



Automated Bijections with Combinatorial Exploration

by

Jón Steinn Elíasson

Thesis of 30 ECTS credits submitted to the School of Computer Science
at Reykjavík University in partial fulfillment
of the requirements for the degree of
Master of Science (M.Sc.) in Computer Science

January 2022

Examining Committee:

Henning Ulfarsson, Supervisor
Assistant Professor, Reykjavík University, Iceland

Christian Bean, Co-advisor
Postdoctoral Researcher, Reykjavík University, Iceland

Jay Pantone, Examiner
Assistant Professor, Marquette University, United States

Émile Nadeau, Examiner
Doctoral Student, Reykjavík University, Iceland

Copyright
Jón Steinn Elíasson
January 2022

Automated Bijections with Combinatorial Exploration

Jón Steinn Elíasson

January 2022

Abstract

Bijections appear in most areas of mathematics. They are of particular importance in the field of combinatorics, where they are a way of enumerating families of objects. The aim of this thesis was to develop a fully automated and domain agnostic bijection search built on top of an existing automated combinatorial specification framework. We define a binary relation on specifications that structurally associates them. When satisfied, a bijection can be constructed between their classes. This theoretical foundation is accompanied by a search algorithm for specifications satisfying this relation. The algorithm utilizes dynamic programming and backtracking. If a bijection is found it can map objects between the classes of the specifications, in both directions. The search algorithm was mainly applied to the domain of permutation patterns, where a total of 189 bijections were discovered, excluding symmetries and compositions. In many cases no previous bijections were known. Some cross-domain bijections were also discovered. As far as we know, this is the first ever fully automated bijection framework, with prior attempts requiring preliminary mathematical work. This work offers substantial structural insight into classes and can be considered a significant innovation in automated mathematics.

Sjálfvirknivæddar gagntækar varpanir með fléttufræðilegri könnun

Jón Steinn Elíasson

janúar 2022

Útdráttur

Gagntækar varpanir koma við sögu á flestum sviðum stærðfræðinnar. Þær eru sérstaklega mikilvægar innan fléttufræði þar sem þær geta talið fjölskyldur hluta. Markmið þessa verkefnis var að þróa leit að gagntækum vörpunum sem er óháð þeim hlutum sem unnið er með, að öllu leyti sjálfvirk og byggð á kerfi sjálfvirkra fléttufræðilegra forskrifta. Við skilgreinum tvístæð vensl á forskriftir sem vensla þær út frá uppbyggingu sem nota má til að mynda gagntæka vörpun þeirra á milli. Þessum fræðilega grunni fylgir leitarreiknirit sem leitar að vensluðum forskriftum. Reikniritið notast við kvika bestun og hopun. Ef gagntæk vörpun finnst þá má varpa stökum flokka forskriftanna í báðar áttir. Leitinni var einkum beitt á umraðanamynstur, þar sem 189 gagntækar varpanir fundust, að undanskildum samhverfum og samskeytingum. Í mörgum tilfellum voru engar gagntækar varpanir þekktar áður. Einnig voru nokkrar gagntækar varpanir uppgötvaðar milli ólíkra hluta. Eftir því sem best er vitað er þetta fyrsta kerfi sem finnur og myndar gagntækar varpanir sem er að öllu leyti sjálfvirkt en fyrri kerfi kröfðust stærðfræðilegs undirbúnings. Þessi vinna gefur mikla innsýn í uppbyggingu flokka og er þýðingarmikil nýjung í heimi sjálfvirkrar stærðfræði.

Automated Bijections with Combinatorial Exploration

Jón Steinn Elíasson

Thesis of 30 ECTS credits submitted to the School of Computer Science
at Reykjavík University in partial fulfillment of
the requirements for the degree of
Master of Science (M.Sc.) in Computer Science

January 2022

Student:

.....
Jón Steinn Elíasson

Examining Committee:

.....
Henning Ulfarsson

.....
Christian Bean

.....
Jay Pantone

.....
Émile Nadeau

The undersigned hereby grants permission to the Reykjavík University Library to reproduce single copies of this Thesis entitled **Automated Bijections with Combinatorial Exploration** and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the Thesis, and except as herein before provided, neither the Thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatsoever without the author's prior written permission.

.....
date

.....
Jón Steinn Elíasson
Master of Science

*I dedicate this thesis to my wife, Sólborg Erla Ingvarsdóttir, and our sons,
Ingvar Sölvi Jónsson and Þorkell Ari Jónsson.*

Acknowledgements

First, I would like to thank my wife Sólborg for all her support and for proofreading this thesis multiple times.

I would also like to thank my advisors Henning and Christian. Henning for all his guidance throughout my studies, his suggestion of this topic and help with the thesis. Christian for assisting with implementations and getting acquainted with the Combinatorial Exploration framework. Furthermore, I would like to thank Émile and Jay for all their help.

Last but not least, I want to thank my mother-in-law and my late father-in-law for helping with the kids during my studies.

Contents

Acknowledgements	xi
Contents	xii
List of Figures	xv
List of Tables	xvii
List of Algorithms	xviii
1 Introduction	1
2 Background	3
2.1 Combinatorics	3
2.2 Permutations	6
2.3 Gridded permutations	9
2.4 Tilings	10
2.5 Combinatorial Exploration	11
2.6 TileScope	12
2.6.1 Assumptions	12
2.6.2 Fusion	13
3 Strategies for TileScope	15
3.1 Column reverse	16
3.2 Column permutation	17
3.3 Sliding	19
4 Parallel specifications	27
4.1 Specification graphs	28
4.2 Parallel specifications	30
4.3 Matching order	33
4.4 Path matching	33
4.5 The parallel algorithm	36
5 Parallel bijections	39
5.1 Object parsing	40
5.2 Parallel bijections	42
5.3 Nonbijective rules	44
5.4 The parallel bijection algorithm	47

6	Bijection search	49
6.1	The bijection search algorithm	49
6.2	Fusion and assumptions	50
7	Results	51
7.1	Cross-domain successes	51
7.2	Permutation classes	52
7.2.1	Avoiding one pattern of size 3	54
7.2.2	Avoiding two patterns of size 3	56
7.2.3	Avoiding three patterns of size 3	56
7.2.4	Avoiding one pattern of size 3 and one of size 4	57
7.2.5	Avoiding one pattern of size 4	57
7.2.6	Experimental classes	58
8	Conclusion	59
8.1	Future work	59
8.2	Open problems	59
8.3	Concluding remarks	60
	Bibliography	61
A	Minimal working example	63
B	Experimental classes	65
B.1	Experimental class I	65
B.2	Experimental class II	66
B.3	Experimental class III	66
B.4	Experimental class IV	66
B.5	Experimental class V	67
B.6	Experimental class VI	68
B.7	Experimental class VII	68
B.8	Experimental class VIII	68
B.9	Experimental class IX	68
B.10	Experimental class X	69
B.11	Experimental class XI	69
B.12	Experimental class XII	69
B.13	Experimental class XIII	69
B.14	Experimental class XIV	70
B.15	Experimental class XV	70
B.16	Experimental class XVI	70
B.17	Experimental class XVII	70
B.18	Experimental class XVIII	71
B.19	Experimental class XIX	71
B.20	Experimental class XX	71
B.21	Experimental class XXI	72
B.22	Experimental class XXII	72
B.23	Experimental class XXIII	72
B.24	Experimental class XXIV	72
B.25	Experimental class XXV	73

B.26 Experimental class XXVI	73
C Experimental setup	75
D Connections created by bijections	77

List of Figures

2.1	A bijection between binary strings of size 3 avoiding consecutive 1's and subsets of $\{1, 2, 3\}$ containing no consecutive numbers.	3
2.2	A grid of size 5 with three raised squares and a path from left to right. . .	4
2.3	A tree representation of a formal grammar for passable $2 \times n$ grids.	5
2.4	The mapping and graphical representation of the permutation 1423.	6
2.5	The permutation 356214 with an occurrence of 4213 circled.	8
2.6	The gridded permutation $87^{(0,2)}1^{(1,0)}6^{(1,2)}2^{(1,0)}4^{(3,1)}3^{(3,0)}5^{(4,2)}$	10
2.7	A 3×2 tiling with $\mathcal{O} = \{1^{(0,0)}, 12^{(1,1)}, 21^{(1,1)}, 3^{(0,1)}1^{(1,0)}2^{(2,1)}\}$ and $\mathcal{R} = \{\{1^{(1,1)}\}, \{2^{(1,1)}1^{(2,0)}\}\}$	11
2.8	A tiling \mathcal{T} with $\text{Grid}(\mathcal{T})$ isomorphic to $\text{Av}_{\geq 1}(132)$	11
2.9	The tiling $((2, 1), \{12^{(0,0)}, 1^{(0,0)}23^{(1,0)}, 123^{(1,0)}\}, \emptyset)$ with and without an assumption.	12
2.10	By rearranging the assumptions on the left we get the assumptions on the right.	13
2.11	A tiling where the fusion strategy applies.	13
3.1	The pair of tilings that motivated the sliding strategy. The one above is the one encountered when mimicking the strategies of a specification for $\text{Av}(1234)$ to a $\text{Av}(1432)$ root while the one below is the one we needed to have identical specification in terms of strategies and structure.	15
3.2	The column reverse of a gridded permutation.	16
3.3	The column reverse of a tiling.	16
3.4	The column permutation of a gridded permutation.	18
3.5	The column permutation of a tiling.	18
3.6	The sliding of $3594^{(0,0)}62^{(1,0)}718^{(2,0)}$	21
3.7	Typical slidable tilings.	22
4.1	The specification graph for a specification for $\text{Av}(132)$	28
4.2	A specification graph on the left and a specification path from the graph, $e_1e_2 \cdots e_7 = 2356235$, on the right with nonequivalent steps, $\{1, 3, 5, 7\}$, as solid arrows.	29
4.3	The paths $2^{\sqcup}4^{\times}2^{\sqcup}4^{\times}$ in the left graph and $2^{\sqcup}5^{\oplus}7^{\times}9^{\sqcup}11^{\times}$ in the right graph are parallel specification paths.	31
4.4	The specification graphs of two parallel specifications.	32
4.5	The recursion tree that shows that the empty rooted paths from the specifications graphs in Figure 4.4 are matchable.	32
5.1	The atomic partitioning of 3241 for the specification graph in Figure 4.1. Double underlining is used to highlight paths to terminal classes. The dotted lines represent paths not taken by parental objects.	41

5.2	The specification graphs of specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.	43
5.3	The parallel map of 132 for the specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.	45
5.4	Two equivalent fusion rules.	46
6.1	Labels B and E falsely matched because of a recursion to their respective parents that then fails to match the next children pair. In this case A and D might not be equinumerous.	50
7.1	The tiling used instead of $\text{Av}(231, 312)$ to deal with the sequence offset of binary strings.	52
7.2	The tiling used instead of $\text{Av}(231, 312, 321)$ to deal with the sequence offset of binary strings with no consecutive 1's.	52
7.3	The tree representation of the formal grammar of the specification found for binary strings avoiding consecutive 1's.	53
7.4	The tree representation of the formal grammar of the specification found for $\text{Av}_{\geq 1}(231, 312, 321)$ after placing and factoring out the top point.	53
7.5	The connections created by bijections found for classes avoiding two patterns of size 3.	56
7.6	The connections created by bijections found for classes avoiding three patterns of size 3.	57
7.7	The connections created by bijections found for the second Wilf class of permutations avoiding one pattern of size 3 and one of size 4.	57
7.8	The connections created by bijections found for the Wilf class of $\text{Av}(1234)$, $\text{Av}(1243)$, $\text{Av}(1432)$ and $\text{Av}(2143)$	58
D.1	The connections created by bijections for experimental class I.	77
D.2	The connections created by bijections for experimental class II.	77
D.3	The connections created by bijections for experimental class III where we failed to connect one permutation class.	77
D.4	The connections created by bijections for experimental class IV.	78
D.5	The connections created by bijections for experimental class V.	78
D.6	The connections created by bijections for experimental class IX.	78
D.7	The connections created by bijections for experimental class X.	78
D.8	The connections created by bijections for experimental class XI.	78
D.9	The connections created by bijections for experimental class XIII.	79
D.10	The connections created by bijections for experimental class XIV.	79
D.11	The connections created by bijections for experimental class XV.	79
D.12	The connections created by bijections for experimental class XVII.	79
D.13	The connections created by bijections for experimental class XVIII.	79
D.14	The connections created by bijections for experimental class XIX.	79
D.15	The connections created by bijections for experimental class XX.	79
D.16	The connections created by bijections for experimental class XXI.	79
D.17	The connections created by bijections for experimental class XXII.	80
D.18	The connections created by bijections for experimental class XXIII.	80
D.19	The connections created by bijections for experimental class XXVI.	80

List of Tables

2.1	The eight symmetry maps for permutations interpreted with reverse and inverse, dihedral group D_4 and an example for $\pi = 1423$	7
2.2	The seven lexicographically minimal representations of symmetry classes of permutation classes avoiding a single classical pattern of size 4. They are grouped into three Wilf classes.	9
2.3	The fusion rules for some common assumptions in terms of generating functions.	14
4.1	The bijections from Figure 4.5 interpreted with matching order.	34
5.1	The non-root classes of specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.	43
5.2	The matching order for the specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.	43
5.3	The map for Fusion interpreted with vertical splitting lines.	46
5.4	The indexed map for a fusion rule.	46
7.1	The inputs and outputs up to size 3 of an automated bijection between binary strings avoiding consecutive 1's and $\text{Grid}(\mathcal{T})$, where \mathcal{T} is the tiling for $\text{Av}_{\geq 1}(231, 312, 321)$ after placing and factoring out the top point. The corresponding permutation is also shown.	54
7.2	The classes and their matching for the parallel specifications of $\text{Av}(123)$ and $\text{Av}(132)$	55
7.3	Rules for the parallel specifications of $\text{Av}(123)$ and $\text{Av}(132)$	56
7.4	The inputs and outputs of size 4 for the bijection between 123 and 132 avoiding permutations.	56

List of Algorithms

3.1	The sliding algorithm	20
4.1	The parallel algorithm	37
5.1	The parallel bijection algorithm.	48

Chapter 1

Introduction

The name combinatorics is derived from the word combinations. It is the field of mathematics concerned with the study of discrete structures. One of its most prominent subfields is enumerative combinatorics. How many unique finite structures of a specific type and size exist? This is the fundamental question of enumerative combinatorics. Two of the most important tools in the enumerative combinatorialist's toolkit are generating functions and bijections.

Permutations are an example of a finite structure. A permutation pattern is subsequence obeying certain conditions. The study of permutation patterns can be traced to MacMahon [1] where he enumerated permutations that can be divided into two decreasing subsequences. In contemporary notation, he was describing the avoidance of the pattern 123. The permutations sortable by a stack were shown to be those avoiding the pattern 231 in Knuth [2]. That was the catalyst for the surge of interest in the study of permutation patterns. A second catalyst was the first bijections between pattern avoiding permutation classes in Simion and Schmidt [3].

The first automated enumeration method, called enumeration schemes, was done by Zeilberger [4] and later extended by Vatter [5]. A more recent approach, Combinatorial Exploration, was introduced by Bean [6]. It is a domain independent automated enumeration framework. Its implementation is called `CombSpecSearcher` and has since been developed further. Given a programmatical interpretation of a finite structure it will attempt to find a structural description which can be translated to a set of equations whose solution is the structure's generating function.

The translation method described by Wood and Zeilberger [7] is an automated way of constructing bijections. In order to apply, it requires an algebraic proof so in a sense, it is not fully automated. A fully automated process of finding and constructing bijections does not exist as far as we know.

Our aim is to incorporate the other essential enumerative combinatorialist's tool, bijections, into Combinatorial Exploration. We implement a bijection based on the structural description output by the framework and furthermore, a search method for bijections in the expanded universe that Combinatorial Exploration uses in its search. We provide a domain independent fully automated method to find and construct bijections. This paper will describe the algorithms used (which have all been implemented in Python) and construct the theoretical foundation needed. As a proof of concept we connect numerous sets with bijections, mainly focusing on sets of permutations.

This thesis is structured as follows. Chapter 2 will cover prerequisites. New strategies for TileScope, a domain specific implementation of Combinatorial Exploration, are introduced in Chapter 3. In Chapter 4 we define criteria that must hold so a

bijection can be constructed and in Chapter 5 we define the resulting bijection. The search method for bijections is described in Chapter 6. We state notable successes in Chapter 7 and reflect on our work in Chapter 8.

Chapter 2

Background

2.1 Combinatorics

A *combinatorial class* is a set \mathcal{C} and a size function $\mathcal{C} \mapsto \mathbb{N} = \{0, 1, 2, \dots\}$ such that for all $n \in \mathbb{N}$, the set

$$\mathcal{C}_n = \{c \in \mathcal{C} \mid \text{size of } c \text{ is } n\}$$

is finite. We refer to an element of a combinatorial class, $c \in \mathcal{C}$, as a *combinatorial object* and its size is denoted by $|c|$. The set of words over a finite alphabet is an example of a combinatorial class where the number of characters in the word serves as a size function.

A *bijection* between sets A and B is an invertible function $f : A \mapsto B$ such that $a = f^{-1}(f(a))$ and $b = f(f^{-1}(b))$ for all $a \in A$ and $b \in B$. The existence of a bijection between sets A and B implies $|A| = |B|$. Suppose A is the set of binary strings of size n avoiding consecutive 1's and B the set of subsets of $\{1, 2, \dots, n\}$ containing no consecutive numbers. By mapping each binary string $b_1b_2 \dots b_n \in A$ to the set containing i if and only if $b_i = 1$, we have a bijection. This bijection can be seen in Figure 2.1 for $n = 3$.

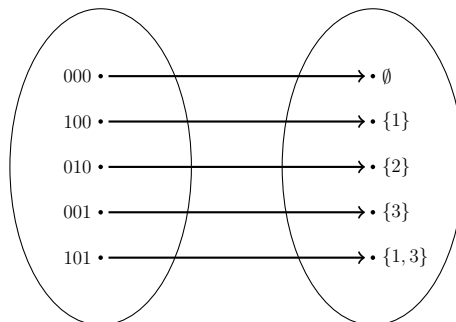


Figure 2.1: A bijection between binary strings of size 3 avoiding consecutive 1's and subsets of $\{1, 2, 3\}$ containing no consecutive numbers.

Given a bijection ϕ between two combinatorial classes \mathcal{C} and \mathcal{D} , we say that ϕ is *size preserving* if $|c| = |\phi(c)|$ for all $c \in \mathcal{C}$. Suppose \mathcal{C} is the set of binary strings that avoid consecutive 0's and \mathcal{D} the set of binary strings that avoid consecutive 1's. The bijection that flips the bits in a binary string is a size preserving bijection between \mathcal{C} and \mathcal{D} .

For a combinatorial class \mathcal{C} , the infinite sequence

$$|\mathcal{C}_0|, |\mathcal{C}_1|, |\mathcal{C}_2|, \dots, |\mathcal{C}_n|, \dots$$

where each term counts the number of elements of a fixed size is called the *counting sequence* of \mathcal{C} . It is the focal point of enumerative combinatorics. We might look for a formula for the terms, a recurrence relation, or an asymptotic equivalence. Another way to present the counting sequence is with a power series

$$\sum_{i=0}^{\infty} |\mathcal{C}_i| z^i$$

called a *generating function* and famously described as a “clothesline on which we hang up a sequence of numbers for display” in Wilf [8]. For example the sequence of Fibonacci numbers $(f_i)_{i=0}^{\infty}$ has the generating function

$$\sum_{i=0}^{\infty} f_i z^i = z + z^2 + 2z^3 + 3z^4 + 5z^5 + 8z^6 + \dots = \frac{z}{1 - z - z^2}.$$

If two combinatorial classes \mathcal{C} and \mathcal{D} share a counting sequence we say they are *isomorphic*, denoted $\mathcal{C} \cong \mathcal{D}$. This is equivalent to the existence of a size preserving bijection between \mathcal{C} and \mathcal{D} . The isomorphic relation is an equivalence relation on the set of all combinatorial classes.

The *symbolic method* in Flajolet and Sedgewick [9] describes how combinatorial classes can be constructed. To that end, they define *atoms*, *constructors* and *combinatorial specifications*. We will demonstrate these concepts, along with *combinatorial rules*, with an example.

Consider a $2 \times n$ with the latter number as its size, where each square can be raised or not, as shown in Figure 2.2. How many unique grids exist such that one can pass from left to right?

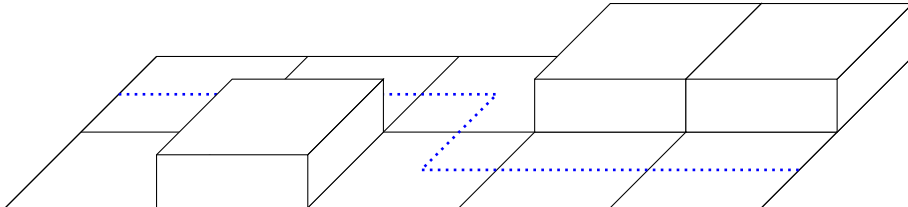


Figure 2.2: A grid of size 5 with three raised squares and a path from left to right.

Let G be the combinatorial class for passable grids. It is either the empty grid or contains at least one vertical slice. This is described by the admissible constructor disjoint union where a combinatorial class is split into disjoint sets. Now we have a combinatorial rule

$$G \cong \{\varepsilon\} \sqcup G_{\geq 1},$$

where ε is the empty grid and $G_{\geq 1}$ is the set of passable grid with a positive size. In terms of generating functions this corresponds to addition, that is

$$G(z) = 1 + G_{\geq 1}(z).$$

Any valid grid consists of a combination of the vertical slices $\begin{smallmatrix} \square \\ \square \end{smallmatrix}$, $\begin{smallmatrix} \square \\ \square \end{smallmatrix}$ and $\begin{smallmatrix} \square \\ \square \end{smallmatrix}$ as one with both raised would block the path. We refer to a combinatorial object of size 1 as an *atom*. The generating functions of the empty object and an atom are 1 and z

respectively. Let A , B and C be the classes of passable grids starting with $\begin{smallmatrix} \square & \square \\ \square & \square \end{smallmatrix}$, $\begin{smallmatrix} \square & \square \\ \square & \blacksquare \end{smallmatrix}$ and $\begin{smallmatrix} \square & \blacksquare \\ \square & \square \end{smallmatrix}$ respectively. Again we have a disjoint union

$$G_{\geq 1} \cong A \sqcup B \sqcup C.$$

We define \circ as the operator that prepends a vertical slice to all grids in a set. This is a Cartesian product, another admissible constructor denoted by \times , where

$$X \times Y = \{(x, y) \mid x \in X \text{ and } y \in Y\}$$

and in terms of generating functions corresponds to multiplication. Any passable grid can follow a vertical slice with neither square raised. Let D and E the classes that can follow $\begin{smallmatrix} \square & \square \\ \square & \blacksquare \end{smallmatrix}$ and $\begin{smallmatrix} \square & \blacksquare \\ \square & \square \end{smallmatrix}$ respectively. Now we have

$$A \cong \{\begin{smallmatrix} \square & \square \\ \square & \square \end{smallmatrix}\} \circ G, \quad B \cong \{\begin{smallmatrix} \square & \square \\ \square & \blacksquare \end{smallmatrix}\} \circ D \text{ and } C \cong \{\begin{smallmatrix} \square & \blacksquare \\ \square & \square \end{smallmatrix}\} \circ E.$$

Following $\begin{smallmatrix} \square & \square \\ \square & \blacksquare \end{smallmatrix}$ can be anything that does not start by blocking the lower square, that is

$$D \cong \{\varepsilon\} \sqcup B \sqcup A$$

and symmetrically we have

$$E \cong \{\varepsilon\} \sqcup C \sqcup A.$$

Now we have our *specification*, a system of rules with each class on the left at most once, defining an unambiguous formal grammar as shown with a tree representation in Figure 2.3.

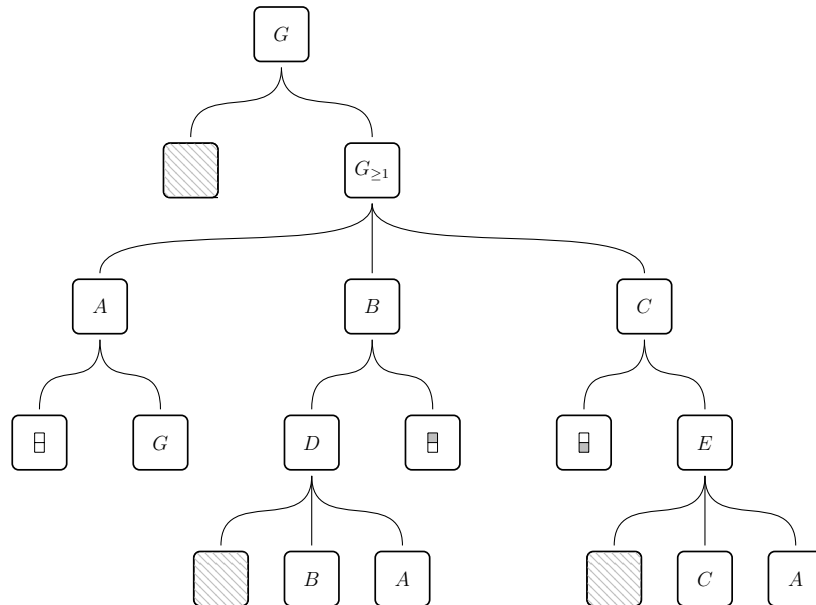


Figure 2.3: A tree representation of a formal grammar for passable $2 \times n$ grids.

This system of rules implies the set of equations on the generating functions. Solving

$$\begin{cases} G(z) = 1 + G_{\geq 1}(z) \\ G_{\geq 1}(z) = A(z) + B(z) + C(z) \\ A(z) = zG(z) \\ B(z) = zD(z) \\ C(z) = zE(z) \\ D(z) = 1 + B(z) + A(z) \\ E(z) = 1 + C(z) + A(z) \end{cases}$$

for $G(z)$ gives

$$G(z) = \frac{1+z}{1-2z-z^2}.$$

The Taylor series for this function, at $z = 0$, is

$$\sum_{i=0}^{\infty} \frac{G^{(i)}(0)}{i!} z^i = 1 + 3z + 7z^2 + 17z^3 + 41z^4 + 99z^5 + \dots$$

which corresponds to the number of grids for each size, e.g., there are 17 and 99 unique passable 2×3 and 2×5 grids respectively. We can even go further using calculus and find the exact formula,

$$|G_n| = \frac{(1 + \sqrt{2})^{n+1} + (1 - \sqrt{2})^{n+1}}{2}.$$

This is not always the case and depends on the type of generating function we have.

In the context of this thesis, we will refer to combinatorial classes as classes and the same goes for combinatorial specifications and rules. We also assume all constructors to be admissible and all rules to entail isomorphism, e.g., there exists a bijection from the left hand side of any rule to the right hand side.

2.2 Permutations

A *permutation* π is a bijection between a set and itself. In our context, the set in question is $[n] = \{1, 2, \dots, n\}$. We adopt a one line notation for permutations, $\pi = \pi_1 \pi_2 \dots \pi_n$ where $\pi_i = j$ if i maps to j . This correspondence for the permutation 1423 can be seen in Figure 2.4 as well as its geometric representation which displays the points $\{(i, \pi(i)) \mid i \in [n]\}$ in the Cartesian plane.

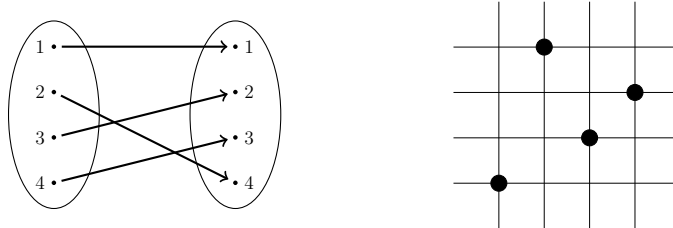


Figure 2.4: The mapping and graphical representation of the permutation 1423.

A permutation on $[n]$ is said to be of size n and \mathcal{S}_n is the set of all permutations of size n . The set of permutations of size 3 is

$$\mathcal{S}_3 = \{123, 132, 213, 231, 312, 321\}.$$

There is a permutation of size 0, namely the empty permutation, denoted ε . This is the unique map from \emptyset to \emptyset . The set of all permutations is denoted by \mathcal{S} . It is a combinatorial class since \mathcal{S}_n is finite for all $n \in \mathbb{N}$, containing $n!$ elements.

