EPTCS 445

Proceedings of the

14th edition of the conference on

Random Generation of Combinatorial Structures

Malosco (BZ), Italy, 8-12 June 2026

Edited by: Srečko Brlek and Luca Ferrari

Published: 7th June 2026
DOI: 10.4204/EPTCS.445
ISSN: 2075-2180
Open Publishing Association

Preface

The fourteenth International Conference on the generation of combinatorial structures (GASCom) took place in Malosco (Italy) from June 8 to 12, 2026. Malosco is a charming village situated at an altitude of 1,041 metres above sea level, making this edition of GASCom the highest ever held. From the very first edition held in Bordeaux (France) this biennial conference steadily increased its international audience until the unfortunate cancellation of the year 2020 edition (due to COVID). Resuming after 2022, the event has been completely revived, as evidenced by the success of the 2024 edition held in Bordeaux.

The series of GASCom conferences brings together researchers in combinatorics, algorithms, probabilities, and more generally, theoretical computer science around the theme of random and exhaustive generation of combinatorial structures. In connection with this main theme, the conference is also interested in contributions in enumerative or analytic combinatorics and interactions with other areas of mathematics, computer science, physics, or biology. The conference is interested in both methods for random or exhaustive generation and in original results on combinatorial or algorithmic questions whose solution has been made possible by an approach involving random or exhaustive generation. The conference featured 3 invited talks and 19 contributed talks. These proceedings include nineteen fully refereed papers, dealing with topics closely related to the generation of combinatorial structures and bijective combinatorics.

The reviewing process was performed with the highest rigour, and we wish to thank all authors who contributed to this special issue and especially the anonymous referees for their generosity in the careful reading of the papers. We would like to extend our special thanks to the editors of the EPTCS for giving us the opportunity to publish these proceedings, as well as to the EPTCS team, whose assistance and collaboration were invaluable in the preparation of this volume.

We dedicate this special issue to Prof. Elena Barcucci, who recently retired from the Dipartimento di Matematica e Informatica "U. Dini" of the Università di Firenze. From the very beginning of GASCom in 1994, Elena was a regular contributor, participant, keynote speaker (Dijon 2006, Montréal 2010), and program committee member as well (14, 18, 22, 24, 26). Elena is an eminent member of the Florentine school, famous for the well-known Florentine rejection algorithm, and the ECO method for enumerating various classes of combinatorial objects.

We enjoyed working on its preparation and hope that you will enjoy reading it.

Guest Editors

Srečko Brlek
Laboratoire LACIM
Université du Québec à Montréal
Montréal (QC) Canada

Luca Ferrari
Dip. di Matematica e Informatica
Università di Firenze
Firenze, Italia

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. i–i, doi:10.4204/EPTCS.445.0

© S. Brlek & L.Ferrari
This work is licensed under the
Creative Commons Attribution License.

Generating Expansions in Cantor Real Bases via a Transducer

Émilie Charlier
Université de Liège, Belgium
echarlier@uliege.be

Representing real numbers using suitable numeration systems - such as integer bases, beta-expansions, or Cantor bases - has long been a central topic in mathematics. In this talk, I will focus on Cantor real bases, and more specifically on automatic Cantor real bases and the properties of real number expansions in this framework.

I will introduce a single transducer associated with a fixed real number r, which computes the B-expansion of r for an infinite family of Cantor real bases B given as input. This point of view contrasts with traditional computational models, where the numeration system is fixed in advance. Under suitable assumptions on the finitely many Pisot numbers occurring in the Cantor real base, only a finite portion of the transducer is visited. I will then present several fundamental results concerning both the structure of this transducer and decidability questions related to these expansions. In particular, for certain classes of Cantor real bases, important combinatorial properties - such as greediness or periodicity of the expansion - can be decided algorithmically. This talk is based on joint work with Pierre Popoli and Michel Rigo.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. ii–ii, doi:10.4204/EPTCS.445.0.1

© É. Charlier
This work is licensed under the
Creative Commons Attribution License.

Q-bonacci Words and Related Structures

Sergey Kirgizov
Université Bourgogne Europe, France
sergey.kirgizov@u-bourgogne.fr

The Q-bonacci (also known as q-decreasing) words are binary, and all their locally maximal factors of the form 0^a1^b satisfy $qa > b$ or $a = 0$. We explore the formation and enumeration of such words in three cases of q: natural, rational and real. When q is natural, such words are enumerated by the classical $(q+1)$-bonacci sequence (next term is the sum of $q+1$ previous terms). When we go from natural to rational and real cases, the situation becomes more and more mesmerizing. Many connections with recent, classical well-known, as well as fairly esoteric mathematical structures will be discussed during this talk.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. iii–iii, doi:10.4204/EPTCS.445.0.2

© S. Kirgizov
This work is licensed under the
Creative Commons Attribution License.

Ascending Polyominoes

Simone Rinaldi
University of Siena, Italy
simone.rinaldi@unisi.it

Convex polyominoes can be refined according to the number of direction changes in monotone paths connecting pairs of cells, leading to the notion of k-convexity. In particular, the cases $k = 1$ and $k = 2$ correspond to L-convex and Z-convex polyominoes, two classical subclasses exhibiting very different combinatorial behaviours.

Between these two extremal families lie several intermediate classes, such as centered and 4-stack polyominoes, which form a natural hierarchy. A striking feature of this hierarchy is the transition in the nature of the generating functions and asymptotic growth: while L-convex and centered polyominoes have rational generating functions and exponential growth of order $(2+\sqrt{2})^n$ and 4^n, respectively, the classes of 4-stack, Z-convex, and general convex polyominoes are described by algebraic generating functions with asymptotics of order $\sqrt{n}4^n$ or $n4^n$.

In this work we focus on Z-convex polyominoes and introduce a finer structural analysis based on the notions of NW- and NE-convexity degree. This refinement yields a decomposition into three disjoint subclasses, denoted by $C(1,2)$, $C(2,1)$, and $C(2,2)$, providing a more detailed understanding of the internal structure of Z-convex polyominoes.

To enumerate these families, we introduce the class of ascending polyominoes, characterized by a simple geometric condition. The enumeration is carried out via the application of the ECO method and a generating tree construction, leading to functional equations for the corresponding generating functions. By solving these equations, we obtain explicit algebraic expressions for the generating functions of ascending polyominoes, centered ascending polyominoes and of the subclasses $C(1,2)$, $C(2,1)$, and $C(2,2)$.

Finally, we derive the asymptotic growth of these families, showing that $C(1,2)$ and $C(2,1)$ have growth of order $n4^n$, while the class $C(2,2)$ exhibits growth of order $\sqrt{n}4^n$, revealing a precise combinatorial transition within the hierarchy of convex polyominoes.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. iv–iv, doi:10.4204/EPTCS.445.0.3

© S. Rinaldi
This work is licensed under the
Creative Commons Attribution License.

Table of Contents

Preface i
Srečko Brlek and Luca Ferrari

Invited Presentation: Generating Expansions in Cantor Real Bases via a Transducer ii
Émilie Charlier

Invited Presentation: Q-bonacci Words and Related Structures iii
Sergey Kirgizov

Invited Presentation: Ascending Polyominoes iv
Simone Rinaldi

Table of Contents v

Touchard-Riordan Polynomials and Schur-positivity of Set Partitions 1
Eli Bagno and David Garber

Random Generation of k-coloured Motzkin Paths 10
Elena Barcucci, Antonio Bernini, Stefano Bilotta and Renzo Pinzani

Fibonacci and Catalan Numbers Meet in Staircase Polyominoes 16
Jean-Luc Baril, José Luis Ramírez, Samuel Ramírez and Diego Villamizar

Computing Joins in the Weak Order of Type B Coxeter Groups: an Algorithmic Approach 21
Riccardo Biagioli and Lorenzo Perrone

Snake Polyominoes of Maximal Area in a Rectangle 29
Alexandre Blondin Massé and Alain Goupil

Entropic Generation of Binary Words 38
Olivier Bodini and Francis Durand

Growing Binary Trees 47
Olivier Bodini, Antoine Genitrini and Khaydar Nurligareev

On the Counting Sequence of Z-convex Polyominoes 57
Luca Castelli and Paolo Massazza

Penrose P2 Tilings: A Study of Fully Leafed Induced Subtrees 66
Mathieu Cloutier, Alain Goupil and Alexandre Blondin Massé

k-Convex Polyominoes by Semi-perimeter 74
Andrew R. Conway and Anthony J. Guttmann

The Genesis Sequence, Tree Records and Endofunctions 87
Enrica Duchi, Adrián Lillo, Pablo Puerto, Mercedes Rosas and Stefan Trandafir

The Curious Case of Reversible Elementary Second Order Cellular Automaton 115 95
Enrico Formenti and Supreeti Kamylia

Generation of Maximal Snake Polyominoes Using a Deep Neural Network 104
Benjamin Gauthier, Alain Goupil and Fadel Toure

(Un)ranking Permutation Classes 114
Nathanaël Hassler and Vincent Vajnovszki

Frequencies of Patterns in Smooth Sequences Over the Alphabet $\{1,3\}$ 121
Damien Jamet, Irène Marcovici, Léo Poirier and Thierry de la Rue

Exhaustive Generation of Genus-One Knot and Link Diagrams via Maps on the Torus 130
Alexander Omelchenko

Permanental Inequalities and Unit Interval Orders 139
Sihong Pan, Mark Skandera and Jiayuan Wang

On the Impossibility of Parabolic Factorization of certain Kazhdan-Lusztig Basis Elements 148
Tommy Parisi, Mark Skandera, Ben Spahiu and Jiayuan Wang

A Class of Multiparameter Signless Stirling Numbers of the First Kind and their q-Analogues 158
Violetta E. Piperigou and Malvina G. Vamvakari

Touchard-Riordan Polynomials and Schur-positivity of Set Partitions

Eli Bagno
Jerusalem College of Technology, 21 HaVaad HaLeumi St., Jerusalem, Israel,
and Michlalah College Jerusalem, 36 Barukh Duvdevani St., Jerusalem, Israel
bagnoe@g.jct.ac.il

David Garber
Holon Institute of Technology, 52 Golomb St., P.O.Box 305, 5810201 Holon, Israel
garber@hit.ac.il

A symmetric function is called *Schur-positive* if it admits an expansion in the Schur basis with nonnegative coefficients. In this paper, we study the Schur-positivity of symmetric functions naturally associated with set partitions, with respect to a descent set function that considers i as descent, if i and $i+1$ share a block in the partition. The Schur expansion involves hook-shaped Young diagrams, and the corresponding coefficients are given by Touchard-Riordan polynomials, which enumerate matchings by their number of crossings.

1 Introduction

Schur-positivity is a central notion in algebraic combinatorics, referring to symmetric functions that can be expressed as a nonnegative linear combination of Schur functions. Since Schur functions form a distinguished basis for the ring of symmetric functions and correspond to irreducible representations of the symmetric group, Schur-positivity often reflects deep algebraic and representation-theoretic structure. Determining whether a given symmetric function is Schur-positive is a major problem in contemporary algebraic combinatorics [13].

Given $D \subseteq \{1,\ldots,n-1\}$, define $\mathscr{F}_D$ to be the fundamental quasi-symmetric function indexed by D (see Definition 2.3 below). Given a combinatorial object σ, one is interested in studying the generating function $\mathscr{Q}_n = \sum \mathscr{F}_{\mathrm{Des}(\sigma)}$, where $\mathrm{Des}(\sigma)$ is a descent set function from the set of the objects σ is taken from, to the set $\mathbb{N}$. We look for functions $\mathscr{Q}_n$ which are symmetric or Schur-positive for each n. When the objects under discussion are permutations, this subject has been developed extensively. Unlike the permutation case, there is still no comparably general framework explaining Schur-positivity for broad families of set partitions, except for some initial results, see [3].

Set partitions can be encoded by restricted growth words, and this encoding has been used extensively in the literature on statistics of set partitions and their generating functions, see e.g. [4] and [6, Sec. 1.7]. This makes the study of Schur-positivity for set partitions a natural extension of the classical permutation setting: one seeks structural conditions on families of set partitions, or on their restricted-growth words, that force the associated quasi-symmetric generating functions to be symmetric and Schur-positive.

Adin and Roichman [2] dealt with the descent-type parameter on matchings (i.e. set partitions having blocks of size at most 2) which they called *geometric descent*: an index $i \in [n-1]$ is a *geometric descent* of a matching M on $[n]$ if one of the following conditions holds:

1. $\{i, i+1\}$ is a matched pair in M.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 1–9, doi:10.4204/EPTCS.445.1

© Eli Bagno and David Garber
This work is licensed under the
Creative Commons Attribution License.

2. The arc containing i intersects the arc containing $i+1$.

3. i is unmatched and $i+1$ is matched.

Moreover, Marmor [8] called a matched pair $\{i, i+1\}$ in a matching *a short chord* and proved that the set of matchings of the set $[n]$ is Schur-positive with respect to the descent function that takes into account only short chords.

Motivated by these works, we define a descent parameter on set partitions, denoted Des, which counts the number of elements i, such that i and $i+1$ share the same block.

We use one of the existing definitions of a crossing number of set partitions that can be found in the literature (namely the one defined in Definition 2.6 below) to extrapolate between Schur-positivity on the entire set of set partitions of $[n]$, denoted $\mathbf{S}et(n)$, and the set of non-crossing set partitions, denoted $\mathscr{NC}(n)$. The former set uses the Touchard-Riordan polynomials as coefficients of the linear combination into Schur functions, while the later uses the Motzkin triangle for this very purpose.

It is worth noting that we have many more results on Schur-positivity of various subsets of set partitions of $[n]$, with respect to another type of descent function. They are gathered in a paper, which is in final steps of preparation.

This paper is organized as follows. In Section 2, we provide a background on Schur-positivity, crossing number of a set partition and its connection to the Touchard-Riordan polynomials and the Motzkin triangle. In Section 3, we prove our main results.

2 Background

2.1 Symmetric functions, quasi-symmetric functions and Schur positivity

In this subsection, we follow [1,7]. Let $\mathbf{x} = \{x_1, x_2, \dots\}$ be a countably infinite set of commuting variables and consider the algebra of formal power series over $\mathbb{Q}$.

A power series $f \in \mathbb{Q}[[\mathbf{x}]]$ is called *symmetric* if it has a bounded degree and it is invariant under permutation of variables. Denote by Sym_n the vector space of symmetric functions, homogeneous of degree n. Bases for Sym_n are indexed by *partitions λ of n* (denoted $\lambda \vdash n$), or equivalently, by Young diagrams with n boxes.

