the comparison made for that edge is valid, and so the slide is successful.

To show that leaf labels are always in the correct positions to slide, it suffices to show that every nondescent leaf label is to the right of the rightmost edge with that label (if any), and that any edges between them are larger. The set of nondescent labels that are also edge labels are precisely $w_{i_1}, \ldots, w_{i_{l-1}}$. So, by Corollary 8.1.11, each such label labels a leaf strictly to the right of the rightmost edge with that label, and all the intermediate edges are larger by the definition of the w_{i_j}'s. Thus, every slide is a valid ψ-slide, and $\mathrm{Tree}(w) \in \mathrm{Slide}^{\psi}(\underline{k})$. □

9.4 Proof of Theorem 8.3.4

Lemma 9.4.1. *Let $\underline{k}$ be a reverse-Catalan composition, and let w be a word of content $\underline{k}$. If $\mathrm{TotalRep}_w(i) + \ell_w(i) < \mathrm{z}(i)$ for some i, then there is some j for which $\mathrm{TotalRep}_w(j) + \ell_w(j) < \mathrm{z}(j)$ and the subword of w right of $\mathbf{j_r}$ contains no entries smaller than j.*

Proof. For such an i, let j be the rightmost entry in w (weakly) smaller than i. If $i = j$, we are done. Otherwise, since $j < i$, $\mathrm{z}(j) \geqslant \mathrm{z}(i)$. In addition, since j is right of every i by definition, every repeat right of all j's is also right of all i's. All but one of the $\ell_w(j)$ copies of j is counted by the term $\mathrm{TotalRep}(i)$, so $\mathrm{TotalRep}_w(j) + \ell_w(j) \leqslant \mathrm{TotalRep}_w(i) + \ell_w(i)$. Thus, we have

$$\mathrm{TotalRep}_w(j) + \ell_w(j) \leqslant \mathrm{TotalRep}_w(i) + \ell_w(i) < \mathrm{z}(i) \leqslant \mathrm{z}(j).$$

So, j is an index for which $\mathrm{TotalRep}_w(j) + \ell_w(j) < \mathrm{z}(j)$, and by construction the subword of w right of $\mathbf{j_r}$ contains no entries smaller than j. □

Lemma 9.4.2. *Let $T = \mathrm{Tree}(w)$, and let i be a nondescent leaf for which $k_i \neq 0$. Then, the edge next to $\mathbf{i_r}$ on the right is larger than i.*

Proof. Since i is a nondescent leaf, that means that it did not label a leaf in Step 5 of Definition 8.1.1. More specifically, the leaf v on the right of $\mathbf{i_r}$ was not labeled at this step, since only i could have labeled it at this stage. So, v is a nondescent leaf too, leaning the edge on its right is larger than i. □

Definition 9.4.3. Consider a tree $T = \mathrm{Tree}(w)$. Fix an i for which $k_i \neq 0$.

Let Rnd (Right Non-Descents) denote the number of nondescent leaves strictly to the right of $\mathbf{i_r}$, and Lnd (Left Non-Descents) the number of nondescent leaves to the left of $\mathbf{i_l}$.

Let z_- denote the number of j for which $k_j = 0$ and $j < i$, and z_+ the number of j for which $k_j = 0$ and $j > i$.

Then, let p_+ be the number of nondescent leaves j which do occur as edge labels in T and $j \geqslant i$. Similarly, let p_- be the number of nondescent leaves j which do occur as edge labels in T and $j < i$.

Note that in the above definition, $z_+ = z(i)$ from Definition 5.3.2.

Using these quantities, we will count the number of nondescent leaves of a tree, relative to a particular choice of i, in two different ways.

Lemma 9.4.4. *Let $\underline{k}$ be a reverse-Catalan composition, w be a word of composition $\underline{k}$, and $T = \mathrm{Tree}(w)$. Fix an $i \in [n]$ such that $k_i \neq 0$. Then,*

$$\mathrm{Lnd} + \mathrm{Rnd} + \ell_w(i) - 1 = z_+ + z_- + p_+ + p_- + 1. \tag{9.1}$$

Proof. We first will count the nondescent leaves of T in terms of their position in the tree.