Given a permutation $\pi = \pi_1\pi_2 \cdots \pi_n$, its *reverse* is

$$r(\pi) = \pi^{\text{rev}} = \pi_n\pi_{n-1} \cdots \pi_1$$

and its *inverse* is

$$i(\pi) = \pi^{-1} = \pi_{i_1}\pi_{i_2} \cdots \pi_{i_n}$$

where $\pi_{i_j} = k$ if $\pi_k = j$. The reverse of 132 is 231 and the inverse of 51324 is 24351. The reverse and inverse functions generate the dihedral group D_4 , that is the symmetries of a square using the graphical representation of a permutation. We define these eight symmetry maps as $\mathbf{sym}_1, \mathbf{sym}_2, \dots, \mathbf{sym}_8$ as is shown in Table 2.1 which includes an example for the permutation 1423 and its symmetries.









\mathbf{sym}_1	π	Identity	$\begin{array}{ c c } \hline A & B \\ \hline D & C \\ \hline \end{array}$	1423	
\mathbf{sym}_2	$r(i(\pi))$	Rotate 90°	$\begin{array}{ c c } \hline D & A \\ \hline C & B \\ \hline \end{array}$	4213	
\mathbf{sym}_3	$i(r(i(r(\pi))))$	Rotate 180°	$\begin{array}{ c c } \hline C & D \\ \hline B & A \\ \hline \end{array}$	2314	
\mathbf{sym}_4	$i(r(\pi))$	Rotate 270°	$\begin{array}{ c c } \hline B & C \\ \hline A & D \\ \hline \end{array}$	2431	
\mathbf{sym}_5	$i(r(i(\pi)))$	Horizontal flip	$\begin{array}{ c c } \hline D & C \\ \hline A & B \\ \hline \end{array}$	4132	
\mathbf{sym}_6	$r(\pi)$	Vertical flip	$\begin{array}{ c c } \hline B & A \\ \hline C & D \\ \hline \end{array}$	3241	
\mathbf{sym}_7	$i(\pi)$	Diagonal flip	$\begin{array}{ c c } \hline C & B \\ \hline D & A \\ \hline \end{array}$	1342	
\mathbf{sym}_8	$r(i(r(\pi)))$	Antidiagonal flip	$\begin{array}{ c c } \hline A & D \\ \hline B & C \\ \hline \end{array}$	3124	

Table 2.1: The eight symmetry maps for permutations interpreted with reverse and inverse, dihedral group D_4 and an example for $\pi = 1423$.

We define $\mathbf{sym}(\pi) = \{\mathbf{sym}_i(\pi) \mid i \in [8]\}$ as the *symmetries of* π . Note that two different symmetry functions can map to the same permutation, e.g., $\mathbf{sym}(1) = \{1\}$. Every permutation has between one and eight symmetries.

Given a finite strictly totally ordered set $(M, <)$ where $M = \{x_1, x_2, \dots, x_n\}$, we define its *sorting* as

$$\mathbf{sort}(M) = (x_{i_1}, x_{i_2}, \dots, x_{i_n})$$

if $x_{i_1} < x_{i_2} < \cdots < x_{i_n}$. Suppose $M = \{1, 5, 2\}$, then $\mathbf{sort}(M) = (1, 2, 5)$. The *indexed subsequence* of a permutation $\pi \in \mathcal{S}_n$ for indices $S \subseteq [n]$ is the sequence

$$\Xi(S, \pi) = \pi_{i_1}\pi_{i_2} \cdots \pi_{i_{|S|}}$$

if $\text{sort}(S) = (i_1, i_2, \dots, i_{|S|})$. Given an indexed subsequence γ of a permutation $\pi \in \mathcal{S}_n$, its *standardization*, $\text{st}(\gamma)$, is the sequence where the i^{th} largest entry has been replaced by i . Let $\pi = 41253 \in \mathcal{S}_5$ and $S = \{1, 3, 4\}$, then $\text{st}(\Xi(S, \pi)) = \text{st}(425) = 213$.

Given two permutations $\pi \in \mathcal{S}_n$ and $\sigma \in \mathcal{S}_k$ we say that π *contains* σ , denoted $\sigma \preceq \pi$, if there exists a subset $S \subseteq [n]$ such that $\text{st}(\Xi(S, \pi)) = \sigma$. If π does not contain σ , it *avoids* it. In the context of containment, we refer to the contained permutation as a (*classical*¹) *pattern* and that permutations contain or avoid a pattern. The permutation 356214 contains the pattern 4213 since

$$\text{st}(\Xi(\{2, 4, 5, 6\}, 356214)) = \text{st}(5214) = 4213.$$

It, however, avoids the pattern 1432 since

$$1432 \notin \{\text{st}(\Xi(S, 356214)) \mid S \subseteq [6]\}.$$

An occurrence of 4213 in 356214 can be seen in Figure 2.5.

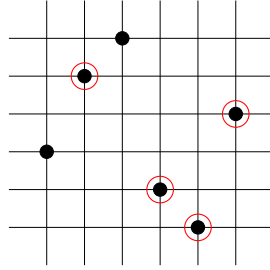


Figure 2.5: The permutation 356214 with an occurrence of 4213 circled.

A set of permutations Π is *closed downwards* with respect to containment if

$$\bigcup_{\pi \in \Pi} \{\sigma \mid \sigma \preceq \pi\} \subseteq \Pi.$$

A set of permutations that is closed downwards is called a *permutation class*. The set $\{\varepsilon, 1, 12, 123\}$ is closed downwards since all patterns contained in any of its elements belong to the set. We can extend this set to include all increasing permutations, $\{\varepsilon, 1, 12, 123, 1234, \dots\}$ and it is still closed downwards. Both sets are therefore permutation classes. Isomorphic permutation classes are said to be *Wilf equivalent* and their equivalence classes are called *Wilf classes*.

A permutation π *avoids* a set of permutations Π if it avoids every $\sigma \in \Pi$. It *contains* Π if it does not avoid it, that is, it contains at least one permutation of Π . Let

$$\text{Av}(\Pi) = \{\pi \in \mathcal{S} \mid \pi \text{ avoids } \Pi\}.$$

We simplify this notation by omitting set brackets, writing $\text{Av}(\sigma_1, \sigma_2)$ instead of $\text{Av}(\{\sigma_1, \sigma_2\})$. Define

$$\text{Av}_n(\Pi) = \{\pi \in \text{Av}(\Pi) \mid |\pi| = n\}$$

for $n \in \mathbb{N}$.

A set of permutations \mathcal{B} is called a *basis* if for all $\pi \in \mathcal{B}$ there does not exist a $\sigma \in \mathcal{B} \setminus \{\pi\}$ such that $\sigma \preceq \pi$. The set $\{12, 21\}$ is a basis while $\{12, 231\}$ is not since $12 \preceq 231$. Permutation classes can be described in terms of a basis avoided, e.g.,

$$\{\varepsilon, 1, 12, 123\} = \text{Av}(21, 1234)$$

¹There are other types of patterns that we do not concern ourselves with in this thesis.

and

$$\{\varepsilon, 1, 12, 123, 1234, \dots\} = \text{Av}(21).$$

Let \mathcal{B} be a basis. We extend the definition of symmetry maps to bases such that

$$\text{sym}_i(\mathcal{B}) = \{\text{sym}_i(\sigma) \mid \sigma \in \mathcal{B}\}$$

and

$$\text{sym}(\mathcal{B}) = \{\text{sym}_i(\mathcal{B}) \mid i \in [8]\}.$$

For any basis \mathcal{B} , the *symmetry class* of $\text{Av}(\mathcal{B})$ is $\{\text{Av}(B) \mid B \in \text{sym}(\mathcal{B})\}$. The symmetry class of any permutation class is a subset of its Wilf class. As symmetries are considered trivial bijections we only concern ourselves with the lexicographically minimal representation of each symmetry class. For example, there are seven symmetry classes and three Wilf classes for singleton bases in \mathcal{S}_4 . They are shown in Table 2.2.

Wilf class	Counting sequence
$\text{Av}(1234)$	1, 2, 6, 23, 103, 513, 2761, 15767, ...
$\text{Av}(1243)$	
$\text{Av}(1432)$	
$\text{Av}(2143)$	
$\text{Av}(1324)$	1, 2, 6, 23, 103, 513, 2762, 15793, ...
$\text{Av}(1342)$	1, 2, 6, 23, 103, 512, 2740, 15485, ...
$\text{Av}(2413)$	

Table 2.2: The seven lexicographically minimal representations of symmetry classes of permutation classes avoiding a single classical pattern of size 4. They are grouped into three Wilf classes.

2.3 Gridded permutations

A *cell* $(a, b) \in \mathbb{N}^2$ is the region within $[0, \infty)^2$ defined by $[a, a + 1) \times [b, b + 1)$. A pair

$$(\pi, P) = (\pi_1 \pi_2 \cdots \pi_n, ((a_1, b_1), (a_2, b_2), \dots, (a_n, b_n)))$$

where P is an n -tuple of cells and π is a permutation is called a *gridded permutation* of size n if for all $i, j \in [n]$, $i < j$ implies $a_i \leq a_j$ and $\pi_i < \pi_j$ implies $b_i \leq b_j$. A more detailed definition is given by Albert, Atkinson, Bouvel, *et al.* [10]. We refer to P as the *positions* of the gridded permutation and π as the *underlying permutation*. We use a one line notation for gridded permutations

$$\pi = \pi_1^{(a_1, b_1)} \pi_2^{(a_2, b_2)} \dots \pi_n^{(a_n, b_n)}$$

where we can exclude all but the last position of consecutive occurrences of the same position. The gridded permutation $87^{(0,2)}1^{(1,0)}6^{(1,2)}2^{(1,0)}4^{(3,1)}3^{(3,0)}5^{(4,2)}$ can be seen in Figure 2.6.

We extend the definition of indexed subsequences to gridded permutations so the sequence produced, $\Xi(S, \pi)$, contains elements of the underlying permutation and positions. Given this extension, we can extend the standardization of an indexed subsequence of a gridded permutation to be applied only to the underlying permutation.

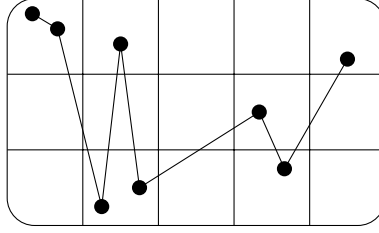


Figure 2.6: The gridded permutation $87^{(0,2)}1^{(1,0)}6^{(1,2)}2^{(1,0)}4^{(3,1)}3^{(3,0)}5^{(4,2)}$.

For example, given a gridded permutation $\pi = 1^{(0,0)}5^{(1,2)}2^{(2,0)}43^{(2,1)}$ and $S = \{1, 2, 5\}$ we have

$$\text{st}(\Xi(S, \pi)) = \text{st}(1^{(0,0)}5^{(1,2)}3^{(2,1)}) = 1^{(0,0)}3^{(1,2)}2^{(2,1)}.$$

Using the extended definition of standardization we extend containment and avoidance to gridded permutations, for both individual and sets of gridded permutations. For example, the gridded permutation $1^{(0,0)}5^{(1,2)}2^{(2,0)}43^{(2,1)}$ contains $1^{(0,0)}2^{(2,0)}3^{(2,1)}$ but avoids $1^{(0,0)}23^{(2,1)}$.

Let \mathcal{G} be the set of all gridded permutations and \mathcal{G}_n be the set of gridded permutations of size n . The set \mathcal{G} is not a combinatorial class since \mathcal{G}_n is infinite for all $n \in \mathbb{Z}^+$, as there are infinitely many cells to choose as positions. If we restrict the choice of cells to be finite, then so are the gridded permutations we can form. Let $\mathcal{G}^{(c,r)}$ be the set of gridded permutation with positions in

$$\{0, 1, \dots, c-1\} \times \{0, 1, \dots, r-1\}$$

and

$$\mathcal{G}_n^{(c,r)} = \{\pi \in \mathcal{G}^{(c,r)} \mid |\pi| = n\}.$$

The set $\mathcal{G}^{(c,r)}$ is a combinatorial class. Define

$$\text{Av}^{(c,r)}(\Pi) = \{\pi \in \mathcal{G}^{(c,r)} \mid \pi \text{ avoids } \Pi\}$$

and

$$\text{Av}_n^{(c,r)}(\Pi) = \{\pi \in \mathcal{G}_n^{(c,r)} \mid \pi \text{ avoids } \Pi\}.$$

2.4 Tilings

A *tiling* is a triple

$$\mathcal{T} = ((c, r), \mathcal{O}, \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_k\})$$

where

$$(c, r) \in \mathbb{Z}^+ \times \mathbb{Z}^+, \mathcal{O} \subseteq \mathcal{G}^{(c,r)} \text{ and } \mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_k\} \subseteq (\mathcal{G}^{(c,r)})^k$$

are called *dimension*, *obstructions* and *requirements* respectively. The gridded permutations in $\text{Av}^{(c,r)}(\mathcal{O})$ that contain $\mathcal{R}_1, \dots, \mathcal{R}_k$ make up the combinatorial class $\text{Grid}(\mathcal{T})$. Define $\text{Grid}_n(\mathcal{T}) = \{\pi \in \text{Grid}(\mathcal{T}) \mid |\pi| = n\}$. A more detailed definition of tilings is given by Bean [6]. An example of a tiling can be seen in Figure 2.7 where obstructions are red and requirements are blue. Cells with 12 and 21 obstruction and a 1 requirement are represented with a larger black point.

Typically, when we want to enumerate $\text{Av}(132)$ we start with a disjoint union constructor since a permutation in $\text{Av}(132)$ is either empty or contains a topmost point.

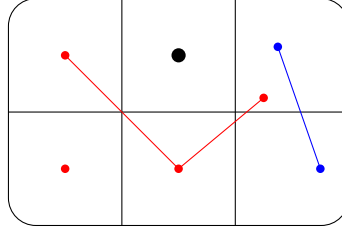


Figure 2.7: A 3×2 tiling with $\mathcal{O} = \{1^{(0,0)}, 12^{(1,1)}, 21^{(1,1)}, 3^{(0,1)}1^{(1,0)}2^{(2,1)}\}$ and $\mathcal{R} = \{\{1^{(1,1)}\}, \{2^{(1,1)}1^{(2,0)}\}\}$.

If it contains a topmost point, it must still avoid 132 on either side. Furthermore, the elements to the left of the topmost point must be greater than those to its right or you have an occurrence of 132 with the topmost point. This is important since it allows us to uniquely map the (standardized) elements to the right of the topmost point from the rule. Now we have a specification

$$\begin{aligned} \text{Av}(132) &\cong \{\varepsilon\} \sqcup \text{Av}_{\geq 1}(132) \\ \text{Av}_{\geq 1}(132) &\cong \text{Av}(132) \times \{\bullet\} \times \text{Av}(132) \end{aligned}$$

and a generating function, obtained by solving the system of equations derived from the specification,

$$\frac{1 - \sqrt{1 - 4z}}{2z}.$$

This is a simple example but given different classes and constructors, we can end up with classes that are difficult to describe rigorously. The purpose of tilings is to provide a sort of language to describe classes with more complicated permutation constraints or limited parts of permutations, e.g., we can describe $\text{Av}_{\geq 1}(132)$ with the tiling shown in Figure 2.8.

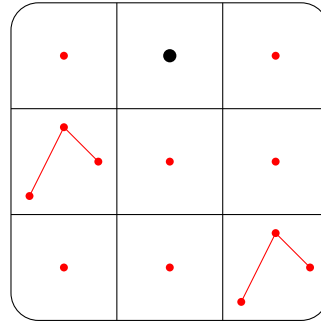


Figure 2.8: A tiling \mathcal{T} with $\text{Grid}(\mathcal{T})$ isomorphic to $\text{Av}_{\geq 1}(132)$.

2.5 Combinatorial Exploration

Combinatorial Exploration [11] is a domain agnostic specification searcher. It uses strategies to generate rules for classes. Strategies are independent of classes and describe an action that may result in rules given an applicable class. The search method is done in a breadth-first manner from the initial class. When run, it uses a predefined set of strategies and expands a universe in hope of finding a specification within it.

To avoid tautologies, the searcher gives classes equivalence labels and tracks rules in terms of parent and children pairs as equivalence labels. The classes that share an equivalence label may or may not be isomorphic. Any rule with a single class on its right hand side that can result in a tautology may have both sides with the same equivalence label. When a specification is found and extracted from the universe, any sequence of equivalence rules is merged into a single rule.

2.6 TileScope

TileScope [12] is an implementation of Combinatorial Exploration for tilings. A root class with a single cell and no requirements is in direct correspondence to a permutation class. The library utilizes Permuta [13], a general permutation library. Most of the strategies TileScope uses can be found in Bean [6]. We will give a brief explanation of those that are not as well as introduce new strategies in Chapter 3. The strategies introduced in this thesis enabled TileScope to discover a specification for $\text{Av}(1432)$.

2.6.1 Assumptions

There are two strategies involving assumptions. One adds assumptions and the other rearranges them. Assumptions represent catalytic variables and are used to track the number of points in cells.

Let $\mathcal{T}_1 = ((2, 1), \{12^{(0,0)}, 1^{(0,0)}23^{(1,0)}, 123^{(1,0)}\}, \emptyset)$ and suppose \mathcal{T}_2 is \mathcal{T}_1 with the assumption that we can count the number of points in $(0, 0)$ as shown in Figure 2.9. Let $T_1(x)$ and $T_2(x, y)$ be the generating function for \mathcal{T}_1 and \mathcal{T}_2 respectively, then $T_1(x) = x^0 + 2x^1 + 5x^2 + 14x^3 + 42x^4 + \dots$ while $T_2(x, y)$ is

$$\begin{aligned} & x^0 y^0 \\ & + x^1 y^0 + x^1 y^1 \\ & + 2x^2 y^0 + 2x^2 y^1 + x^2 y^2 \\ & + 5x^3 y^0 + 5x^3 y^1 + 3x^3 y^2 + x^3 y^3 \\ & + 14x^4 y^0 + 14x^4 y^1 + 9x^4 y^2 + 4x^4 y^3 + x^4 y^4 \\ & + \dots \end{aligned}$$

and $T_1(x) = T_2(x, 1)$.

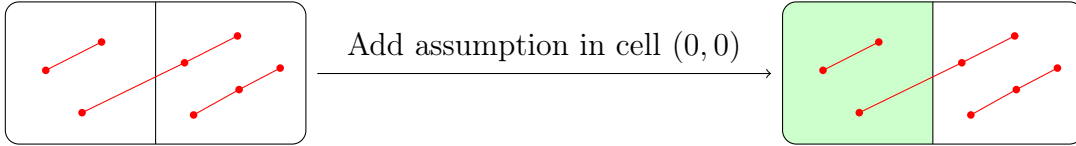


Figure 2.9: The tiling $((2, 1), \{12^{(0,0)}, 1^{(0,0)}23^{(1,0)}, 123^{(1,0)}\}, \emptyset)$ with and without an assumption.

When the set of cells of one assumption is a subset of the set of cells of another we can rearrange them. Figure 2.10 demonstrates assumption rearrangement for the tiling $((3, 1), \{12^{(0,0)}, 1^{(0,0)}23^{(1,0)}, 12^{(1,0)}3^{(2,0)}\}, \emptyset)$ where the left one has y as a catalytic variable for points in $(0, 0)$ and z for points in either $(0, 0)$ or $(1, 0)$ while in the right one z only tracks those in $(1, 0)$ after the assumptions have been rearranged. If $T_1(x, y, z)$ is the generating function before the rearrangement and $T_2(x, y, z)$ after then $T_1(x, y, z) = T_2(x, yz, z)$.

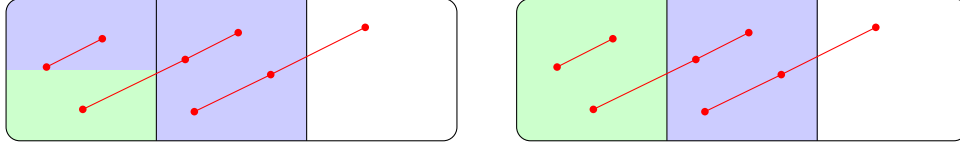


Figure 2.10: By rearranging the assumptions on the left we get the assumptions on the right.

2.6.2 Fusion

The fusion strategy tries to merge adjacent columns or rows in a tiling. To simplify our explanation we will look at how it works for a single row (or a single column) tiling as they are the most common scenario where fusion is applied. It is possible to fuse columns c and $c + 1$ in a tiling \mathcal{T} if for every gridded permutation

$$\alpha \alpha_1^{(c,0)} \alpha_2^{(c,0)} \dots \alpha_i^{(c,0)} \beta_1^{(c+1,0)} \beta_2^{(c+1,0)} \dots \beta_j^{(c+1,0)} \beta$$

in $\text{Grid}(\mathcal{T})$, where the subsequences α and β are outside of said columns, the class $\text{Grid}(\mathcal{T})$ also contains

$$\begin{aligned} & \alpha \alpha_1^{(c,0)} \alpha_2^{(c,0)} \dots \alpha_i^{(c,0)} \beta_1^{(c,0)} \beta_2^{(c,0)} \dots \beta_j^{(c,0)} \beta, \\ & \alpha \alpha_1^{(c,0)} \alpha_2^{(c,0)} \dots \alpha_i^{(c,0)} \beta_1^{(c,0)} \beta_2^{(c,0)} \dots \beta_j^{(c+1,0)} \beta, \\ & \vdots \\ & \alpha \alpha_1^{(c+1,0)} \alpha_2^{(c+1,0)} \dots \alpha_i^{(c+1,0)} \beta_1^{(c+1,0)} \beta_2^{(c+1,0)} \dots \beta_j^{(c+1,0)} \beta. \end{aligned}$$

An example of a fusable tiling is shown in Figure 2.11. The rules, in terms of generating functions, depend on the tiling's assumptions. Some common scenarios are shown in Table 2.3

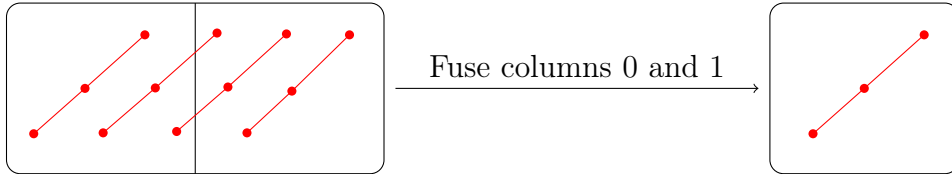


Figure 2.11: A tiling where the fusion strategy applies.

TileScope requires the fusion strategy to find specifications for multiple classes, e.g., $\text{Av}(123)$, $\text{Av}(1234)$, $\text{Av}(1243)$ and $\text{Av}(1432)$.

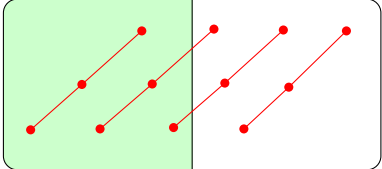
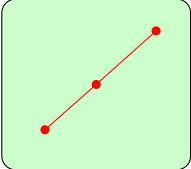
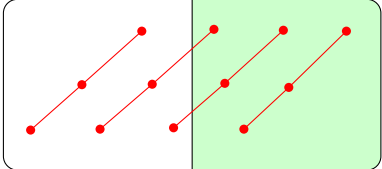
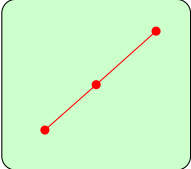
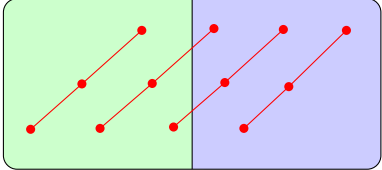
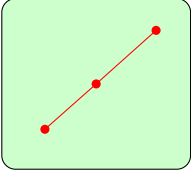
\mathcal{T}_1	\mathcal{T}_2	Rule
		$T_1(x, y) = \frac{yT_2(x, y) - T_2(x, 1)}{y - 1}$
		$T_1(x, y) = \frac{yT_2(x, y) - T_2(x, 1)}{y - 1}$
		$T_1(x, y, z) = \frac{yT_2(x, y) - zT_2(y, z)}{y - z}$

Table 2.3: The fusion rules for some common assumptions in terms of generating functions.

Chapter 3

Strategies for TileScope

This chapter introduces three new strategies for TileScope, column reverse, column permutation and sliding. The first two, although perfectly valid strategies on their own, are used to extend sliding to any columns. The sliding strategy, as previously mentioned, allowed for the first discovery of a specification for $\text{Av}(1432)$ by TileScope.

Using a specification for $\text{Av}(1234)$ discovered by TileScope and applying the same strategies at every step starting with $\text{Av}(1432)$, we had a matching tree except for one tiling that we failed to expand in the same manner as the one for $\text{Av}(1234)$. By looking at the tree we had created from the $\text{Av}(1432)$ root, we could see what tiling was needed instead of the one we failed to expand in order to complete the specification in the same way. This pair of tilings, shown in Figure 3.1, whose inconspicuous equivalence we failed to disprove, is the motivation for the sliding strategy. The process of applying the same strategies to another root further motivated the automated bijection search in Chapters 4, 5 and 6.

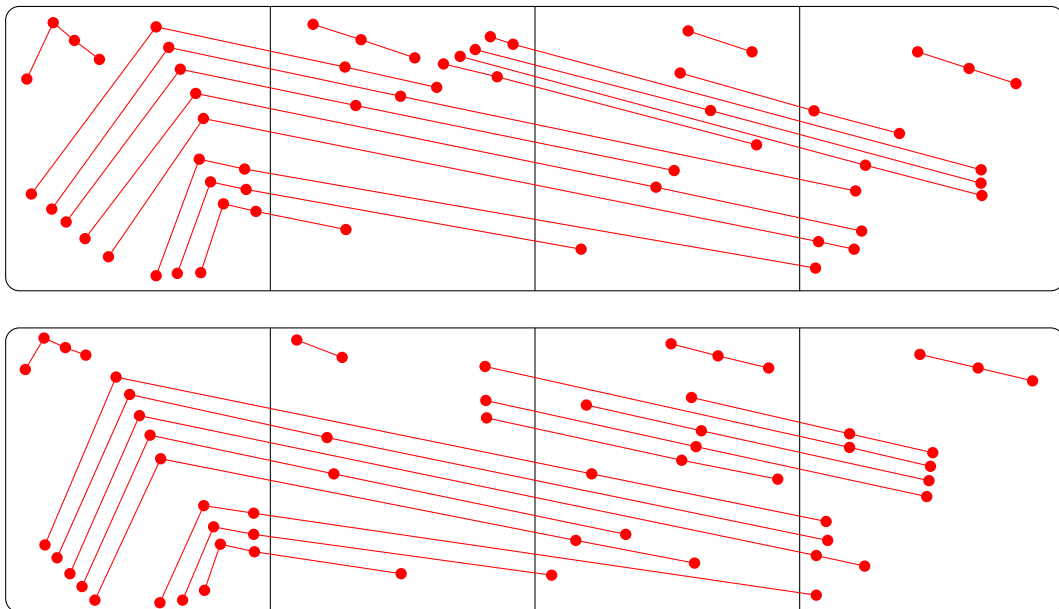


Figure 3.1: The pair of tilings that motivated the sliding strategy. The one above is the one encountered when mimicking the strategies of a specification for $\text{Av}(1234)$ to a $\text{Av}(1432)$ root while the one below is the one we needed to have identical specification in terms of strategies and structure.

3.1 Column reverse

Definition 3.1. For a gridded permutation $\pi = \pi_1^{(x_1, y_1)} \pi_2^{(x_2, y_2)} \dots \pi_n^{(x_n, y_n)} \in \mathcal{G}_n$ define its *column reverse* for $[a, b]$, where $a, b \in \mathbb{N}$ and $a \leq b$, as the mapping $\text{rev}_{[a, b]}$ where $\text{rev}_{[a, b]}(\pi)$ is π where its longest subsequence whose columns all belong to interval

$$\pi_i^{(x_i, y_i)} \pi_{i+1}^{(x_{i+1}, y_{i+1})} \dots \pi_{i+k}^{(x_{i+k}, y_{i+k})}$$

has been replaced by

$$\pi_{i+k}^{(a+b-x_{i+k}, y_{i+k})} \dots \pi_{i+1}^{(a+b-x_{i+1}, y_{i+1})} \pi_i^{(a+b-x_i, y_i)}.$$

The column reverse can be extended to subsequences of gridded permutations and tilings such that it is applied to all of its obstructions and requirements. Suppose $\pi = 5^{(0,1)} 2^{(0,0)} 4^{(1,1)} 1^{(1,0)} 6^{(2,2)} 3^{(3,1)}$, then

$$\text{rev}_{[1,2]}(\pi) = 5^{(0,1)} 2^{(0,0)} 6^{(1,2)} 1^{(2,0)} 4^{(2,1)} 3^{(3,1)},$$

as shown in Figure 3.2. An example for tilings can be seen in Figure 3.3.

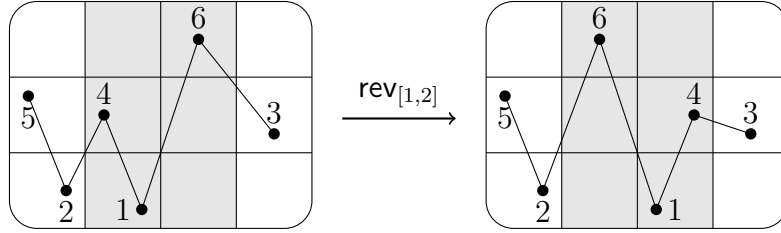


Figure 3.2: The column reverse of a gridded permutation.