The celebrated Schur basis of Sym_n is essential for our needs. Given a partition $\lambda \vdash n$, a *standard Young tableau (SYT) of shape λ* is obtained by filling the boxes of the corresponding Young diagram bijectively with the elements of $\{1, \dots, n\}$ so that rows and columns increase. In a *semistandard Young tableau (SSYT) of shape λ*, we demand that the entries are positive integers such that rows weakly increase and columns strictly increase. We write $\mathrm{SYT}(\lambda)$ or $\mathrm{SSYT}(\lambda)$ for the set of standard or semistandard Young tableaux of shape λ, respectively.

In the following definition, we present the notion of a *descent set* of an SYT T:

Definition 2.1. *The* descent set of a standard Young tableau T *is defined to be:*

$$\mathrm{Des}(T) = \{i \mid i+1 \text{ is in a lower row than } i \text{ in } T\}.$$

For example, if $T = \begin{array}{|c|c|c|}\hline 1 & 3 & 5 \\ \hline 2 & 4 & 7 \\ \hline 6 \\ \cline{1-1} \end{array}$, then $\mathrm{Des}(T) = \{1,3,5\}$.

The *Schur function corresponding to a partition* $\lambda \vdash n$ is: $s_\lambda = \sum_{T \in \mathrm{SSYT}(\lambda)} \prod_{i \in T} x_i$. For example, the set $\mathrm{SSYT}(3,1)$ contains inter alia the following semistandard fillings of the shape $\lambda = \ydiagram{3,1}$:

$$T_1 = \begin{array}{|c|c|c|}\hline 1&1&1\\\hline 2\\\cline{1-1}\end{array}, \quad T_2 = \begin{array}{|c|c|c|}\hline 1&2&2\\\hline 2\\\cline{1-1}\end{array}, \quad T_3 = \begin{array}{|c|c|c|}\hline 1&2&3\\\hline 4\\\cline{1-1}\end{array},$$

which contribute to $s_{(3,1)}$ the monomials $m_1 = x_1^3 x_2$, $m_2 = x_1 x_2^3$ and $m_3 = x_1 x_2 x_3 x_4$ respectively. It is well-known [12, Chap. 7] that the set $\{s_\lambda\}_{\lambda \vdash n}$ is a basis of Sym_n.

For each $0 \leq k \leq n$, define the *hook diagram* $\lambda_k = (n-k, 1^k)$ (where 1^k stands for k parts of size 1), and note that:

$$\{\mathrm{Des}(T) : T \in \mathrm{SYT}(\lambda_k)\} = \{D \subseteq [n-1] : |D| = k\}.$$

A symmetric function is called *Schur-positive* if all the coefficients in its expansion in the basis of Schur functions are nonnegative. Determining whether a given symmetric function is Schur-positive is a major problem in contemporary algebraic combinatorics [13].

Quasi-symmetric functions extend symmetric functions. Here is the formal definition:

Definition 2.2. *A* quasi-symmetric function *is a formal power series g of bounded degree, satisfying that any two of its monomials $x_{i_1}^{n_1} \dots x_{i_k}^{n_k}$ (where $i_1 < \dots < i_k$) and $x_{j_1}^{n_1} \dots x_{j_k}^{n_k}$ (where $j_1 < \dots < j_k$) have the same coefficient in g.*

Clearly, every symmetric function is quasi-symmetric, but not conversely; for example, $\sum_{i<j} x_i^2 x_j$ is quasi-symmetric but not symmetric.

The fundamental quasi-symmetric functions, which are defined now, serve as a basis for the vector space of homogeneous quasi-symmetric functions of degree n:

Definition 2.3. *For each subset $D \subseteq [n-1]$ define the* fundamental quasi-symmetric function

$$\mathscr{F}_{n,D}(\mathbf{x}) := \sum_{\substack{i_1 \leq i_2 \leq \dots \leq i_n \\ i_j < i_{j+1} \text{ if } j \in D}} x_{i_1} x_{i_2} \cdots x_{i_n}.$$

Let $\mathscr{B}$ be a (multi-)set of combinatorial objects, equipped with a *descent function* $\mathrm{Des} : \mathscr{B} \to P([n-1])$, which associates to each element $b \in \mathscr{B}$ a subset $\mathrm{Des}(b) \subseteq [n-1]$. Define the quasi-symmetric function

$$\mathscr{Q}_n(\mathscr{B}) := \sum_{b \in \mathscr{B}} m(b, \mathscr{B}) \mathscr{F}_{n,\mathrm{Des}(b)},$$

where $m(b, \mathscr{B})$ is the multiplicity of the element $b \in \mathscr{B}$.

Gessel showed that (see [12, Theorem 7.19.7]):

Theorem 2.4 (Gessel)**.** *For every partition $\lambda \vdash n$, $\mathscr{Q}_n(\mathrm{SYT}(\lambda)) = s_\lambda$.*

For another substantial example, given $A \subseteq \mathbf{S}_n$, where $\mathbf{S}_n$ is the Symmetric group on n elements, define the quasi-symmetric function $\mathscr{Q}_n(A) := \sum_{\pi \in A} \mathscr{F}_{n,\mathrm{Des}(\pi)}$, where $\mathrm{Des}(\pi) := \{i : \ \pi(i) > \pi(i+1)\}$ is the ordinary descent set of π.

2.2 The crossing number and Touchard-Riordan polynomials

Let n be a positive integer and let $\mathbf{S}et(n)$ be the set of all set partitions of $[n]$. Let $\pi = \{B_1, B_2, \ldots, B_k\} \in \mathbf{S}et(n)$ be a set partition of $[n]$ into blocks $B_1, \ldots, B_k$, and assume that the elements in each block B_i are ordered increasingly. The partition π can be depicted graphically as a linear graph made of the n points $\{1, \ldots, n\}$, drawn on a (virtual) horizontal line, ordered increasingly by their labels. For every block $B_j = \{j_1, \ldots, j_t\}$, we join every two points which correspond to consecutive elements, j_k and j_{k+1}, from the same block by a semicircular arc. In the case that $j_{k+1} = j_k + 1$, we join j_k and j_{k+1} by a straight line instead of an arc.

Example 2.5. *Let $\pi = \{\{1,3,7\},\{2,4,5,6,8\}\}$ be a partition of the set $[8]$. Then its linear graph is presented in Figure 1.*

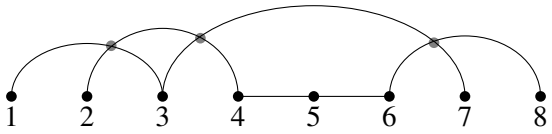


Figure 1: An example of a linear graph

There are several different definitions for the notion of crossing number of a set partition. In this work, we choose to concentrate on the following version, see e.g. [11]:

Definition 2.6. *Let π be a set partition of the set $[n]$, presented as a linear graph as above. For $r < s$, two arcs (i_r, j_r) and (i_s, j_s)* cross *each other if $i_r < i_s < j_r < j_s$. We denote by* $\mathrm{cr}(\pi)$ *the number of such crossings and call it the* crossing number *of the set partition π.*

In Example 2.5, the crossings are depicted as gray points, so we have $\mathrm{cr}(\pi) = 3$.

The crossing number can be similarly defined on matchings of $[n]$, which are a subset of $\mathbf{S}et(n)$.

Definition 2.7. *A partition $\pi \in \mathbf{S}et(n)$ in which each block contains at most 2 elements is called a* matching. *If n is even and all the blocks are of size 2, then π is called a* perfect matching. *A matching of the set $[n]$ can be depicted as a* chord diagram*, which is a circle containing n points, labeled by the numbers $1, \ldots, n$ ordered clockwise on the circle. Each block of size 2 is represented by a chord connecting its two elements, see Figure 2.*

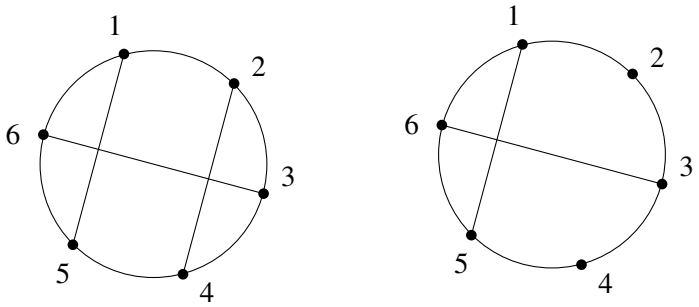


Figure 2: Left: a perfect matching, right: a non-perfect matching

Definition 2.8. *The* crossing number of a matching *M, denoted* $\mathrm{cr}(M)$*, is the crossing number of its corresponding partition as defined above, which can also be seen as the number of pairs of crossing chords in the chord diagram.*

The generating function of all perfect matchings on $2m$ points with respect to the crossing number is given by Touchard-Riordan polynomials, which are defined as follows:

$$\sum_M q^{\mathrm{cr}(M)} = T_{2m}(q) = \frac{1}{(1-q)^m}\sum_{i=0}^{m}(-1)^i\left(\binom{2m}{m-i}-\binom{2m}{m-i-1}\right)q^{\binom{i+1}{2}},$$

where M runs through all perfect matchings on $2m$ points. Since isolated points have no effect on the crossing number, we can easily obtain the generating function for all the matchings on n points (not necessarily perfect), as follows. Let $T_{n,j}(q)$ be the generating function for the number of matchings on n points with j chords with respect to the crossing number. Then we clearly have $p = n-2j$ unmatched points, and thus we have: $T_{n,j}(q) = \binom{n}{2j}T_{2j}(q)$.

2.3 Non-crossing matchings and Motzkin numbers

Matchings M satisfying $\mathrm{cr}(M) = 0$, i.e. having no crossings, deserve special attention since they are counted by the *Motzkin numbers*, as originally introduced by Motzkin [9] (these numbers have other various combinatorial interpretations, see [5] and Sequence A001006 in OEIS [10]). The recursion defining Motzkin numbers is:

$$M_n = M_{n-1} + \sum_{i=0}^{n-2} M_i M_{n-2-i} = \frac{2n+1}{n+2}M_{n-1} + \frac{3n-3}{n+2}M_{n-2},$$

and the first Motzkin numbers are: $M_0 = 1$, $M_1 = 1,\ 2,\ 4,\ 9,\ 21,\ 51,\ 127,\ 323$.

Example 2.9. *As $M_4 = 9$, there are 9 ways to draw non-crossing chords between 4 points on the circle, which are drawn in Figure 3.*

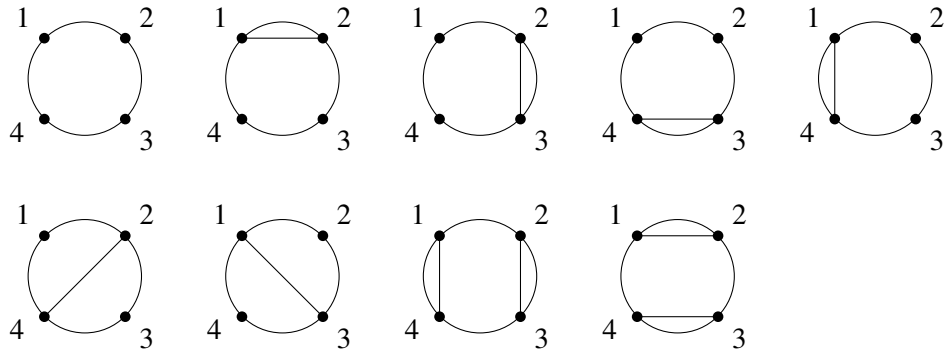


Figure 3: The 9 ways to draw non-crossing chords between 4 points on the circle

There is a refinement of Motzkin numbers, called *Motzkin triangle* (see Sequence A055151 in the OEIS [10]), whose first 9 lines are presented in Table 1.

n \ k	0	1	2	3	4
0	1				
1	1				
2	1	1			
3	1	3			
4	1	6	2		
5	1	10	10		
6	1	15	30	5	
7	1	21	70	35	
8	1	28	140	140	14

Table 1: Table form of sequence A055151

The element $M_{n,k}$ in the Motzkin triangle counts the number of ways to draw k non-crossing chords between n points on the circle. For example, the fifth line of the triangle "1 6 2" counts one diagram of 4 points on a circle without chords, 6 diagrams with one chord and 2 diagrams with 2 chords (see Example 2.9).

Remark 2.10. *Note that the combinatorial explicit connection between Touchard-Riordan polynomials and Motzkin numbers is given by $T_{2n}(0) = M_{2n}$. We use this observation in Corollary 3.3(3)-(4) below. We could not find an explicit appearance in the literature of this connection.*

3 Schur-positivity of set partitions with respect to consecutive elements sharing a block

Let $\mathbf{S}et(n,\ell)$ be the set of all set partitions of the set $[n]$ with ℓ blocks (and respectively, we denote by $\mathscr{NC}(n,\ell)$ the non-crossing partitions of $[n]$ having ℓ blocks). For $\pi \in \mathbf{S}et(n,\ell)$, define:

$$\mathrm{Des}(\pi) = \{i \in [n-1] \mid i \text{ and } i+1 \text{ are in the same block}\}.$$

Note that this set function is not sparse in the sense of Definition 1.7 of Marmor [8].

Our main result addresses the Schur-positivity of set partitions having ℓ blocks with respect to the parameter Des we have just defined:

Theorem 3.1.

$$\sum_{\pi \in \mathbf{S}et(n,\ell)} q^{\mathrm{cr}(\pi)} \mathscr{F}_{n,\mathrm{Des}(\pi)} = \sum_{k=0}^{n-1} T_{n-1-k,n-k-\ell}(q) s_{(n-k,1^k)} \tag{1}$$

Proof. Let $\mathscr{D}_{n,j}$ be the set of chord diagrams having n points and j chords, and let $\mathbf{S}et(n,\ell,k)$ be the set of set partitions of $[n]$ with ℓ blocks and k descents.

One can consider both sides of Equation (1) as polynomials in the variable q whose coefficients are in the ring of quasi-symmetric functions. To prove the equality, we have to show that the corresponding coefficients are equal. We do that with the aid of Gessel's Theorem 2.4, by invoking a bijection

$$\mathrm{SYT}(n-k,1^k) \times \mathscr{D}_{n-1-k,n-k-\ell} \to \mathbf{S}et(n,\ell,k),$$

which preserves the crossing number.