First, $\ell_w(i)$ as defined above is the number of edges in the rightmost consecutive group of $\mathbf{i}$'s. So, there are $\ell_w(i) - 1$ nondescent leaves between $\mathbf{i_l}$ and $\mathbf{i_r}$. Then, Lnd and Rnd denote the number of nondescent leaves left of $\mathbf{i_l}$ and right of $\mathbf{i_r}$, respectively, by definition. So, the total number of nondescent leaves is $\mathrm{Lnd} + \mathrm{Rnd} + \ell_w(i) - 1$.

Alternatively, we can count the nondescent leaves in terms of their label values. First, any j for which $k_j = 0$ is a nondescent leaf, which contributes $z_+ + z_-$. Next, we have $p_+ + p_-$ nondescent

leaves j which do occur as edge labels, split up by whether and $j \geqslant i$ or $j < i$. Finally, c is also a nondescent leaf not included in the above categories. Since every leaf label either does label an edge or it does not, we have that the total number of nondescent leaves is $z_+ + z_- + p_+ + p_- + 1$.

Thus, we have proven (9.1). □

Lemma 9.4.5. *The p_+ nondescent leaves that slide in T and are weakly larger than i all occur to the right of $\mathbf{i_r}$.*

Proof. For $j = i$, clearly the leaf i appears right of $\mathbf{i_r}$ (regardless of whether i is a nondescent leaf). Then, note that for all such $j > i$, the leaf j must be to the right of $\mathbf{i_r}$, or else there is an edge $\mathbf{j}$ left of $\mathbf{i_r}$, and the rightmost such edge $\mathbf{j}$ and the edge on its right (which is larger than j by Lemma 9.4.2) would form a $23{-}\overline{2}{-}1$ pattern with $\mathbf{i_r}$. □

Theorem 8.3.4. Let $\underline{k}$ be a reverse-Catalan composition of n, and let w be a word of composition $\underline{k}$. Then, $\mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$ if and only if $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$ and

$$\mathrm{TotalRep}_w(i) + \ell_w(i) \geqslant \mathrm{z}(i) \tag{9.2}$$

for all i such that $k_i > 0$.

Proof. ($\Longrightarrow$) The pattern avoidance condition is given by Lemma 8.1.8.

We will prove the second condition by the contrapositive. That is, suppose $\mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$, $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$, but there is some edge label i for which w does not satisfy (8.1). By Lemma 9.4.1, we may assume that no edges smaller than i occur to the right of $\mathbf{i_r}$.

By definition, we have $z(i) = z_+$. Next, consider the branch B starting at $\mathbf{i_r}$, and recall that any branch of a tree has one more leaf than internal edge. Let $|B|$ denote the number of internal edges in this branch. Then if D is the number of descent leaves in B, we have

$$\mathrm{Rnd} + D = |B| + 1.$$

Each descent leaf x is just to the right of the rightmost $\mathbf{x}$ by definition. So, rewriting the above equation as $\mathrm{Rnd} - 1 = |B| - D$ we find that there are $\mathrm{Rnd} - 1$ remaining edges, which all are either repeats of edge labels to their right, or are the rightmost edges with a label, where the label is a nondescent leaf to the right of i. We also assumed that no edges smaller than $\mathbf{i}$ occur right of $\mathbf{i_r}$, so $\mathrm{Rnd} - 1$ equals the total number of repeats right of $\mathbf{i_r}$ plus the number of nondescent leaves in $\mathrm{Tree}(w)$ larger than i that label an edge (which by Lemma 9.4.5 all occur right of $\mathbf{i_r}$). Symbolically, $\mathrm{Rnd} - 1 = \mathrm{TotalRep}_w(i) + p_+$, or equivalently, $\mathrm{TotalRep}_w(i) = \mathrm{Rnd} - p_+ - 1$. Using our assumption that (8.1) is not satisfied, and applying Lemma 9.4.4, we have:

$$\begin{aligned}
\mathrm{TotalRep}_w(i) + \ell_w(i) &< z(i) \\
\mathrm{Rnd} - p_+ - 1 + \ell_w(i) &< z_+ \\
\mathrm{Rnd} + \ell_w(i) - z_+ - p_+ &< 1 \\
z_- + p_- - \mathrm{Lnd} + 2 &< 1 \\
z_- + p_- + 1 &< \mathrm{Lnd}.
\end{aligned}$$