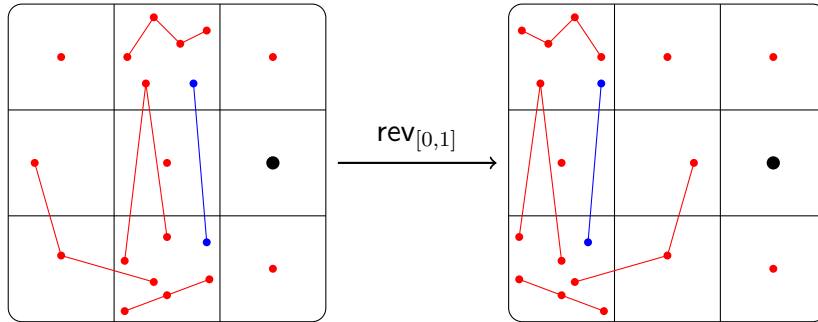


Figure 3.3: The column reverse of a tiling.

Lemma 3.2. Let $\pi, \sigma \in \mathcal{G}$ such that $\sigma \preceq \pi$, then $\text{rev}_{[a, b]}(\sigma) \preceq \text{rev}_{[a, b]}(\pi)$.

Proof. If π has no elements within the columns then neither does σ and both are mapped to themselves. If σ has no elements within the columns then the mapping has no effect on the pattern which remains in $\text{rev}_{[a, b]}(\pi)$.

Let $\{i, i+1, \dots, i+k\}$ be the indices of elements of π with columns in $[a, b]$. Let $\sigma = \text{st}(\Xi(A_1 \cup L \cup A_2, \pi))$ where $L \subseteq \{i, i+1, \dots, i+k\}$ and A_1 and A_2 are the

indices in π on either side of the columns corresponding to the occurrence of σ , then

$$\begin{aligned}
\text{rev}_{[a,b]}(\sigma) &= \text{rev}_{[a,b]}(\text{st}(\Xi(A_1 \cup L \cup A_2, \pi))) \\
&= \text{st}(\text{rev}_{[a,b]}(\Xi(A_1 \cup L \cup A_2, \pi))) \\
&= \text{st}(\text{rev}_{[a,b]}(\Xi(A_1, \pi)\Xi(L, \pi)\Xi(A_2, \pi))) \\
&= \text{st}(\Xi(A_1, \pi)\text{rev}_{[a,b]}(\Xi(L, \pi))\Xi(A_2, \pi)) \\
&= \text{st}(\Xi(A_1, \text{rev}_{[a,b]}(\pi))\text{rev}_{[a,b]}(\Xi(L, \pi))\Xi(A_2, \text{rev}_{[a,b]}(\pi))) \\
&= \text{st}(\Xi(A_1, \text{rev}_{[a,b]}(\pi))\Xi(\{2i + k - \ell \mid \ell \in L\}, \text{rev}_{[a,b]}(\pi))\Xi(A_2, \text{rev}_{[a,b]}(\pi))) \\
&= \text{st}(\Xi(A_1 \cup \{2i + k - \ell \mid \ell \in L\} \cup A_2, \text{rev}_{[a,b]}(\pi)))
\end{aligned}$$

which shows that $\text{rev}_{[a,b]}(\sigma)$ is the standardization of a subsequence of $\text{rev}_{[a,b]}(\pi)$. \square

Proposition 3.3. Let \mathcal{T} be a tiling, then $|\text{Grid}_n(\mathcal{T})| = |\text{Grid}_n(\text{rev}_{[a,b]}(\mathcal{T}))|$ for all $n \in \mathbb{N}$.

Proof. By Lemma 3.2 all occurrences of requirements are preserved so the only way that $\text{rev}_{[a,b]}(\pi)$ is not in $\text{Grid}(\text{rev}_{[a,b]}(\mathcal{T}))$ is if there exists an obstruction o that is avoided by π while $\text{rev}_{[a,b]}(\pi)$ contains $\text{rev}_{[a,b]}(o)$. Suppose that there is such an obstruction, then $\text{rev}_{[a,b]}(\text{rev}_{[a,b]}(\pi)) = \pi$ must contain $\text{rev}_{[a,b]}(\text{rev}_{[a,b]}(o)) = o$ which contradicts π avoiding o . Therefore $\pi \in \text{Grid}(\mathcal{T})$ implies $\text{rev}_{[a,b]}(\pi) \in \text{Grid}(\text{rev}_{[a,b]}(\mathcal{T}))$. With $\text{rev}_{[a,b]}$ being its own inverse it is a bijection between

$$\text{Grid}_n(\mathcal{T}) \text{ and } \{\text{rev}_{[a,b]}(\pi) \mid \pi \in \text{Grid}_n(\mathcal{T})\} \subseteq \text{Grid}_n(\text{rev}_{[a,b]}(\mathcal{T}))$$

and thus $|\text{Grid}_n(\mathcal{T})| \leq |\text{Grid}_n(\text{rev}_{[a,b]}(\mathcal{T}))|$. Analogously, in the other direction, we have $|\text{Grid}_n(\mathcal{T})| \geq |\text{Grid}_n(\text{rev}_{[a,b]}(\mathcal{T}))|$. \square

3.2 Column permutation

Let $\pi = \pi_1^{(x_1, y_1)} \pi_2^{(x_2, y_2)} \dots \pi_n^{(x_n, y_n)} \in \mathcal{G}_n$ and define $\chi_x(\pi) = \pi_1^{(x, y_1)} \pi_2^{(x, y_2)} \dots \pi_n^{(x, y_n)}$ for $x \in \mathbb{N}$, e.g.,

$$\chi_1(1^{(0,0)}3^{(2,3)}2^{(5,2)}) = 1^{(1,0)}3^{(1,3)}2^{(1,2)}.$$

We extend this map to subsequences of gridded permutations.

Definition 3.4. Let $\pi = \pi_1^{(x_1, y_1)} \pi_2^{(x_2, y_2)} \dots \pi_n^{(x_n, y_n)} \in \mathcal{G}_n^{(c, r)}$ and $\sigma \in \mathcal{S}_k$ for $k \geq c$. The *column permutation* σ of π is

$$C_\sigma(\pi) = \chi_0(\Xi(A_{\sigma_1}, \pi))\chi_1(\Xi(A_{\sigma_2}, \pi)) \dots \chi_{k-1}(\Xi(A_{\sigma_k}, \pi))$$

where $A_i = \{j \in [n] \mid x_j = i - 1\}$ for $i \in [k]$ are the sets of indices in column i .

Note that A_1, A_2, \dots, A_k can include empty sets and σ can be larger than the number of columns the gridded permutation spans, but not vice versa. We extend this definition to tilings such that it is applied to all of its obstructions and requirements. Let $\sigma = 4312$ and

$$\pi = 6^{(0,1)}2^{(0,0)}5^{(1,1)}3^{(1,0)}8^{(1,2)}4^{(2,1)}7^{(3,2)}1^{(3,0)},$$

then $A_1 = \{1, 2\}$, $A_2 = \{3, 4, 5\}$, $A_3 = \{6\}$, $A_4 = \{7, 8\}$ and we have

$$\begin{aligned}
C_\sigma(\pi) &= \chi_0(\Xi(A_{\sigma_1}, \pi)) \chi_1(\Xi(A_{\sigma_2}, \pi)) \chi_2(\Xi(A_{\sigma_3}, \pi)) \chi_3(\Xi(A_{\sigma_4}, \pi)) \\
&= \chi_0(\Xi(A_4, \pi)) \chi_1(\Xi(A_3, \pi)) \chi_2(\Xi(A_1, \pi)) \chi_3(\Xi(A_2, \pi)) \\
&= \chi_0(\Xi(\{7, 8\}, \pi)) \chi_1(\Xi(\{6\}, \pi)) \chi_2(\Xi(\{1, 2\}, \pi)) \chi_3(\Xi(\{3, 4, 5\}, \pi)) \\
&= \chi_0(7^{(3,2)} 1^{(3,0)}) \chi_1(4^{(2,1)}) \chi_2(6^{(0,1)} 2^{(0,0)}) \chi_3(5^{(1,1)} 3^{(1,0)} 8^{(1,2)}) \\
&= 7^{(0,2)} 1^{(0,0)} 4^{(1,1)} 6^{(2,1)} 2^{(2,0)} 5^{(3,1)} 3^{(3,0)} 8^{(3,2)}
\end{aligned}$$

as is shown in Figure 3.4. An example for tilings can be seen in Figure 3.5.

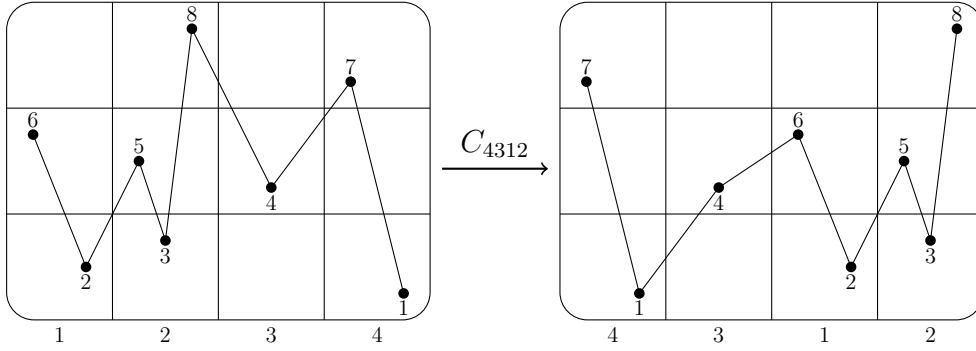


Figure 3.4: The column permutation of a gridded permutation.

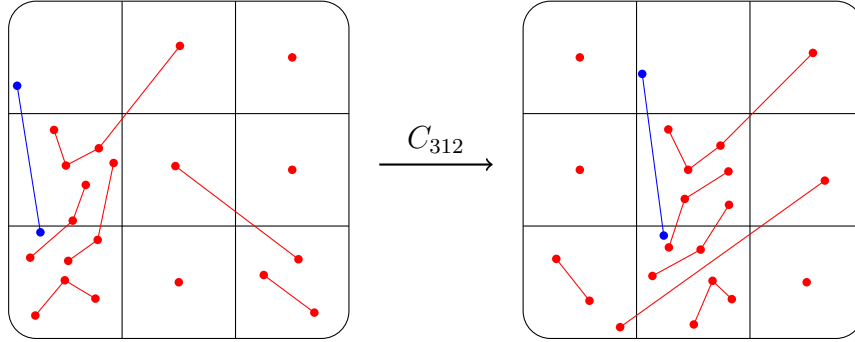


Figure 3.5: The column permutation of a tiling.

Let $\pi = \pi_1 \pi_2 \cdots \pi_n \in \mathcal{S}_n$ and $A = (i_1, i_2, \dots, i_k)$ a finite sequence of size k containing elements from $[n - 1]$ and define the *adjacent swap* of π (relative to A) as

$$\text{AdjSwap}_A(\pi) = \begin{cases} \pi_1 \pi_2 \cdots \pi_{i_1-1} \pi_{i_1+1} \pi_{i_1} \pi_{i_1+2} \pi_{i_1+3} \cdots \pi_n & \text{if } k = 1, \\ \text{AdjSwap}_{(i_2, i_3, \dots, i_k)}(\text{AdjSwap}_{(i_1)}(\pi)) & \text{otherwise.} \end{cases}$$

Let $\pi = 1423$, then $\text{AdjSwap}_{(1,2)}(\pi) = \text{AdjSwap}_{(2)}(4123) = 4213$.

Lemma 3.5. Let $\pi = 12 \cdots n \in \text{Av}_n(21)$ then

$$\{\text{AdjSwap}_A(\pi) \mid A \text{ is a finite sequence of elements from } [n - 1]\} = \mathcal{S}_n$$

It is well known that permutation groups can be generated by adjacent swaps, see e.g., [14].

Proposition 3.6. Let $\mathcal{T} = ((c, r), \mathcal{O}, \{\mathcal{R}_1, \dots, \mathcal{R}_k\})$ be a tiling. For all $\sigma \in \mathcal{S}_c$ and $n \in \mathbb{N}$ we have $|\text{Grid}_n(\mathcal{T})| = |\text{Grid}_n(C_\sigma(\mathcal{T}))|$.

Proof. Swapping two adjacent columns i and $i + 1$ can be expressed as compositions of bijections, $\text{rev}_{[i,i]} \circ \text{rev}_{[i+1,i+1]} \circ \text{rev}_{[i,i+1]}$ and by Lemma 3.5 we can produce any permutation of columns given adjacent swaps. \square

3.3 Sliding

Sliding is a generalization (to the setting of tilings) of the sliding lemma described in Babson and West [15, Lemma 2.2].

Definition 3.7. Let $\pi = \pi_1^{(x_1,0)} \pi_2^{(x_2,0)} \dots \pi_n^{(x_n,0)} \in \mathcal{G}_n^{(c,1)}$. We define *sliding* as the map $\Upsilon : \mathcal{G}_n^{(c,1)} \rightarrow \mathcal{G}_n^{(c,1)}$ defined as follows.

- a) If π has no elements in $(1, 0)$, we move whatever is in $(0, 0)$ to $(1, 0)$ and we are done.
- b) If π has elements in $(1, 0)$, we start by shifting all elements in $(2, 0), (3, 0), \dots, (c - 1, 0)$ one cell to the right and then we do the following until we have depleted the elements of π in $(0, 0)$:
 - i. If the rightmost element in $(0, 0)$ is larger than all elements in $(1, 0)$, we move the rightmost element of $(1, 0)$ to $(2, 0)$ as the leftmost element and then we move the rightmost element of $(0, 0)$ to $(1, 0)$ as the leftmost element.
 - ii. If the rightmost element in $(0, 0)$ has a larger element in $(1, 0)$, then we move the rightmost element of $(0, 0)$ to the right of the smallest element in $(1, 0)$ that is larger than it and that element is moved to be the leftmost element in $(2, 0)$.

Finally we shift all elements one cell to the left.

This mapping is outlined in Algorithm 3.1¹. An example for $3594^{(0,0)}62^{(1,0)}718^{(2,0)}$ can be seen in Figure 3.6.

Definition 3.8. A pair of tilings

$$(\mathcal{T}_1, \mathcal{T}_2) = (((c, 1), \mathcal{O}_1, \{\mathcal{R}_1, \dots, \mathcal{R}_k\}), ((c, 1), \mathcal{O}_2, \{\mathcal{R}'_1, \dots, \mathcal{R}'_k\}))$$

is *slidable* if the following conditions are satisfied.

- The tilings share requirements, that is $\{\mathcal{R}_1, \dots, \mathcal{R}_k\} = \{\mathcal{R}'_1, \dots, \mathcal{R}'_k\}$.
- No gridded permutation in $\mathcal{R}_1 \cup \dots \cup \mathcal{R}_k$ has positions in columns 0 and 1.
- There exists an $n > 1$ such that the obstructions $12 \dots n^{(0,0)}$, $12 \dots (n-1)^{(0,0)}n^{(1,0)}$ and $12^{(1,0)}$ belong to \mathcal{O}_1 , $12 \dots n^{(1,0)}$, $1^{(0,0)}23 \dots n^{(1,0)}$ and $12^{(0,0)}$ belong to \mathcal{O}_2 and no other obstructions that are entirely within columns 0 and 1 belong to either \mathcal{O}_1 or \mathcal{O}_2 .

¹Every algorithm in this thesis is accompanied by a Python implementation that can be found in the appropriate repository at <https://github.com/PermutaTriangle>.

Algorithm 3.1 The sliding algorithm

Input: A gridded permutation $\pi = \pi_1^{(x_1,0)} \pi_2^{(x_2,0)} \dots \pi_n^{(x_n,0)} \in \mathcal{G}_n^{(m,1)}$.
Output: A gridded permutation in $\mathcal{G}_n^{(m,1)}$ with $(0,0)$ slid through $(1,0)$.

```

1: procedure SLIDE( $\pi$ )
2:   if  $\{i \mid x_i = 1\} = \emptyset$  then
3:     return  $\pi_1^{(x_1+1-\text{sign}(x_1),0)} \pi_2^{(x_2+1-\text{sign}(x_2),0)} \dots \pi_n^{(x_n+1-\text{sign}(x_n),0)}$   $\triangleright$  shift column 0
4:   end if
5:    $\triangleright$  Shift elements that are not in the first two columns
6:   for  $i \leftarrow |\{j \mid x_j < 2\}| + 1, n$  do
7:      $x_i \leftarrow x_i + 1$ 
8:   end for
9:    $\triangleright$  Deplete the elements in  $(0,0)$ 
10:  while  $\{j \mid x_j = 0\} \neq \emptyset$  do
11:     $i \leftarrow \max \{j \mid x_j = 0\}$ 
12:     $A \leftarrow \{j \mid x_j = 1 \text{ and } \pi_j > \pi_i\}$ 
13:     $k \leftarrow |\{j \mid x_j < 2\}|$ 
14:    if  $A = \emptyset$  then
15:       $x_k \leftarrow 2$ 
16:       $x_i \leftarrow 1$ 
17:    else
18:       $j \leftarrow \max(A)$ 
19:       $\alpha \leftarrow \pi_i$ 
20:       $\beta \leftarrow \pi_j$ 
21:      for  $z \leftarrow i, j - 1$  do
22:         $\pi_z^{(x_z,0)} \leftarrow \pi_{z+1}^{(x_{z+1},y_{z+1})}$ 
23:      end for
24:       $\pi_{j-1} \leftarrow \alpha$ 
25:      for  $z \leftarrow j, k - 1$  do
26:         $\pi_z^{(x_z,0)} \leftarrow \pi_{z+1}^{(x_{z+1},y_{z+1})}$ 
27:      end for
28:       $\pi_k^{(x_k,0)} \leftarrow \beta^{(2,0)}$ 
29:    end if
30:  end while
31:  return  $\pi_1^{(x_1-1,0)} \pi_2^{(x_2-1,0)} \dots \pi_n^{(x_n-1,0)}$ 
32: end procedure

```

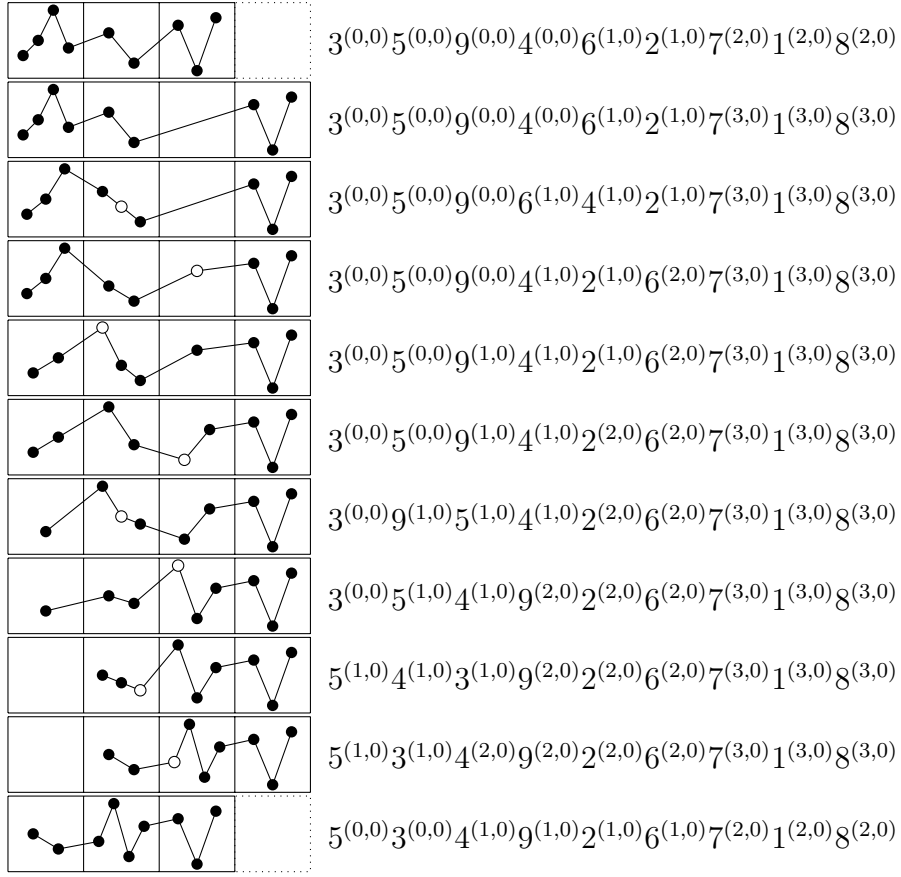


Figure 3.6: The sliding of $3594^{(0,0)}62^{(1,0)}718^{(2,0)}$.

- For any obstruction $\pi = \pi_1^{(x_1,0)} \pi_2^{(x_2,0)} \dots \pi_n^{(x_n,0)} \in \mathcal{O}_1$ with at least one element outside columns 0 and 1 the subsequence $\Xi(\{i \in [w] \mid x_i \in \{0,1\}\}, \pi)$ is either empty or in

$$\{j(j+1)^{(0,0)}, j^{(0,0)}(j+1)^{(1,0)}, j^{(0,0)}, j^{(1,0)} \mid j \in \mathbb{Z}^+\}.$$

Similarly, for \mathcal{O}_2 , the subsequence must be either empty or in

$$\{j(j+1)^{(1,0)}, j^{(0,0)}(j+1)^{(1,0)}, j^{(0,0)}, j^{(1,0)} \mid j \in \mathbb{Z}^+\}.$$

- Any obstruction entirely outside columns 0 and 1 in \mathcal{O}_1 must exist in \mathcal{O}_2 and vice versa.
- Let $o_1 = \pi_1^{(0,0)} \pi_2^{(x_2,0)} \dots \pi_s^{(x_s,0)}$ and $o_2 = \pi_1^{(1,0)} \pi_2^{(x_2,0)} \dots \pi_s^{(x_s,0)}$ where $x_2, s > 1$ then $o_1 \in \mathcal{O}_1$, $o_2 \in \mathcal{O}_1$, $o_1 \in \mathcal{O}_2$ and $o_2 \in \mathcal{O}_2$ are equivalent. That is, any crossing obstruction with one point in the first two columns must exist in both tilings and with that point in both columns.
- Let

$$\begin{aligned} o_1 &= \pi_1^{(0,0)} \pi_2^{(0,0)} \pi_3^{(x_3,0)} \dots \pi_s^{(x_s,0)} \\ o_2 &= \pi_1^{(0,0)} \pi_2^{(1,0)} \pi_3^{(x_3,0)} \dots \pi_s^{(x_s,0)} \\ o_3 &= \pi_1^{(1,0)} \pi_2^{(1,0)} \pi_3^{(x_3,0)} \dots \pi_s^{(x_s,0)} \end{aligned}$$

where $s > 2$, $x_3 > 1$ and $\pi_1 + 1 = \pi_2$, then $o_1 \in \mathcal{O}_1$, $o_2 \in \mathcal{O}_1$, $o_2 \in \mathcal{O}_2$ and $o_3 \in \mathcal{O}_2$ are equivalent. That is, any crossing obstruction with $j(j+1)$ in the first two columns (and at least one point outside of those columns) must exist in both tilings where $j(j+1)$ has been spread between the first two columns in all possible ways, excluding the one that collides with the 12 obstruction.

An example of two slidable tilings can be seen in Figure 3.7.

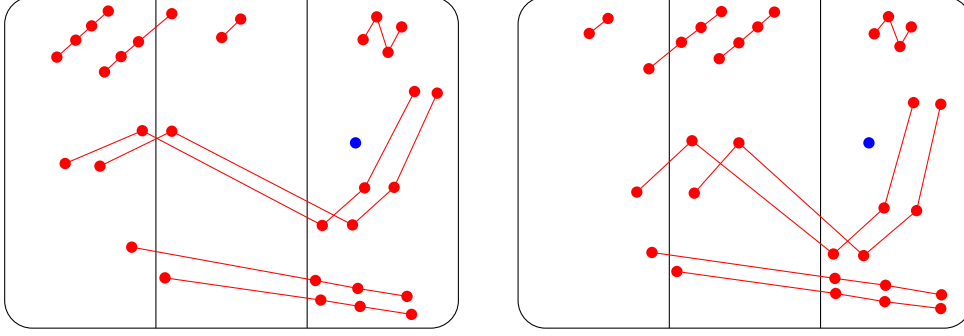


Figure 3.7: Typical slidable tilings.

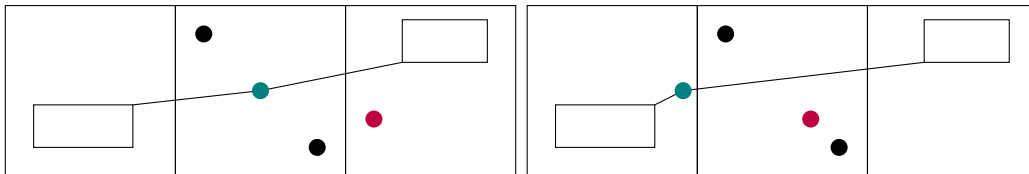
Lemma 3.9. Let $(\mathcal{T}_1, \mathcal{T}_2)$ be slidable and $i \in \mathbb{N}$, then $\Upsilon(\pi) \in \text{Grid}_i(\mathcal{T}_2)$ for every $\pi \in \text{Grid}_i(\mathcal{T}_1)$.

Proof. If π contains no elements within the first two columns it is mapped to itself and since the tilings share requirements and are identical outside the first two columns then $\Upsilon(\pi) \in \text{Grid}_i(\mathcal{T}_2)$. If π contains no elements in column 1 then whatever is in column 0 is moved to column 1. Since π avoided $12 \cdots n$ in column 0 then $\Upsilon(\pi)$ will avoid $12 \cdots n$ in column 1 and any crossing obstruction in \mathcal{T}_2 with one or two points in column 1 would have existed with those points in column 0 in \mathcal{T}_1 and therefore $\Upsilon(\pi) \in \text{Grid}_i(\mathcal{T}_2)$.

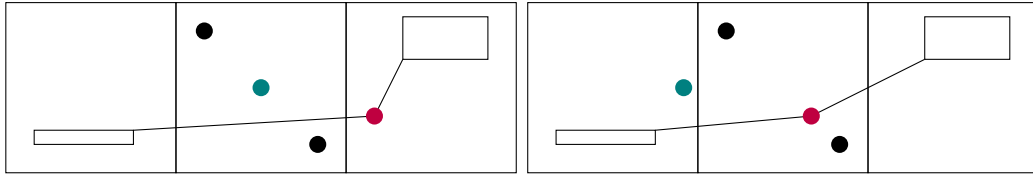
Now suppose π has elements in column 1. By the end of the mapping all columns are shifted one to the left but we will refer to their indices before the shift in the mapped tiling. Note that the mapping is defined by placing elements in-order into column 1 so we will never form an increasing pattern there.

Suppose after a single step in the map that we form the first $12 \cdots n$ (spread in any way across the first 3 columns). Note that it can contain at most one point in column 1 as that column is decreasing.

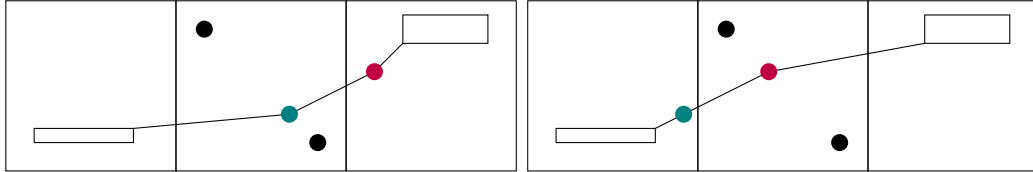
If the $12 \cdots n$ pattern does contain the newly moved point in column 1 and not the one we moved to column 2, then the pattern must have been there before the step was taken.



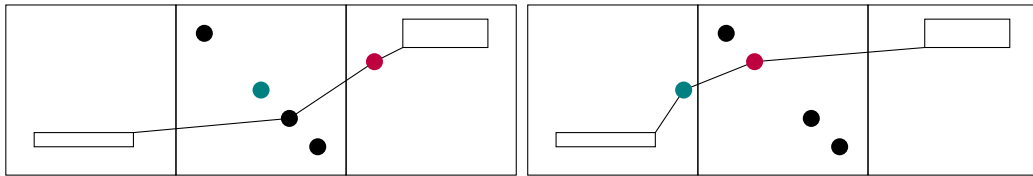
Suppose this pattern includes the newly moved element in column 2 and no elements in column 1 then the pattern must have existed before regardless of whether the new element in column 2 was the smallest element in column 1 before the move or not.