We start with the case $k = 0$, i.e., we present a bijection from the set of pairs

$$P = \left\{ \left(\boxed{1}\boxed{2}\boxed{\cdots}\boxed{n}\,,\, C \right) \mid C \in \mathscr{D}_{n-1,n-\ell} \right\}$$

to $\mathbf{S}et(n,\ell,0)$ which, as defined above, is the set of set partitions of $[n]$ with ℓ blocks and no descents. Let $C \in \mathscr{D}_{n-1,n-\ell}$, i.e., a chord diagram on $n-1$ points with $n-\ell$ chords. The bijection has two steps: First, for each chord $\{i,j\}$ of C with $i < j$, we create the block $\{i, j+1\}$. Next, we unify every two intersecting blocks. The points unmatched by chords contribute singletons to the resulting set partition.

Since the crossing number depends solely on the arcs that connect consecutive elements in the blocks of the partition, it is easy to see that the resulting set partition, consisting of ℓ blocks, has the same number of crossings as in the chord diagram.

In the opposite direction, when we are given a set partition with no descents, it is straightforward to recover the chord diagram which is mapped to it.

For example, for the chord diagram in the left part of Figure 4, which is an element of $\mathscr{D}_{7,3}$, we first construct the blocks $\{1,4\},\{2,6\},\{4,8\}$, and then we send it to the set partition

$$\{\{1,4,8\},\{2,6\},\{3\},\{5\},\{7\}\} \in \mathbf{S}et(8,5,0).$$

Note that there are two crossings in both the chord diagram and the corresponding set partition.

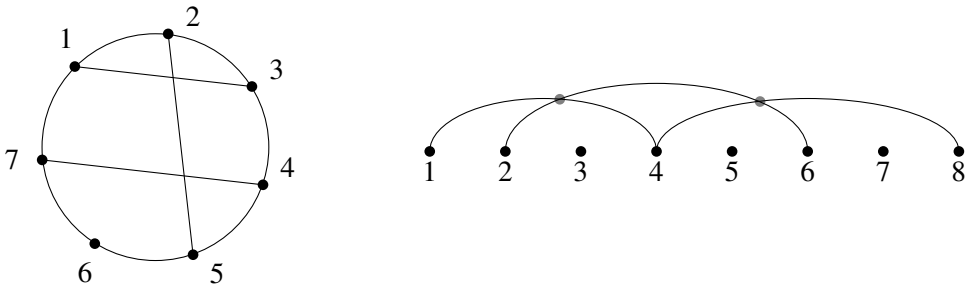


Figure 4: Left: A chord diagram. Right: The linear graph of the resulting set partition.

We turn now to the case $k \geq 1$. Clearly, every standard Young tableau of shape $(n-k,1^k)$ has exactly k descents. Given a pair $(T,C) \in \mathrm{SYT}(n-k,1^k) \times \mathscr{D}_{n-1-k,n-k-\ell}$, we first define a function $\varphi : [n] \to [n-k]$ by:

$$\varphi(j) = j - |\{i \in \mathrm{Des}(T) \mid i < j\}|.$$

Next, we apply the procedure described above for the case $k = 0$ for the chord diagram C having $n-1-k$ points on a circle with $n-k-\ell$ chords, resulting in a partition of the set $[n-k]$, having ℓ blocks. Finally, we obtain a set partition of $[n]$ by replacing each element of $[n-k]$ by the elements in its preimage by φ, see Example 3.2 below.

Note that in the last step, while inflating the set $[n-k]$ to the set $[n]$ using φ, the crossing number of the set partition does not change, as we count only crossings of arcs, while the inflation contributes only straight segments.

In the opposite direction, note that when we are given a set partition, one can easily recover the descent set, and the associated function φ. Then, it is straightforward to recover also the chord diagram which is mapped to it. □

Example 3.2. *Given a standard Young tableau T, with* $\mathrm{Des}(T) = \{1,2,5,7,9\} \subseteq [13]$ *and a chord diagram C as in Figure 4 above, we first define* $\varphi : [13] \to [8]$*, using the tableau T, as follows:*

$$\varphi(1) = \varphi(2) = \varphi(3) = 1,\ \varphi(4) = 2,\ \varphi(5) = \varphi(6) = 3,\ \varphi(7) = \varphi(8) = 4,$$

$$\varphi(9) = \varphi(10) = 5,\ \varphi(11) = 6,\ \varphi(12) = 7 \text{ and } \varphi(13) = 8.$$

Next, we use the chord diagram C to obtain the set partition, as computed in the case $k = 0$:

$$\{\{1,4,8\},\{2,6\},\{3\},\{5\},\{7\}\}.$$

Finally, we replace each element of the set $[8]$ *by the elements in its preimage by φ, to obtain the set partition:*

$$\{\{\underbrace{1,2,3}_{\varphi^{-1}(1)},\ \underbrace{7,8}_{\varphi^{-1}(4)},\ \underbrace{13}_{\varphi^{-1}(8)}\},\{\underbrace{4}_{\varphi^{-1}(2)},\ \underbrace{11}_{\varphi^{-1}(6)}\},\{\underbrace{5,6}_{\varphi^{-1}(3)}\},\{\underbrace{9,10}_{\varphi^{-1}(5)}\},\{\underbrace{12}_{\varphi^{-1}(7)}\}\} \in \mathbf{S}et(13,5,5).$$

Plugging $q = 1$ in Equation (1) induces an explicit Schur-positivity result for the set of set partitions, while substituting $q = 0$ reduces us to a Schur-positivity result for non-crossing partitions, where elements from the Motzkin triangle replace the coefficients of the Touchard-Riordan polynomials:

Corollary 3.3.

1. $\mathscr{Q}_n(\mathbf{S}et(n,\ell)) = \sum_{k=0}^{n-1} T_{n-1-k,n-\ell-k}(1)s_{(n-k,1^k)}$,
2. $\mathscr{Q}_n(\mathbf{S}et(n)) = \sum_{k=0}^{n-1}\sum_{\ell=1}^{n-k} T_{n-1-k,n-\ell-k}(1)s_{(n-k,1^k)}$,
3. $\mathscr{Q}_n(\mathscr{NC}(n,\ell)) = \sum_{k=0}^{n-1} M_{n-1-k,n-\ell-k}s_{(n-k,1^k)}$,
4. $\mathscr{Q}_n(\mathscr{NC}(n)) = \sum_{k=0}^{n-1} M_{n-k-1}s_{(n-k,1^k)}$.

Acknowledgments

We would like to express our gratitude to Yuval Roichman for fruitful discussions.

References

[1] Ron M. Adin, Eli Bagno & Yuval Roichman (2018): *Block decomposition of permutations and Schur-positivity*. *Journal of Algebraic Combinatorics* 47, pp. 603–622, doi:10.1007/s10801-017-0788-9.

[2] Ron M. Adin & Yuval Roichman (2023): *Cyclic descents, matchings and Schur-positivity*. *Electronic Journal of Combinatorics* 30(2), #P2.41, doi:10.37236/11761.

[3] Eli Bagno, David Garber, Toufik Mansour & Amir Safadi (2026): *Pattern avoidance and Schur-positivity in restricted-growth words of type B*. *Discrete Applied Mathematics* 386, pp. 116–138, doi:10.1016/j.dam.2026.01.020.

[4] Yue Cai & Margaret A. Readdy (2017): *q-Stirling numbers: A new view*. Advances in Applied Mathematics 86, pp. 50–80, doi:10.1016/j.aam.2016.11.007.

[5] Robert Donaghey & Louis W. Shapiro (1977): *Motzkin numbers*. Journal of Combinatorial Theory, Series A 23(3), pp. 291–301, doi:10.1016/0097-3165(77)90020-6.

[6] Ömer Eğecioğlu & Adriano M. Garsia (2021): *Lessons in Enumerative Combinatorics*. Graduate Texts in Mathematics 290, Springer, doi:10.1007/978-3-030-71250-1.

[7] Zachary Hamaker, Brendan Pawlowski & Bruce E. Sagan (2020): *Pattern avoidance and quasisymmetric functions*. Algebraic Combinatorics 3(2), pp. 365–388. doi:10.5802/alco.96.

[8] Avichai Marmor (2024): *Schur-positivity of short chords in matchings*. Algebraic Combinatorics 7(3), pp. 887–914. doi:10.5802/alco.351.

[9] Theodore S. Motzkin (1948): *Relations between hypersurface cross ratios, and a combinatorial formula for partitions of a polygon, for permanent preponderance, and for non-associative products*. Bulletin of the American Mathematical Society 54, pp. 352–360, doi:10.1090/s0002-9904-1948-09002-4.

[10] OEIS Foundation Inc. (2026): *The Online Encyclopedia of Integer Sequences*. Available at `https://oeis.org`.

[11] Svetlana Poznanović & Catherine Yan (2009): *Crossings and nestings of two edges in set partitions*. SIAM Journal on Discrete Mathematics 23(2), pp. 787–804, doi:10.1137/070705222.

[12] Richard P. Stanley (1999): *Enumerative Combinatorics, Vol. 2*. Cambridge University Press, doi:10.1017/CBO9780511609589.

[13] Richard P. Stanley (2000): *Positivity problems and conjectures in algebraic combinatorics*. In V. Arnold, M. Atiyah, P. Lax & B. Mazur, editors: Mathematics: Frontiers and Perspectives, American Math. Society, Providence, RI, pp. 295–319.

Random Generation of k-coloured Motzkin Paths

Elena Barcucci
University of Florence
Italy
elena.barcucci@unifi.it

Antonio Bernini
University of Florence
Italy
antonio.bernini@unifi.it

Stefano Bilotta
University of Florence
Italy
stefano.bilotta@unifi.it

Renzo Pinzani
University of Florence
Italy
renzo.pinzani@unifi.it

We study k-coloured Motzkin paths, namely Motzkin paths in which horizontal steps can be coloured in k different ways, and investigate their connection with the number of prefixes ending at odd height from both an analytical and a combinatorial point of view. Moreover, the combinatorial approach provides a random generation algorithm for k-coloured Motzkin paths in linear-time.

1 Foreword

The research presented in this contribution was initiated in early 2003 by two of the present authors, Elena Barcucci and Renzo Pinzani, together with Alberto Del Lungo, a past student of theirs, at that time full professor at the University of Siena. He suddenly died on June 3rd 2003, when he was only 38 years old. The pain of his death for the two was so high that they completely forgot the manuscript, describing some results about the mentioned research, which Alberto gave them a few days before his death. During last summer, Elena, while recovering the documents in her room because of her retirement, found the manuscript. As a consequence, Elena and Renzo asked the co-authors Antonio Bernini and Stefano Bilotta, two other past students of theirs, for help in revising and completing the results. They accepted with enthusiasm and so the work was completed.

2 Introduction

A *Motzkin path* is a lattice path in the plane that starts at $(0,0)$, ends on the x-axis, never goes below it, and consists of three types of steps: an up step $(1,1)$, a down step $(1,-1)$, and a horizontal step $(1,0)$. Motzkin paths are classical objects in enumerative combinatorics enumerated by the famous Motzkin numbers, originally introduced by T. Motzkin [10]. They have been extensively investigated due to their connections with Catalan structures, continued fractions, and various combinatorial sequences [9, 12]. Several extensions have been considered, including coloured Motzkin paths and weighted variations, which enrich the combinatorial structure while preserving recursive decompositions [8]. In addition, we also recall their role in the generation of cross bifix-free words [3], together with the study of their generating function under pattern-avoidance constraints [6].

Many efficient algorithms to randomly generate Motzkin words of fixed size are known. Several of them are based on rejection methods, as in [1], while Barcucci et al. [4, 5] proposed an efficient algorithm based on an anticipated rejection method, known as the Florentine algorithm [2].

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 10–15, doi:10.4204/EPTCS.445.2

© E. Barcucci, A. Bernini, S. Bilotta & R. Pinzani
This work is licensed under the
Creative Commons Attribution License.

In this paper we consider k-coloured Motzkin paths, namely Motzkin paths in which horizontal steps can be coloured in k different ways, while rise and fall steps remain uncoloured. Let m_n denote the number of k-coloured Motzkin paths of length n. We are interested in the enumeration of left factors (prefixes) of such paths according to the parity of their final height. More precisely, let $f_n^{(o)}$ (resp. $f_n^{(e)}$) be the number of prefixes of length n ending at odd (resp. even) height, and let f_n be the total number of prefixes of length n.

In this paper we consider a relation between k-coloured Motzkin paths and prefixes ending at odd height, namely

$$f_{n+1}^{(o)} = (n+1)m_n, \qquad \text{for } n \geq 0.$$

We provide two different proofs of this identity. The first one is analytic and is based on the functional equation satisfied by the generating function of k-coloured Motzkin paths. The second one is purely combinatorial and relies on a bijective transformation together with Raney's lemma [11] on cyclic shifts of integer sequences.

Besides giving two independent proofs of the same enumerative identity, the combinatorial approach also leads to a random generation algorithm for k-coloured Motzkin paths and a linear-time complexity analysis of it.

3 Enumeration of Motzkin prefixes

We give two different proofs of the following

Theorem 1. *For $n \geq 0$,*

$$f_{n+1}^{(o)} = (n+1)m_n.$$

Analytical proof (sketch)

It is not difficult to verify that

$$f_{n+1}^{(e)} = 2f_n^{(o)} + kf_n^{(e)},$$

and

$$f_{n+1}^{(o)} = kf_n^{(o)} + 2f_n^{(e)} - m_n.$$

with initial conditions $f_0^{(e)} = 1,\ f_0^{(o)} = 0,\ f_1^{(e)} = k,\ f_1^{(o)} = 1$.

We now define the generating functions

$$E(t) = \sum_{n\geq 0} f_n^{(e)} t^n, \qquad O(t) = \sum_{n\geq 0} f_n^{(o)} t^n, \qquad M(t) = \sum_{n\geq 0} m_n t^n.$$

From the above recurrence relations we obtain

$$\begin{cases} E(t) = 2t\,O(t) + kt\,E(t) + 1, \\ O(t) = kt\,O(t) + 2t\,E(t) - t\,M(t), \end{cases}$$

whose solution is given by

$$E(t) = \frac{2tO(t)+1}{1-kt},$$

and

$$O(t) = \frac{(1-kt)tM(t) - 2t}{4t^2 - (1-kt)^2}. \tag{1}$$

Moreover, we recall the functional equation satisfied by $M(t)$:

$$M(t) = 1 + ktM(t) + t^2 M(t)^2. \tag{2}$$

Setting $\overline{M}(t) = tM(t)$ and performing some straightforward manipulations, we obtain

$$t\frac{d}{dt}\overline{M}(t) = \frac{\overline{M}(t)}{1 - kt - 2t\overline{M}(t)}.$$

We have $f_{n+1}^{(o)} = (n+1)m_n$ if and only if $O(t) = t\frac{d}{dt}\overline{M}(t)$. This latter identity follows from equations (1) and (2).