Thus, the total number of nondescent leaves left of $\mathbf{i_l}$ is greater than the number of nondescent leaf labels smaller than i (including c). Since the caterpillar labeling algorithm labels nondescent leaves from left to right, and always chooses one of the two smallest unused labels, there is a nondescent leaf $d \geqslant i$ to the left of $\mathbf{i_l}$. If $d = i$, then i cannot label the edges $\mathbf{i_l}$ through $\mathbf{i_r}$, which contradicts our assumption. So, we may assume $d > i$. Let $\mathbf{j}$ be the edge immediately to the left of $\mathbf{i_l}$. By definition of $\mathbf{i_l}$, $\mathbf{j} \neq i$. We now consider the two cases of whether or not $\mathbf{j}$ is larger than i:

Case 1: $\mathbf{j} > i$. In this case, $\mathbf{j}$ must be a descent leaf, which in particular means that d is to the left of $\mathbf{j}$ as well. Let $\mathbf{e}$ be the edge to the right of d. If $\mathbf{e} \neq i$, then in the slide algorithm $\mathbf{e}$ will compare i (or something smaller if a smaller leaf than i exists right of $\mathbf{e}$) to d, and since $d > i$ this would be an invalid slide. So, we must have $\mathbf{e} = i$. Then, since $\mathbf{e} < \mathbf{j}$, there must be a nondescent leaf f between $\mathbf{e}$ and $\mathbf{j}$. Consider the step where d was labeled in the caterpillar labeling algorithm. At that step, d and i were the two smallest remaining labels, since d was the

one chosen by the algorithm, $i < d$, and neither had yet been used. So, $f > i$. Thus, we can repeat the above argument, replacing d with f. Since the tree is finite, we will eventually reach a step where $\mathbf{e} \neq i$, since there is a $\mathbf{j} \neq i$ between d and $\mathbf{i_l}$. Thus, we do not have a valid slide tree.

Case 2: $\mathbf{j} < i$. Moreover, we have $\mathbf{j} < i < d$. We again let $\mathbf{e}$ be the edge to the right of d. Since $\mathbf{e}$ cannot be both i and $\mathbf{j}$, when $\mathbf{e}$ slides in the slide algorithm it will see at least one of the leaves i and j to its right, but since it is sliding to d it will compare something smaller than d to d, which is not allowed. Thus, we again do not have a valid slide tree.

We have now shown that in both cases we cannot have a valid slide tree, which contradicts our original assumption. Thus, if $\mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$, w must satisfy (8.1) for all i.

($\Longleftarrow$) We induct on how far $\underline{k}$ is from being right-justified. More precisely, we induct on $\#\{(i < j) \,|\, k_i \neq 0, k_j = 0\}$. The base case is the right justified case given in Theorem 1.2.4. For the inductive step, it suffices to show that if the Proposition is true for $\underline{k}' = (\ldots, 0, l, \ldots)$, then it is also true for $\underline{k} = (\ldots, l, 0, \ldots)$.

Let $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$, for some w satisfying (8.1) for all i where $k_i \neq 0$, and where $\underline{k}$ is some non-right justified composition with $k_i = 0$ and $k_{i-1} \neq 0$ for some i. Let $\underline{k}' = (k_1, \ldots, k_{i-2}, 0, k_{i-1}, k_{i+1}, \ldots, k_n)$, and let $w' = (i, i-1)w$ be the word obtained from w by replacing all $i-1$'s with i's. Then, since the letters of w and w' have all the same relative orders, and the only change in (8.1) for w' is that for our chosen i, the right-hand side is 1 smaller. So, w' satisfies the necessary condition, and moreover, (8.1) is strict for the chosen i. Thus, by the inductive hypothesis, $T' = \mathrm{Tree}(w') \in \mathrm{Cat}^{\psi}(\underline{k}')$.