Finally suppose the pattern involves elements from both columns 1 and 2. Here two scenario can arise. One were the pattern involves both moved elements and the other only the one moved to column 2. In the first one the pattern remains as we move both points together when we backtrack.



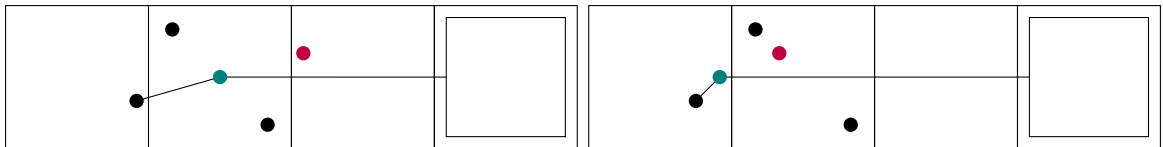
In the latter, there is a smaller element in column 1 than the moved one that belongs to the pattern. When we go back, the moved element goes back from column 1 to column 0 but that is between the element in column 1 that formed the pattern and the element that was moved to column 2, preserving the pattern.



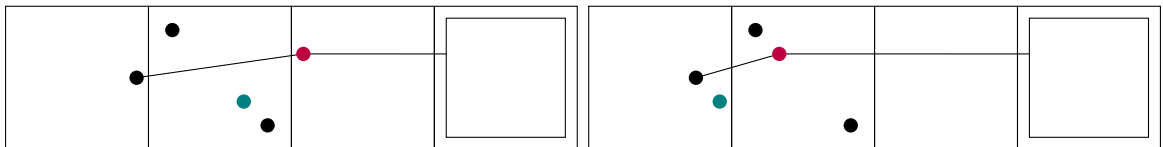
In the case of the first crossing obstructions forming for the first time with one element in the first 3 columns and some outside, then wherever that element was before must have been an occurrence of that pattern as well.

Now suppose we form the crossing pattern with the elements $k(k+1)$ in the first three columns (spread in any way) for the first time. This can happen in several ways. Note that the two increasing elements can not both be in column 1 as it is always decreasing.

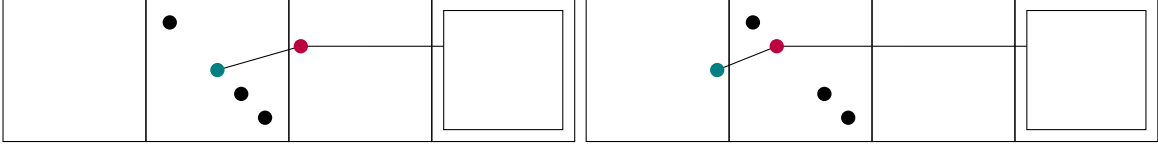
Suppose the $k(k+1)$ part in the first 3 columns occurs between the first two, then the pattern must have existed before with $k(k+1)$ in column 0.



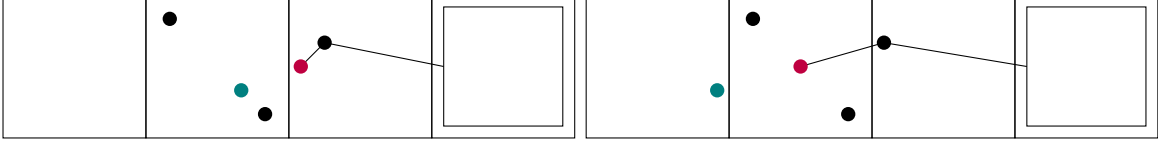
If the pattern occurs with $k(k+1)$ in columns 0 and 2, then it must have existed with $k(k+1)$ in columns 0 and 1 before.



If the pattern occurs with $k(k+1)$ in columns 1 and 2 they must be the two elements that were moved since they are adjacent in value and increasing and therefore the pattern must have existed with $k(k+1)$ in columns 0 and 1 before.



If the pattern occurs with $k(k+1)$ entirely within column 2, then the pattern must have existed with $k(k+1)$ in columns 1 and 2 before.



The trivial bijection of shifting the entire gridded permutation one column to the left (with column 0 being depleted after the mapping) will not create any of these patterns. Since no obstruction from \mathcal{T}_2 can occur, $\Upsilon(\pi)$ is in $\text{Grid}_i(\mathcal{T}_2)$. \square

Definition 3.10. Let $\pi = \pi_1^{(x_1,0)} \pi_2^{(x_2,0)} \dots \pi_n^{(x_n,0)} \in \mathcal{G}_n^{(c,1)}$. We define *inverse sliding* as the map $\Upsilon^{-1} : \mathcal{G}_n^{(c,1)} \rightarrow \mathcal{G}_n^{(c,1)}$ defined as follows.

- a) If π has no elements in $(0,0)$, we move whatever is in $(1,0)$ to $(0,0)$ and we are done.
- b) If π has elements in $(0,0)$, we start by shifting all elements one cell to the right and then we do the following until we have depleted the elements of π in $(2,0)$:
 - i. If the leftmost element in $(2,0)$ is smaller than all elements in $(1,0)$, we move the leftmost element of $(1,0)$ to $(0,0)$ as the rightmost element and then we move the leftmost element of $(2,0)$ to $(1,0)$ as the rightmost element.
 - ii. If the leftmost element in $(2,0)$ has a smaller element in $(1,0)$, then we move the leftmost element in $(2,0)$ to the left of the largest element in $(1,0)$ that is smaller than it and that element is moved to be the rightmost element in $(0,0)$.

Finally we shift all elements in $(3,0), (4,0), \dots$ one cell to the left.

Lemma 3.11. Let $(\mathcal{T}_1, \mathcal{T}_2)$ be slidable and $i \in \mathbb{N}$, then $\Upsilon^{-1}(\pi) \in \text{Grid}_i(\mathcal{T}_1)$ for every $\pi \in \text{Grid}_i(\mathcal{T}_2)$.

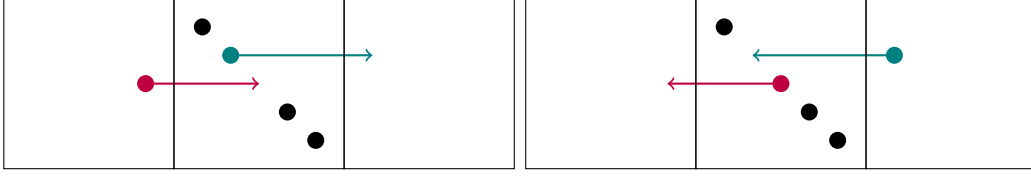
Proof. The argument when $(0,0)$ is empty is an analog of the case in Lemma 3.9 when $(1,0)$ is empty so we assume that π has elements in $(0,0)$. The avoidance of $12 \dots n$ and any crossing obstruction is an analog of the same case in Lemma 3.9, viewing the images from right to left instead. \square

Proposition 3.12. Let $(\mathcal{T}_1, \mathcal{T}_2)$ be slidable, then $|\text{Grid}_i(\mathcal{T}_1)| = |\text{Grid}_i(\mathcal{T}_2)|$ for all $i \in \mathbb{N}$.

Proof. Let $\pi \in \text{Grid}_i(\mathcal{T}_1)$ and $\pi' \in \text{Grid}_i(\mathcal{T}_2)$. A step in Υ where we move an element larger than all the elements in $(1,0)$ is cancelled out by the corresponding move in Υ^{-1} and vice versa.



A step in Υ where we move an element not larger than all elements in $(1, 0)$ is cancelled out by the corresponding move in Υ^{-1} and vice versa.



Both maps move one point into and one out of $(1, 0)$ at every step, so the number of elements in $(1, 0)$ is fixed. Therefore the number of elements in the other column of the first two must be fixed. When mapping π with Υ we take a certain amount of steps and when we map back with Υ^{-1} we take the same amount of steps, each cancelling out the steps of Υ and therefore $\Upsilon^{-1}(\Upsilon(\pi)) = \pi$ and similarly $\Upsilon(\Upsilon^{-1}(\pi')) = \pi'$ and thus, Υ is a bijection between $\text{Grid}_i(\mathcal{T}_1)$ and $\text{Grid}_i(\mathcal{T}_2)$. \square

We can extend the Definition 3.8 of slidable tilings to any nonequal columns c_1 and c_2 instead of 0 and 1. The conditions would replace any constraints involving the first two columns with c_1 and c_2 and allow for subsequences on either side and between.

Proposition 3.13. Let $(\mathcal{T}_1, \mathcal{T}_2)$ be slidable in columns c_1 and c_2 , then $|\text{Grid}_i(\mathcal{T}_1)| = |\text{Grid}_i(\mathcal{T}_2)|$ for all $i \in \mathbb{N}$.

Proof. Suppose the tilings have c columns and let $\sigma = \sigma_1 \sigma_2 \cdots \sigma_c \in \mathcal{S}_c$ where $\sigma_1 = c_1$ and $\sigma_2 = c_2$. We can permute the columns so c_1 and c_2 are the first two, slide them and permute back, that is we can use a composition of bijections, $C_{\sigma^{-1}} \circ \Upsilon \circ C_{\sigma}$, to map between the two. \square

Chapter 4

Parallel specifications

The concept of parallelism¹ in specifications is a binary relation over $\mathfrak{A} \times \mathfrak{A}$ where \mathfrak{A} is the set of all specifications. The purpose of parallelism is to structurally match specifications with the aim of constructing a bijection between them. In order to define this relation we will need to formalize some notation and define supporting concepts.

A combinatorial class that is empty or contains a single element of size less than 2 is called a *terminal class*. Let

$$\begin{aligned}\mathcal{C}^{(1)} &\cong \mathcal{C}^{(11)} \circ_1 \mathcal{C}^{(12)} \circ_1 \dots \circ_1 \mathcal{C}^{(1n_1)} \\ \mathcal{C}^{(2)} &\cong \mathcal{C}^{(21)} \circ_2 \mathcal{C}^{(22)} \circ_2 \dots \circ_2 \mathcal{C}^{(2n_2)} \\ &\vdots \\ \mathcal{C}^{(k)} &\cong \mathcal{C}^{(k1)} \circ_k \mathcal{C}^{(k2)} \circ_k \dots \circ_k \mathcal{C}^{(kn_k)}\end{aligned}$$

be the rules of a specification $\check{\mathcal{C}}$ for a class $\mathcal{C}^{(1)}$ where $\circ_1, \circ_2, \dots, \circ_k$ are constructors and each $\mathcal{C}^{(ij)}$ is a terminal class or in $\{\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \dots, \mathcal{C}^{(k)}\}$ for

$$(i, j) \in \mathcal{D}(\check{\mathcal{C}}) = \{(a, b) \mid a \in [k], b \in [n_a]\}.$$

For any $i \in [k]$ where $n_i = 1$ the constructor \circ_i is a unary operator and in the case of equivalence rules, $\mathcal{C}^{(i)} \cong \mathcal{C}^{(i1)}$, the constructor \circ_i is an identity unary operator Θ . To describe this specification we will use the notation

$$\check{\mathcal{C}} = (L, O, R, D) = \left(\begin{pmatrix} \mathcal{C}^{(1)} \\ \mathcal{C}^{(2)} \\ \vdots \\ \mathcal{C}^{(k)} \end{pmatrix}, \begin{pmatrix} \circ_1 \\ \circ_2 \\ \vdots \\ \circ_k \end{pmatrix}, \begin{pmatrix} R_{11} & R_{12} & \dots & R_{1t} \\ R_{21} & R_{22} & \dots & R_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ R_{k1} & R_{k2} & \dots & R_{kt} \end{pmatrix}, \begin{pmatrix} n_1 \\ n_2 \\ \vdots \\ n_k \end{pmatrix} \right)$$

where $t = \max\{n_1, n_2, \dots, n_k\}$, $R_{ij} = \mathcal{C}^{(ij)}$ for $(i, j) \in \mathcal{D}(\check{\mathcal{C}})$ and the remaining entries of R are the empty set (and of no importance). If we revisit the specification for $\text{Av}(132)$ from Section 2.4 with this notation we have

$$\check{\mathcal{C}} = \left(\begin{pmatrix} \text{Av}(132) \\ \text{Av}_{\geq 1}(132) \end{pmatrix}, \begin{pmatrix} \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & \text{Av}_{\geq 1}(132) & \emptyset \\ \text{Av}(132) & \{\bullet\} & \text{Av}(132) \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \end{pmatrix} \right).$$

¹Not to be confused with the programming paradigm.

4.1 Specification graphs

Definition 4.1. Let $\check{\mathcal{C}} = (L, O, R, D)$ be a specification with k rules. The *specification graph* of $\check{\mathcal{C}}$ is the rooted directed multigraph $\mathfrak{G}(\check{\mathcal{C}}) = (V, E, r, \text{src}, \text{dst}, \text{op})$ with vertices

$$V = \bigcup_{i=1}^k \{L_i, R_{i1}, R_{i2}, \dots, R_{iD_i}\},$$

labelled edges

$$E = \left[\sum_{i=1}^k D_i \right],$$

a root $r = L_1$ and mappings from edge labels to their sources, destinations and constructors,

$$(\text{src}(e), \text{dst}(e), \text{op}(e)) = (L_{\alpha(e)}, R_{\alpha(e)\beta(e)}, O_{\alpha(e)})$$

with

$$\alpha(e) = \min \{i \in [k] \mid D_1 + D_2 + \dots + D_k \geq e\}$$

and

$$\beta(e) = e - \sum_{i=1}^{\alpha(e)-1} D_i.$$

The edge labels are defined to share order with the lexicographical order of $\mathcal{D}(\check{\mathcal{C}})$. That is, edge i will have a destination R_{jk} if (j, k) is the i^{th} largest entry in $\mathcal{D}(\check{\mathcal{C}})$. The source and constructor for said edge are the corresponding class in L and constructor in O . An example for the aforementioned specification for $\text{Av}(132)$ can be seen in Figure 4.1 where the root is drawn with double circle. The destinations of labels are

$$\begin{pmatrix} \text{dst}(1) & \text{dst}(2) & \emptyset \\ \text{dst}(3) & \text{dst}(4) & \text{dst}(5) \end{pmatrix} = \begin{pmatrix} \{\varepsilon\} & \text{Av}_{\geq 1}(132) & \emptyset \\ \text{Av}(132) & \{\bullet\} & \text{Av}(132) \end{pmatrix} = R.$$

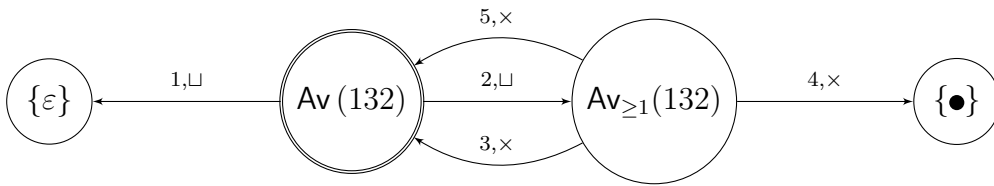


Figure 4.1: The specification graph for a specification for $\text{Av}(132)$.

A path in a rooted multigraph can be described as a sequence of edges². A rooted path is any such sequence that starts with an edge from the root. There is an empty rooted path, ε , corresponding to going nowhere from the root. For a finite rooted path $p = e_1 e_2 \dots e_n$, of size $|p| = n$, in a specification graph we define $\text{tail}(p)$ as $\text{dst}(e_n)$ if $n > 0$ and the root otherwise and refer to it as the *tail* of p . Define \oplus as the operator that concatenates two paths (or edges), that is

$$\alpha_1 \alpha_2 \dots \alpha_n \oplus \beta_1 \beta_2 \dots \beta_k = \alpha_1 \alpha_2 \dots \alpha_n \beta_1 \beta_2 \dots \beta_k.$$

²Using a sequence of vertices is ambiguous in multigraphs.

Definition 4.2. Let $(V, E, r, \text{src}, \text{dst}, \text{op})$ be a specification graph. Given a finite rooted path p , its *path expansion* is the set

$$p^+ = \{p \oplus e \mid e \in E, \text{dst}(e) \neq \emptyset \text{ and } \text{src}(e) = \text{tail}(p)\}.$$

If a path ends in a terminal class, its path expansion is the empty set. Note that we never expand to empty classes and that expansions only depend on the tail of the path. In the graph from Figure 4.1 we have $\varepsilon^+ = \{1, 2\}$, $2^+ = \{23, 24, 25\}$, $24^+ = \emptyset$ and $2523^+ = \{25231, 25232\}$.

Definition 4.3. A *specification path* in a specification graph $(V, E, r, \text{src}, \text{dst}, \text{op})$ is a finite rooted path $p = e_1 e_2 \cdots e_m \in E^m$ in the graph such that $\text{tail}(p)$ is a terminal class or there exists a $j \in [m]$ such that $\text{src}(e_j) = \text{tail}(p)$.

The paths 1, 24, 23, 25, 23232, 232323 and 252324 are examples of specification paths in the specification graph in Figure 4.1. In fact, all finite rooted paths in the graph except the paths ε and 2 are specification paths. We will use the notation $e_1^{\text{op}(e_1)} e_2^{\text{op}(e_2)} \cdots e_m^{\text{op}(e_m)}$ interchangeably with the one without the constructors, opting for the inclusion of constructors when their relevance is of importance.

Let $\check{\mathcal{C}}$ be a specification and $p = e_1 e_2 \cdots e_m \in E^m$ be a specification path in $\mathfrak{G}(\check{\mathcal{C}}) = (V, E, r, \text{src}, \text{dst}, \text{op})$. The *nonequivalent steps* of p , denoted p^* , is the set of edge indices of edges in p that do not correspond to a equivalence rule, that is $p^* = \{i \in [m] \mid \text{op}(e_i) \neq \ominus\}$. Suppose we have a specification

$$\check{\mathcal{C}} = \left(\begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix}, \begin{pmatrix} \sqcup \\ \ominus \\ \times \\ \ominus \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & B \\ C & \emptyset \\ \{c\} & D \\ A & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \\ 2 \\ 1 \end{pmatrix} \right)$$

where c is an atom. For the specification path $p = 2356235$ in $\mathfrak{G}(\check{\mathcal{C}})$ the nonequivalent steps are $p^* = \{1, 3, 5, 7\}$. The specification graph for $\check{\mathcal{C}}$, as well as the path p where the nonequivalent steps are solid can be seen in Figure 4.2.

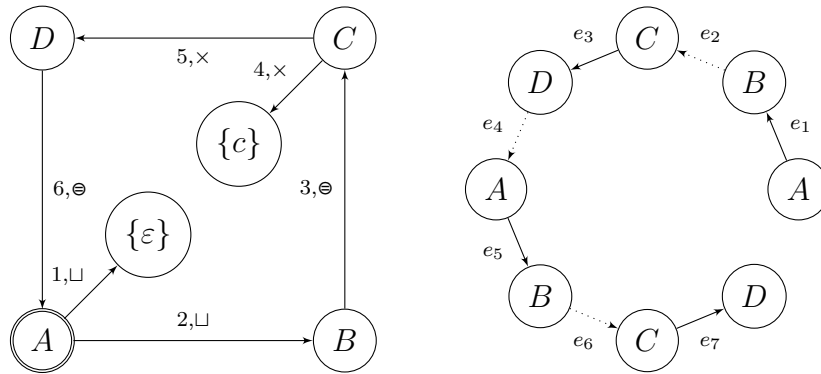


Figure 4.2: A specification graph on the left and a specification path from the graph, $e_1 e_2 \cdots e_7 = 2356235$, on the right with nonequivalent steps, $\{1, 3, 5, 7\}$, as solid arrows.

Given a path $p = e_1 e_2 \cdots e_n$ in a specification graph $(V, E, r, \text{src}, \text{dst}, \text{op})$ we define the *left strip* of p as

$$\text{lstrip}(p) = \begin{cases} \text{lstrip}(e_2 e_3 \cdots e_n) & \text{if } n > 0 \text{ and } \text{op}(e_1) = \ominus, \\ p & \text{otherwise} \end{cases}$$

and the *right extend* of p as

$$\text{rextend}(p) = \begin{cases} \text{rextend}(q) & \text{if } p^+ = \{q\} \text{ and } \text{tail}(p) \cong \text{tail}(q), \\ p & \text{otherwise.} \end{cases}$$

Left strip removes any consecutive edges corresponding to equivalence rules from the left end of a path. Right extend will add the equivalent step to the right end of a path until its tail is not the left hand side of an equivalence rule. In the graph from Figure 4.2 we have $\text{lstrip}(356) = 56$ and $\text{rextend}(35) = 356$.

4.2 Parallel specifications

Definition 4.4. Two constructors \circ_1 and \circ_2 are equivalent, $\circ_1 \equiv \circ_2$, if

$$\mathcal{C}^{(1)} \circ_1 \mathcal{C}^{(2)} \circ_1 \dots \circ_1 \mathcal{C}^{(n)} \cong \mathcal{D}^{(1)} \circ_2 \mathcal{D}^{(2)} \circ_2 \dots \circ_2 \mathcal{D}^{(n)}$$

for all classes $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \dots, \mathcal{C}^{(n)}$ and $\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, \dots, \mathcal{D}^{(n)}$ in the domain of \circ_1 and \circ_2 respectively³ where there exists a $\pi \in \mathcal{S}_n$ such that $\mathcal{C}^{(i)} \cong \mathcal{D}^{(\pi_i)}$ for all $i \in [n]$.

Before defining *parallel specification paths* formally, we will give a brief informal description. Two specification paths in two different specifications are parallel if they have equivalent constructors at every step and end in either isomorphic terminal classes or a recursion at an equal distance, ignoring equivalence steps.

Definition 4.5. Let $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ be two specifications with specification graphs

$$\mathfrak{G}(\check{\mathcal{C}}) = (V_1, E_1, r_1, \text{src}_1, \text{dst}_1, \text{op}_1) \text{ and } \mathfrak{G}(\check{\mathcal{D}}) = (V_2, E_2, r_2, \text{src}_2, \text{dst}_2, \text{op}_2).$$

Let $\alpha = \alpha_1 \alpha_2 \dots \alpha_n$ be a specification path in $\mathfrak{G}(\check{\mathcal{C}})$ and $\beta = \beta_1 \beta_2 \dots \beta_k$ in $\mathfrak{G}(\check{\mathcal{D}})$. We say that α and β are *parallel specification paths* if all of the following conditions are met.

- i. The size, ignoring equivalences, is the same, that is $|\alpha^*| = |\beta^*| = s$.
- ii. For all $q \in [s]$ we have $\text{op}_1(\alpha_{i_q}) \equiv \text{op}_2(\beta_{j_q})$ where $\text{sort}(\alpha^*) = (i_1, i_2, \dots, i_s)$ and $\text{sort}(\beta^*) = (j_1, j_2, \dots, j_s)$.
- iii. Either $\text{tail}(\alpha_n)$ and $\text{tail}(\beta_n)$ are isomorphic terminal classes or there exists a $(i, j) \in [n] \times [k]$ such that $\text{src}_1(\alpha_i) = \text{dst}_1(\alpha_n)$, $\text{src}_2(\beta_j) = \text{dst}_2(\beta_k)$ and

$$|\{\ell \in \alpha^* \mid i \leq \ell \leq n\}| = |\{\ell \in \beta^* \mid j \leq \ell \leq k\}|.$$

We write $\alpha \parallel \beta$ to indicate that specification paths α and β are parallel and $\alpha \nparallel \beta$ to indicate that they are not. Suppose we have specifications

$$\check{\mathcal{C}} = \left(\begin{pmatrix} A \\ B \end{pmatrix}, \begin{pmatrix} \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & B \\ \{c\} & A \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right)$$

³For our purposes it would suffice to restrict the domains to classes of specifications.

and

$$\check{\mathcal{D}} = \left(\begin{pmatrix} E \\ F \\ G \\ H \\ I \\ J \end{pmatrix}, \begin{pmatrix} \sqcup \\ \sqcup \\ \ominus \\ \times \\ \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} F & G \\ \{\varepsilon\} & E \\ H & \emptyset \\ \{d\} & I \\ \{\varepsilon\} & J \\ \{d\} & I \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \\ 1 \\ 2 \\ 2 \\ 2 \end{pmatrix} \right)$$

where c and d are atoms. For the specification graphs $\mathfrak{G}(\check{\mathcal{C}})$ and $\mathfrak{G}(\check{\mathcal{D}})$ in Figure 4.3, we have $2^{\sqcup}4^{\times}2^{\sqcup}4^{\times} \parallel 2^{\sqcup}5^{\ominus}7^{\times}9^{\sqcup}11^{\times}$ and $2^{\sqcup}4^{\times}1^{\sqcup} \not\parallel 1^{\sqcup}$.

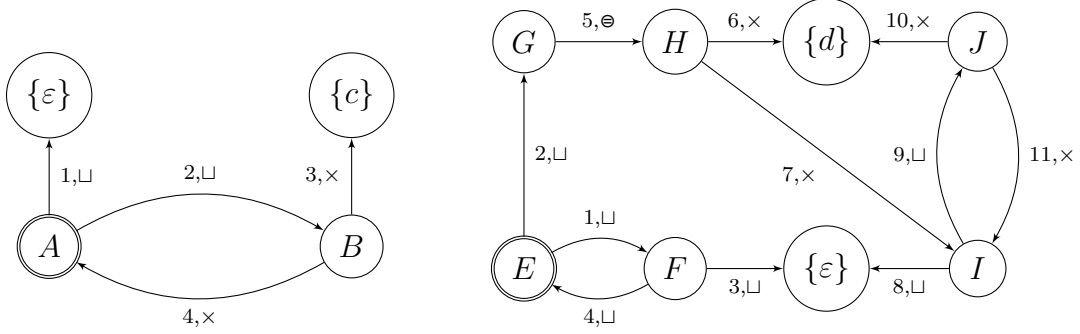


Figure 4.3: The paths $2^{\sqcup}4^{\times}2^{\sqcup}4^{\times}$ in the left graph and $2^{\sqcup}5^{\ominus}7^{\times}9^{\sqcup}11^{\times}$ in the right graph are parallel specification paths.

Before defining parallelism in specifications, we introduce the idea informally and provide a supporting definition. The relation is satisfied if, recursively, every path pair is either parallel or we can pair the expansions of both such that each pair is parallel, starting from the empty rooted paths and ignoring equivalences.

Let $\mathfrak{G}(\check{\mathcal{C}})$ and $\mathfrak{G}(\check{\mathcal{D}})$ be two specification graphs. The rooted paths α in $\mathfrak{G}(\check{\mathcal{C}})$ and β in $\mathfrak{G}(\check{\mathcal{D}})$ are said to be *matchable* if $\text{rextend}(\alpha) \parallel \text{rextend}(\beta)$ or if there exists a bijection $\phi : \text{rextend}(\alpha)^+ \mapsto \text{rextend}(\beta)^+$ such that a and $\phi(a)$ are matchable for all $a \in \text{rextend}(\alpha)^+$.

Definition 4.6. Two specifications $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ are *parallel* if the empty rooted paths in their respective specification graphs are matchable.

We use $\check{\mathcal{C}} \parallel \check{\mathcal{D}}$ and $\check{\mathcal{C}} \not\parallel \check{\mathcal{D}}$ to indicate that specifications are or are not parallel. Take for example the specifications

$$\check{\mathcal{C}} = \left(\begin{pmatrix} A \\ B \end{pmatrix}, \begin{pmatrix} \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & B & \emptyset \\ \{c\} & A & A \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \end{pmatrix} \right)$$

and

$$\check{\mathcal{D}} = \left(\begin{pmatrix} E \\ F \\ G \\ H \end{pmatrix}, \begin{pmatrix} \sqcup \\ \times \\ \ominus \\ \sqcup \end{pmatrix}, \begin{pmatrix} F & \{\varepsilon\} & \emptyset \\ G & \{d\} & E \\ H & \emptyset & \emptyset \\ \{\varepsilon\} & F & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right)$$

where c and d are atoms. Their specification graphs are shown in Figure 4.4 and a matchable recursion tree for their empty rooted paths is shown in Figure 4.5.

The parallel relation is reflexive and symmetric for both specification paths and specifications. This is a natural consequence of the definition of specification paths.

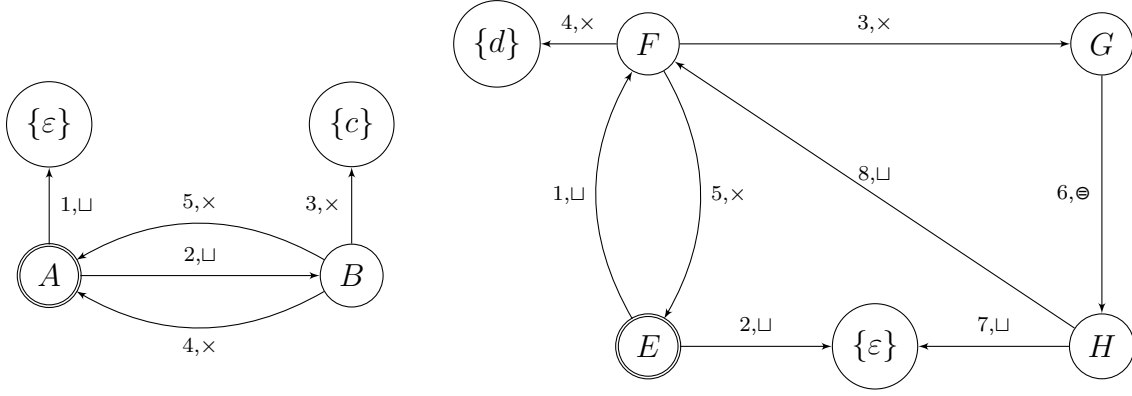


Figure 4.4: The specification graphs of two parallel specifications.