Combinatorial proof (sketch)

Let P be a k-coloured Motzkin factor whose last point has odd height $h = 2i+1$, for $i \geq 1$. Then P can be factorized as

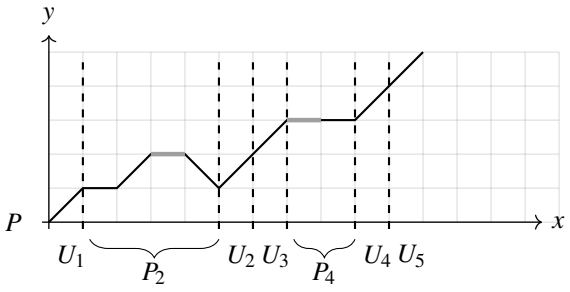
$$P = P_1 U_1 P_2 U_2 \dots P_{2i+1} U_{2i+1} P_{2i+2}$$

where P_ℓ is a k-coloured Motzkin path (possibly the empty path) and U_ℓ is a rise step. This decomposition highlights the rise steps that actually contribute to the final height of P.

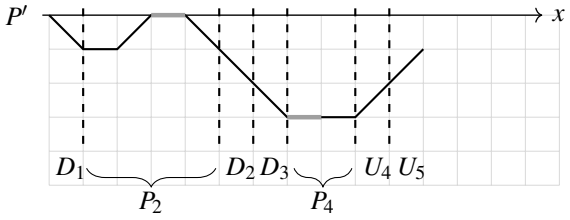
We perform a transformation Γ (based on a classical reflection argument) by turning the first $i+1$ rise steps U_ℓ into $i+1$ fall steps, denoted by D_ℓ, obtaining the path P':

$$P' = P_1 D_1 P_2 D_2 \dots P_{i+1} D_{i+1} P_{i+2} U_{i+2} \dots P_{2i+1} U_{2i+1} P_{2i+2}.$$

The following figure shows an example of the action of Γ on a 2-coloured Motzkin prefix of final height 5 (so that $i = 2$), together with its explicit decomposition (note that in the figure P' is weakly below the x-axis, but in general this is not the case).



$P_1 = P_3 = P_5 = P_6 = \varepsilon$ (empty path)



One can prove that this transformation defines a bijection between the set of k-coloured Motzkin prefixes of length n having odd final height and the set of paths of length n ending at ordinate -1.

We encode rise, fall, and horizontal steps (independently of their colour) by the integers 1, -1, and 0, respectively. Each path can then be read as a sequence of integers, and the one corresponding to P' clearly has sum equal to -1. From the Raney's Lemma [11] (which can be found also in [9, Chapter 7, p. 360]) it is possible to deduce the following:

Lemma 1. *If $(x_1,\ldots,x_{n+1})$ is a sequence of integers whose sum is -1, exactly one of its cyclic shifts*

$$(x_2,x_3,\ldots,x_{n+1},x_1),(x_3,x_4,\ldots,x_{n+1},x_1.x_2),\ldots,(x_{n+1},x_1,\ldots,x_{n-1},x_n)$$

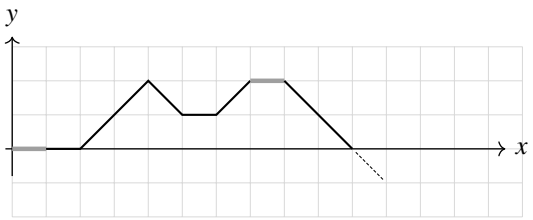
has all of its partial sums up to the n-th entry non-negative.

From the above Lemma there exists an unique cyclic shift of the encoding of P' which is a k-coloured Motzkin path except that for the last integer. Moreover, the identification of this unique cyclic shift can be performed in linear time: it suffices to locate the index immediately following the position where the first minimum of the partial sums is attained.

Since Γ is a bijection and each cyclic shift has final height equal to -1, then each of the $n+1$ cyclic shifts has a different corresponding k-coloured Motzkin prefix with odd final height. Therefore:

$$f_{n+1}^{(0)} = (n+1)m_n.$$

Continuing the example shown above, the encoding of P is $(1,0,1,0,-1,1,1,0,0,1,1)$ and the encoding of P' is $(-1,0,1,0,-1,-1,-1,0,0,1,1)$. Its unique cycle shift corresponding to the k-coloured Motzkin path is $(0,0,1,1,-1,0,1,0,-1,-1,-1)$ illustrated below:



Note that, each other cyclic shift different from $\mathscr{S}$ corresponds to a k-coloured Motzkin prefix different from P.

4 The random generation algorithm

From the previous combinatorial proof we obtain an algorithm to randomly generate a k-coloured Motzkin path:

1. use the Florentine algorithm [4] to randomly generate a k-coloured Motzkin prefix F of length $n+1$;
2. while F has even final height, return to step 1;
3. perform the Γ transformation on F;
4. perform the cyclic shifts of Lemma 1 and identify the unique Motzkin path of length n associated to the $n+1$ cyclic shifts of length $n+1$.

4.1 The complexity

The expected number of extracted characters (or calls to *random*) for generating a k-coloured Motzkin left factor of length n is [4]

$$2n - \frac{1}{2}\sqrt{(k+2)\pi(n+1)} + \frac{6+k}{4} + O(1).$$

Therefore, the first step can be performed in linear time. For the second step, establishing that the final height of the newly generated prefix F is clearly performed in constant time. Nevertheless, the generation of F having final odd height depends on the ratio $\frac{f_n^{(0)}}{f_n}$.

From Theorem 1 we have $\frac{f_n^{(0)}}{f_n} = \frac{nm_{n-1}}{f_n}$, and from the recursive construction of k-coloured Motzkin left factors we have $m_n = (k+2)f_n - f_{n+1}$. Thus,

$$\frac{f_n^{(0)}}{f_n} = \frac{n\big((k+2)f_{n-1} - f_n\big)}{f_n}.$$

Being $f(t) = \sum_{n\geq 0} f_n t^n = \frac{1}{2t}\left(\sqrt{\frac{1+t}{1-3t}} - 1\right)$, see for example [4], it is easy to prove (via the Darboux's method [7]) that

$$f_n \sim \frac{1}{\sqrt{k+2}} \frac{(k+2)^n}{\sqrt{\pi n}}.$$

Therefore,

$$\frac{f_n^{(0)}}{f_n} \sim \ldots \sim \frac{n}{\sqrt{n-1}}\left(\frac{1}{\sqrt{n}+\sqrt{n-1}}\right) \sim \frac{1}{2}.$$

In the following table, we show some numerical results for $k = 1$:

n	$f_n^{(0)}/f_n$
1	0.5
2	0.5
3	0.4653
4	0.4574
5	0.46875
6	0.47195
7	0.47600
8	0.47856
9	0.48081
10	0.48257
20	0.49096
30	0.49310
40	0.49539
50	0.49630

Therefore, the average computational cost of the first and second steps grows asymptotically as $4n$. The third and fourth steps can be performed in linear time. Then, the average complexity of the algorithm is linear in n.

5 Conclusions

The results herein contained are surely not up to date. Indeed better performing algorithms do exist in order to generate Motzkin prefixes [2] which could improve step 1 of our algorithm. Nevertheless the authors believe that these results give a new contribution to the field of the random generation of combinatorial objects.

References

[1] N. Alonso (2023): *Uniform random generations and rejection method(I) with binomial majorant*. *arXiv:2303.09338*, pp. 1–25, doi:10.48550/arXiv.2303.09338.

[2] A. Bacher (2018): *Improving the Florentine algorithms: recovering algorithms for Motzkin and Schröder paths*. *arXiv:1802.06030*, doi:10.48550/arXiv.1802.06030.

[3] E. Barcucci, S. Bilotta, E. Pergola, R. Pinzani & J. Succi (2016): *Cross-bifix-free sets generation via Motzkin paths*. *RAIRO Theor. Inform. Appl. (RAIRO:ITA)* 50, pp. 81–91, doi:10.1051/ita/2016008.

[4] E. Barcucci, R. Pinzani & R. Sprugnoli (1994): *The random generation of directed animals*. *Theor. Comput. Sci* 127(1–3), pp. 333–350, doi:10.1016/0304-3975(94)90046-9.

[5] E. Barcucci, R. Pinzani & R. Sprugnoli (1995): *The random generation of underdiagonal walks*. *Discrete Math.* 139, pp. 3–18, doi:10.1016/0012-365X(94)00121-X.

[6] C. Bean, A. Bernini, M. Cervetti & L. Ferrari (2022): *On the generating functions of pattern-avoiding Motzkin paths*. *J. Symbolic Comput.* 113, pp. 126–138, doi:10.1016/j.jsc.2022.02.006.

[7] G. Darboux (1878): *Sur l'approximation des fonctions de très-grands nombres et sur une classe étendue de développements en série*. *J. Math. Pures Appl.* 4, pp. 377–416.

[8] E. Deutsch (1999): *Dyck path enumeration*. *Discrete Math.* 204, pp. 13–33, doi:10.1016/S0012-365X(98)00371-9.

[9] R. L. Graham, D. E. Knuth & O. Patashnik (1994): *Concrete Mathematics*. Addison-Wesley, Reading, Massachusetts.

[10] T. Motzkin (1948): *Relations between hypersurface cross ratios, and a combinatorial formula for partitions of a polygon, for permanent preponderance, and for non-associative products*. *Bulletin of the American Mathematical Society* 54(4), pp. 352–360, doi:10.1090/S0002-9904-1948-09002-4.

[11] G. N. Raney (1960): *Functional composition patterns and power series reversion*. *Trans. Amer. Math. Soc* 94, pp. 441–451, doi:10.2307/1993433.

[12] R. P. Stanley (1999): *Enumerative Combinatorics, Volume 2*. Cambridge University Press, doi:10.1017/CBO9780511609589.

Fibonacci and Catalan Numbers Meet in Staircase Polyominoes

Jean-Luc Baril
LIB, Université Bourgogne Europe,
Dijon, France
barjl@u-bourgogne.fr

José L. Ramírez
Departamento de Matemáticas,
Universidad Nacional de Colombia
Bogotá, Colombia
jlramirezr@unal.edu.co

Samuel Ramírez
samramirezra@unal.edu.co

Diego Villamizar
Department of Mathematics,
Xavier University of Louisiana
New Orleans, LA 70125
dvillami@xula.edu

We study Fibonacci (staircase) polyominoes, a class of column-convex polyominoes whose lower boundary is a staircase with unit vertical steps. We derive multivariate generating functions that refine Turban's Fibonacci-number enumeration by tracking additional perimeter and area parameters. The proofs use a catalytic functional equation and, in a perimeter specialization, the kernel method, leading to explicit closed forms and Catalan-number coefficient formulas.

1 Introduction

The enumeration of polyomino classes defined by convexity and directionality constraints is a classical topic in enumerative combinatorics; see [5] and the references therein. A *polyomino* is a finite edge-connected union of unit squares in $\mathbb{Z}^2$.

Among the many families studied with respect to parameters such as area and perimeter [2, 3, 6], Turban [8] introduced a particularly natural class of *Fibonacci polyominoes*. These are column-convex polyominoes whose lower boundary is a staircase path with unit vertical steps (see Figure 1). Turban proved that the number of such polyominoes of area n equals the Fibonacci number F_n, and considered variants allowing staircase steps of arbitrary heights [8, 9].

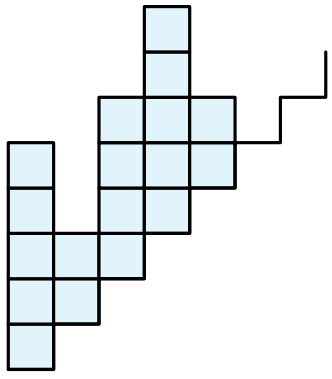


Figure 1: Fibonacci polyomino of area 18.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 16–20, doi:10.4204/EPTCS.445.3

© J.-L. Baril & J. L. Ramírez & S. Ramírez & D. Villamizar
This work is licensed under the
Creative Commons Attribution License.

In this paper we refine Turban's enumeration by deriving multivariate generating functions that track additional parameters.

2 Perimeter and area

For a Fibonacci polyomino P, let $\texttt{area}(P)$ denote its area (number of cells). Write $\texttt{sper}(P)$ for the semiperimeter, i.e., half of the number of unit edges on the boundary of P. We further split it into the horizontal and vertical contributions: $\texttt{xper}(P)$ is half of the number of horizontal boundary edges (or number of columns of P) and $\texttt{yper}(P)$ is half of the number of vertical boundary edges. Thus $\texttt{sper}(P) = \texttt{xper}(P) + \texttt{yper}(P)$.

Let $\boldsymbol{F}$ be the class of all Fibonacci polyominoes. For $m,n,k \in \mathbb{Z}_{>0}$, let $\boldsymbol{F}_{m,n,k}$ be the subset consisting of those P with $2m$ horizontal boundary edges, $2n$ vertical boundary edges, and area k, with $n \geq m$ and $k \geq 2m-1$. Equivalently, $\boldsymbol{F} = \bigcup_{m,n,k\geq 1} \boldsymbol{F}_{m,n,k}$. We encode these statistics by the trivariate generating function

$$F(x,y,q) = \sum_{P\in\boldsymbol{F}} x^{\texttt{xper}(P)} y^{\texttt{yper}(P)} q^{\texttt{area}(P)} = \sum_{m,n,k\geq 1} \left(\sum_{P\in\boldsymbol{F}_{m,n,k}} 1 \right) x^m y^n q^k.$$

To keep track of the height of the first column, let $F_h(x,y,q)$ be the generating function of Fibonacci polyominoes whose first column has height exactly $h \geq 1$, so that $F(x,y,q) = \sum_{h\geq 1} F_h(x,y,q)$. Introducing a catalytic variable s marking this height, we set

$$F(x,y,q;s) = \sum_{h\geq 1} F_h(x,y,q)\, s^h, \qquad \text{so that} \qquad F(x,y,q;1) = F(x,y,q).$$

For brevity, write $F(s) := F(x,y,q;s)$ and $F(1) := F(x,y,q)$.