Next, we will use this to show that $T = \mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$. We will first show that the same leaf labels are arranged the same way in T as in T', except possibly for swapping the leaves i and $i-1$. So, consider what could go wrong in the slide algorithm by either leaving the leaves alone or swapping $i-1$ and i. If we do not swap the leaves i and $i-1$, then every comparison made in the slide algorithm is the same in both trees, and so all slides are valid. The only possible issue is that $i-1$ may not be able to label the appropriate edges. That is, in T', $i-1$ might be to the left of an edge $\mathbf{i}$, or there may be an edge between the leaf $i-1$ and the rightmost edge $\mathbf{i}$ that is smaller

than $i-1$. Alternatively, if we do swap i and $i-1$, then $i-1$ is in the correct position, and every comparison using at most one of i and $i-1$ is the same comparison. So, the only possible issue is if some edge in T' compares i to $i-1$, then T would not be a valid caterpillar tree. We consider different cases on where leaf $i-1$ can be relative to leaf i and $\mathbf{i_r}$ in T'.

Case 1: $i-1$ is between i and $\mathbf{i_r}$ in T'. In this case, we do not swap i and $i-1$ when forming T. In the slide algorithm, every edge between $\mathbf{i_r}$ and i is contracted before i slides, so every edge between $i-1$ and $\mathbf{i_r}$ is also contracted. Thus, $i-1$ can slide across $\mathbf{i_r}$ in T.

Case 2: $i-1$ is to the right of i. In this case, we swap i and $i-1$. Since $i-1$ is to the right of i in T', no edge compared the two in T'. So, no edge compares $i-1$ and i against each other in T either, and so all comparisons in T are valid as well.

Case 3: $i-1$ is to the left of $\mathbf{i_r}$. In this case, we swap i and $i-1$. We will show that the furthest left that $i-1$ can be in T' is immediately to the left of $\mathbf{i_l}$. By a similar argument to the one we made in the forwards direction of this proof, the number of nondescent leaves right of $\mathbf{i_r}$ is at least the total number of repeats right of $\mathbf{i_r}$, plus the number of nondescent edge labels larger than i (which by Lemma 9.4.5 all occur to the right of $\mathbf{i_r}$), plus one (because there is one more leaf than edge). That is, $\mathrm{Rnd} \geqslant \mathrm{TotalRep}_w(i) + p_+ + 1$. As noted above, for w' and our chosen i, (8.1) is strict. So, combining these two inequalities and using Lemma 9.4.4, we have:

$$\begin{aligned}
\mathrm{Rnd} - p_+ - 1 + \ell_w(i) &\geqslant \mathrm{TotalRep}_w(i) + \ell_w(i) \geqslant z_+ + 1 \\
\mathrm{Rnd} + \ell_w(i) - z_+ - p_+ &\geqslant 2 \\
z_- + p_- - \mathrm{Lnd} + 2 &\geqslant 2 \\
z_- + p_- &\geqslant \mathrm{Lnd}.
\end{aligned}$$

Thus, the number of nondescent leaves left of $\mathbf{i_l}$ is at most the total number of nondescent leaves (excluding c) less than i. We again note that in our $\mathrm{Tree}()$ algorithm, we label the nondescent leaves from left to right, and at each step, we chose one of the two smallest remaining unused labels. So, the first time $i-1$ (the largest nondescent label smaller than i) would appear as an

option would be the rightmost nondescent leaf left of $\mathbf{i_l}$. If the leaf immediately to the left of $\mathbf{i_l}$ is a nondescent leaf, then we are done. Otherwise, let v be the rightmost nondescent leaf left of $\mathbf{i_l}$. Since any leaves between v and $\mathbf{i_l}$ are descent leaves, the edges between them (including $\mathbf{i_l}$ itself) are strictly descreasing from left to right. Thus, the edge on the right of v is strictly larger than $i-1$. So, when we label v in Step 7 of Definition 8.1.1, our options are two labels, the larger of which is at most $i-1$, both of whom are smaller than the edge to the right. So, we will choose the smaller of the two, and not label v by $i-1$.