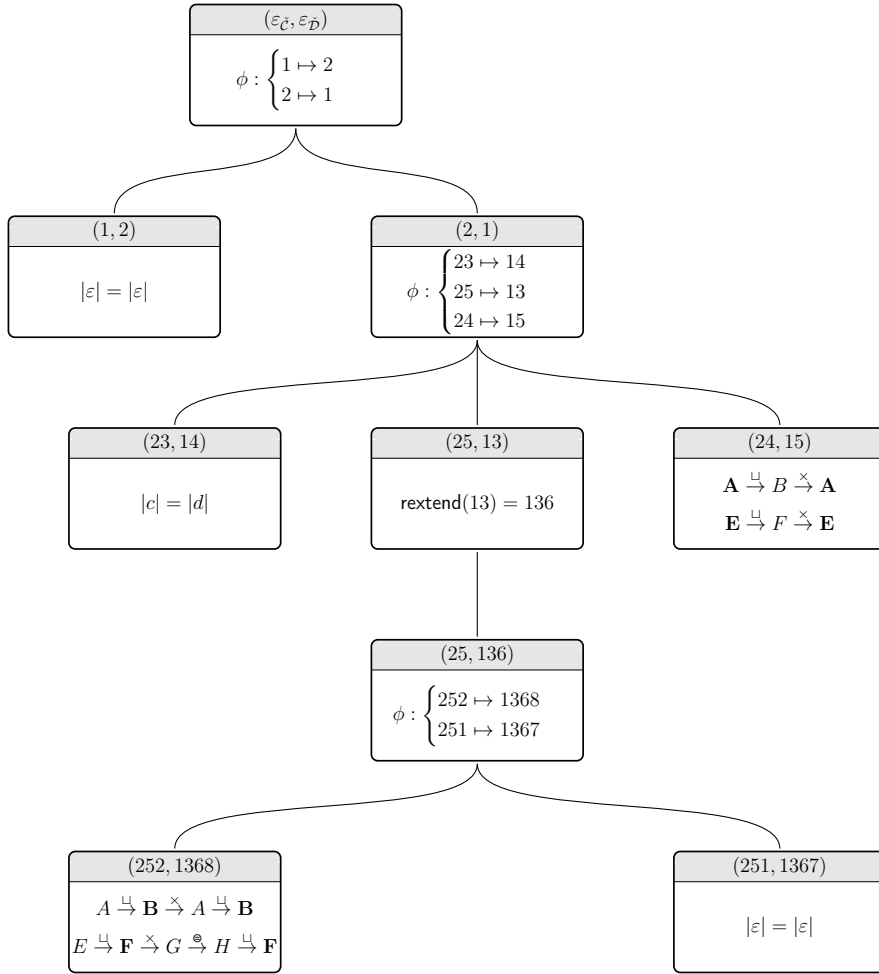


Figure 4.5: The recursion tree that shows that the empty rooted paths from the specifications graphs in Figure 4.4 are matchable.

For specifications we can use the identity map and the inverse of the bijection to match path expansions. The parallel relation is, however, not transitive for specification paths. A counterexample would be a path a parallel to b and b parallel to c , both with recursion but at different positions in b (ignoring equivalences). We will later show that the parallel relation is transitive for specifications but first we need to introduce a systematic way to capture the bijection between the path expansions.

4.3 Matching order

Later when we look to construct bijections from parallel specifications in Chapter 5 the bijections used to pair path expansions will become important. Let α and β be matchable paths in each specification graph of two parallel specifications. Suppose they are matchable because there was a bijection between $\text{rextend}(\alpha)^+$ and $\text{rextend}(\beta)^+$, such that every pair was matchable. The only difference between the paths in $\text{rextend}(\alpha)^+$ is the last edge and the same goes for $\text{rextend}(\beta)^+$. Therefore, what we have is a bijection between the last edge of paths in $\text{rextend}(\alpha)^+$ and $\text{rextend}(\beta)^+$.

Suppose we have rooted paths p_1 and p_2 in $\mathfrak{G}(\check{\mathcal{C}})$ and p_3 and p_4 in $\mathfrak{G}(\check{\mathcal{D}})$ such that p_1 and p_3 are matchable because of a bijection between $\text{rextend}(p_1)^+$ and $\text{rextend}(p_3)^+$ and p_2 and p_4 because of a bijection between $\text{rextend}(p_2)^+$ and $\text{rextend}(p_4)^+$. If

$$\text{tail}(\text{rextend}(p_1)) = \text{tail}(\text{rextend}(p_2)) \text{ and } \text{tail}(\text{rextend}(p_3)) = \text{tail}(\text{rextend}(p_4))$$

then, in terms of last edges in their expansions, the domain and codomain of the two bijections are identical as the expansion only depends on tails. By Definition 4.6 there is nothing stopping us from using different bijections in terms of last edges (if more than one exist) for the same tail pair if it comes up multiple times but then again, there is nothing stopping us from using the same one every time and that is precisely what we will do. This way, we can relate pairs of classes, in each specification, that are on the left of a nonequivalence rule to a bijection between edges from them. We refer to this mapping as the *matching order* of the pair of classes.

Definition 4.7. Let $\check{\mathcal{C}} = (L^{(1)}, O^{(1)}, R^{(1)}, D^{(1)})$ and $\check{\mathcal{D}} = (L^{(2)}, O^{(2)}, R^{(2)}, D^{(2)})$ be two parallel specifications with n and m rules respectively. Let

$$(\mathcal{C}, \mathcal{D}) \in \{L_1^{(1)}, L_2^{(1)}, \dots, L_n^{(1)}\} \times \{L_1^{(2)}, L_2^{(2)}, \dots, L_m^{(2)}\}$$

be a pair of classes that are tails of paths $\text{rextend}(\alpha)$ and $\text{rextend}(\beta)$. Suppose α and β are matchable because of a path expansion bijection $\phi : \text{rextend}(\alpha)^+ \mapsto \text{rextend}(\beta)^+$ with

$$\text{rextend}(\alpha)^+ = \{\text{rextend}(\alpha) \oplus a_i \mid i \in [k]\} \text{ and } \text{rextend}(\beta)^+ = \{\text{rextend}(\beta) \oplus b_i \mid i \in [k]\}$$

where a_i and b_i are edges and $k = |\text{rextend}(\alpha)^+| = |\text{rextend}(\beta)^+|$. The *matching order*, $\Gamma_{\mathcal{C}, \mathcal{D}}$, for \mathcal{C} and \mathcal{D} is a bijection such that $\Gamma_{\mathcal{C}, \mathcal{D}}(a_i) = b_j$ if $\phi(\text{rextend}(\alpha) \oplus a_i) = \text{rextend}(\beta) \oplus b_j$.

The bijections shown in the recursion tree in Figure 4.5 interpreted with matching order for classes of specifications can be seen in Table 4.1. There are no recurring pairs in this example but if there were, one would have to choose a fixed bijection and which one does not matter. The benefit of using the matching order is that it allows us to extend this pairing to paths of any finite size.

4.4 Path matching

Define

$$\mathcal{P}(\check{\mathcal{C}}) = \{p \mid p \text{ is a finite rooted path in } \mathfrak{G}(\check{\mathcal{C}}) \text{ and } p = \text{rextend}(p)\}$$

Paths	Tails	Path expansion bijections	Matching order	Expansion tails
$(\varepsilon_{\check{\mathcal{C}}}, \varepsilon_{\check{\mathcal{D}}})$	(A, E)	$\begin{cases} 1 \mapsto 2 \\ 2 \mapsto 1 \end{cases}$	$\begin{cases} 1 \mapsto 2 \\ 2 \mapsto 1 \end{cases}$	$\begin{matrix} \{\varepsilon\} & \{\varepsilon\} \\ B & F \end{matrix}$
$(2, 1)$	(B, F)	$\begin{cases} 23 \mapsto 14 \\ 25 \mapsto 13 \\ 24 \mapsto 15 \end{cases}$	$\begin{cases} 3 \mapsto 4 \\ 5 \mapsto 3 \\ 4 \mapsto 5 \end{cases}$	$\begin{matrix} \{c\} & \{d\} \\ A & G \\ A & E \end{matrix}$
$(25, 135)$	(A, H)	$\begin{cases} 252 \mapsto 1368 \\ 251 \mapsto 1367 \end{cases}$	$\begin{cases} 2 \mapsto 8 \\ 1 \mapsto 7 \end{cases}$	$\begin{matrix} \{\varepsilon\} & \{\varepsilon\} \\ B & F \end{matrix}$

Table 4.1: The bijections from Figure 4.5 interpreted with matching order.

for a specification $\check{\mathcal{C}}$. These are the paths that do not have a tail on the left hand side of an equivalence rule. Given a specification $\check{\mathcal{D}}$, parallel to $\check{\mathcal{C}}$, we can extend the matching of edges to paths in $\mathcal{P}(\check{\mathcal{C}})$ and $\mathcal{P}(\check{\mathcal{D}})$. We will mostly do so sequentially for each edge but special attention must be paid to equivalence rules.

Let $\alpha = \alpha_1 \alpha_2 \cdots \alpha_n$ be a path in $\mathfrak{G}(\check{\mathcal{C}})$ and $\beta = \beta_1 \beta_2 \cdots \beta_k$ in $\mathfrak{G}(\check{\mathcal{D}})$ where $\check{\mathcal{C}} \parallel \check{\mathcal{D}}$. Define

$$\Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\alpha, \beta) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\text{lstrip}(\alpha), \text{rextend}(\beta))$$

where

$$\psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\alpha, \beta) = \begin{cases} \beta & \text{if } n = 0, \\ \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\alpha_2 \alpha_3 \cdots \alpha_n, \beta \oplus \Gamma_{\text{src}(\alpha_1), \text{tail}(\beta)}(\alpha_1)) & \text{otherwise.} \end{cases}$$

The purpose of $\Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}$ is to apply $\psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}$ after handling removing or adding equivalence steps while $\psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}$ converts an edge in one specification graph to a corresponding one in the other. For $\psi_{\check{\mathcal{D}}, \check{\mathcal{C}}}$ we would replace Γ with Γ^{-1} .

Definition 4.8. Let $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ be parallel specifications and $p \in \mathcal{P}(\check{\mathcal{C}})$. The *path matching* of p in $\mathcal{P}(\check{\mathcal{D}})$ is $\gamma_{\check{\mathcal{C}}, \check{\mathcal{D}}}(p) = \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(p, \varepsilon)$.

As an example, consider the path 242523 in $\mathcal{P}(\check{\mathcal{C}})$ in Figure 4.4. Its path matching in $\mathcal{P}(\check{\mathcal{D}})$ and its image path matching in $\mathcal{P}(\check{\mathcal{C}})$ are

$$\begin{aligned} \gamma_{\check{\mathcal{C}}, \check{\mathcal{D}}}(242523) &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(242523, \varepsilon) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(242523, \varepsilon) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(42523, 1) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(42523, 1) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(2523, 15) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(2523, 15) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(523, 151) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(523, 151) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(23, 1513) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(23, 15136) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(3, 151368) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(3, 151368) \\ &= \Psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\varepsilon, 1513684) = \psi_{\check{\mathcal{C}}, \check{\mathcal{D}}}(\varepsilon, 1513684) \\ &= 1513684 \end{aligned}$$

and

$$\begin{aligned}
\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{1513684}) &= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{1513684}, \varepsilon) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{1513684}, \varepsilon) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{513684}, 2) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{513684}, 2) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{13684}, 24) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{13684}, 24) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{3684}, 242) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{3684}, 242) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{684}, 2425) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{684}, 2425) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{4}, 24252) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\mathbf{4}, 24252) \\
&= \Psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\varepsilon, 242523) = \psi_{\check{\mathcal{D}},\check{\mathcal{C}}}(\varepsilon, 242523) \\
&= 242523.
\end{aligned}$$

The matching order of any class pair will always relate a terminal class to another terminal class such that they are isomorphic. Therefore the map $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}$ will preserve terminal class tails in the sense that their tails are isomorphic terminal classes. For each nonequivalent step in a path p in the domain of $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}$ a nonequivalent step is constructed for $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}(p)$. Any other edges added are equivalent steps⁴. Therefore the number of nonequivalent steps is preserved.

Proposition 4.9. The mapping $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}} : \mathcal{P}(\check{\mathcal{C}}) \mapsto \mathcal{P}(\check{\mathcal{D}})$ is a bijection for parallel specifications $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ with $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}} = \gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}^{-1}$.

Proof. Let $(\alpha, \beta) = (\alpha_1\alpha_2\cdots\alpha_a, \beta_1\beta_2\cdots\beta_b) \in \mathcal{P}(\check{\mathcal{C}}) \times \mathcal{P}(\check{\mathcal{D}})$ such that

$$\alpha = A_1 \oplus \alpha_{i_1} \oplus A_2 \oplus \alpha_{i_2} \oplus \cdots \oplus A_k \oplus \alpha_{i_k} \oplus A_{k+1}$$

and

$$\beta = B_1 \oplus \beta_{j_1} \oplus B_2 \oplus \beta_{j_2} \oplus \cdots \oplus B_k \oplus \beta_{j_k} \oplus B_{k+1}$$

where $\text{sort}(\alpha^*) = (\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_k})$ and $\text{sort}(\beta^*) = (\beta_{j_1}, \beta_{j_2}, \dots, \beta_{j_k})$ and A_i and B_i are paths containing only equivalent steps for $i \in [k+1]$. The equivalent steps are uniquely determined by the nonequivalent steps and the map will ignore them for elements in the domain and add them for elements in the codomain.

Suppose that $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}(\alpha) = \beta$, then

$$\Gamma_{\text{src}(\alpha_{i_\ell}), \text{src}(\beta_{j_\ell})}(\alpha_{i_\ell}) = \beta_{j_\ell} \text{ and } \Gamma_{\text{src}(\alpha_{i_\ell}), \text{src}(\beta_{j_\ell})}^{-1}(\beta_{j_\ell}) = \alpha_{i_\ell}$$

for all $\ell \in [k]$. The map $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}$ will ignore any equivalent steps in β and convert β_{j_ℓ} to α_{i_ℓ} for all $\ell \in [k]$. Since the equivalent steps in α are uniquely determined by the nonequivalent steps $\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_k}$, the equivalent steps added by $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}$ will be the same as in α and in the same positions, that is $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}(\beta) = \alpha$.

Analogously, we have $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}(\alpha) = \beta$ assuming $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}(\beta) = \alpha$. Since $\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}(\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}(\alpha)) = \alpha$ and $\gamma_{\check{\mathcal{C}},\check{\mathcal{D}}}(\gamma_{\check{\mathcal{D}},\check{\mathcal{C}}}(\beta)) = \beta$ the map is a bijection. \square

Proposition 4.10. The parallel relation for specifications is transitive.

Proof. Let $\check{\mathcal{A}}, \check{\mathcal{B}}$ and $\check{\mathcal{C}}$ be specifications such that $\check{\mathcal{A}} \parallel \check{\mathcal{B}}$ and $\check{\mathcal{B}} \parallel \check{\mathcal{C}}$. The empty rooted paths, $\varepsilon_{\check{\mathcal{A}}}$ and $\varepsilon_{\check{\mathcal{C}}}$, are matchable if $\text{rextend}(\varepsilon_{\check{\mathcal{A}}}) \parallel \text{rextend}(\varepsilon_{\check{\mathcal{C}}})$ or there exists a bijection

$$\phi : \text{rextend}(\varepsilon_{\check{\mathcal{A}}})^+ \mapsto \text{rextend}(\varepsilon_{\check{\mathcal{C}}})^+$$

⁴The complement of nonequivalent steps.

such that p and $\phi(p)$ are matchable for all $p \in \text{rextend}(\varepsilon_{\check{\mathcal{A}}})^+$. Suppose we arrive, via this expansion, at paths a and c in the specification graphs of $\check{\mathcal{A}}$ and $\check{\mathcal{C}}$ respectively. If the tail of either $\text{rextend}(a)$ or $\text{rextend}(c)$ is a terminal class then the tail of the other must be as well since $\gamma_{\check{\mathcal{B}},\check{\mathcal{C}}}(\gamma_{\check{\mathcal{A}},\check{\mathcal{B}}}(\text{rextend}(a))) = c$ and both maps preserve terminal classes. Furthermore they must be isomorphic and thus the paths are parallel. If they include a recursion at the same distance they are parallel but that is not implied as the parallel relation is not transitive for specification paths. Suppose the paths are not parallel, then we can use the bijection

$$\gamma_{\check{\mathcal{B}},\check{\mathcal{C}}} \circ \gamma_{\check{\mathcal{A}},\check{\mathcal{B}}} : \text{rextend}(a)^+ \mapsto \text{rextend}(c)^+$$

to expand further. Since there are finite possible pairs from each specification, we always end up with a recursion at an equal distance or in terminal classes. \square

4.5 The parallel algorithm

The parallel algorithm checks if two specifications are parallel and if so, returns the matching order. The graphs are expanded as trees simultaneously⁵ and lineal descendants from the root form a path. At any node⁶ in the tree is the tail of its path from the root. A node corresponding to the path q , is the parent of the nodes corresponding to the paths in q^+ , which are called children. Matching orders are stored in a dictionary with class pair keys. We alter the matching order such that both the domain and codomain are standardized, e.g., the i^{th} largest edge label is replaced by i . If we can traverse the trees down to matched specification paths, we have parallel specifications.

The algorithm uses dynamic programming to match pairs, remembering both successes and failures. The matching order is done with an iterative backtracking to avoid nesting recursions and expanding unnecessarily. If we cannot match the first child to some child of the other node, there is no point trying to match the second child. Whether the first and third child match for given nodes has nothing to do with what children were matched previously so we remember child matching failures to limit each pair to a single attempt.

The algorithm takes two specifications as input where each specification is represented by a quadruple (r, o, ϕ, n) where r is the root class and o, ϕ and n are maps from classes to constructors, n -tuple of non-empty children and the number of non-empty children respectively. The pseudocode can be seen in Algorithm 4.1 where we use a **stack** with **size**, **pop** and **push** operations.

Let n and k be the number of classes in the two specifications and m the largest number of children in both specifications. Each pair of classes is never expanded more than once. Each expansion will at most attempt to match each pair of children. By the rule of product there are nk pairs of classes and at most m^2 pairs of children. Therefore the worst case time complexity is $\mathcal{O}(nkm^2)$.

⁵Not concurrently.

⁶We refer to vertices as nodes in algorithmic context.

Algorithm 4.1 The parallel algorithm

Input: Two specifications $\check{\mathcal{C}} = (r_1, o_1, \alpha, n_1)$ and $\check{\mathcal{D}} = (r_2, o_2, \beta, n_2)$.

Output: Boolean (0 or 1) and children order for matched pair

```
1:  $(S, F, A, M) \leftarrow (\emptyset, \emptyset, \emptyset, \emptyset)$  ▷ Successes, Failures, Ancestors, Map
2: procedure P( $\mathcal{C}, \mathcal{D}$ )
3:   if  $(\mathcal{C}, \mathcal{D}) \in S \cup F$  then
4:     return  $|\{(\mathcal{C}, \mathcal{D})\} \cap S|$ 
5:   end if
6:   if  $|\mathcal{C}| = |\mathcal{D}| = 1$  then
7:     return  $2 - |\{ |c| \mid c \in \mathcal{C} \cup \mathcal{D} \}|$ 
8:   end if
9:    $(V, U) \leftarrow (\{\mathcal{C}\}, \{\mathcal{D}\})$ 
10:  while  $o_1(\mathcal{C}) = \ominus$  do
11:     $(V, \mathcal{C}) \leftarrow (V \cup \{\alpha_1(\mathcal{C})\}, \alpha_1(\mathcal{C}))$ 
12:  end while
13:  while  $o_2(\mathcal{D}) = \ominus$  do
14:     $(U, \mathcal{D}) \leftarrow (U \cup \{\beta_1(\mathcal{D})\}, \beta_1(\mathcal{D}))$ 
15:  end while
16:  if  $V \times U \cap A \neq \emptyset$  then
17:    return 1
18:  end if
19:  if  $n_1(\mathcal{C}) \neq n_2(\mathcal{D})$  or  $o_1(\mathcal{C}) \neq o_2(\mathcal{D})$  then
20:    return 0
21:  end if
22:   $(B, A, m) \leftarrow (\emptyset, A \cup V \times U, (1, 2, \dots, n_1(\mathcal{C})))$ 
23:   $s \leftarrow \text{stack}[(1, n_1(\mathcal{C}), \{n_1(\mathcal{C})\}), \dots, (1, 2, \{2\}), (1, 1, \{1\})]$ 
24:  while  $\text{SIZE}(s) > 0$  do
25:     $(i_1, i_2, u) \leftarrow \text{POP}(s)$ 
26:    if  $(i_1, i_2) \in B$  or  $P(\alpha_{i_1}(\mathcal{C}), \beta_{i_2}(\mathcal{D})) \neq 1$  then
27:       $B \leftarrow B \cup \{(i_1, i_2)\}$ 
28:      continue
29:    end if
30:     $m_{i_2} \leftarrow i_1$ 
31:    if  $i_1 = n_1(\mathcal{C})$  then
32:       $(A, S, M) \leftarrow (A \setminus V \times U, S \cup \{(\mathcal{C}, \mathcal{D})\}, M \cup \{((\mathcal{C}, \mathcal{D}), m)\})$ 
33:      return 1
34:    end if
35:    for  $j \in [n_1(\mathcal{C})] \setminus u$  in decreasing order do
36:       $\text{PUSH}(s, (i_1 + 1, j, u \cup \{j\}))$ 
37:    end for
38:  end while
39:   $(A, F) \leftarrow (A \setminus V \times U, F \cup \{(\mathcal{C}, \mathcal{D})\})$ 
40:  return 0
41: end procedure
42: return P( $r_1, r_2$ ),  $M$ 
```

Chapter 5

Parallel bijections

Given an object of a root class in a specification, we need a systematic way of breaking it down into objects of terminal classes and given such objects of terminal classes, a systematic way to reconstruct the object of the root class. The idea behind our bijection is simple. We take an object belonging to the root of a specification, break it down and reconstruct it with a parallel specification.

Let $\check{\mathcal{C}} = (L, (\circ_1, \circ_2, \dots, \circ_k), R, D)$ be a specification. In order for us to achieve this bijection, every rule $L_i \cong R_{i1} \circ_i R_{i2} \circ_i \dots \circ_i R_{iD_i}$ must be accompanied by a bijection between L_i and $R_{i1} \circ_i R_{i2} \circ_i \dots \circ_i R_{iD_i}$. A specification along with such a collection of bijections is said to be *parsable*.

Take for example a rule $\mathcal{C} \cong \mathcal{C}^{(1)} \times \mathcal{C}^{(2)}$. Any bijection for this rule would map an object of \mathcal{C} to a pair of objects (c_1, c_2) with $c_1 \in \mathcal{C}^{(1)}$ and $c_2 \in \mathcal{C}^{(2)}$. It is not as simple for disjoint union as the object will belong to exactly one of the resulting classes. Suppose $\mathcal{C} \cong \mathcal{C}^{(1)} \sqcup \mathcal{C}^{(2)}$ and we have an object $c \in \mathcal{C}$ and a bijection ϕ for this rule, how does $\phi(c)$ look exactly? If this was set union, the bijection would be the identity map but there would be no information about which set the mapped element belongs to. Unique colors, m_1 and m_2 , are used to mark the elements in Flajolet and Sedgewick [9], that is

$$\mathcal{C}^{(1)} \sqcup \mathcal{C}^{(2)} = (\{m_1\} \times \mathcal{C}^{(1)}) \cup (\{m_2\} \times \mathcal{C}^{(2)}).$$

This way, $\phi(c)$ would be a pair of a color and an element in $\mathcal{C}^{(1)} \cup \mathcal{C}^{(2)}$. We propose a different approach, which ultimately achieves the same thing. Instead of marking with colors, we mark with edges from the specification graph. We further extend this marking to any constructor. The bijective nature of the map remains as the edges are uniquely determined but will need some trivial transformation for elements to belong to the original set.

Given a specification $\check{\mathcal{C}} = (L, (\circ_1, \circ_2, \dots, \circ_k), R, D)$ accompanied by a collection of bijections for each of its rules, we refer to $(\phi_1, \phi_2, \dots, \phi_k)$ as its *parser* where ϕ_i maps an element of L_i , to an element of $R_{i1} \circ_i R_{i2} \circ_i \dots \circ_i R_{iD_i}$, using the corresponding bijection extended with the corresponding edge labels. If we look at the specification for Av(132) with the specification graph from Figure 4.1 and let (ϕ_1, ϕ_2) be its parser, then we have $\phi_1(\varepsilon) = (\varepsilon, 1)$, $\phi_1(321) = (321, 2)$ and

$$\phi_2(534612) = ((\text{st}(534), 3), (\text{st}(6), 4), (\text{st}(12), 5)) = ((312, 3), (1, 4), (12, 5)).$$

5.1 Object parsing

Let $\check{\mathcal{C}} = (L, O, R, D)$ be a parsable specification with k rules, a parser $(\phi_1, \phi_2, \dots, \phi_k)$, a specification graph $\mathfrak{G}(\check{\mathcal{C}})$ and an object c in the root class. We define $\omega_{\check{\mathcal{C}}}(c) = \varphi_{\check{\mathcal{C}}}(c, \varepsilon)$ as the *atomic partitioning* of c in $\check{\mathcal{C}}$ where

$$\varphi_{\check{\mathcal{C}}}(c, p) = \begin{cases} \{p\} & \text{if } \text{tail}(p) \text{ is a terminal class} \\ \bigcup_{j=1}^{\ell} \varphi_{\check{\mathcal{C}}}(c_j, p \oplus e_j) & \text{if } \text{tail}(p) = L_i \text{ and } \phi_i(c) = ((c_1, e_1), \dots, (c_\ell, e_\ell)) \end{cases}$$

Let $\check{\mathcal{C}}$ be the aforementioned specification for $\mathbf{Av}(132)$ and using the specification graph in Figure 4.1 and starting with $\pi = 3241 \in \mathbf{Av}(132)$ we have

$$\begin{aligned} \omega_{\check{\mathcal{C}}}(\pi) &= \varphi_{\check{\mathcal{C}}}(3241, \varepsilon) = \varphi_{\check{\mathcal{C}}}(3241, 2) \\ &= \varphi_{\check{\mathcal{C}}}(21, 23) \cup \{24\} \cup \varphi_{\check{\mathcal{C}}}(1, 25) = \{24\} \cup \varphi_{\check{\mathcal{C}}}(21, 232) \cup \varphi_{\check{\mathcal{C}}}(1, 252) \\ &= \{24, 2324, 2524\} \cup \varphi_{\check{\mathcal{C}}}(\varepsilon, 2323) \cup \varphi_{\check{\mathcal{C}}}(1, 2325) \cup \varphi_{\check{\mathcal{C}}}(\varepsilon, 2523) \cup \varphi_{\check{\mathcal{C}}}(\varepsilon, 2525) \\ &= \{24, 2324, 2524, 23231, 25231, 25251\} \cup \varphi_{\check{\mathcal{C}}}(1, 2325) \\ &= \{24, 2324, 2524, 23231, 25231, 25251, 232524\} \cup \varphi_{\check{\mathcal{C}}}(\varepsilon, 232523) \cup \varphi_{\check{\mathcal{C}}}(\varepsilon, 232525) \\ &= \{24, 2324, 2524, 23231, 25231, 25251, 232524, 2325231, 2325251\} \end{aligned}$$

This is shown with a tree in Figure 5.1. Note that the image of this map is not all subsets of the set of paths that end in terminal classes. The set $\{24\}$ is not the atomic partitioning of any object since it requires siblings that would not be generated. Since every step taken involves a bijective rule then $\omega_{\check{\mathcal{C}}}$ is a bijection between the objects of the root class and the possible atomic partitionings.