A decomposition according to the possible configurations of the final columns yields the functional equation

$$F(s) = \frac{xyqs}{1-yqs} + \frac{xy(qs)^2}{1-qs} F(1) + \left(\frac{x(yqs)^2}{1-yqs} - \frac{xy(qs)^2}{1-qs} \right) F(sq). \tag{1}$$

Iterating (1) (with the convention that an empty product equals 1) and then setting $s=1$ gives an explicit closed form for $F(x,y,q)$.

Theorem 2.1. *The generating function of nonempty Fibonacci polyominoes according to horizontal semiperimeter, vertical semiperimeter, and area is*

$$F(x,y,q) = \frac{\sum_{\ell\geq 0} \left(\frac{(xy)^{\ell+1} q^{(\ell+1)^2} (y-1)^\ell}{1-yq^{\ell+1}} \prod_{i=0}^{\ell-1} \frac{1}{(1-yq^{i+1})(1-q^{i+1})} \right)}{1 - \sum_{\ell\geq 0} \left(\frac{(xy)^{\ell+1} q^{(\ell+1)(\ell+2)} (y-1)^\ell}{1-q^{\ell+1}} \prod_{i=0}^{\ell-1} \frac{1}{(1-yq^{i+1})(1-q^{i+1})} \right)}.$$

As a consistency check, specializing (1) at $s = x = y = 1$ gives, with $F(q) := F(1,1,q)$,

$$F(q) = \frac{q}{1-q} + \frac{q^2}{1-q} F(q), \qquad \text{hence} \qquad F(q) = \frac{q}{1-q-q^2}.$$

Therefore the coefficient of q^n is the Fibonacci number F_n, recovering Turban's enumeration by area.

2.1 Perimeter

We now specialize to perimeter enumeration by discarding the area statistic, that is, by setting $q = 1$ in (1). In this specialization, the series $F(1) = F(x, y, 1)$ is precisely the bivariate generating function that counts Fibonacci polyominoes by horizontal and vertical semiperimeter.

Setting $q = 1$ in (1) gives a linear functional equation in $F(s)$:

$$\left(1 - \frac{x(sy)^2}{1-sy} + \frac{xys^2}{1-s}\right) F(s) = \frac{xys}{1-ys} + \frac{xys^2}{1-s} F(1).$$

Let $p(m,n)$ be the number of Fibonacci polyominoes whose boundary has $2m$ horizontal edges and $2n$ vertical edges, and set

$$P(x,y) := \sum_{m,n \geq 1} p(m,n) x^m y^n.$$

By definition, $P(x,y) = F(1)$. We compute it via the kernel method [1, 7]. Denote the kernel by

$$K(s) := 1 - \frac{x(sy)^2}{1-sy} + \frac{xys^2}{1-s}.$$

Choosing $s = s_0(x,y)$ so that $K(s_0) = 0$ cancels the left-hand side and yields an explicit expression for $F(1)$ from the right-hand side. We take the *small* root (analytic at $x = 0$), namely

$$s_0 = \frac{1 + y + \sqrt{(1-y)(1-y-4xy)}}{2y\big(1 + x(1-y)\big)}.$$

Theorem 2.2. *The generating function for nonempty Fibonacci polyominoes according to horizontal and vertical semiperimeter is*

$$P(x,y) = \frac{1 + x - y - 3xy - (1+x)\sqrt{(1-y)(1-y-4xy)}}{2x}.$$

The first few terms of the expansion of $P(x,y)$ in powers of y are

$$P(x,y) = xy + \left(x^2 + x\right) y^2 + \left(2x^3 + 3x^2 + x\right) y^3 + \left(5x^4 + \mathbf{9x^3} + 5x^2 + x\right) y^4 + \left(14x^5 + 29x^4 + 21x^3 + 7x^2 + x\right) y^5 + O\left(y^6\right).$$

The bold coefficients in the above expansion correspond to the Fibonacci polyominoes shown in Figure 2.

We denote by $C_n = \frac{1}{n+1}\binom{2n}{n}$ the nth Catalan number, and let

$$C(z) = \sum_{n \geq 0} C_n z^n = \frac{1 - \sqrt{1-4z}}{2z}$$

be its generating function. From the previous theorem we obtain an explicit expression for the coefficients of $P(x,y)$ in terms of Catalan numbers.

Theorem 2.3. *For $m \geq 1$ and $n \geq 2$, we have*

$$p(m,n) = [x^m y^n] P(x,y) = C_m \binom{n-2}{m-1} + C_{m-1} \binom{n-2}{m-2}.$$

In particular, for $n \geq 2$, $p(n,n) = C_{n-1}$.

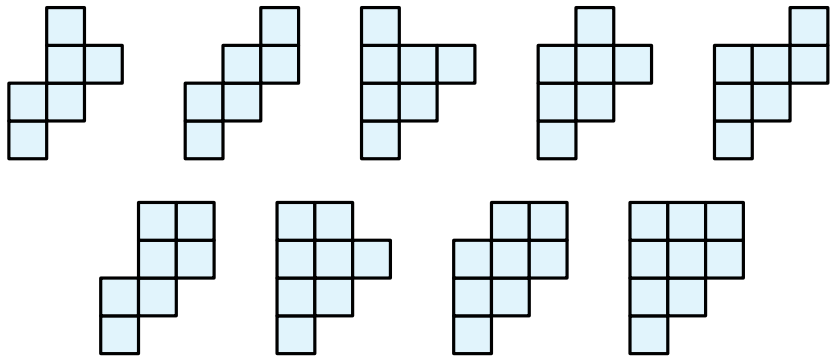


Figure 2: Fibonacci polyominoes with 6 horizontal steps and 8 vertical steps.

Notice that for the case $n = m$ we can give a combinatorial argument. Let P be a Fibonacci polyomino whose boundary has exactly $2n$ horizontal and $2n$ vertical edges. This implies that P has n columns. Observe that the bottom cell of each column produces (on its right side) a vertical edge lying into the boundary. So, P has exactly n other vertical edges in the boundary, which implies that the number of cells in the ith column is at most $n-i+1$. Moreover, this also implies that there is no descent in the polyomino, which means that the number of cells in the ith column is at most equal to the number plus one of cells in the $(i+1)$th column. Let a_i, $1 \leq i \leq n$, be the number of cells in the column i. Then we have $2 \leq a_i$ for $1 \leq i \leq n-1$, $a_n = 1$, and $a_i - 2 \leq a_{i+1} - 1$, which proves that the word $b_1 b_2 \cdots b_{n-1} = (a_{n-1}-2)(a_{n-2}-2)\cdots(a_1-2)$ is a Catalan word, that is $0 \leq b_i \leq n-1$ and $b_{i+1} \leq b_i + 1$. It is well known that these sequences are enumerated by the $(n-1)$th Catalan number (cf. [4]).

For example, the Catalan words of length $n = 3$ are $000, 001, 010, 011$, and 012. The corresponding Fibonacci polyominoes are shown in Figure 3.

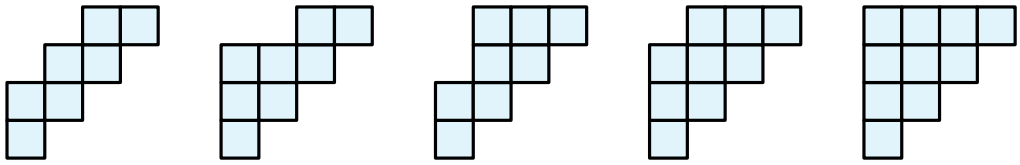


Figure 3: Fibonacci polyominoes with 8 horizontal steps and 8 vertical steps.

Notice that in the case $n = m$, n is also the number of rows and the polyominoes are parallelogram polyominoes having the same number of rows and columns, so the upper boundary is an elevated Dyck path of length $2n$ and, consequently, they are enumerated by the $(n-1)$-th Catalan number.

References

[1] Cyril Banderier, Mireille Bousquet-Mélou, Alain Denise, Philippe Flajolet, Danièle Gardy & Dominique Gouyou-Beauchamps (2002): *Generating functions for generating trees*. *Discrete Mathematics* 246(1–3), pp. 29–55, doi:10.1016/S0012-365X(01)00250-3.

[2] Aubrey Blecher, Charlotte Brennan, Arnold Knopfmacher & Toufik Mansour (2017): *The perimeter of words*. *Discrete Mathematics* 340(10), pp. 2456–2465, doi:10.1016/j.disc.2017.06.003.

[3] Mireille Bousquet-Mélou & Andrew Rechnitzer (2003): *The site-perimeter of bargraphs*. *Advances in Applied Mathematics* 31(1), pp. 86–112, doi:10.1016/S0196-8858(02)00553-5.

[4] David Callan, Toufik Mansour & José L. Ramírez (2021): *Statistics on Bargraphs of Catalan Words*. *Journal of Automata, Languages and Combinatorics* 26(3–4), pp. 177–196, doi:10.25596/jalc-2021-177.

[5] Anthony J. Guttmann, editor (2009): *Polygons, Polyominoes and Polycubes*. *Lecture Notes in Physics* 775, Springer, Dordrecht, doi:10.1007/978-1-4020-9927-4.

[6] Toufik Mansour & Armend Sh. Shabani (2019): *Enumerations on bargraphs*. *Discrete Mathematics Letters* 2, pp. 65–94. Available at https://www.dmlett.com/archive/DML19_v2_p.65_94.pdf.

[7] Helmut Prodinger (2004): *The kernel method: a collection of examples*. *Séminaire Lotharingien de Combinatoire* 50, p. B50f. Available at https://www.mat.univie.ac.at/~slc/wpapers/s50proding.html. 19 pp.

[8] Loïc Turban (2000): *Lattice animals on a staircase and Fibonacci numbers*. *Journal of Physics A: Mathematical and General* 33(13), pp. 2587–2595, doi:10.1088/0305-4470/33/13/311.

[9] Loïc Turban (2006): *Lattice animals on a staircase and generalized Fibonacci numbers*. doi:10.48550/arXiv.cond-mat/0106595. arXiv:cond-mat/0106595.

Computing Joins in the Weak Order of Type B Coxeter Groups: an Algorithmic Approach

Riccardo Biagioli
Dipartimento di Matematica
Università di Bologna, Italy
riccardo.biagioli2@unibo.it

Lorenzo Perrone
Dipartimento di Matematica
Università di Bologna, Italy
lorenzo.perrone8@unibo.it

We present an algorithm for computing the join of two elements in the weak order of the Coxeter group of type B. This extends Markowsky's algorithm for computing joins of standard permutations to signed permutations, and allows us to confirm a conjecture of Dyer concerning a geometric interpretation of these joins.

1 Introduction

In a very influential paper, Dyer [4] proposed several conjectures regarding the weak order of Coxeter groups. One of these conjectures, which will be discussed in detail in the following sections, has been the primary motivation for the present study.

It is well known that the Coxeter system of type B_n, whose Coxeter graph is displayed in Figure 1 admits a combinatorial interpretation as the group S_n^B of signed permutations, namely the set of bijections $\sigma : [\pm n] \to [\pm n]$ satisfying the symmetric property $\sigma(-a) = -\sigma(a)$ for every $a \in [n]$, where we denote $[n] = \{1, \dots, n\}$ and $[\pm n] = \{-n, \dots, -1, 1, \dots n\}$. We refer to [2, §8.1] for any undefined notation.

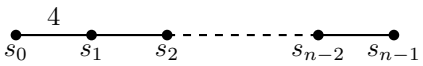


Figure 1: The Coxeter graph of type B_n.

To avoid the use of minus signs, we write $\bar{i}$ instead of $-i$. Let $i \neq \pm j \in [\pm n]$; we denote by $(i\ j) \in S_n^B$ the *transposition* that swaps i with j and $\bar{i}$ with $\bar{j}$, and by $\left(i\ \bar{i}\right)$ the one that exchanges i with $\bar{i}$. Note that, naturally, $(i\ j)$, $(j\ i)$, $\left(\bar{i}\ \bar{j}\right)$ and $\left(\bar{j}\ \bar{i}\right)$ all represent the same signed permutation. Through this combinatorial interpretation, we take as set of Coxeter generators for S_n^B the set $S_B = \{s_0 = \left(1\ \bar{1}\right), s_1 = (1\ 2), \dots, s_{n-1} = (n-1\ n)\}$.

In what follows, we will use both the *complete notation* of a signed permutation

$$\sigma = \sigma(\bar{n}) \cdots \sigma(\bar{1})\sigma(1) \cdots \sigma(n)$$

and its *window notation* $\sigma = \sigma(1)\sigma(2) \cdots \sigma(n)$ which exploits the symmetry $\sigma(\bar{i}) = \overline{\sigma(i)}$. We will specify which notation we use only when it is strictly necessary.

Recall that, given a Coxeter system (W,S), its set of reflections is $T = \bigcup_{w \in W} wSw^{-1}$ and the *left-reflection set* of an element $w \in W$ is $T_L(w) = \{t \in T \mid \ell(tw) < \ell(w)\}$. The *(right) weak order* on (W,S) is a partial order defined by the prefix property: for any $u, v \in W$, we have $u \leq_R v$ if and only if a reduced expression for u is a prefix of a reduced expression for v. It can also be characterized in terms of left-reflection sets, namely

$$u \leq_R v \text{ if and only if } T_L(u) \subseteq T_L(v). \tag{1}$$

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 21–28, doi:10.4204/EPTCS.445.4

© R. Biagioli & L. Perrone
This work is licensed under the
Creative Commons Attribution License.

The poset $(W, \leq_R)$ is a meet-semilattice, and in particular it is a lattice whenever the group W is finite. On the other hand, it is never a lattice when W is infinite. For further details on $(W, \leq_R)$, we refer the reader to [2, Chapter 3].

In the combinatorial description (S_n^B, S_B), the set of reflections corresponds to the set of transpositions $T = \{(i\ j) \mid i \neq j \in [\pm n]\}$ introduced above. Moreover, in this interpretation, it is useful to introduce the following well known statistics of a signed permutation $\sigma \in S_n^B$:

- $\mathrm{Inv}(\sigma) = \{(i,j) \in [n] \times [n] \mid i < j, \sigma(i) > \sigma(j)\}$, the set of *inversions* of σ;
- $\mathrm{Neg}(\sigma) = \{i \in [n] \mid \sigma(i) < 0\}$, the set of *negative entries* of σ;
- $\mathrm{Nsp}(\sigma) = \{\{i,j\} \subseteq [n] \mid i \neq j, \sigma(i) + \sigma(j) < 0\}$, the set of *negative sum pairs* of σ.