So, we have shown that if $i-1$ is left of $\mathbf{i_r}$, then the edge to the immediate right of $i-1$ is an edge labeled $\mathbf{i}$. So, the (first) label that will compare something to $i-1$ is i itself, which will compare some other label $l \notin \{i-1, i\}$ to $i-1$. After this point, every edge between $i-1$ and i is contracted, so no other label can compare one to the other. If we swap $i-1$ and i in T, then instead $i-1$ will compare l to i, which is fine for the ψ-slide algorithm. Thus, swapping the leaves $i-1$ and i when forming T does not introduce any problems.

Therefore, in any case, we can take the valid caterpillar tree $T' \in \mathrm{Cat}^{\psi}(\underline{k}')$ to form another tree T with edge sequence w without introducing any disallowed slides, meaning $T \in \mathrm{Cat}^{\psi}(\underline{k})$. □

9.5 Proof of Theorem 8.4.2

Lemma 9.5.1. *Let $\underline{k}$ be a reverse-Catalan composition, and let w be a word of content $\underline{k}$. If $\mathrm{BigRep}_w(i) < \mathrm{z}(i)$ for some i, then there is some j for which $\mathrm{BigRep}_w(j) < \mathrm{z}(j)$ and the subword of w right of the rightmost j contains no entries smaller than j.*

Proof. For such an i, let j be the rightmost entry in w (weakly) smaller than i. Since $j \leqslant i$, $\mathrm{z}(j) \geqslant \mathrm{z}(i)$. Then, since j is the rightmost entry right of i, there are no entries l with $j < l < i$ right of j. So, everything after j is also larger than i, so $\mathrm{BigRep}_w(j) \leqslant \mathrm{BigRep}_w(i)$. Thus, we have $\mathrm{BigRep}_w(j) \leqslant \mathrm{BigRep}_w(i) < \mathrm{z}(i) \leqslant \mathrm{z}(j)$. The second condition is satisfied by definition of how we chose j. □

Finally, we prove the full classification for the ω caterpillar trees.

Lemma 9.5.2. *Let w be a word of content $\underline{k}$ and $T = \mathrm{Tree}(w) \in \mathrm{Slide}^{\psi}(\underline{k})$. If in Step 7 of Definition 8.1.1, the second smallest label was ever used for a leaf, then $T \notin \mathrm{Slide}^{\omega}(\underline{k})$.*

Proof. Let v be a leaf that was labeled by the second smallest remaining unused label in Step 7 of Definition 8.1.1, and $\mathbf{e}$ the edge on its right. For this to have happened, we must have $\mathbf{e} < v$. However, since v is a nondescent leaf, $\mathbf{e}$ slides before the edge to the left of v, and so $\mathbf{e}$ slides to v. Since $\mathbf{e} < v$, $\mathbf{e}$ is not a valid ω-slide, and thus $T \notin \mathrm{Slide}^{\omega}(\underline{k})$. □

As a consequence of this, we know that in any ω caterpillar tree, the nondescent leaves always increase from left to right.

Lemma 9.5.3. *Let $T \in \mathrm{Cat}^{\psi}(\underline{k})$, and i a label such that $k_i \neq 0$. Then the number of leaves right of $\mathbf{i_r}$ that do not label any edges weakly right of $\mathbf{i_r}$ is $\mathrm{TotalRep}_w(i) + 1$.*

Proof. Let B be the branch starting at $\mathbf{i_r}$. First, any branch has one more leaf than internal edges. Each distinct edge label in B corresponds to exactly one leaf label. So, the number of leaves that do not label any edges in B is the number of repeated edges in B, plus one. That is, there are $\mathrm{TotalRep}_w(i) + 1$ such leaves. □

Theorem 8.4.2. Let $\underline{k}$ be a reverse-Catalan composition, and let w be a word of composition $\underline{k}$. Then, $\mathrm{Tree}(w) \in \mathrm{Cat}^{\omega}(\underline{k})$ if and only if $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$ and

$$\mathrm{BigRep}_w(i) \geqslant \mathrm{z}(i) \tag{9.3}$$

for all i such that $k_i > 0$.