To define the inverse of $\omega_{\check{\mathcal{C}}}$ we start by providing a few supporting maps. Given a set of terminal ending paths S we define $T(S) = \{(c, p) \mid p \in S \text{ and } c = \text{tail}(p)\}$. Given set of paths and object pairs P we define $\text{nxt}(P)$ as the longest path in P , breaking ties with lexicographic order of edge labels as tuples. We define the *next siblings* as

$$\text{sib}(P) = \{(c, e_1 e_2 \dots e_{n-1} \oplus e') \in P \mid \text{nxt}(P) = e_1 e_2 \dots e_n\}.$$

This mapping is undefined for a singleton set where the only path is the empty one. Suppose

$$\text{sib}(P) = \{(c_1, (p \oplus q_1)), \dots, (c_r, (p \oplus q_r))\}$$

where $q_1 < q_2 < \dots < q_r$, then the *next parent* is

$$\text{par}(P) = (\phi_i^{-1}((c_1, q_1), (c_2, q_2), \dots, (c_r, q_r)), p)$$

if $\text{tail}(p) = L_i$. Note that the way edge labels are defined guarantees consistency between order of labels and the right hand side of rules.

Given an atomic partitioning Q we define $\omega_{\check{\mathcal{C}}}^{-1}(Q) = \vartheta(T(Q))$ where

$$\vartheta(P) = \begin{cases} c & \text{if } |P| = 1 \text{ and } (c, \varepsilon) \in P \\ \vartheta((P \setminus \text{sib}(P)) \cup \{\text{par}(P)\}) & \text{otherwise} \end{cases}$$

Suppose we start with the set of terminal endings from $\omega_{\check{\mathcal{C}}}(\pi)$ in the previous example. We start by extending the set to include the terminal classes, that is

$$\begin{aligned} P = T(\omega_{\check{\mathcal{C}}}(\pi)) &= \{(1, 24), (1, 2324), (1, 2524), (\varepsilon, 23231), (\varepsilon, 25231), \\ &\quad (\varepsilon, 25251), (1, 232524), (\varepsilon, 2325231), (\varepsilon, 2325251)\}. \end{aligned}$$

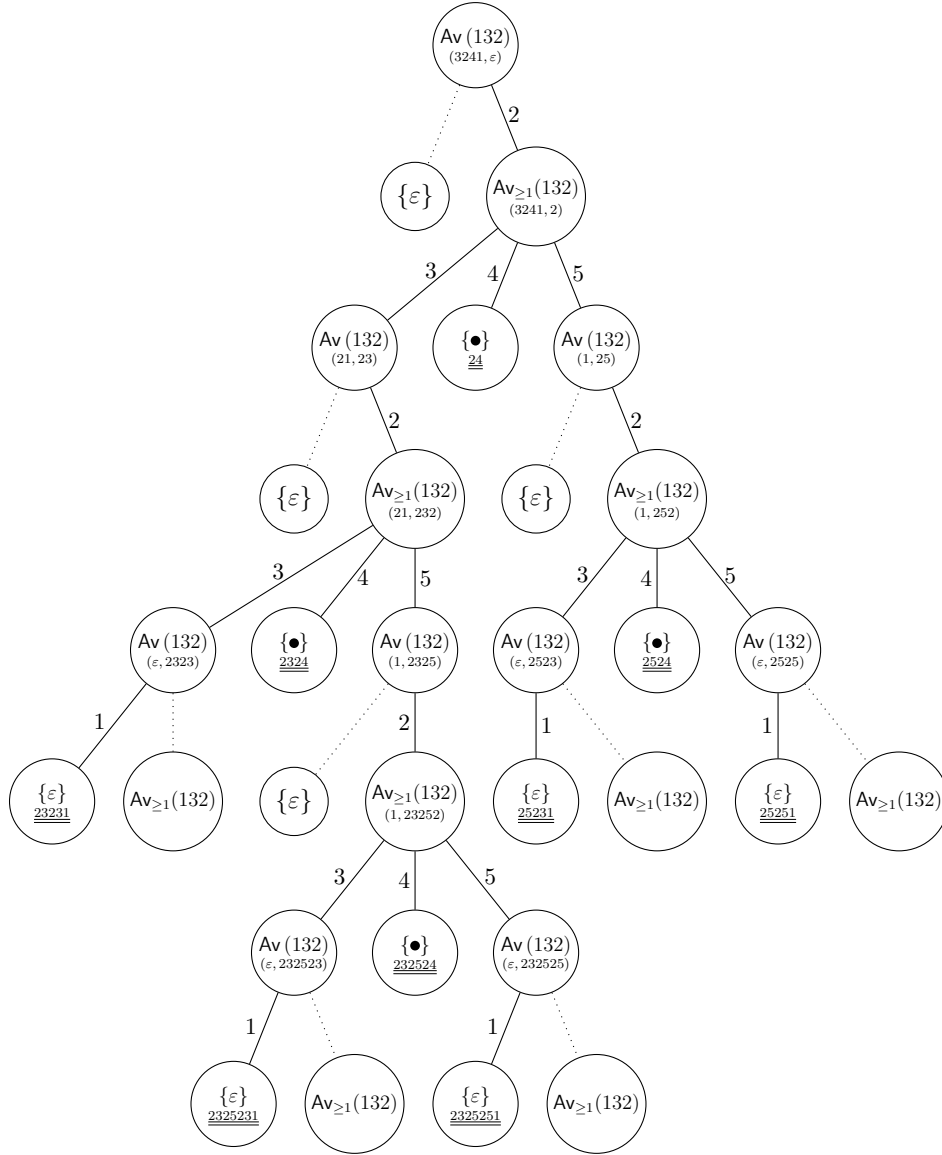


Figure 5.1: The atomic partitioning of 3241 for the specification graph in Figure 4.1. Double underlining is used to highlight paths to terminal classes. The dotted lines represent paths not taken by parental objects.

Now we have $\text{sib}(P) = \{(\varepsilon, 2325251)\}$ and the next step in the algorithm will replace $(\varepsilon, 2325251)$ with $(\varepsilon, 232525)$. The same applies for the next step, replacing $(\varepsilon, 2325231)$ with $(\varepsilon, 232523)$. At this point we have

$$P = \{(1, 24), (1, 2324), (1, 2524), (\varepsilon, 23231), (\varepsilon, 25231), (\varepsilon, 25251), (1, 232524), (\varepsilon, 232523), (\varepsilon, 232525)\}.$$

Now we have $\text{sib}(P) = \{(1, 232523), (\varepsilon, 232524), (\varepsilon, 232525)\}$ and next parent

$$\text{par}(P) = (\phi_2^{-1}((\varepsilon, 232523), (1, 232524), (\varepsilon, 232525)), 23252) = (1, 23252).$$

We replace the siblings with the parent and get

$$P = \{(1, 24), (1, 2324), (1, 2524), (\varepsilon, 23231), (\varepsilon, 25231), (\varepsilon, 25251), (1, 23252)\}.$$

For the next four steps, $\text{sib}(P)$ will be a singleton set with no changes to the object itself and thus we can remove the last step of each of their path, arriving at

$$P = \{(1, 24), (1, 2324), (1, 2524), (\varepsilon, 2323), (\varepsilon, 2523), (\varepsilon, 2525), (1, 2325)\}.$$

Next we have $\text{sib}(P) = \{(\varepsilon, 2323), (1, 2324), (1, 2325)\}$ with parent $(21, 232)$ and after that, $\text{sib}(P) = \{(\varepsilon, 2523), (1, 2524), (\varepsilon, 2525)\}$ with parent $(1, 252)$. These two parents are next, with no siblings and parent $(21, 23)$ and $(1, 25)$ respectively and we arrive at

$$P = \{(21, 23), (1, 24), (1, 25)\}.$$

Now we have

$$\begin{aligned} \vartheta(P) &= \vartheta(P \setminus \{(21, 23), (1, 24), (1, 25)\} \cup \{\phi_2^{-1}((21, 3), (1, 4), (1, 5))\}, 2) \\ &= \vartheta(\{(3241, 2)\}) = \vartheta(\{(3241, \varepsilon)\}) = 3241. \end{aligned}$$

Looking at Figure 5.1, what we are essentially doing is taking the longest path that ends in a terminal class and gathering all siblings required to map in the other direction and taking a step backward, repeating until we are back at the root.

5.2 Parallel bijections

Definition 5.1. If $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ are two parallel specifications with path matching $\gamma_{\check{\mathcal{C}}, \check{\mathcal{D}}}$ and root classes \mathcal{C} and \mathcal{D} , then their *parallel map* is $\mathfrak{P} : \mathcal{C} \mapsto \mathcal{D}$ where, for any $c \in \mathcal{C}$, we have $\mathfrak{P}(c) = \omega_{\check{\mathcal{D}}}^{-1}(\{\gamma_{\check{\mathcal{C}}, \check{\mathcal{D}}}(p) \mid p \in \omega_{\check{\mathcal{C}}}(c)\})$.

As an example we will look at the specifications $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$ for $\mathcal{C}^{(1)} = \text{Av}(231, 321)$ and $\mathcal{D}^{(1)} = \text{Av}(132, 312)$, found by TileScope. The specifications can be described as

$$\check{\mathcal{C}} = \left(\begin{pmatrix} \mathcal{C}^{(1)} \\ \mathcal{C}^{(2)} \\ \mathcal{C}^{(3)} \\ \mathcal{C}^{(4)} \\ \mathcal{C}^{(5)} \end{pmatrix}, \begin{pmatrix} \sqcup \\ \ominus \\ \times \\ \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & \mathcal{C}^{(2)} & \emptyset \\ \mathcal{C}^{(3)} & \emptyset & \emptyset \\ \mathcal{C}^{(1)} & \{1\} & \mathcal{C}^{(4)} \\ \{\varepsilon\} & \mathcal{C}^{(5)} & \emptyset \\ \mathcal{C}^{(4)} & \{1\} & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \\ 3 \\ 2 \\ 2 \end{pmatrix} \right)$$

and

$$\check{\mathcal{D}} = \left(\begin{pmatrix} \mathcal{D}^{(1)} \\ \mathcal{D}^{(2)} \\ \mathcal{D}^{(3)} \\ \mathcal{D}^{(4)} \\ \mathcal{D}^{(5)} \end{pmatrix}, \begin{pmatrix} \sqcup \\ \ominus \\ \times \\ \sqcup \\ \times \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & \mathcal{D}^{(2)} & \emptyset \\ \mathcal{D}^{(3)} & \emptyset & \emptyset \\ \mathcal{D}^{(1)} & \{1\} & \mathcal{D}^{(4)} \\ \{\varepsilon\} & \mathcal{D}^{(5)} & \emptyset \\ \{1\} & \mathcal{D}^{(4)} & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \\ 3 \\ 2 \\ 2 \end{pmatrix} \right)$$

where the rules in both are (in order) top most point placement, row and column separation, factoring, top most point placement and factoring. The classes are summarized in Table 5.1.

The specification graphs for these two specifications are shown in Figure 5.2. Their matching order are shown in Table 5.2. Parallel specifications can be dissimilar in structure when the number of classes grows. For demonstrational purposes we chose specifications with few classes that are prone to result in similarly structured specifications.

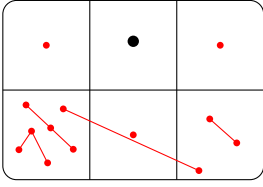
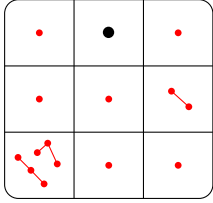
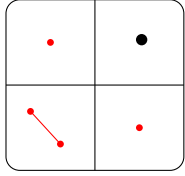
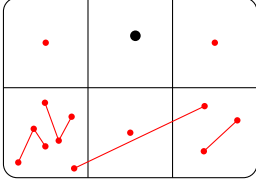
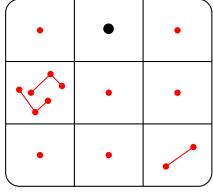
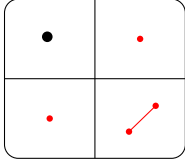
$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$	$\mathcal{C}^{(4)}$	$\mathcal{C}^{(5)}$
		$\text{Av}(21)$	
$\mathcal{D}^{(2)}$	$\mathcal{D}^{(3)}$	$\mathcal{D}^{(4)}$	$\mathcal{D}^{(5)}$
		$\text{Av}(12)$	

Table 5.1: The non-root classes of specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.

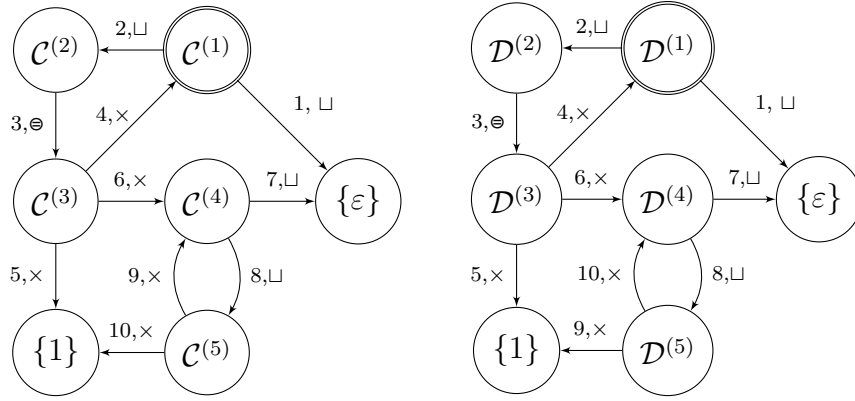


Figure 5.2: The specification graphs of specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.

$\Gamma_{\mathcal{C}^{(1)}, \mathcal{D}^{(1)}}$	$1 \rightarrow 1$
	$2 \rightarrow 2$
	$4 \rightarrow 4$
$\Gamma_{\mathcal{C}^{(3)}, \mathcal{D}^{(3)}}$	$5 \rightarrow 5$
	$6 \rightarrow 6$
$\Gamma_{\mathcal{C}^{(4)}, \mathcal{D}^{(4)}}$	$7 \rightarrow 7$
	$8 \rightarrow 8$
$\Gamma_{\mathcal{C}^{(5)}, \mathcal{D}^{(5)}}$	$9 \rightarrow 10$
	$10 \rightarrow 9$

Table 5.2: The matching order for the specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.

Since we are working with tilings, the objects we map are gridded permutations. The root classes only have a single cell so the bijection to permutations is trivial. Let $\pi = 132^{(0,0)} \in \mathcal{C}^{(1)}$. As can be seen in the upper tree of Figure 5.3, its atomic partitioning is

$$\omega_{\tilde{\mathcal{C}}}(\pi) = \{235, 236(10), 234235, 236897, 2342341, 2342367\}.$$

The path matching of 236(10) in $\mathcal{P}(\check{\mathcal{D}})$ is

$$\begin{aligned}
\gamma_{\check{\mathcal{D}}}(236(10)) &= \Psi_{\check{\mathcal{D}}}(236(10), \varepsilon) = \psi_{\check{\mathcal{D}}}(236(10), \varepsilon) \\
&= \Psi_{\check{\mathcal{D}}}(36(10), 2) = \psi_{\check{\mathcal{D}}}(6(10), 23) \\
&= \Psi_{\check{\mathcal{D}}}((10), 236) = \psi_{\check{\mathcal{D}}}((10), 236) \\
&= \Psi_{\check{\mathcal{D}}}(\varepsilon, 2369) = \psi_{\check{\mathcal{D}}}(\varepsilon, 2369) \\
&= 2369.
\end{aligned}$$

We do the same for the other paths and get

$$\{\gamma_{\check{\mathcal{D}}}(p) \mid p \in \omega_{\check{\mathcal{C}}}(c)\} = \{235, 2369, 234235, 2368(10)7, 2342341, 2342367\}.$$

We then extend this set to contain the tail of every path and work our way up to the root of $\check{\mathcal{D}}$, using the inverse of every rule's bijection, as is shown in Figure 5.3 and end up with $\mathfrak{P}(132) = 231$.

Proposition 5.2. The parallel map is a bijection for parallel specifications.

Proof. The function \mathfrak{P} is a composition of a bijection and a bijection lifted to a powerset. \square

Corollary 5.3. Let \mathcal{C} and \mathcal{D} be the root classes of two parallel specifications, then $\mathcal{C} \cong \mathcal{D}$.

5.3 Nonbijective rules

There are rules that are not inherently bijective but still allow us to count a left hand side from the right hand side. We will discuss a solution to nonbijective rules where there is a map that is not injective. As an example we will look at fusion but this can be generalized beyond that.

Suppose we have class \mathcal{C} on the left hand side of a rule and that ϕ is a mapping of an element of \mathcal{C} to an element of the rule's right hand side that is not injective. Define \bar{c} as $\{c' \in \mathcal{C} \mid \phi(c') = \phi(c)\}$.

Definition 5.4. A rule with a class \mathcal{C} as its left hand side and a mapping ϕ from its left to its right hand side is *indexable* if \bar{c} is a strictly totally ordered finite set for every $c \in \mathcal{C}$ for some binary relation.

Given an indexable rule with a map ϕ from \mathcal{C} we can convert it into a bijection $\hat{\phi}$ by extending the codomain to be the Cartesian product of itself and the natural numbers. Let $\hat{\phi}(c) = (\phi(c), i)$ if c is the $i + 1^{\text{th}}$ largest¹ element in \bar{c} . Its inverse, $\hat{\phi}^{-1}$, will map $(\phi(c), i)$ to c_i if $\text{sort}(\bar{c}) = (c_1, c_2, \dots, c_n)$

As an example we will look at the fusion rule in Figure 2.11 with tilings

$$\mathcal{T}_1 = ((2, 1), \{123^{(0,0)}, 12^{(0,0)}3^{(1,0)}, 1^{(0,0)}23^{(1,0)}, 123^{(1,0)}\}, \emptyset)$$

and

$$\mathcal{T}_2 = ((1, 1), \{123^{(0,0)}\}, \emptyset).$$

¹The offset is done for zero based indexing.

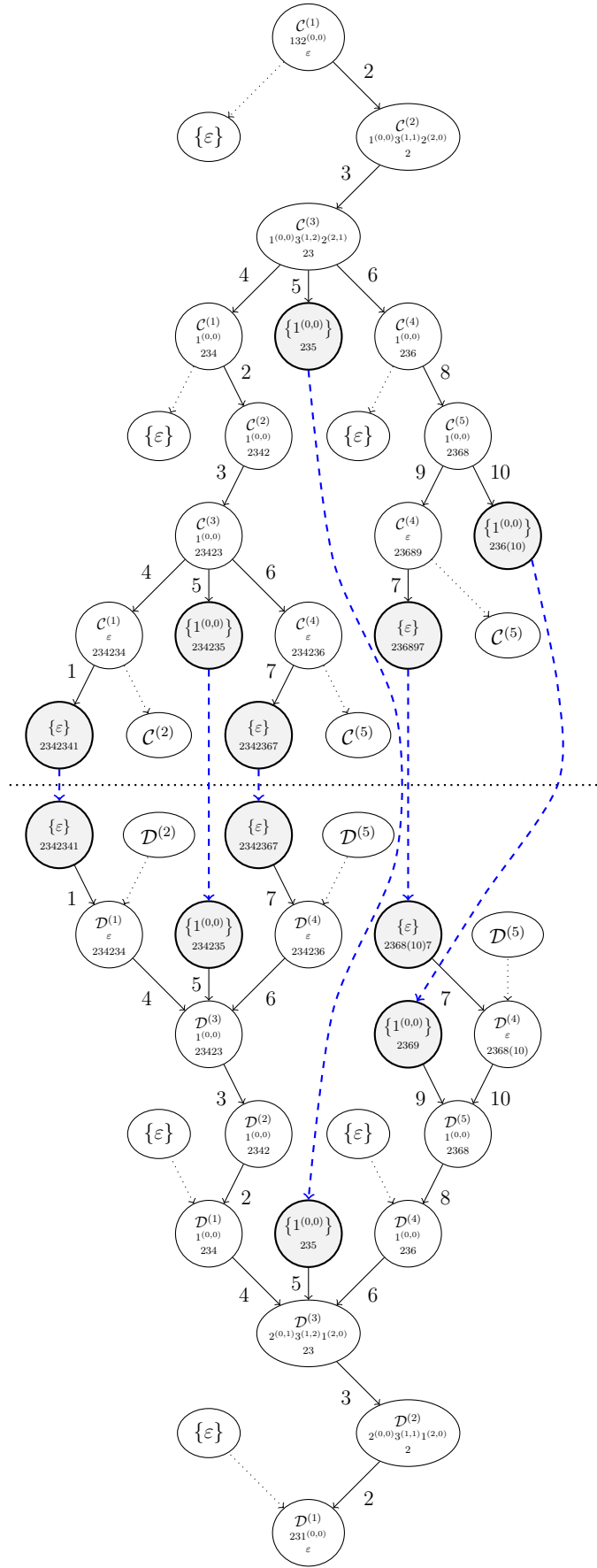


Figure 5.3: The parallel map of 132 for the specifications found for the permutation classes $\text{Av}(231, 321)$ and $\text{Av}(132, 312)$ by TileScope.

Any gridded permutation in $\text{Grid}(\mathcal{T}_1)$ is a permutation from $\text{Av}(123)$ spread across the first two cells in all possible ways. They all map to their underlying permutation within the single cell of \mathcal{T}_2 . The elements that map to the same element can be thought of as the underlying permutation with a vertical line drawn between its points in all possible ways as demonstrated in Table 5.3. We can use the index of the line that splits them and $<$ as a binary relation and that makes the set of elements that map to the same element a strictly totally ordered finite set for all such sets within $\text{Grid}(\mathcal{T}_1)$.

$$\begin{array}{c|c} \pi_1\pi_2\cdots\pi_n & \pi_1\pi_2\cdots\pi_n^{(1,0)} \\ \pi_1|\pi_2\cdots\pi_n & \pi_1^{(0,0)}\pi_2\cdots\pi_n^{(1,0)} \\ \vdots & \vdots \\ \pi_1\pi_2\cdots\pi_n| & \pi_1\pi_2\cdots\pi_n^{(0,0)} \end{array}$$

Table 5.3: The map for Fusion interpreted with vertical splitting lines.

Let $c = 1^{(0,0)}32^{(1,0)} \in \text{Grid}(\mathcal{T}_1)$, then

$$\bar{c} = \{132^{(0,0)}, 13^{(0,0)}2^{(1,0)}, 1^{(0,0)}32^{(1,0)}, 132^{(1,0)}\}$$

and

$$\text{sort}(\bar{c}) = (132^{(1,0)}, 1^{(0,0)}32^{(1,0)}, 13^{(0,0)}2^{(1,0)}, 132^{(0,0)}) .$$

The bijection from all elements in \bar{c} are shown in Table 5.4. These bijections would then be further extended by the edge label when used in a parallel bijection.

c	$\phi(c)$	i	$\hat{\phi}(c)$	$\hat{\phi}^{-1}(\phi(c), i)$
$132^{(1,0)}$	$132^{(0,0)}$	0	$(132^{(0,0)}, 0)$	$132^{(1,0)}$
$1^{(0,0)}32^{(1,0)}$	$132^{(0,0)}$	1	$(132^{(0,0)}, 1)$	$1^{(0,0)}32^{(1,0)}$
$13^{(0,0)}2^{(1,0)}$	$132^{(0,0)}$	2	$(132^{(0,0)}, 2)$	$13^{(0,0)}2^{(1,0)}$
$132^{(0,0)}$	$132^{(0,0)}$	3	$(132^{(0,0)}, 3)$	$132^{(0,0)}$

Table 5.4: The indexed map for a fusion rule.

Suppose we have two indexable rules that are nonbijective from left hand sides \mathcal{C} and \mathcal{D} . In order for the two of them to be equivalent, they additionally need to satisfy an existence of a bijection between $\{\bar{c} \mid c \in \mathcal{C}\}$ and $\{\bar{d} \mid d \in \mathcal{D}\}$ that preserves cardinality.

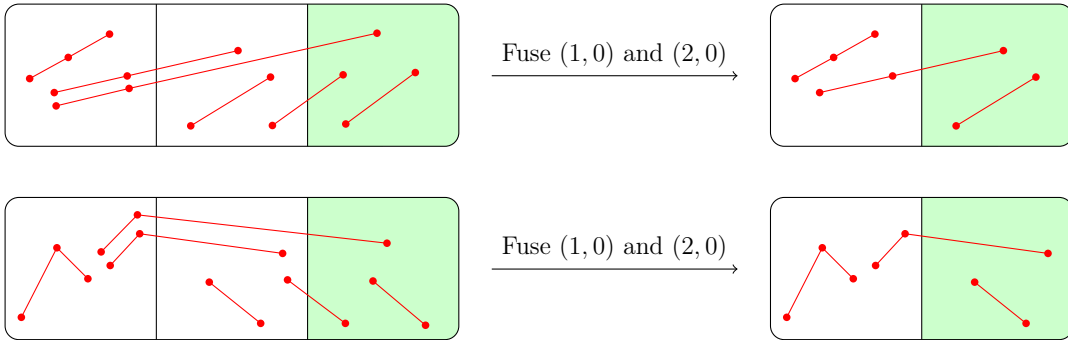


Figure 5.4: Two equivalent fusion rules.

The two fusion rules shown in Figure 5.4 are an example of two equivalent rules that are nonbijective. Combinatorial Exploration found the generating function of both right hand sides to be

$$T(x, y) = \frac{C(x)}{1 - xyC(x)} \quad \text{where } C(x) = \frac{1 - \sqrt{1 - 4x}}{2x}$$

is the Catalan generating function. Let T_1 and T_2 be the generating functions for the upper and lower left hand sides respectively. The assumptions are identical in both so we have

$$T_1(x, y) = \frac{yT(x, y) - T(x, 1)}{y - 1} = T_2(x, y).$$

Since the generating functions are identical for both right hand sides, then there are equally many gridded permutations of size n with k points in cell $(1, 0)$. For each such gridded permutation in either class there are a total of $k + 1$ corresponding gridded permutations in their respective left hand sides that would map to it. Therefore a bijection does exist between the sets of elements that map to the same element in the two left hand sides. The rules are equivalent because the generating functions for the left hand sides are the same (given that the right hand sides are) and this bijection exists.

5.4 The parallel bijection algorithm

The parallel map is implemented with a parse tree where the original object is broken down into atoms for the domain's specification. While doing so we follow along in the codomain's specification and then build back up to the root in the codomain's specifications. The matching order is used to pair the correct children.

The algorithm can be seen in Algorithm 5.1. It uses the specifications as objects and assumes they have access to their root classes and rule look-ups \mathcal{R}_1 and \mathcal{R}_2 , respectively, from their left hand side classes. Any rule object knows its classes and can map both forward (ϕ) and backward (ϕ^{-1}). The matching order will be a function that both filters out any empty classes and sorts a tuple of broken down objects and their corresponding classes by the matched order between the classes. In the case where a forward map only maps to a subset of its right hand side classes (e.g., disjoint union) then the objects for the classes that are not being used is set to `null`.

Suppose we have an object of size n and two parallel specifications where the one with more rules has r rules. We will always reach an atom before exhausting every rule and thus reduce the object by at least one at that point. The worst time complexity of the map is therefore $\mathcal{O}(nr)$.

Algorithm 5.1 The parallel bijection algorithm.

Input: Two parallel specifications $\check{\mathcal{C}}$ and $\check{\mathcal{D}}$, their matching order M and an object x of the root of $\check{\mathcal{C}}$.

Output: The corresponding object in the codomain.

```

1: procedure MAP( $o, r_1, r_2$ )
2:   if lhs( $r_1$ ) is a non-empty terminal class then
3:     return minimum sized object of lhs( $r_2$ )
4:   end if
5:   if  $r_2$  is an equivalence rule then
6:     if  $r_1$  is not an equivalence rule then
7:       return  $\phi_{r_2}^{-1}(\text{MAP}(o, r_1, \mathcal{R}_2(\text{rhs}(r_2))))$ 
8:     end if
9:     return  $\phi_{r_2}^{-1}(\text{MAP}(\phi_{r_1}(o), \mathcal{R}_1(\text{rhs}(r_1), \mathcal{R}_2(\text{rhs}(r_2))))$ 
10:  end if
11:  if  $r_1$  is an equivalence rule then
12:    return MAP( $\phi_{r_1}(o), \mathcal{R}_1(\text{rhs}(r_1), r_2)$ )
13:  end if
14:  ( $p, c$ )  $\leftarrow M(\phi_{r_1}(o), \text{rhs}(r_1))$  ▷ list of objects and list of classes
15:   $i \leftarrow 1$ 
16:   $c' \leftarrow []$ 
17:  for  $j \leftarrow 1$ , number of right hand side classes in  $r_2$  do
18:    if rhs( $r_2, j$ ) =  $\emptyset$  then ▷ the  $j^{\text{th}}$  rhs class
19:       $c' \leftarrow c' + [\text{null}]$ 
20:    else
21:      if  $p_i = \text{null}$  then
22:         $c' \leftarrow c' + [\text{null}]$ 
23:      else
24:         $c' \leftarrow c' + [\text{MAP}(p_i, \mathcal{R}_1(c_i), \mathcal{R}_2(\text{rhs}(r_2, j)))]$ 
25:      end if
26:       $i \leftarrow i + 1$ 
27:    end if
28:  end for
29:  return  $\phi_{r_2}^{-1}(c')$ 
30: end procedure
31: return MAP( $x, \mathcal{R}(\text{root}(\check{\mathcal{C}})), \mathcal{R}(\text{root}(\check{\mathcal{D}}))$ )

```

Chapter 6

Bijection search

We know how to check if two specifications are parallel and if so, how to map objects between them but now we turn our attention to searching for parallel specifications. This is done by using Combinatorial Exploration's search for specification with further steps taken once it has finished.