The computation of these statistics provides a direct formula for the Coxeter length:

$$\ell(\sigma) = |\mathrm{Inv}(\sigma)| + |\mathrm{Neg}(\sigma)| + |\mathrm{Nsp}(\sigma)|.$$

This identity reflects a decomposition of the left-reflection set of σ into three disjoint subsets:

$$T_L(\sigma) = \left\{(i\ j) \mid (i,j) \in \mathrm{Inv}(\sigma^{-1})\right\} \sqcup \left\{\left(i\ \bar{i}\right) \mid i \in \mathrm{Neg}(\sigma^{-1})\right\} \sqcup \left\{\left(i\ \bar{j}\right) \mid \{i,j\} \in \mathrm{Nsp}(\sigma^{-1})\right\}. \quad (2)$$

A proof of last equality can be found in [2, Chapter 8] and [8]. It is important to observe that the left reflection set $T_L(w)$ uniquely identifies the element w. Therefore, to determine the join of two elements in $(W, \leq_R)$, it suffices to compute its left reflection set. For instance, Theorem 10-3.25 of [6] gives two descriptions of the left reflection set of the join of two elements in terms of closure operations. As we will see below, the conjecture that motivated our work is formulated in a similar way.

In [4], Dyer introduced the extended weak order, a partial order generalizing the weak order, and conjectured that it is a lattice even when the underlying Coxeter group is infinite. In the same paper, he formulated a conjectural description of the join of two elements in this poset. It is not yet known whether this conjecture holds even for finite Coxeter systems. It has been proved for Coxeter groups of types I and A in [1], and verified for the exceptional types F_4, H_3, and H_4 using SageMath [7]. Establishing this conjecture for Coxeter systems of type B was the main motivation for the present paper. For the original statement, we refer the reader to [4, §2.8]. Here, we give an equivalent formulation for finite Coxeter groups in terms of the *Bruhat preclosure* recently introduced by Dermenjian in [3, §3].

For each Coxeter system (W,S), we recall that the Bruhat graph $B(W)$ is the directed graph having W as vertex set, and where for any $u, v \in W$ there is an arrow $u \xrightarrow{t} v$ whenever $v = tu$ and $\ell(u) < \ell(v)$.

Definition 1 (A-Bruhat path) *Let W be a Coxeter group; consider the Bruhat graph $B(W)$ associated with W and $A \subseteq T$. An A-*Bruhat path *is any directed path in $B(W)$ starting from the identity element e of W, whose edges are labeled by elements in the set A. If a set $A \subseteq T$ can be written as a union of two left-reflection sets, i.e. $A = T_L(u) \cup T_L(v)$ for some $u, v \in W$, then we use the expression (u,v)-Bruhat paths instead of A-Bruhat paths.*

Definition 2 (Bruhat preclosure) *Given a set of reflection $A \subseteq T$; its Bruhat preclosure is*

$$\overline{A} = \{t \in T \mid \textit{there is an } A\textit{-Bruhat path from } e \textit{ to } t\}.$$

If $A = \overline{A}$ we say it is preclosed.

Observe that this operation is not a closure operator: in general, $\overline{\overline{A}} \neq \overline{A}$. Indeed, counterexamples were found by Dermenjian in Coxeter groups of types H_3 and F_4; see [3, §3]. Consequently, the following conjecture is of a rather different nature from Theorem 10-3.25, where the operations involved are proper closure operators.

Conjecture 3 *Let W be a finite Coxeter group and $u, v \in W$. Then*

$$T_L(u \vee_R v) = \overline{T_L(u) \cup T_L(v)}.$$

In general, computing the join of two elements in the poset $(W, \leq_R)$ can be quite difficult; see for instance Figure 2. Conjecture 3 states that the left-reflection set of the join $u \vee_R v$ can be determined by considering all reflections reached by (u,v)-Bruhat paths.

In Section 2, we present the main result in this extended abstract which is an algorithmic way to compute the join of two signed permutations starting from their complete notation. This is a generalization of the algorithm presented in [5] for the symmetric group. This algorithm will serve as a key tool in a forthcoming article, where we establish the validity of Dyer's conjecture for Coxeter groups of type B.

Observe that, thanks to equation (2), the weak order on S_n^B can be characterized as:

$$\sigma \leq_R \tau \iff \mathrm{Inv}(\sigma^{-1}) \subseteq \mathrm{Inv}(\tau^{-1}),\ \mathrm{Neg}(\sigma^{-1}) \subseteq \mathrm{Neg}(\tau^{-1}),\ \mathrm{Nsp}(\sigma^{-1}) \subseteq \mathrm{Nsp}(\tau^{-1}). \tag{3}$$

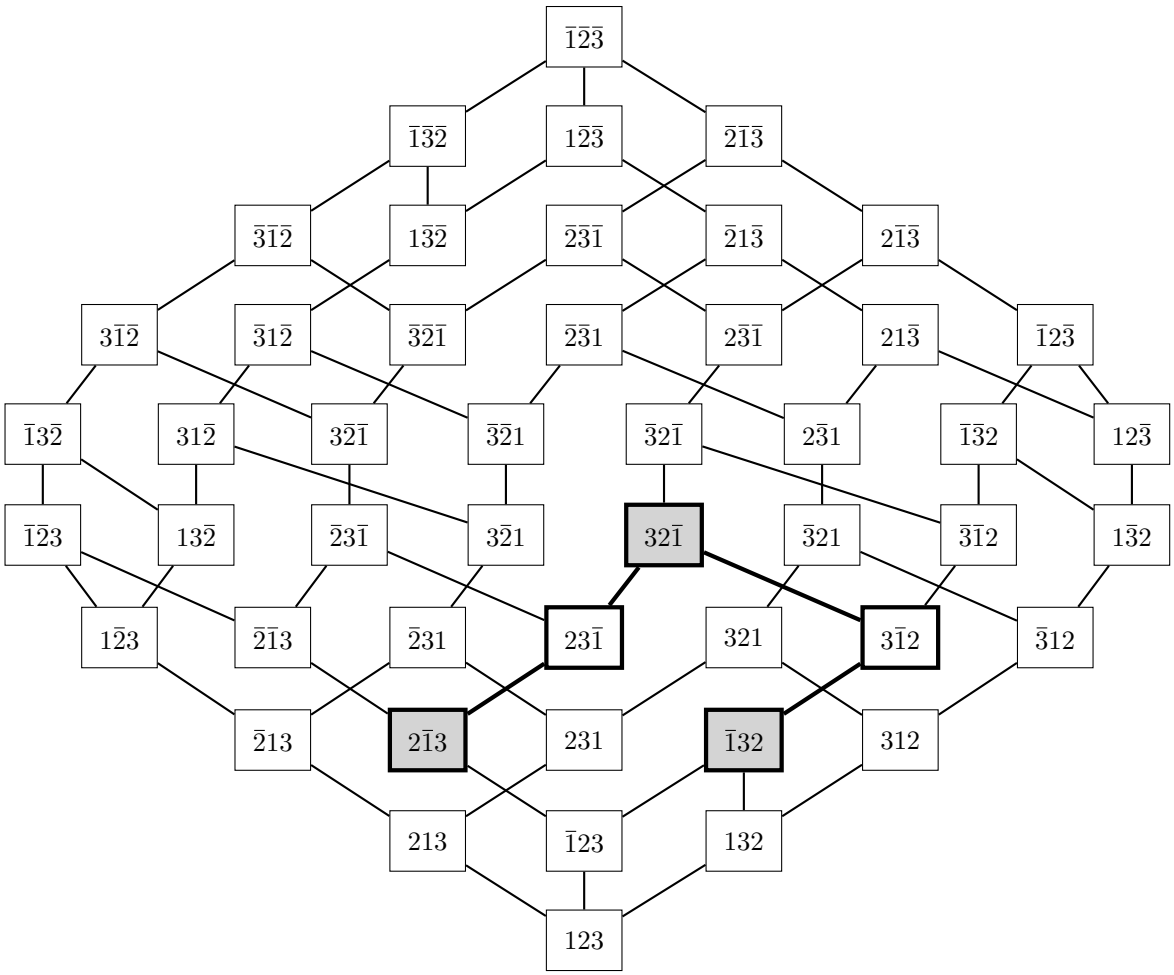


Figure 2: The Hasse diagram of $(S_3^B, \leq_R)$, where signed permutations are in window notation and the join operation of $2\bar{1}3$ and $\bar{1}32$ is highlighted.

2 Algorithm

In this section, we describe an algorithm to compute the join of two elements in $(S_n^B, \leq_R)$. Note that when restricted to standard permutations this coincides with the algorithm in [5, §2]; therefore it can be considered a generalization of it. For simplicity and readability in the definition of the algorithm, we use the complete notation for the two input signed permutations.

Definition 4 (n-cut) *The n-*cut *of $\sigma \in S_n^B$ is the pair (B_σ, B_σ^*), where B_σ and B_σ^* are the strings made of all entries different from $\bar{n}$, respectively, on the left and on the right of the entry n in the complete notation of σ. More precisely, if $k := |\sigma^{-1}(n)|$, we have:*

1. *if $\sigma^{-1}(n) < 0$, then*

$$B_\sigma = \sigma(\bar{n})\sigma(\overline{n-1})\cdots\sigma(\overline{k+1}),$$
$$B_\sigma^* = \sigma(\overline{k-1})\cdots\sigma(\bar{1})\sigma(1)\cdots\sigma(k-1)\sigma(k+1)\cdots\sigma(n);$$

2. *if $\sigma^{-1}(n) > 0$, then*

$$B_\sigma = \sigma(\bar{n})\sigma(\overline{n-1})\cdots\sigma(\overline{k+1})\sigma(\overline{k-1})\cdots\sigma(\bar{1})\sigma(1)\cdots\sigma(k-1),$$
$$B_\sigma^* = \sigma(k+1)\cdots\sigma(n).$$

Moreover, in case 1, we write $B_\sigma^* = B_\sigma^{*'}B_\sigma^{*''}$, where

$$B_\sigma^{*'} = \sigma(\overline{k-1})\cdots\sigma(\bar{1})\sigma(1)\cdots\sigma(k-1), \qquad B_\sigma^{*''} = \sigma(k+1)\cdots\sigma(n),$$

while in case 2, we write $B_\sigma = B'_\sigma B''_\sigma$, where

$$B'_\sigma = \sigma(\bar{n})\sigma(\overline{n-1})\cdots\sigma(\overline{k+1}), \qquad B''_\sigma = \sigma(\overline{k-1})\cdots\sigma(\bar{1})\sigma(1)\cdots\sigma(k-1).$$

Thus, in case 1, we express the complete notation of the signed permutation as $\sigma = B_\sigma n B_\sigma^{*'}\bar{n}B_\sigma^{*''}$, while in case 2, we can write $\sigma = B'_\sigma \bar{n} B''_\sigma n B_\sigma^*$.

Example 5 *Consider $\sigma = \bar{3}4\bar{1}\bar{2}21\bar{4}3$ and $\tau = 2\bar{1}\bar{4}3\bar{3}41\bar{2}$ in S_4^B. Their 4-cuts are*

$$(B_\sigma, B_\sigma^*) = (\bar{3}, \bar{1}\bar{2}213), \qquad (B_\tau, B_\tau^*) = (2\bar{1}3\bar{3}, 1\bar{2}).$$

Furthermore, we have $\sigma^{-1}(4) < 0$, so, $B_\sigma^{*'} = \bar{1}\bar{2}21$ and $B_\sigma^{*''} = 3$; while $\tau^{-1}(4) > 0$, so, $B'_\tau = 2\bar{1}$ and $B''_\tau = 3\bar{3}$. Hence, $\sigma = B_\sigma 4 B_\sigma^{*''}\bar{4}B_\sigma^{*'}$ and $\tau = B'_\tau \bar{4} B''_\tau 4 B_\tau^*$.

Remark 6 *Note that the n-cut of a signed permutation $\sigma \in S_n^B$ determines it uniquely except when $\sigma^{-1}(n) = \pm 1$, in which case the sign of the entry $\pm n$ is not encoded in the cut. For instance, $\sigma = 4\bar{1}3\bar{2}$ and $\tau = \bar{4}\bar{1}3\bar{2}$ share the same 4-cut, but they differ in the sign of the entry 4.*

We define the map $\rho : S_n^B \to S_{n-1}^B$ as $\rho(\sigma) := B_\sigma B_\sigma^* \in S_{n-1}^B$. For brevity, we write ρ_σ instead of $\rho(\sigma)$. In particular, for $k \in [n-1]$

$$\rho_\sigma(k) = \begin{cases} \sigma(k), & \text{if } 0 < k < |\sigma^{-1}(n)|, \\ \sigma(k+1), & \text{if } |\sigma^{-1}(n)| \leq k \leq n-1. \end{cases}$$

To avoid any misunderstanding, by ρ_σ^{-1} we always denote the inverse of ρ_σ in the group S_{n-1}^B; moreover, observe that $\rho_\sigma^{-1} \neq \rho_{\sigma^{-1}}$, for instance, if $\sigma = 4213$, then $\rho_\sigma^{-1} = 213 \neq 321 = \rho_{\sigma^{-1}}$.

The following lemma is fundamental for understanding the map ρ; specifically, it details how the statistics of a signed permutation are transformed under its application.

Lemma 7 *Let $\sigma \in S_n^B$; then*

$$\begin{aligned}\mathrm{Inv}(\rho_\sigma^{-1}) &= \mathrm{Inv}(\sigma^{-1}) \setminus \{(k,n) \mid k \in [n-1]\};\\ \mathrm{Neg}(\rho_\sigma^{-1}) &= \mathrm{Neg}(\sigma^{-1}) \setminus \{n\};\\ \mathrm{Nsp}(\rho_\sigma^{-1}) &= \mathrm{Nsp}(\sigma^{-1}) \setminus \{\{k,n\} \mid k \in [n-1]\}.\end{aligned}$$

Example 8 *Consider $\sigma = 3\bar{6}15\bar{4}2 \in S_6^B$. Then $\sigma^{-1} = 36\bar{1}5\bar{4}\bar{2}$, so we can compute*

$$\begin{aligned}\mathrm{Inv}(\sigma^{-1}) &= \{(1,3),(1,4),(1,6),(2,3),(2,4),(2,5),(2,6),(3,4),(3,6),(5,6)\},\\ \mathrm{Neg}(\sigma^{-1}) &= \{4,6\},\\ \mathrm{Nsp}(\sigma^{-1}) &= \{\{1,4\},\{3,4\},\{3,6\},\{4,5\},\{4,6\}\}.\end{aligned}$$

Moreover, $\rho_\sigma = 31\bar{5}\bar{4}2$ and $\rho_\sigma^{-1} = 25\bar{1}\bar{4}3$. Therefore, as stated in Lemma 7, we have

$$\begin{aligned}\mathrm{Inv}(\rho_\sigma^{-1}) &= \{(1,3),(1,4),(2,3),(2,4),(2,5),(3,4)\},\\ \mathrm{Neg}(\rho_\sigma^{-1}) &= \{4\},\\ \mathrm{Nsp}(\rho_\sigma^{-1}) &= \{\{1,4\},\{3,4\},\{4,5\}\}.\end{aligned}$$

For simplicity, we identify strings with their underlying sets of elements. In particular, given two strings $B = b_1 b_2 \cdots b_k$ and $B' = b'_1 b'_2 \cdots b'_h$, we write $B \subseteq B'$ meaning that $\{b_1,\dots,b_k\} \subseteq \{b'_1,\dots,b'_h\}$. Similarly, $B \cup B'$ denotes the set $\{b_1,\dots,b_k\} \cup \{b'_1,\dots,b'_h\}$.