Proof. ($\implies$) The avoidance condition is satisfied by Lemma 8.1.8 and the fact that $\mathrm{Cat}^{\omega}(\underline{k}) \subseteq \mathrm{Cat}^{\psi}(\underline{k})$. Next, assume that $\mathrm{Tree}(w)$ is a valid ω-slide tree. Suppose, aiming for contradiction, that $\mathrm{BigRep}_w(i) < \mathrm{z}(i)$ for some i with $k_i \neq 0$. Then by Lemma 9.5.1, there is some j for which $\mathrm{BigRep}_w(j) < \mathrm{z}(j)$ and there are no entries smaller than j to the right of j in w. We will show that j cannot perform an ω $\underline{k}$-slide.

For j to be able to perform an ω slide, it is necessary for there to be a leaf m to the right of the edge $\mathbf{j}$ with $m < \mathbf{j}$. Such an m would be a nondescent leaf, since there is no edge $m < \mathbf{j}$ to the right of j. Note that for any l, if $k_l = 0$, then l is a nondescent leaf. This means that at least $\mathrm{z}(j)$ nondescent leaf labels are larger than j. By Lemma 9.5.2, this means the $z(j)$ rightmost nondescent leaves are all larger than j. Since $\mathbf{j_r}$ has no edges smaller than j to its right, $\mathrm{BigRep}_w(j) = \mathrm{TotalRep}_w(j)$. So, by Lemma 9.5.3, there are $\mathrm{BigRep}_w(j) + 1$ leaves right of $\mathbf{j_r}$ that do not label an edge weakly right of $\mathbf{j_r}$. Since we assumed that $\mathrm{z}(j) \geqslant \mathrm{BigRep}_w(j) + 1$, there is no room for a nondescent label $m < j$ to the right of $\mathbf{j_r}$, and so j cannot perform an ω-slide. Thus, $\mathrm{Tree}(w) \notin \mathrm{Cat}^{\omega}(\underline{k})$, and this concludes the forwards implication.

($\Longleftarrow$) Suppose that $w \in \mathrm{Av}_{\underline{k}}(2{-}1{-}2, 23{-}\overline{2}{-}1)$ and $\mathrm{BigRep}_w(i) \geqslant \mathrm{z}(i)$ for all i where $k_i \neq 0$. Since (8.2) is a stronger condition than (8.1), Theorem 1.2.5 gives us that $\mathrm{Tree}(w) \in \mathrm{Cat}^{\psi}(\underline{k})$. We want to show that $\mathrm{Tree}(w) \in \mathrm{Cat}^{\omega}(\underline{k})$ as well. To do this, it suffices to show that for each j that labels an edge, $\mathbf{j_r}$ is a valid ω-slide, since that ensures that j is larger than the largest thing it ever compares in the slide algorithm.

By Lemma 9.5.3, there are $\mathrm{TotalRep}_w(j) + 1$ leaf labels right of $\mathbf{j_r}$ that do not label an edge weakly right of $\mathbf{j_r}$. If any of these leaves label an edge to the left of j, they must be smaller than j, since $\mathrm{Tree}(w) \in \mathrm{Slide}^{\psi}(\underline{k})$. So, any of these $\mathrm{TotalRep}_w(j) + 1$ leaves that are larger than j must not label any edges at all. Since there are exactly $\mathrm{z}(j)$ of these in the whole tree, and $\mathrm{TotalRep}_w(j) + 1 \geqslant \mathrm{BigRep}_w(j) + 1 > \mathrm{z}(j)$, there is at least one leaf to the right of j smaller than j, so j is a valid ω-slide. Thus, $\mathrm{Tree}(w) \in \mathrm{Cat}^{\omega}(\underline{k})$. $\square$

Bibliography

[1] Eric Babson and Einar Steingrímsson. Generalized permutation patterns and a classification of the Mahonian statistics. *Sém. Lothar. Combin.*, 44(B44b), 2000.

[2] David Bevan. Permutation patterns: basic definitions and notation, 2015.

[3] Joshua Brakensiek, Christopher Eur, Matt Larson, and Shiyue Li. Kapranov degrees, 2024.

[4] Guido Castelnuovo. Numero delle involuzioni razionali giacenti sopra una curva di dato genere. *Atti Accad. Naz. Lincei Rend. Lincei Mat. Appl.*, pages 130–133, 1889.

[5] Renzo Cavalieri. Moduli spaces of pointed rational curves, July 2016. Lecture Notes from Combinatorial Algebraic Geometry program at the Fields Institute.