6.1 The bijection search algorithm

When finding a bijection between classes we expand universes¹ of rules for both until they contain at least one specification each. We can expand further but usually universes will contain multiple specifications as soon as they contain one. Once that is done we gather the information we need in our search and store in better suited data structures.

The universes will contain rules in the form of equivalence labels $(p, (c_1, c_2, \dots, c_n))$ where the p is the equivalence label of the parent and c_i is an equivalence label of child i . Any equivalence label p can have multiple rules

$$\begin{aligned} &\left(p, \left(c_1^{(1)}, c_2^{(1)}, \dots, c_{n_i}^{(1)}\right)\right) \\ &\left(p, \left(c_1^{(2)}, c_2^{(2)}, \dots, c_{n_2}^{(2)}\right)\right) \\ &\vdots \\ &\left(p, \left(c_1^{(k)}, c_2^{(k)}, \dots, c_{n_k}^{(k)}\right)\right) \end{aligned}$$

in the universe. Depending on which one is looked at, there may be some equivalence rules needed to reach the children of that rule. That means we do not have to worry about equivalence rules in our search.

The algorithm works similarly to the parallel algorithm in Section 4.5. The main difference is that we ignore equivalence rules and for any parent pair p_1 and p_2 from each universe, we gather every possible matching of their children. This is done since p_1 and p_2 can match with some specific children at one point while p_1 could match another equivalence label in the other universe later using another rule but our specifications need a single rule for any parent. The complexity of finding these matches is the same as for the parallel algorithm but now the number of rules is significantly larger. The

¹Strategies used to expand universes are defined in packs prior to expanding.

complexity of the specification search dominates our algorithm so the complexity of the search itself is the same as that of the specification search.

Once we have found all matched equivalence labels and assuming the roots have matched we start looking within the collection of matched labels for specifications that are consistent in the sense that only a single rule contains any equivalence label as its left hand side. This is done with backtracking starting from the two roots. This stage will guard against any false positives from recursion as shown in Figure 6.1.

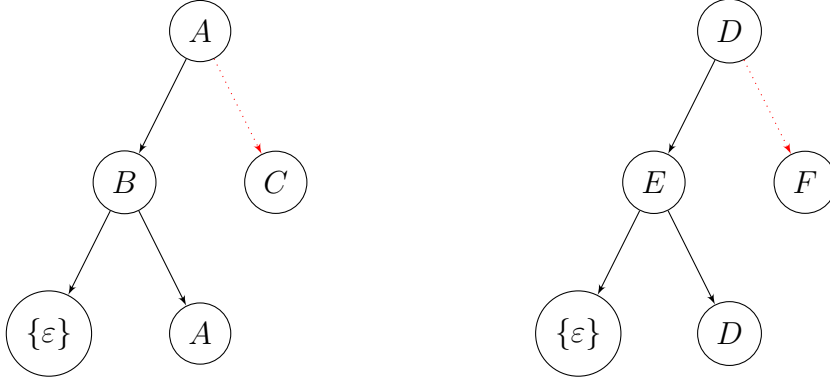


Figure 6.1: Labels B and E falsely matched because of a recursion to their respective parents that then fails to match the next children pair. In this case A and D might not be equinumerous.

If all of those stages are successful we pass the rules (in terms of equivalence labels) of the specifications found to a specification extractor that handles creating any required path of equivalence rules. A minimal working example on how to use this search can be seen in Appendix A.

6.2 Fusion and assumptions

When dealing with universes constructed from packs involving fusion or assumption strategies we need some slight alterations to the base algorithms.

As discussed, we need to extend the map for fusion rules to include indices as the base map is not bijective. On top of that we can have matching rules where one has the fused assumption on the left while the other on the right in which case we reverse the indices in the second specification's backward map. If there are assumptions on both sides we track the assumptions down with breadth first search until we find the parents and decide whether to reverse the indices based on where they end up in the parents. This is done after the specifications have been constructed.

In the search itself, fusion and assumptions rules may be within the same equivalence label. That is a systemic choice of Combinatorial Exploration to avoid tautologies. This will require us to extract the rules within each pair of matching equivalence labels given their children and grandparents and validate if the paths are in fact valid.

In the case of multiple assumptions, their correspondence between tilings of the two specifications can matter. This can not be done for the class they appear in but instead we label assumptions as soon as they appear and make sure any atoms containing assumptions have the same label.

Chapter 7

Results

The search method we have developed and the resulting bijections work between different domains. The problem, however, is the lack of existing implementations of domains using Combinatorial Exploration. With Combinatorial Exploration being a product of a research group that is mainly interested in permutation patterns, unsurprisingly, the most supported domain is permutations with TileScope. As a consequence, we will mostly demonstrate bijections found between permutation classes.

7.1 Cross-domain successes

There is a simple example implementation of words provided in the repository for Combinatorial Exploration. A word over an alphabet Σ is any sequence of elements from Σ . A pattern in a word is another word that occurs as a consecutive subsequence. A class can be described as a triple $\mathcal{W} = (p, \Sigma, A)$, where p is a required prefix, Σ is the alphabet and A is the set of patterns to avoid. For example $(\varepsilon, \{0, 1\}, \{11\})$ is the set of binary strings that avoid consecutive 1's, that is

$$(\varepsilon, \{0, 1\}, \{11\}) = \{\varepsilon, 0, 1, 00, 01, 10, 000, 001, 010, 100, 101, \dots\}.$$

The bijection search will easily find bijections between sets such as $(\varepsilon, \{0, 1\}, \{11\})$ and $(\varepsilon, \{a, b\}, \{bb\})$, that are identical given a bijection of the letters of the alphabet. The same goes for the trivial sets $(\varepsilon, \{0\}, \emptyset)$ and $\text{Av}(12)$, where there is only a single element of each size. The implementation for words has few strategies and limited effort was allocated to finding bijections between words and permutation classes. We will demonstrate two such bijections in greater detail but there is a caveat that needs to be addressed.

Consider the class $\mathcal{W} = (\varepsilon, \{a, b\}, \emptyset)$. For any fixed size n , there are 2^n unique words as we have a choice between two letters for each position in any sequence. This produces the counting sequence $1, 2, 4, 8, 16, 32, 64, \dots$ while the permutation class we want to match with, $\text{Av}(231, 321)$, has the counting sequence $1, 1, 2, 4, 8, 16, 32, \dots$ which is off by one. We could use $\text{Av}_{\geq 1}(231, 312)$ instead, which shares this sequence with the words if we start from $n = 1$, but that brings another problem. Words of size n would map to permutations of size $n + 1$ while parallel specifications require atoms to match. What we can do instead is place the topmost point of $\text{Av}_{\geq 1}(231, 312)$ and factor it out. That leaves us with a tiling root \mathcal{T} where

$$\begin{aligned}\text{Av}(231, 312) &\cong \{\varepsilon\} \sqcup \text{Av}_{\geq 1}(231, 312), \\ \text{Av}_{\geq 1}(231, 312) &\cong \{\bullet\} \times \text{Grid}(\mathcal{T}).\end{aligned}$$

The tiling in question here is

$$\mathcal{T} = ((2, 1), \{2^{(0,0)}1^{(1,0)}, 21^{(1,0)}, 231^{(0,0)}, 321^{(0,0)}\}, \emptyset),$$

shown in Figure 7.1. Now we can find and construct bijections between \mathcal{W} and $\text{Grid}(\mathcal{T})$.

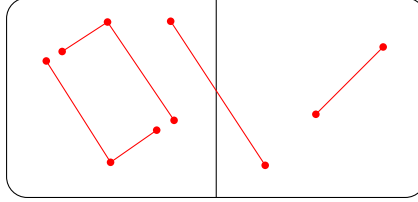


Figure 7.1: The tiling used instead of $\text{Av}(231, 312)$ to deal with the sequence offset of binary strings.

It will map the words to gridded permutations in $\mathcal{G}^{(2,1)}$ instead of $\mathcal{G}^{(1,1)}$ but there is a trivial bijection from these gridded permutations to permutations in $\text{Av}_{\geq 1}(231, 312)$.

The other bijection found was between $\mathcal{W} = (\varepsilon, \{0, 1\}, \{11\})$ and $\text{Av}(231, 312, 321)$. These are counted by the Fibonacci numbers but here we have the same issue as before, with one sequence being off by one. We remedy this in the same way as before, leaving us with a tiling root that can be seen in Figure 7.2. This tiling can have at most a single point in cell $(1, 0)$ and if it includes one, it must be greater than all the points in cell $(0, 0)$. We can map any such gridded permutation to a permutation with a trivial bijection. Given a gridded permutation of size n that contains no point in cell $(1, 0)$ we append $n + 1$ to the underlying permutation. If it contains a point in $(1, 0)$ we place $n + 1$ immediately prior to the last element.

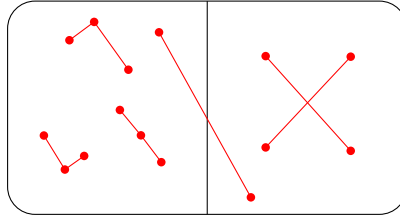


Figure 7.2: The tiling used instead of $\text{Av}(231, 312, 321)$ to deal with the sequence offset of binary strings with no consecutive 1's.

Figure 7.3 and Figure 7.4 show the two parallel specifications that were found. Note that there is an empty class in the specification for words. Table 7.1 shows the inputs and outputs up to size 3 for the bijection constructed and corresponding permutation with the mapping described above.

7.2 Permutation classes

The experiments we carried out can be separated into known Wilf classes and *experimental classes*.

The experiments for known Wilf classes includes bijections found for all the permutation classes shown to be Wilf classes with bijections in Simion and Schmidt [3], excluding symmetries. These classes are split over the next five subsections.

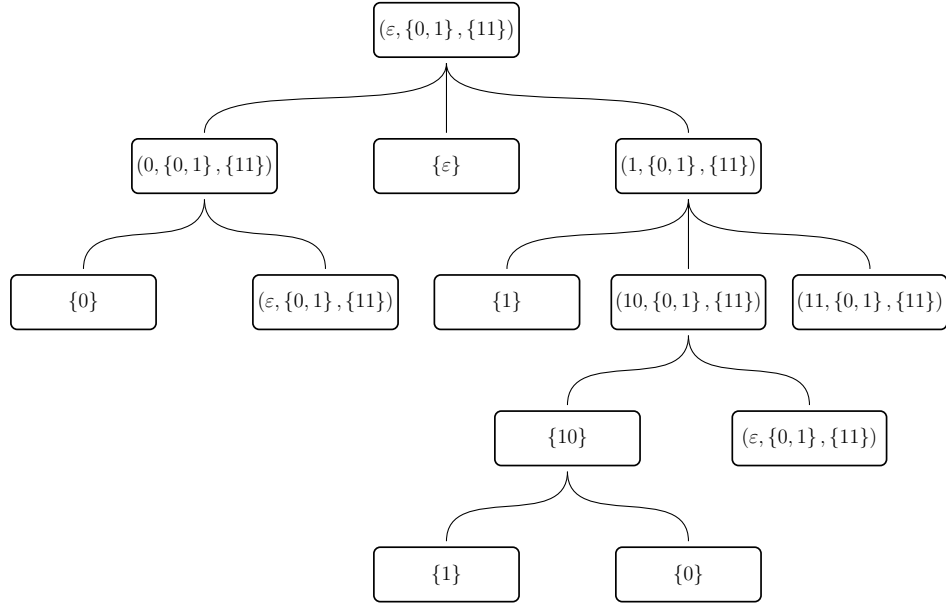


Figure 7.3: The tree representation of the formal grammar of the specification found for binary strings avoiding consecutive 1's.

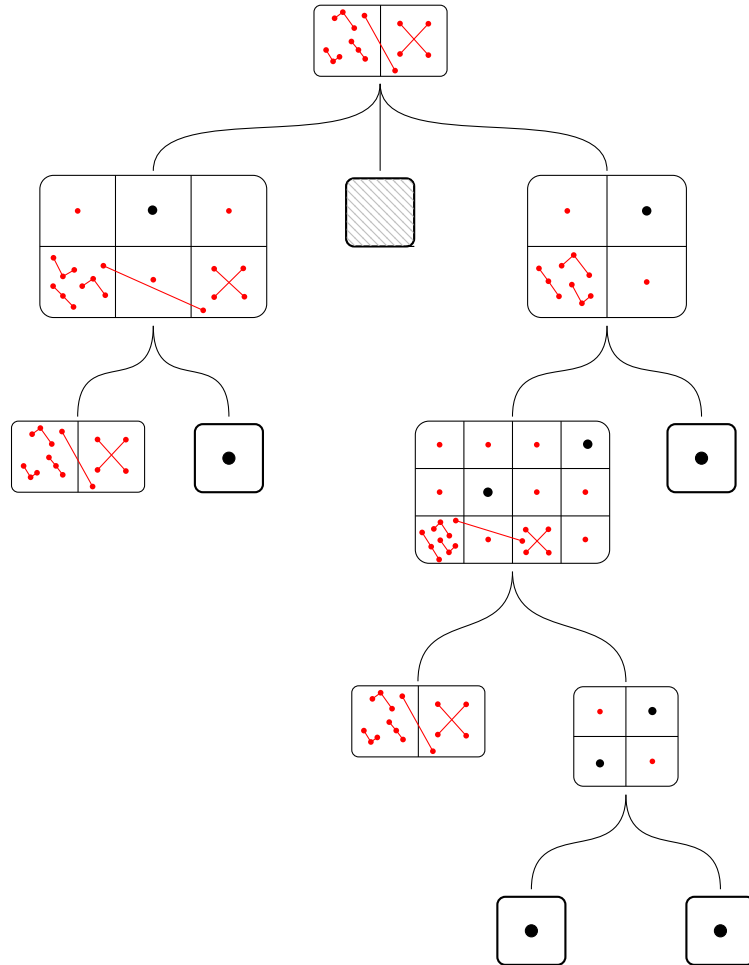


Figure 7.4: The tree representation of the formal grammar of the specification found for $\text{Av}_{\geq 1}(231, 312, 321)$ after placing and factoring out the top point.

Word	Gridded permutation	Permutation
ε	ε	1
0	$1^{(0,0)}$	12
1	$1^{(1,0)}$	21
00	$12^{(0,0)}$	123
01	$21^{(0,0)}$	213
10	$1^{(0,0)}2^{(1,0)}$	132
000	$123^{(0,0)}$	1234
001	$213^{(0,0)}$	2134
010	$132^{(0,0)}$	1324
100	$12^{(0,0)}3^{(1,0)}$	1243
101	$21^{(0,0)}3^{(1,0)}$	2143

Table 7.1: The inputs and outputs up to size 3 of an automated bijection between binary strings avoiding consecutive 1's and $\text{Grid}(\mathcal{T})$, where \mathcal{T} is the tiling for $\text{Av}_{\geq 1}(231, 312, 321)$ after placing and factoring out the top point. The corresponding permutation is also shown.

An experimental class is a collection of permutation classes that share the first 12 terms in their counting sequence. For the experimental classes we focused on classes avoiding patterns of size 4. We started with those avoiding eleven patterns and worked our way downwards. Symmetries and classes with a *regular insertion encoding*, defined in Albert, Linton, and Ruškuc [16], were omitted.

7.2.1 Avoiding one pattern of size 3

The parallel specification we found for $\text{Av}(123)$ and $\text{Av}(132)$ have an identical tree structure and even share strategies. They are

$$\check{\mathcal{C}} = \left(\begin{pmatrix} \mathcal{C}^{(1)} \\ \mathcal{C}^{(2)} \\ \mathcal{C}^{(3)} \\ \mathcal{C}^{(4)} \\ \mathcal{C}^{(5)} \\ \mathcal{C}^{(6)} \\ \mathcal{C}^{(7)} \end{pmatrix}, \begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ o_5 \\ o_6 \\ o_7 \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & \mathcal{C}^{(2)} & \emptyset \\ \mathcal{C}^{(3)} & \{1\} & \emptyset \\ \mathcal{C}^{(4)} & \emptyset & \emptyset \\ \{\varepsilon\} & \mathcal{C}^{(5)} & \mathcal{C}^{(6)} \\ \mathcal{C}^{(7)} & \{1\} & \emptyset \\ \mathcal{C}^{(4)} & \mathcal{C}^{(8)} & \emptyset \\ \mathcal{C}^{(4)} & \emptyset & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \\ 1 \\ 3 \\ 2 \\ 2 \\ 1 \end{pmatrix} \right)$$

and

$$\check{\mathcal{D}} = \left(\begin{pmatrix} \mathcal{D}^{(1)} \\ \mathcal{D}^{(2)} \\ \mathcal{D}^{(3)} \\ \mathcal{D}^{(4)} \\ \mathcal{D}^{(5)} \\ \mathcal{D}^{(6)} \\ \mathcal{D}^{(7)} \end{pmatrix}, \begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ o_5 \\ o_6 \\ o_7 \end{pmatrix}, \begin{pmatrix} \{\varepsilon\} & \mathcal{D}^{(2)} & \emptyset \\ \mathcal{D}^{(3)} & \{1\} & \emptyset \\ \mathcal{D}^{(4)} & \emptyset & \emptyset \\ \{\varepsilon\} & \mathcal{D}^{(5)} & \mathcal{D}^{(6)} \\ \mathcal{D}^{(7)} & \{1\} & \emptyset \\ \mathcal{D}^{(4)} & \mathcal{D}^{(8)} & \emptyset \\ \mathcal{D}^{(4)} & \emptyset & \emptyset \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \\ 1 \\ 3 \\ 2 \\ 2 \\ 1 \end{pmatrix} \right)$$

where classes are listed in Table 7.2 in matched pairs and the constructor's rules can be seen in Table 7.3.

Tilings in the specification for $\text{Av}(123)$	Tilings in the specification for $\text{Av}(132)$
$\mathcal{C}^{(1)}$ 	$\mathcal{D}^{(1)}$
$\mathcal{C}^{(2)}$ 	$\mathcal{D}^{(2)}$
$\mathcal{C}^{(3)}$ 	$\mathcal{D}^{(3)}$
$\mathcal{C}^{(4)}$ 	$\mathcal{D}^{(4)}$
$\mathcal{C}^{(5)}$ 	$\mathcal{D}^{(5)}$
$\mathcal{C}^{(6)}$ 	$\mathcal{D}^{(6)}$
$\mathcal{C}^{(7)}$ 	$\mathcal{D}^{(7)}$
$\mathcal{C}^{(8)}$ 	$\mathcal{D}^{(8)}$

Table 7.2: The classes and their matching for the parallel specifications of $\text{Av}(123)$ and $\text{Av}(132)$.

The inputs and outputs of size 4 for the bijection are shown in Table 7.4 and it seems to agree with the Simion and Schmidt bijection in [3] which we leave as an open problem in Conjecture 8.1.

Constructor	Rule
o_1	Place bottommost point (+)
o_2	Factor (\times)
o_3	Add assumption in cell (1, 0)
o_4	Place bottommost in row (+)
o_5	Factor (\times)
o_6	Factor (\times)
o_7	Fuse columns 1 and 2

Table 7.3: Rules for the parallel specifications of $\text{Av}(123)$ and $\text{Av}(132)$.

$\text{Av}_4(123)$	$\text{Av}_4(132)$
4321	4321
3421	3421
3241	3241
4231	4231
2431	2341
3214	3214
4213	4213
2413	2314
4312	4312
3412	3412
2143	2134
3142	3124
4132	4123
1432	1234

Table 7.4: The inputs and outputs of size 4 for the bijection between 123 and 132 avoiding permutations.

7.2.2 Avoiding two patterns of size 3

The only Wilf class with more than one permutation class is the class consisting of $\text{Av}(123, 132)$, $\text{Av}(132, 213)$ and $\text{Av}(132, 231)$. The connections created by bijections found can be seen in Figure 7.5.

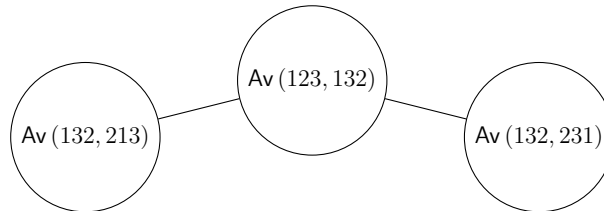


Figure 7.5: The connections created by bijections found for classes avoiding two patterns of size 3.

7.2.3 Avoiding three patterns of size 3

The connections created by bijections found for the only Wilf class (containing more than one permutation class) can be seen in Figure 7.6.

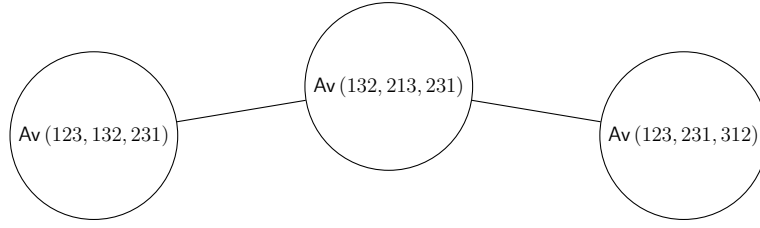


Figure 7.6: The connections created by bijections found for classes avoiding three patterns of size 3.

7.2.4 Avoiding one pattern of size 3 and one of size 4

There are two Wilf classes with more than one permutation class. The first consists of $\text{Av}(132, 4312)$ and $\text{Av}(132, 4231)$ which we found a bijection between and the connections created by bijections found for the second can be seen in Figure 7.7.

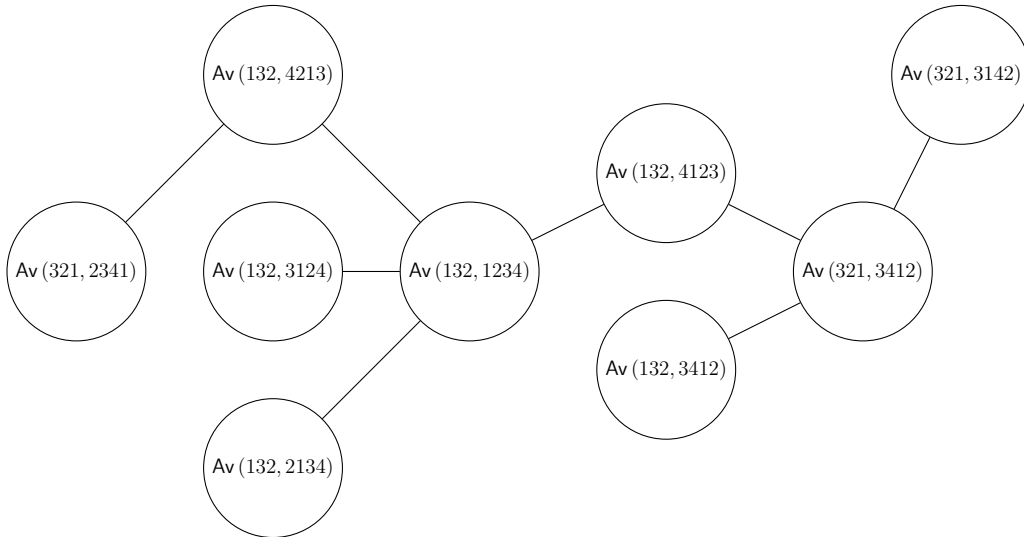


Figure 7.7: The connections created by bijections found for the second Wilf class of permutations avoiding one pattern of size 3 and one of size 4.

7.2.5 Avoiding one pattern of size 4

There are two Wilf classes with more than one permutation class. We did not manage to find bijections between $\text{Av}(1342)$ and $\text{Av}(2413)$ which make up one of those Wilf classes. That is because TileScope requires another technique¹ to enumerate them which is not compatible with our bijection searcher. The other consists of $\text{Av}(1234)$, $\text{Av}(1243)$, $\text{Av}(1432)$ and $\text{Av}(2143)$. TileScope lacks the strategies to solve $\text{Av}(2143)$ so that was out of reach while the connections created by bijections found for the others are shown in Figure 7.8.

¹The technique is called the *table method* which allows some rules in the universe to be used backwards.

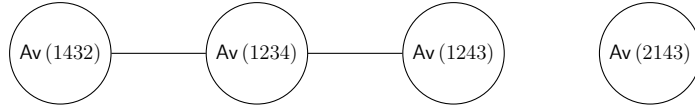


Figure 7.8: The connections created by bijections found for the Wilf class of $\text{Av}(1234)$, $\text{Av}(1243)$, $\text{Av}(1432)$ and $\text{Av}(2143)$.

7.2.6 Experimental classes

The experimental classes we ran our bijection searcher for can be seen in Appendix B. They are grouped by Roman numerals and figures contain their list indices.

The experiments were carried out by comparing the first class against all others, then the second to all but the first and so on. A disjoint-set data structure was used to keep track of confirmed equivalences. As a consequence, the experiments favored lower numbers to have higher degrees. The setup for the experiments can be seen in Appendix C.

Every experimental class was confirmed to be a Wilf class with the exception of experimental class III where we failed to find a bijection between

$$\text{Av}(1243, 1342, 1423, 1432, 2143, 2413, 3142, 3412, 4231)$$

and any other class. We leave that as an open problem. The connections created by bijections found for each experimental class can be seen in Figures D.1-D.19 in Appendix 8. Experimental classes with only two permutation classes are omitted since all of them were successfully shown to be Wilf classes and no information is contained in their figure.

Chapter 8

Conclusion

8.1 Future work

For experimental classes we considered bases with eleven, ten or nine size 4 patterns. Combinatorial Exploration can solve them quickly and all the experiments were performed on a local machine. A dedicated machine is needed to continue this experiment for classes avoiding eight patterns or less of size 4.

Many ideas on improvements and applications arose while working on this thesis and some could not be explored further due to time restrictions. We will go over some of the ideas that did not come to fruition. This will mostly be conveyed from the perspective of permutations but could be generalized for other domains.

Further analysis of the resulting bijections could be insightful. We could compare them to previously known bijections, track permutation statistics preserved or even attempt to visualize them. Matched non-root tilings are equinumerous and could inspire more rules about tilings.

Instead of searching for specifications individually and then trying to find parallel specifications from the extended universe of those searches, it would be preferable to develop our own search method that is intended for finding parallel specifications. This would open the doors for parallel specification related heuristics, where the expansion of a class with a strategy could be evaluated against the other universe. We could even look into some domain specific machine learning related heuristics. This would allow us to find bijections without having specifications where equal classes in each universe can be paired without expansion where the identity map would be applied.

We could produce conjectures on equivalent tilings if we find a partial match. Suppose we know the root classes are isomorphic and at some point, in the trees, we can match all but one children pair. This pair could be isomorphic but we lack the strategies to match them. These would require humans to look at and try to prove but in return, could help formalize some theory about tiling equivalences, which would help find more specifications and bijections.

8.2 Open problems

As mentioned earlier, the bijection found between $\text{Av}(123)$ and $\text{Av}(132)$ seems to agree with the Simion and Schmidt bijection in [3]. This has been confirmed for sizes $n \leq 10$ but no effort went into finding a proof and we leave it as a conjecture.

Conjecture 8.1. The parallel bijection for the specifications in subsection 7.2.1 agrees with the Simion and Schmidt bijection in [3].

Another open problem is to use our framework to find a bijection between

$$\text{Av}(1243, 1342, 1423, 1432, 2143, 2413, 3142, 3412, 4231)$$

and any of the following classes.

- $\text{Av}(1243, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4231)$
- $\text{Av}(1243, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4231)$
- $\text{Av}(1243, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4231)$
- $\text{Av}(1243, 1342, 2143, 2413, 2431, 3142, 3412, 4132, 4231)$
- $\text{Av}(1324, 1342, 2143, 2341, 2413, 2431, 3142, 3241, 3412)$

Multiple sets of strategies were attempted which all failed to find a bijection. It remains the only class in an experimental class that we tried but did not find a bijection for.

8.3 Concluding remarks

We believe this thesis presents the first ever fully automated way of constructing bijections between combinatorial classes. It is built on top of the specification search by Bean [6]. The thesis contains both the theoretical and algorithmic foundations to find and create such a bijection and is accompanied by open source implementations.

Unlike the translation method in Wood and Zeilberger [7], our automated bijection does not require any previous mathematical work, e.g., algebraic proof. It only depends on sufficient implementation of strategies for domains.

As we discussed in Section 8.1, the search is done from a universe that is expanded with a completely different goal in mind. We believe this to be a limiting factor of finding bijections. In contrast, its strength is finding a bijection without any supplied information, given that a domain has adequate strategies.

A plethora of articles have been written about bijections between sets. This process has now been entirely automated. Nothing will ever replace the art of a human constructed bijection or the structural insight they might provide, nor do we intend to. However, combinatorial classes can become immensely complex and tedious to work with by hand. Others might be of limited interest. The automated bijection can replace the human mind in those cases. There are after all infinite bases for which we can define permutation classes. It could provide some insight into how one might go about constructing a bijection by hand.