The next proposition highlights some properties of the n-cut of a signed permutation and the map ρ, in connection with the weak order. These results will be crucial in the proof of Theorem 13.

Proposition 9 *Let $\sigma, \tau \in S_n^B$ and suppose $\sigma \leq_R \tau$ in $(S_n^B, \leq_R)$; then the following hold:*

1. *$B_\sigma^* \subseteq B_\tau^*$;*
2. *$\rho_\sigma \leq_R \rho_\tau$ in $(S_{n-1}^B, \leq_R)$; that is, the map ρ is a surjective morphism of posets.*

The converse implication of the first statement in Proposition 9 is not true in general. Indeed, if we consider $\sigma = 34\bar{1}2$ and $\tau = 1\bar{4}23$, with 4-cuts $(B_\sigma, B_\sigma^*) = (\bar{2}1\bar{3}3, \bar{1}2)$ and $(B_\tau, B_\tau^*) = (\bar{2}\bar{3}, \bar{1}123)$, then $B_\sigma^* \subseteq B_\tau^*$, but $\sigma \not\leq_R \tau$, since, for example, $1 \in \mathrm{Neg}(\sigma^{-1}) \setminus \mathrm{Neg}(\tau^{-1})$.

In the following example we collect some computations relative to multiple applications of the map ρ. It is clear that if $\sigma \in S_n^B$ then $\rho_\sigma^i \in S_{n-i}^B$. The symbol ε denotes the empty string.

Example 10 *Consider $\sigma = \bar{4}\bar{1}5\bar{2}3, \tau = \bar{4}5\bar{1}3\bar{2} \in S_5^B$; one can check that $\sigma \leq_R \tau$, for instance using* (3)*. It is not hard to verify the validity of the statements of Proposition 9 in each row of the table.*

i	ρ_σ^i	ρ_τ^i	$(B_{\rho_\sigma^i}, B^*_{\rho_\sigma^i})$	$(B_{\rho_\tau^i}, B^*_{\rho_\tau^i})$
0	$\bar{3}2\bar{5}14\bar{4}\bar{1}5\bar{2}3$	$2\bar{3}1\bar{5}44\bar{5}\bar{1}3\bar{2}$	$(\bar{3}214\bar{4}\bar{1}, \bar{2}3)$	$(2\bar{3}14\bar{4}, \bar{1}3\bar{2})$
1	$\bar{3}214\bar{4}\bar{1}\bar{2}3$	$2\bar{3}14\bar{4}\bar{1}3\bar{2}$	$(\bar{3}21, \bar{1}\bar{2}3)$	$(2\bar{3}1, \bar{1}3\bar{2})$
2	$\bar{3}21\bar{1}\bar{2}3$	$2\bar{3}1\bar{1}3\bar{2}$	$(21\bar{1}\bar{2}, \varepsilon)$	$(21\bar{1}, \bar{2})$
3	$21\bar{1}\bar{2}$	$21\bar{1}\bar{2}$	$(\varepsilon, 1\bar{1})$	$(\varepsilon, 1\bar{1})$
4	$1\bar{1}$	$1\bar{1}$		

We now describe an algorithm that, given $\sigma, \tau \in S_n^B$, produces another signed permutation $J(\sigma,\tau)$. In the following description, we use the complete notation for signed permutations.

Given $\sigma \in S_n^B$, for any $2 \leq i \leq n$, we define

$$H_i^\sigma = \{a \in [\pm(i-1)] \mid \sigma^{-1}(a) > \sigma^{-1}(i)\}.$$

Observe that the underlying set of the string $B^*_{\rho_\sigma^{n-i}}$ coincides with H_i^σ.

Algorithm 1

Consider $\sigma, \tau \in S_n^B$.

1. Set
$$\pi_1 = \begin{cases} \bar{1}1, & \text{if } \sigma^{-1}(1) > 0 \text{ and } \tau^{-1}(1) > 0, \\ 1\bar{1}, & \text{otherwise.} \end{cases}$$
2. For each $i \in \{2,3,\ldots,n\}$:
   (a) Compute H_i^σ and H_i^τ;
   (b) Define C^*_{i-1} as the shortest suffix of π_{i-1} such that
   $$H_i^\sigma \cup H_i^\tau \subseteq C^*_{i-1}$$
   and call C_{i-1} the string such that $\pi_{i-1} = C_{i-1}C^*_{i-1}$.
   (c) Define π_i by placing i in the complete notation of π_{i-1} (consequently, $\bar{i}$ is placed in the opposite entry) as follows:
   - if $|C^*_{i-1}| = 0$, place i on the right of the last entry of π_{i-1};
   - if $|C^*_{i-1}| = i-1$, place i immediately on the left of C^*_{i-1} in case $\sigma^{-1}(i) > 0$ and $\tau^{-1}(i) > 0$; otherwise, place i immediately on the right of C_{i-1};
   - if $|C^*_{i-1}| \neq 0$ and $|C^*_{i-1}| \neq i-1$, place i immediately on the left of the first entry of C^*_{i-1}.
3. Set $J(\sigma,\tau) := \pi_n$.

Remark 11 *The definition of π_i in Algorithm 1 can be visualized as follows:*

$$\pi_i = \begin{cases} C_{i-1}iC^{*'}_{i-1}\bar{i}C^{*''}_{i-1}, & \text{if } |C^*_{i-1}| \geq i, \\ C'_{i-1}\bar{i}C''_{i-1}iC^*_{i-1}, & \text{if } |C^*_{i-1}| \leq i-2, \\ C_{i-1}\bar{i}iC^*_{i-1}, & \text{if } |C^*_{i-1}| = i-1 \text{ and } \sigma^{-1}(i), \tau^{-1}(i) > 0, \\ C_{i-1}i\bar{i}C^*_{i-1}, & \text{otherwise,} \end{cases}$$

where

$$C_{i-1} = C'_{i-1}C''_{i-1} \text{ with } |C'_{i-1}| = |C^*_{i-1}|$$

and

$$C^*_{i-1} = C^{*'}_{i-1}C^{*''}_{i-1} \text{ with } |C^{*''}_{i-1}| = |C_{i-1}|.$$

Moreover, we observe that the i-cut of π_i verifies $(B_{\pi_i}, B^*_{\pi_i}) = (C_{i-1}, C^*_{i-1})$, so $\rho_{\pi_i} = \pi_{i-1}$.

Example 12 *Let $\sigma = 435\bar{1}\bar{2}, \tau = \bar{2}135\bar{4} \in S_5^B$. We apply Algorithm 1. Since $\sigma^{-1}(1) < 0$, we have $\pi_1 = 1\bar{1}$. The following table displays the subsequent computations for i from 2 to 5.*

i	H_i^σ	H_i^τ	$H_i^\sigma \cup H_i^\tau$	C_{i-1}^*	π_i
2	$\{\bar{1},1\}$	$\{1\}$	$\{\bar{1},1\}$	$1\bar{1}$	$21\bar{1}\bar{2}$
3	$\{\bar{2},\bar{1}\}$	$\emptyset$	$\{\bar{2},\bar{1}\}$	$\bar{1}\bar{2}$	$21\bar{3}3\bar{1}\bar{2}$
4	$\{\bar{1},\bar{2},3\}$	$\{\bar{3},\bar{2},\bar{1},1,2,3\}$	$\{\bar{3},\bar{2},\bar{1},1,2,3\}$	$21\bar{3}3\bar{1}\bar{2}$	$421\bar{3}3\bar{1}\bar{2}\bar{4}$
5	$\{\bar{2},\bar{1}\}$	$\{\bar{4}\}$	$\{\bar{4},\bar{2},\bar{1}\}$	$\bar{1}\bar{2}\bar{4}$	$421\bar{5}\bar{3}35\bar{1}\bar{2}\bar{4}$

To better understand the notation used in Remark 11, we note that in the previous computations we find:

- $C_1^{*'} = 1\bar{1}$ *and* $C_1^{*''} = \varepsilon$;
- $C_2^{*'} = 21\bar{1}\bar{2}$ *and* $C_2^{*''} = \varepsilon$;
- $C_3^{*'} = 21\bar{3}3\bar{1}\bar{2}$ *and* $C_3^{*''} = \varepsilon$;
- $C_4' = 421$ *and* $C_4'' = \bar{3}3$.

In conclusion, we obtain $J(\sigma,\tau) = \pi_5 = 35\bar{1}\bar{2}\bar{4}$.

The main result of this extended abstract is the following theorem, which shows that Algorithm 1 correctly computes the join of two signed permutations in $(S_n^B, \leq_R)$.

Theorem 13 *Let* $\sigma,\tau \in S_n^B$*; then*

$$J(\sigma,\tau) = \sigma \vee_R \tau. \tag{4}$$

We conclude this section by sketching our proof of Theorem 13. We prove by induction on i that $\pi_i = \rho_\sigma^{n-i} \vee_R \rho_\tau^{n-i} \in S_i^B$; in particular, we show that

$$\begin{aligned}
\mathrm{Inv}((\rho_\sigma^{n-i})^{-1}) \cup \mathrm{Inv}((\rho_\tau^{n-i})^{-1}) &\subseteq \mathrm{Inv}(\pi_i^{-1}),\\
\mathrm{Neg}((\rho_\sigma^{n-i})^{-1}) \cup \mathrm{Neg}((\rho_\tau^{n-i})^{-1}) &\subseteq \mathrm{Neg}(\pi_i^{-1}),\\
\mathrm{Nsp}((\rho_\sigma^{n-i})^{-1}) \cup \mathrm{Nsp}((\rho_\tau^{n-i})^{-1}) &\subseteq \mathrm{Nsp}(\pi_i^{-1}),
\end{aligned}$$

getting $\rho_\sigma^{n-i}, \rho_\tau^{n-i} \leq_R \pi_i$, which implies $\rho_\sigma^{n-i} \vee_R \rho_\tau^{n-i} \leq_R \pi_i$. Then, in order to obtain $\pi_i \leq_R \rho_\sigma^{n-i} \vee_R \rho_\tau^{n-i}$, it is sufficient to prove that for any $\beta \in S_i^B$ such that $\rho_\sigma^{n-i}, \rho_\tau^{n-i} \leq_R \beta$, one has $\pi_i \leq_R \beta$. This is done by using Lemma 7 and Proposition 9 on $\rho_\sigma^{n-i}, \rho_\tau^{n-i} \leq_R \beta$. The case $i = n$, yields $\pi_n = J(\sigma,\tau) = \sigma \vee_R \tau$.

3 Final remarks

The main importance of Theorem 13 and of its proof is that they provide an explicit description of the inversions, negative entries, and negative sum pairs of the join $\sigma \vee_R \tau$ in terms of those of σ and τ. This characterization reduces the proof of the following result to three cases and yields the inclusion $T_L(\sigma \vee_R \tau) \subseteq \overline{T_L(\sigma) \cup T_L(\tau)}$.

Theorem 14 *Let* $\sigma,\tau \in S_n^B$ *and* $t \in T_L(\sigma \vee_R \tau)$*. Then there exists a palindromic* (σ,τ)*-Bruhat path reaching* t*. In particular,* $T_L(\sigma \vee_R \tau) \subseteq \overline{T_L(\sigma) \cup T_L(\tau)}$.

The reverse inclusion follows from [3, Theorem 4.21] of Dermenjian. Together, the two inclusions establish the validity of Conjecture 3 for Coxeter groups of type B.

We conclude this extended abstract with a related open problem. As noted above, our algorithm generalizes the one defined by Markowsky for computing the join of two permutations in the weak order of the symmetric group S_n. Indeed, S_n is isomorphic to the subgroup of S_n^B consisting of signed

permutations with no negative entries. Furthermore, $(S_n, \leq_R)$ is a sublattice of $(S_n^B, \leq_R)$ and Algorithm 1, when restricted to this subgroup, coincides exactly with Markowsky's algorithm.

Within S_n^B, one can also consider S_n^D, the subgroup of index 2 consisting of all signed permutations with an even number of negative entries, see [2, Chapter 8]. It is then natural to ask whether the same approach can be used to compute joins in type D. In general, however, Algorithm 1 does not compute the join of two signed permutations in the poset $(S_n^D, \leq_D)$.

As a Coxeter system, S_n^D can be generated by $S_D = \{s_0^D = (\bar{1}\ \bar{2}), s_1 = (1\ 2), \ldots, s_{n-1} = (n-1\ n)\}$ and the main difference with S_n^B is the absence of reflections of the form $(a\ \bar{a})$. We denote by $\leq_D$ the (right) weak order defined on the Coxeter system (S_n^D, S_D) and by $\leq_B$ the one induced by S_n^B on the subgroup S_n^D. The two orders do not coincide. More precisely, given $\sigma, \tau \in S_n^D$ one has $\sigma \leq_B \tau$ implies $\sigma \leq_D \tau$, but the converse implication is not true. One reason for this is that the left-reflection set of $\sigma \in S_n^D$ is given by $\{(i\ j) \mid (i,j) \in \mathrm{Inv}(\sigma^{-1})\} \sqcup \{(i\ \bar{j}) \mid \{i,j\} \in \mathrm{Nsp}(\sigma^{-1})\}$, which is a subset of its left-reflection set in S_n^B described in (2). The following example shows that Algorithm 1, when applied to elements of S_n^D, may fail to produce their joins in $(S_n^D, \leq_D)$.