[6] Renzo Cavalieri, Maria Gillespie, and Leonid Monin. Projective embeddings of $\overline{M}_{0,n}$ and parking functions. *J. Combin. Theory Ser. A*, 182, 12 2019.

[7] Gavril Farkas and Carl Lian. Linear series on general curves with prescribed incidence conditions. 2021. arXiv:2105.09340.

[8] William Fulton. *Young tableaux*. London Math Soc. Student Texts **35**. Cambridge University Press, 1997.

[9] Maria Gillespie. Variations on a theme of Schubert calculus. In *Recent Trends in Algebraic Combinatorics*, Association for Women in Mathematics Series, pages 115–158. Springer, 2019. Edited by Hélène Barcelo, Gizem Karaali, and Rosa Orellana.

[10] Maria Gillespie, Sean T. Griffin, and Jake Levinson. Degenerations and multiplicity-free formulas for products of ψ and ω classes on $\overline{M}_{0,n}$. *Math. Z.*, 304(56), 2023.

[11] Maria Gillespie, Sean T. Griffin, and Jake Levinson. Lazy tournaments and multidegrees of a projective embedding of $\overline{M}_{0,n}$. *Comb. Theory*, 3(1), 2023.

[12] Maria Gillespie and Andrew Reimer-Berg. A generalized RSK for enumerating linear series on n-pointed curves. *Algebraic Combinatorics*, 6(1):1–16, 2023.

[13] Marvin Anas Hahn and Shiyue Li. Intersecting psi-classes on tropical hassett spaces. *Comb. Theory*, 2(3), 2022.

[14] Allen Hatcher. *Algebraic topology*. Cambridge Univ. Press, Cambridge, 2000.

[15] Mikhail M Kapranov. Veronese curves and Grothendieck-Knudsen moduli space $\overline{M}_{0,n}$. *J. Algebraic Geom.*, 2:239–262, 1993.

[16] Sean Keel and Jenia Tevelev. Equations for $\overline{M}_{0,n}$. *Internat. J. Math.*, 20(09):1159–1184, 2009.

[17] Michael Kerber and Hannah Markwig. Intersecting psi-classes on tropical $\overline{M}_{0,n}$. *Int. Math. Res. Not. IMRN*, 2009(2):221–240, 12 2008.

[18] Joachim Kock and Israel Vainsencher. *An Invitation to Quantum Cohomology*. Birkhäuser, Boston, MA, 2007.

[19] Carl Lian and Saskia Solotko. Curves in projective space and RSK, 2025.

[20] Leonid Monin and Julie Rana. Equations of $\overline{M}_{0,n}$. In Gregory G. Smith and Bernd Sturmfels, editors, *Combinatorial Algebraic Geometry: Selected Papers From the 2016 Apprenticeship Program*, volume 80 of *Fields Inst. Commun.*, pages 113–132. Springer, New York, 2017.

[21] David Mumford. *Towards an Enumerative Geometry of the Moduli Space of Curves*, pages 271–328. Birkhäuser Boston, Boston, MA, 1983.

[22] Lara Pudwell. Enumeration schemes for permutations avoiding barred patterns. *Electron. J. Combin.*, 17(R29), 2010.

[23] Andrew Reimer-Berg. Insertion algorithms and pattern avoidance on trees arising in the Kapranov embedding of $\overline{M}_{0,n+3}$, 2026+. In preparation.

[24] Victor Reiner and Mark Shimozono. Percentage-avoiding, northwest shapes and peelable tableaux. *Journal of Combinatorial Theory, Series A*, 82(1):1–73, 1998.

[25] Rob Silversmith. Cross-ratio degrees and perfect matchings. *Proc. Amer. Math. Soc.*, 150, 9 2022.

[26] Richard Stanley and Sergey Fomin. *Enumerative Combinatorics: Volume 2.* Cambridge University Press, 1999.

[27] Einar Steingrímsson. Generalized permutation patterns – a short survey. In Steve Linton, Nik Ruškuc, and Vincent Vatter, editors, *Permutation Patterns*, London Math. Soc. Lecture Note Ser., pages 137–152. Cambridge Univ. Press, Cambridge, 2010.