To demonstrate the potential of our automated bijections we have connected numerous permutation classes with bijections, paying homage to Simion and Schmidt [3]. Our hope is that this work will encourage others to extend Combinatorial Exploration into more domains and continue this work, or inspire different approaches to fully automated bijections.

Bibliography

- [1] P. MacMahon, *Combinatory Analysis*. London: Cambridge University Press, 1916.
- [2] D. E. Knuth, *The Art of Computer Programming. Vol. 1: Fundamental Algorithms*. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont, 1969.
- [3] R. Simion and F. Schmidt, “Restricted permutations”, *European Journal of Combinatorics*, vol. 6, pp. 383–406, 1985.
- [4] D. Zeilberger, “Enumeration schemes and, more importantly, their automatic generation”, *Annals of Combinatorics*, vol. 2, pp. 185–195, 1998.
- [5] V. Vatter, “Enumeration schemes for restricted permutations”, *Combinatorics, Probability and Computing*, vol. 17, no. 1, pp. 137–159, 2008.
- [6] C. Bean, “Finding structure in permutation patterns”, Ph.D. dissertation, Reykjavik University, 2018. [Online]. Available: <https://opinvindi.is/bitstream/handle/20.500.11815/1184/phd-bean-2018.pdf>.
- [7] P. M. Wood and D. Zeilberger, “A translation method for finding combinatorial bijections”, *Annals of Combinatorics*, vol. 13, no. 3, pp. 383–402, 2009.
- [8] H. Wilf, *generatingfunctionology*, second. Academic Press, 1994.
- [9] P. Flajolet and R. Sedgewick, *Analytic combinatorics*. Cambridge University Press, 2009.
- [10] M. Albert, M. Atkinson, M. Bouvel, N. Ruškuc, and V. Vatter, “Geometric grid classes of permutations”, *Transactions of the American Mathematical Society*, vol. 365, no. 11, pp. 5859–5881, 2013.
- [11] É. Nadeau, C. Bean, H. Ulfarsson, J. S. Eliasson, and J. Pantone, *Permutatriangle/comb_spec_searcher*, version 4.0.0, Jun. 2021. DOI: 10.5281/zenodo.4946832. [Online]. Available: <https://doi.org/10.5281/zenodo.4946832>.
- [12] T. K. Magnusson, É. Nadeau, C. Bean, H. Ulfarsson, J. S. Eliasson, and J. Pantone, *Permutatriangle/tilings*, version 3.0.0, Jun. 2021. DOI: 10.5281/zenodo.4948344. [Online]. Available: <https://doi.org/10.5281/zenodo.4948344>.
- [13] R. P. Ardal, T. K. Magnusson, É. Nadeau, B. J. Kristinsson, B. A. Gudmundsson, C. Bean, H. Ulfarsson, J. S. Eliasson, M. Tannock, A. B. Bjarnason, J. Pantone, and A. B. Arnarson, *Permutatriangle/permuta*, version 2.0.4, Apr. 2021. DOI: 10.5281/zenodo.4725759. [Online]. Available: <https://doi.org/10.5281/zenodo.4725759>.
- [14] S. M. Johnson, “Generation of permutations by adjacent transposition”, *Mathematics of Computation*, vol. 17, no. 83, pp. 282–285, 1963.

- [15] E. Babson and J. West, “The permutations $123\ p\ 4\ \dots\ pm$ and $321\ p\ 4\ \dots\ pm$ are wilf-equivalent”, *Graphs and Combinatorics*, vol. 16, no. 4, pp. 373–380, 2000.
- [16] M. H. Albert, S. Linton, and N. Ruškuc, “The insertion encoding of permutations”, *The Electronic Journal of Combinatorics*, vol. 12, no. 1, R47, 2005.

Appendix A

Minimal working example

Listing A.1 shows a minimal example of how one can search for bijection and then, use its map and inverse map. In this example the two symmetric classes $Av(231)$ and $Av(132)$ are used.

```
1 from comb_spec_searcher.bijection import ParallelSpecFinder
2 from comb_spec_searcher.isomorphism import Bijection
3 from tilings.strategies import BasicVerificationStrategy
4 from tilings.tilescope import TileScope, TileScopePack
5
6 pack = TileScopePack.row_and_col_placements(row_only=True)
7 pack = pack.add_verification(BasicVerificationStrategy(), replace=True)
8 searcher1 = TileScope("231", pack)
9 searcher2 = TileScope("132", pack)
10
11 specs = ParallelSpecFinder(searcher1, searcher2).find()
12 bijection = Bijection.construct(*specs)
13 for n in range(6):
14     for gp in bijection.domain.generate_objects_of_size(n):
15         print(f"{gp} -> {bijection.map(gp)}")
```

Listing A.1: A minimal example to find a bijection.

Appendix B

Experimental classes

B.1 Experimental class I

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 3142, 3412)
3. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 3412, 4132)
4. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 3412)
5. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3412, 4132)
6. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412)
7. Av (1243, 1324, 1342, 1423, 2413, 2431, 3142, 3412, 4132)
8. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412)
9. Av (1243, 1324, 1342, 1432, 2143, 2413, 3142, 3412, 4132)
10. Av (1243, 1324, 1342, 1432, 2143, 2431, 3142, 3412, 4132)
11. Av (1243, 1324, 1342, 1432, 2413, 2431, 3142, 3412, 4132)
12. Av (1243, 1324, 1342, 2143, 2413, 2431, 3142, 3412, 4132)
13. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132)
14. Av (1243, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4132)
15. Av (1243, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132)
16. Av (1243, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132)
17. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412)
18. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132)
19. Av (1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132)
20. Av (1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132)
21. Av (1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4132)

B.2 Experimental class II

1. Av (1243, 1324, 1342, 1423, 2413, 2431, 3142, 4132, 4231)
2. Av (1243, 1324, 1342, 1432, 2413, 2431, 3142, 4132, 4231)
3. Av (1243, 1324, 1342, 2143, 2413, 2431, 3142, 4132, 4231)
4. Av (1243, 1324, 1432, 2143, 2413, 2431, 3142, 4132, 4231)
5. Av (1243, 1342, 1423, 1432, 2413, 2431, 3142, 4132, 4231)
6. Av (1243, 1342, 1423, 2143, 2413, 2431, 3142, 4132, 4231)
7. Av (1243, 1342, 1432, 2143, 2413, 2431, 3142, 4132, 4231)
8. Av (1324, 1342, 1423, 2143, 2413, 2431, 3142, 4132, 4231)
9. Av (1324, 1342, 1432, 2143, 2413, 2431, 3142, 4132, 4231)
10. Av (1324, 1342, 2314, 2341, 2413, 2431, 3142, 3241, 3412)

B.3 Experimental class III

1. Av (1243, 1342, 1423, 1432, 2143, 2413, 3142, 3412, 4231)
2. Av (1243, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4231)
3. Av (1243, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4231)
4. Av (1243, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
5. Av (1243, 1342, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
6. Av (1324, 1342, 2143, 2341, 2413, 2431, 3142, 3241, 3412)

B.4 Experimental class IV

1. Av (1243, 1324, 1342, 1423, 2413, 2431, 3142, 3412, 4231)
2. Av (1243, 1324, 1342, 1432, 2413, 2431, 3142, 3412, 4231)
3. Av (1243, 1324, 1342, 1432, 2431, 3142, 3412, 4132, 4231)
4. Av (1243, 1324, 1342, 2143, 2413, 2431, 3142, 3412, 4231)
5. Av (1243, 1324, 1342, 2413, 2431, 3142, 3412, 4132, 4231)
6. Av (1243, 1342, 1423, 1432, 2413, 2431, 3412, 4132, 4231)
7. Av (1243, 1342, 1423, 2413, 2431, 3142, 3412, 4132, 4231)
8. Av (1243, 1342, 1432, 2143, 2431, 3142, 3412, 4132, 4231)
9. Av (1243, 1342, 1432, 2413, 2431, 3142, 3412, 4132, 4231)

10. Av (1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4231)
11. Av (1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
12. Av (1324, 1342, 1432, 2143, 2431, 3142, 3412, 4132, 4231)
13. Av (1324, 1342, 2143, 2314, 2341, 2413, 2431, 3142, 3412)
14. Av (1324, 1342, 2143, 2314, 2341, 2413, 2431, 3241, 3412)
15. Av (1324, 1342, 2143, 2314, 2413, 2431, 3142, 3241, 3412)
16. Av (1324, 1342, 2143, 2314, 2413, 2431, 3142, 3412, 4231)

B.5 Experimental class V

1. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3412, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2413, 3142, 3412, 4231)
3. Av (1243, 1324, 1342, 1423, 1432, 2413, 3412, 4132, 4231)
4. Av (1243, 1324, 1342, 1423, 1432, 2431, 3412, 4132, 4231)
5. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3412, 4231)
6. Av (1243, 1324, 1342, 1423, 2143, 2413, 3142, 3412, 4231)
7. Av (1243, 1324, 1342, 1423, 2413, 2431, 3412, 4132, 4231)
8. Av (1243, 1324, 1342, 1432, 2143, 2431, 3142, 3412, 4231)
9. Av (1243, 1324, 1342, 1432, 2143, 3142, 3412, 4132, 4231)
10. Av (1243, 1324, 1342, 1432, 2413, 2431, 3412, 4132, 4231)
11. Av (1243, 1324, 1342, 1432, 2413, 3142, 3412, 4132, 4231)
12. Av (1243, 1324, 1342, 2143, 2413, 3142, 3412, 4132, 4231)
13. Av (1243, 1324, 1342, 2143, 2431, 3142, 3412, 4132, 4231)
14. Av (1243, 1324, 1432, 2413, 2431, 3142, 3412, 4132, 4231)
15. Av (1243, 1324, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
16. Av (1243, 1342, 1423, 1432, 2143, 2431, 3412, 4132, 4231)
17. Av (1243, 1342, 1423, 2143, 2413, 2431, 3412, 4132, 4231)
18. Av (1243, 1342, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
19. Av (1243, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
20. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4231)
21. Av (1324, 1342, 1423, 1432, 2143, 2413, 3142, 3412, 4231)

- 22. Av (1324, 1342, 1423, 1432, 2143, 2413, 3412, 4132, 4231)
- 23. Av (1324, 1342, 1423, 1432, 2143, 2431, 3412, 4132, 4231)
- 24. Av (1324, 1342, 1423, 2143, 2413, 2431, 3412, 4132, 4231)
- 25. Av (1324, 1342, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
- 26. Av (1324, 1342, 1432, 2143, 2413, 3142, 3412, 4132, 4231)
- 27. Av (1324, 1342, 2143, 2314, 2341, 2413, 3142, 3412, 4231)

B.6 Experimental class VI

- 1. Av (1324, 1342, 1423, 2413, 2431, 3142, 3412, 4132, 4231)
- 2. Av (1324, 1342, 1432, 2413, 2431, 3142, 3412, 4132, 4231)

B.7 Experimental class VII

- 1. Av (1324, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4231)
- 2. Av (1324, 1342, 1423, 1432, 2413, 2431, 3412, 4132, 4231)

B.8 Experimental class VIII

- 1. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4231)
- 2. Av (1324, 1342, 1423, 1432, 2413, 2431, 3142, 4132, 4231)

B.9 Experimental class IX

- 1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 4231)
- 2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 4132, 4231)
- 3. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 4231)
- 4. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 4132, 4231)
- 5. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 4231)
- 6. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 4231)
- 7. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 4132, 4231)
- 8. Av (1243, 1324, 1342, 1432, 2143, 2431, 3142, 4132, 4231)
- 9. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 4132, 4231)
- 10. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4231)
- 11. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 4132, 4231)

B.10 Experimental class X

1. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 4123)
2. Av (1234, 1243, 1324, 1342, 1423, 2314, 2341, 3124, 4123)
3. Av (1234, 1243, 1342, 1423, 2134, 2314, 2341, 3124, 4123)
4. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 4132)
5. Av (1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124, 4123)
6. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 4132)
7. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 4132)
8. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4132)
9. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4132)

B.11 Experimental class XI

1. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4231)
2. Av (1243, 1342, 1423, 1432, 2143, 2413, 3412, 4132, 4231)
3. Av (1243, 1342, 1432, 2143, 2413, 3142, 3412, 4132, 4231)

B.12 Experimental class XII

1. Av (1243, 1324, 1342, 1423, 2134, 2314, 2341, 3412, 4123)
2. Av (1243, 1342, 1423, 2134, 2314, 2341, 3124, 3412, 4123)

B.13 Experimental class XIII

1. Av (1243, 1324, 1342, 1423, 2143, 2413, 3412, 4132, 4231)
2. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3412, 4231)
3. Av (1243, 1324, 1342, 1432, 2143, 2413, 3142, 3412, 4231)
4. Av (1243, 1324, 1342, 1432, 2143, 2431, 3412, 4132, 4231)
5. Av (1243, 1324, 1342, 2143, 2413, 2431, 3412, 4132, 4231)
6. Av (1243, 1324, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
7. Av (1243, 1324, 1432, 2143, 2413, 2431, 3412, 4132, 4231)

B.14 Experimental class XIV

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2431, 3412, 4132)
2. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3412, 4132)
3. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3412, 4132)
4. Av (1243, 1324, 1432, 2143, 2413, 2431, 3142, 3412, 4132)

B.15 Experimental class XV

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 3412, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2431, 3412, 4231)
3. Av (1243, 1324, 1342, 1423, 2143, 2431, 3412, 4132, 4231)
4. Av (1243, 1324, 1342, 1432, 2143, 2413, 3412, 4132, 4231)

B.16 Experimental class XVI

1. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 4132, 4231)
2. Av (1243, 1324, 1342, 1432, 2143, 2413, 3142, 4132, 4231)

B.17 Experimental class XVII

1. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 3412)
2. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 3124, 3412)
3. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 3412, 4123)
4. Av (1234, 1243, 1324, 1342, 1423, 2134, 2341, 3412, 4123)
5. Av (1234, 1243, 1324, 1342, 1423, 2314, 2341, 3124, 3412)
6. Av (1234, 1243, 1324, 1342, 1423, 2314, 2341, 3412, 4123)
7. Av (1234, 1243, 1324, 1342, 2134, 2314, 2341, 3412, 4123)
8. Av (1234, 1243, 1324, 1342, 2134, 2341, 3124, 3412, 4123)
9. Av (1234, 1243, 1324, 1342, 2314, 2341, 3124, 3412, 4123)
10. Av (1234, 1243, 1342, 1423, 2134, 2314, 2341, 3124, 3412)
11. Av (1234, 1243, 1342, 1423, 2134, 2314, 2341, 3412, 4123)
12. Av (1234, 1243, 1342, 1423, 2314, 2341, 3124, 3412, 4123)
13. Av (1234, 1324, 1342, 1423, 2314, 2341, 3124, 3412, 4123)
14. Av (1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124, 3412)
15. Av (1243, 1324, 1342, 1423, 2314, 2341, 3124, 3412, 4123)

B.18 Experimental class XVIII

1. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142)
3. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 4132)

B.19 Experimental class XIX

1. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3412, 4132, 4231)
3. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4231)
4. Av (1243, 1324, 1342, 1423, 2413, 2431, 3142, 3412, 4132, 4231)
5. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
6. Av (1243, 1324, 1342, 1432, 2143, 2431, 3142, 3412, 4132, 4231)
7. Av (1243, 1324, 1342, 1432, 2413, 2431, 3142, 3412, 4132, 4231)
8. Av (1243, 1324, 1342, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
9. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
10. Av (1243, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4132, 4231)
11. Av (1243, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
12. Av (1243, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
13. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
14. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
15. Av (1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
16. Av (1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
17. Av (1324, 1342, 2143, 2314, 2341, 2413, 2431, 3142, 3241, 3412)

B.20 Experimental class XX

1. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 4132, 4231)
2. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 4132, 4231)
3. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 4132, 4231)
4. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4132, 4231)
5. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4132, 4231)

B.21 Experimental class XXI

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132)
3. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4132)
4. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132)
5. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132)
6. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4132)
7. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4132)

B.22 Experimental class XXII

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 3142, 3412, 4231)
3. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 3412, 4132, 4231)
4. Av (1243, 1324, 1342, 1423, 1432, 2143, 2431, 3412, 4132, 4231)
5. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3412, 4132, 4231)
6. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
7. Av (1243, 1324, 1342, 1432, 2143, 2413, 3142, 3412, 4132, 4231)
8. Av (1243, 1324, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)

B.23 Experimental class XXIII

1. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124, 3412)
2. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 3412, 4123)
3. Av (1234, 1243, 1324, 1342, 1423, 2314, 2341, 3124, 3412, 4123)
4. Av (1234, 1243, 1342, 1423, 2134, 2314, 2341, 3124, 3412, 4123)
5. Av (1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124, 3412, 4123)

B.24 Experimental class XXIV

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 4132, 4231)

B.25 Experimental class XXV

1. Av (1234, 1243, 1324, 1342, 1423, 2134, 2314, 2341, 3124, 4123)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 4132)

B.26 Experimental class XXVI

1. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4231)
2. Av (1243, 1324, 1342, 1423, 1432, 2143, 2413, 2431, 3412, 4132, 4231)
3. Av (1243, 1324, 1342, 1423, 1432, 2413, 2431, 3142, 3412, 4132, 4231)
4. Av (1243, 1324, 1342, 1423, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
5. Av (1243, 1324, 1342, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
6. Av (1243, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)
7. Av (1324, 1342, 1423, 1432, 2143, 2413, 2431, 3142, 3412, 4132, 4231)

Appendix C

Experimental setup

Listing A.1 shows how experimental classes were evaluated. The example was used for experimental class XIV.

```
1 import itertools
2 from permata.misc import UnionFind
3 from comb_spec_searcher.bijection import ParallelSpecFinder
4 from tilings.tilescope import TileScopePack, TileScope
5 from tilings.strategies import BasicVerificationStrategy
6
7
8 def class_bijections(bases, packs):
9     uf = UnionFind(len(bases))
10    connections = []
11    for packs in itertools.product(packs, packs):
12        for idx1, basis1 in enumerate(bases):
13            for idx2_offset, basis2 in enumerate(bases[idx1 + 1 :]):
14                idx2 = idx1 + idx2_offset + 1
15                if uf.find(idx1) == uf.find(idx2):
16                    continue
17                t1 = TileScope(basis1, packs[0])
18                t2 = TileScope(basis2, packs[1])
19                if ParallelSpecFinder(t1, t2).find() is not None:
20                    uf.unite(idx1, idx2)
21                    connections.append((idx1, idx2))
22    return len(connections) == len(bases) - 1, connections
23
24
25 bases = [
26     "0132_0213_0231_0312_0321_1032_1320_2301_3021",
27     "0132_0213_0231_0312_1032_1302_1320_2301_3021",
28     "0132_0213_0231_0321_1032_1302_1320_2301_3021",
29     "0132_0213_0321_1032_1302_1320_2031_2301_3021",
30 ]
31 packs = [
32     TileScopePack.row_and_col_placements(row_only=True).add_verification(
33         BasicVerificationStrategy(), replace=True
34     ),
35     # Extend as needed
36 ]
37 connected, connections = class_bijections(bases, packs)
38 print(f"Confirmed Wilf class: {connected}")
39 print(", ".join(f"({a},{b})" for a, b in connections))
```

Listing C.1: Experimental setup for evaluating experimental classes.

Appendix D

Connections created by bijections

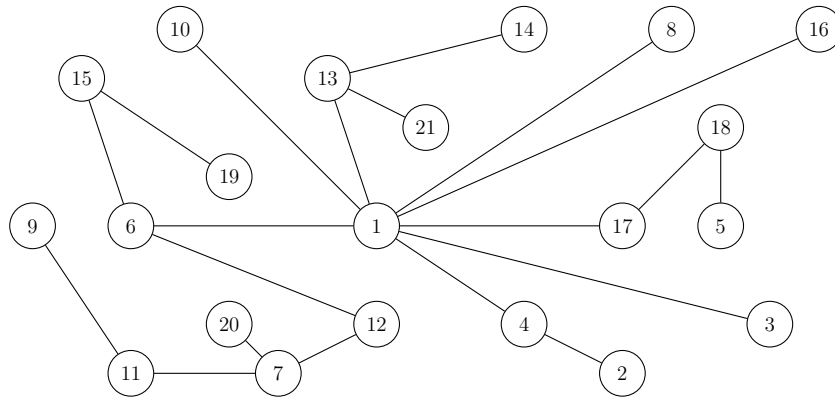


Figure D.1: The connections created by bijections for experimental class I.

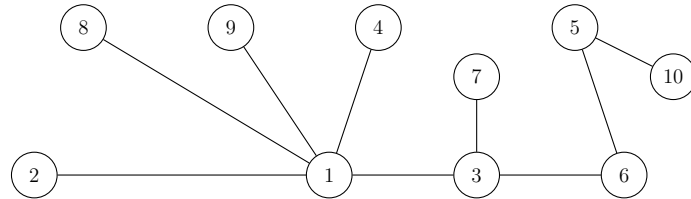


Figure D.2: The connections created by bijections for experimental class II.

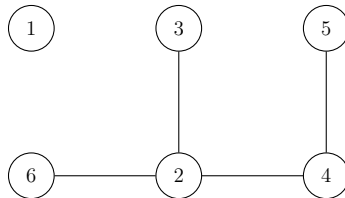


Figure D.3: The connections created by bijections for experimental class III where we failed to connect one permutation class.

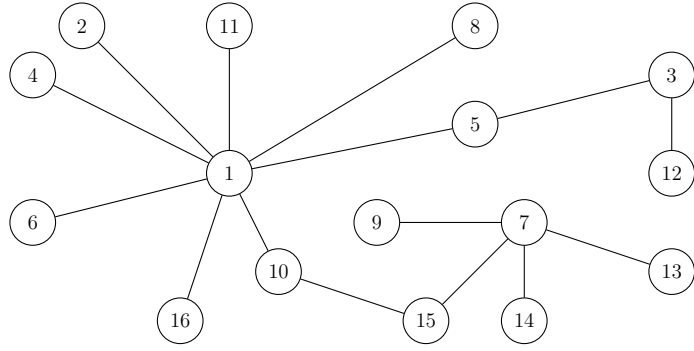


Figure D.4: The connections created by bijections for experimental class IV.

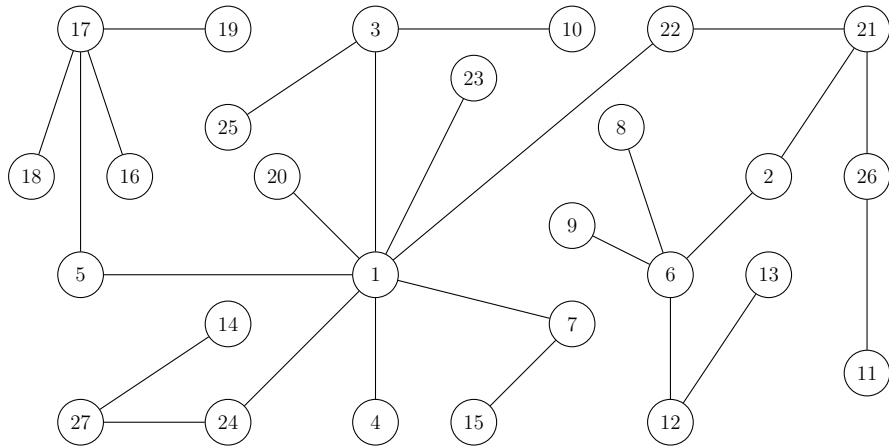


Figure D.5: The connections created by bijections for experimental class V.

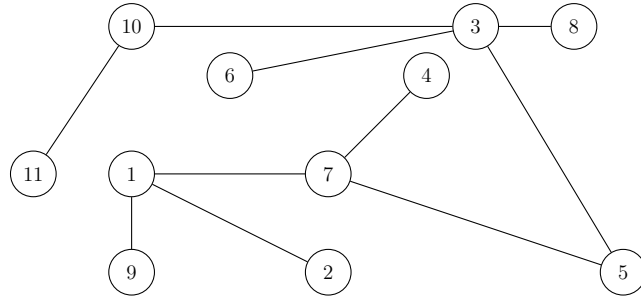


Figure D.6: The connections created by bijections for experimental class IX.

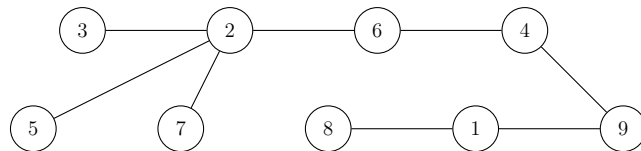


Figure D.7: The connections created by bijections for experimental class X.



Figure D.8: The connections created by bijections for experimental class XI.

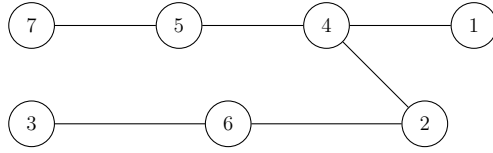


Figure D.9: The connections created by bijections for experimental class XIII.



Figure D.10: The connections created by bijections for experimental class XIV.

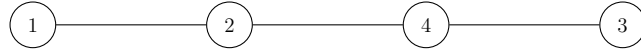


Figure D.11: The connections created by bijections for experimental class XV.

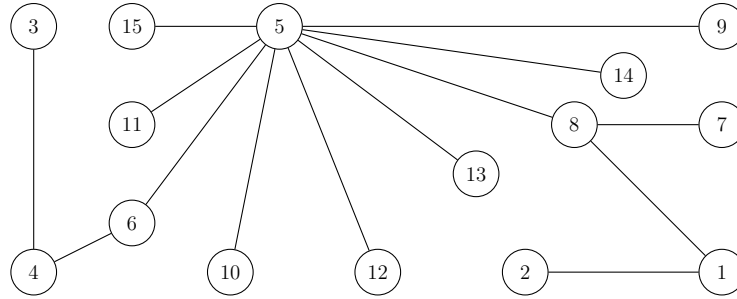


Figure D.12: The connections created by bijections for experimental class XVII.

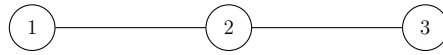


Figure D.13: The connections created by bijections for experimental class XVIII.

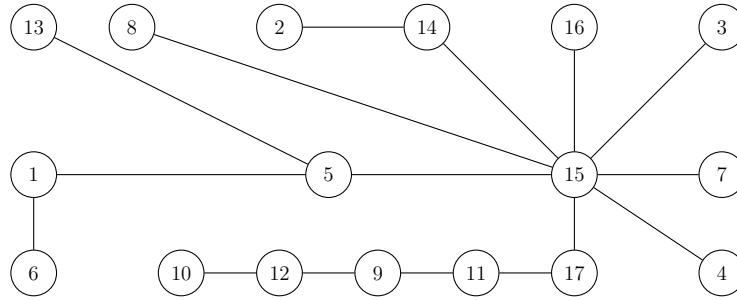


Figure D.14: The connections created by bijections for experimental class XIX.

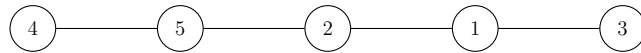


Figure D.15: The connections created by bijections for experimental class XX.

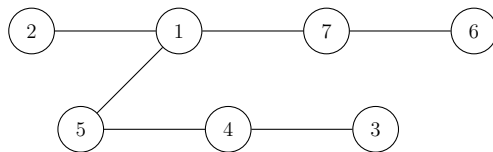


Figure D.16: The connections created by bijections for experimental class XXI.

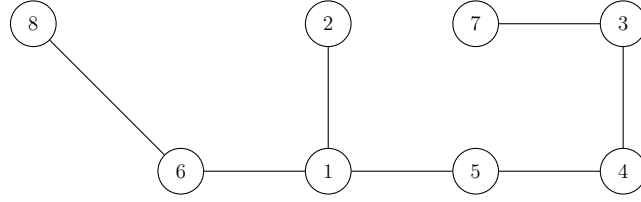


Figure D.17: The connections created by bijections for experimental class XXII.

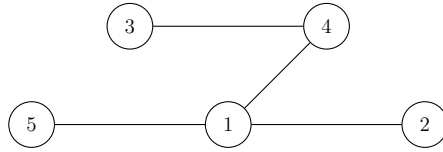


Figure D.18: The connections created by bijections for experimental class XXIII.

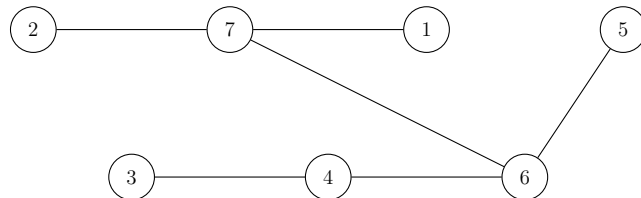


Figure D.19: The connections created by bijections for experimental class XXVI.