Example 15 *Consider the two signed permutations $\sigma = 2314$ and $\tau = \bar{1}3\bar{2}4$ in S_4^D. Their join in $(S_4^D, \leq_D)$ is $\sigma \vee_D \tau = \bar{3}1\bar{2}4$, but if we apply Algorithm 1 we get $\pi_1 = 1\bar{1}$ and*

i	H_i^σ	H_i^τ	π_i
2	$\{1\}$	$\{\bar{1}, 1\}$	$21\bar{1}\bar{2}$
3	$\{1\}$	$\{\bar{2}\}$	$231\bar{1}\bar{3}\bar{2}$
4	$\emptyset$	$\emptyset$	$\bar{4}231\bar{1}\bar{3}\bar{2}4$

So, the join of σ and τ in S_4^B is $\sigma \vee_B \tau = \bar{1}\bar{3}\bar{2}4$ which is not an element of S_4^D.

At present, an analog to Algorithm 1 that operates directly on the complete notation of S_n^D remains elusive, and we propose its construction as an open problem. Such an algorithm would likely be a key instrument for extending our approach from [1] and obtaining a complete proof of Dyer's conjecture for type D_n.

References

[1] R. Biagioli & L. Perrone (2026): *On a Conjecture of Dyer on the Join in the Weak Order of a Coxeter Group*. Journal of Combinatorics 17(4), pp. 507–525, doi:10.4310/JOC.260511230537.

[2] A. Björner & F. Brenti (2005): *Combinatorics of Coxeter Groups*. Graduate Texts in Mathematics 231, Springer, doi:10.1007/3-540-27596-7.

[3] Aram Dermenjian (2025): *Bruhat Preclosure*. arXiv:2512.08711.

[4] M. Dyer (2019): *On the Weak Order of Coxeter Groups*. Canadian Journal of Mathematics 71(2), pp. 299–336, doi:10.4153/CJM-2017-059-0.

[5] G. Markowsky (1994): *Permutation lattices revised*. Mathematical Social Sciences 27(1), pp. 59–72, doi:10.1016/0165-4896(94)00731-4.

[6] N. Reading (2016): *Finite Coxeter Groups and the Weak Order*. In Friedrich Grätzer, Georgeand Wehrung, editor: *Lattice Theory: Special Topics and Applications: Volume 2*, Springer International Publishing, pp. 489–561, doi:10.1007/978-3-319-44236-5_10.

[7] The Sage Developers (2025): *SageMath*. `https://www.sagemath.org`.

[8] H. Yu (2024): *The weak order on the hyperoctahedral group and the monomial basis for the Hopf algebra of signed permutations*. Discrete Mathematics 347(6):113942, doi:10.1016/j.disc.2024.113942.

Snake Polyominoes of Maximal Area in a Rectangle

Alexandre Blondin Massé
Université du Québec à Montréal
Montréal, Canada
blondin_masse.alexandre@uqam.ca

Alain Goupil
Université du Québec à Trois-Rivières
Trois-Rivières, Canada
alain.goupil@uqtr.ca

Given a discrete rectangle R of dimensions $h \times w$, let $\mathbf{W}^{\text{snake}}_{h\times w}$ be the set of snake-like polyominoes contained in R represented as binary matrices, *i.e.* polyominoes whose underlying simple graph is a chain with respect to the 4-adjacency relation. We present an algorithm that generates $\mathbf{W}^{\text{snake}}_{h\times w}$ for any h and w. Also, let $a_{\max}(h,w)$ the maximal area that can be realized by an element of $\mathbf{W}^{\text{snake}}_{h\times w}$. We provide exact formulas of $a_{\max}(h,w)$ for $h \in \{1,2,3,4,5\}$ and $w \in \mathbb{N}^*$.

1 Introduction

Following a previous result ([1]) stating that the number of cells of maximal degree 2 in a 2D word with a rectangle R of dimensions $h \times w$ is at most $2/3$ the area of R plus a small constant that depends on h and w, we now investigate the maximal area of snake polyominoes contained in a rectangle of dimensions $h \times w$. We show that this family of polyominoes possess sufficient structure to provide explicit expressions for their maximal area in terms of h and w, for height $h \in \{1,2,3,4,5\}$. Our underlying conjecture is that there exists an explicit expression for the maximal area of snake-like polyominoes in any given rectangle of arbitrary dimensions $h \times w$ that can be expressed in terms of h and w. Due to space restriction, many proofs of the stated propositions and lemmas have been ommitted, but will be included in an upcoming extended version.

2 2D Words

For $a, b \in \mathbb{N}$, let $[\![a,b]\!] = \{n \in \mathbb{N} \mid a \leq n \leq b\}$. Also, if $a, b, m \in \mathbb{N}$, with $m > 0$, we write $a \equiv_m b$ when a and b are congruent modulo m. The definitions and notations that are used here for 2-dimensional words is adapted from [2, 3].

Let A be a finite alphabet and $h, w \in \mathbb{N}$. A *2-dimensional word, or 2D-word, W of dimensions $h \times w$ on A* is a matrix of h rows and w columns with entries in A. The dimensions h and w are called respectively the *height* and *width* of W. The entry of W at row $i \in [\![1,h]\!]$ and column $j \in [\![1,w]\!]$ is denoted by $W[i,j]$. Also, given $i \in [\![1,h]\!]$ (resp. $j \in [\![1,w]\!]$), the 1-dimensional word, 1*D*-word, obtained by taking the i-th row (resp. j-th column) of W is denoted by $W[i,_]$ (resp. $W[_,j]$). The 2D word W is called *empty* if $h = 0$ or $w = 0$. For each $h \in \mathbb{N}$ (resp. $w \in \mathbb{N}$), there exists a unique empty word of height h (resp. width w), denoted by $\varepsilon_{h\times 0}$ (resp. $\varepsilon_{0\times w}$). The set of all 2D words of dimensions $h \times w$ (resp. of height h, of width w) on A is denoted by $\mathbf{W}^A_{h\times w}$ (resp. $\mathbf{W}^A_{h\times_}$, $\mathbf{W}^A_{_\times w}$). Given $h, w_1, w_2 \in \mathbb{N}$, the *horizontal concatenation* of $U \in \mathbf{W}^A_{h\times w_1}$ and $V \in \mathbf{W}^A_{h\times w_2}$ is the word $U \oplus V \in \mathbf{W}^A_{h\times(w_1+w_2)}$ defined by

$$(U \oplus V)[i,j] = \begin{cases} U[i,j], & \text{if } j \leq w_1; \\ V[i,j-w_1], & \text{if } j > w_1. \end{cases}$$

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 29–37, doi:10.4204/EPTCS.445.5

© A. Blondin Massé & A. Goupil
This work is licensed under the
Creative Commons Attribution License.

The *vertical concatenation* $U \ominus V \in \mathbf{W}^A_{(h_1+h_2)\times w}$ of $U \in \mathbf{W}^A_{h_1\times w}$ and $V \in \mathbf{W}^A_{h_2\times w}$ is defined similarly for $h_1, h_2, w \in \mathbb{N}$. Given $W \in \mathbf{W}^A_{h\times w}$ and $m, n \in \mathbb{Q}$ such that $mh, nw \in \mathbb{N}$, the (m,n)*-th power of* W is the word $W^{m\times n} \in \mathbf{W}^A_{mh\times nw}$ such that $W^{m\times n}[i,j] = W[(i-1 \bmod h)+1, (j-1 \bmod w)+1]$. One might easily prove that $(\mathbf{W}^A_{h\times_}, \oplus, \varepsilon_{h\times 0})$ and $(\mathbf{W}^A_{_\times w}, \ominus, \varepsilon_{0\times w})$ are monoids. We denote by $\tau(W)$ the 2D word obtained from W by transposing its underlying matrix.

Since 2D words can be seen as discrete rectangles filled with letters, we encode for convenience their *north*, *west*, *east* and *south* sides, or directions, with the alphabet $D = \{\mathtt{n}, \mathtt{w}, \mathtt{e}, \mathtt{s}\}$. Also, for sake of compactness, we write subsets of D as words, without, braces and commas, so that the subset $\{\mathtt{n}, \mathtt{e}, \mathtt{s}\}$ is denoted by `nes`. For convenience, we consider two bijective endofunctions on D. The first one is defined by $\rho(\mathtt{n}) = \mathtt{w}$, $\rho(\mathtt{w}) = \mathtt{s}$, $\rho(\mathtt{e}) = \mathtt{n}$ and $\rho(\mathtt{s}) = \mathtt{e}$, corresponding geometrically with a quarter-turn counterclockwise rotation. The second one is defined by $\tau(\mathtt{n}) = \mathtt{w}$, $\tau(\mathtt{w}) = \mathtt{n}$, $\tau(\mathtt{e}) = \mathtt{s}$ and $\tau(\mathtt{s}) = \mathtt{e}$, corresponding geometrically with a reflection along a diagonal axis from the north-west corner to the south-east corner. Let W, U be 2D words. U is a *factor* of W when the rectangle of U is a subrectangle of the rectangle of W. More formally, we say that U is a *factor* of W if there exist 2D words R_1, R_2, R_3, C_1, C_2, C_3, W_{11}, W_{12}, W_{13}, W_{21}, W_{23}, W_{31}, W_{32} and W_{33} such that

$$W = \begin{matrix} R_1 \\ \ominus \\ R_2 \\ \ominus \\ R_3 \end{matrix} = C_1 \oplus C_2 \oplus C_3, \qquad \begin{aligned} R_1 &= W_{11} \oplus W_{12} \oplus W_{13} \\ R_2 &= W_{21} \oplus U \oplus W_{23} \\ R_3 &= W_{31} \oplus W_{32} \oplus W_{33} \\ C_1 &= W_{11} \ominus W_{21} \ominus W_{31} \\ C_2 &= W_{12} \ominus U \ominus W_{32} \\ C_3 &= W_{13} \ominus W_{23} \ominus W_{33} \end{aligned} \tag{1}$$

Given any $H \subseteq D$ and any U, W such that U is a factor of W, we say that U *is an* H*-factor of* W if the condition $\mathtt{n} \in H$ implies R_1 is empty, the condition $\mathtt{w} \in H$ implies C_1 is empty, the condition $\mathtt{e} \in H$ implies C_3 is empty and the condition $\mathtt{s} \in H$ implies R_3 is empty. Roughly speaking, U is an H-factor of W if, for each $d \in H$, U is on the border of W in the direction d.

From now on, we fix the alphabet $A = \{\square, \blacksquare\}$. Given $W \in \mathbf{W}^A_{h\times w}$, the *graph of* W, denoted by $G[W]$, is the subgraph of the grid graph $G_{h\times w}$ induced by $\{(i,j) \mid W[i,j] = \blacksquare\}$. Any ordered pair $(i,j) \in [\![1,h]\!] \times [\![1,w]\!]$ is called a *cell of* W. A cell (i,j) is on the *n-boundary (resp. w-boundary, e-boundary, s-boundary)* of W if $i = 1$ (resp. $j = 1$, $j = w$, $i = h$), or simply on the *boundary* of W if it is on the d-boundary of W, for some $d \in D$. Otherwise (i,j) is called an *internal* cell. We say that the cell (i,j) of W is *occupied* if $W[i,j] = \blacksquare$ and that it is *empty* if $W[i,j] = \square$. The number of occupied cells in W is denoted $|W|_\blacksquare$ and is also called the *area* of W. The *neighborhood of* (i,j) *in* W is defined by

$$N_W(i,j) = \{(i',j') \in [\![1,h]\!] \times [\![1,w]\!] \mid (i-i')^2 + (j-j')^2 = 1\}, \tag{2}$$

while the *degree* of (i,j) in W is $\deg_W(i,j) = \mathrm{Card}\{(i',j') \in N_W(i,j) \ : \ W[i',j'] = \blacksquare\}$. A connected component $C \subseteq [\![1,h]\!] \times [\![1,w]\!]$ of $G[W]$ is called a *component of* W. The number of components of W is denoted by $c(W)$. We say that W is *n-inscribed (resp. w-inscribed, e-inscribed, s-inscribed)* if $\blacksquare$ occurs in $W[1,_]$ (resp. $W[_,1]$, $W[_,w]$, $W[h,_]$). Finally, we say that W is *inscribed* if it is d-inscribed for all $d \in D$.

3 Snake Words and Snake Factors

A *snake word* is an inscribed binary word W on A such that $G[W]$ is a chain. The set of all snake words of dimensions $h \times w$ on A is denoted by $\mathbf{W}^{\mathrm{snake}}_{h\times w}$. Similarly, a *snake forest word* is a binary word W on A

such that $G[W]$ is a forest of chains. A binary word W is called a *snake factor* if there exists a snake word W' such that W is a factor of W'. The set of all snake factors of dimensions $h \times w$ on A is denoted by $\mathbf{W}^{\text{s-f}}_{h\times w}$. The following two observations are immediate:

Proposition 1.

(i) A snake factor is a snake forest word.
(ii) There exist snake forest words that are not snake factors.

A *sided word of dimensions* $h \times w$ is a pair $S = (W, H)$, where $W \in \mathbf{W}^A_{h\times w}$ and $H \subseteq D$ is called the *hull* of S. Without ambiguity, the area and number of components are naturally extended from words to sided words by setting $|S|_\blacksquare = |W|_\blacksquare$ and $c(S) = c(W)$. Similarly, the definitions of *height*, *width*, *cell*, *degree of a cell*, *component*, *snake word*, *snake forest* and *inscription* on 2D words are naturally extended to sided words.

The definitions of *horizontal concatenation*, *vertical concatenation*, *factor* and *snake factor* for sided words, however, need to be adjusted. Let $S = (W, H)$ and $S' = (W', H')$ be two sided words. Then the *horizontal concatenation* of S and S' is the partial binary operations $S \oplus S' = (W \oplus W', H \cup H')$ defined whenever S and S' have the same height, $\mathtt{e} \notin H$, $\mathtt{w} \notin H'$ and $H - \{\mathtt{w}\} = H' - \{\mathtt{e}\}$. The *vertical concatenation* $S \ominus S'$ of S and S' is defined similarly. We say that S is a *factor* of S' if $H \subseteq H'$ and W is an H-factor of W'. Finally, $S = (W, H)$ is called a *snake factor* if there exists an inscribed sided word $S' = (W', D)$ such that S is a factor of S' and W is a snake factor of W'.

Let $S = (W, H)$ be a sided word and (i, j) a cell of S. The *number of outside liberties of* (i, j) *in* S, denoted by $\ell^{\text{out}}_S(i, j)$, is defined by

$$\begin{aligned} \ell^{\text{out}}_S(i,j) &= \mathbb{I}(i=1)\mathbb{I}(\mathtt{n} \notin H) + \mathbb{I}(j=1)\mathbb{I}(\mathtt{w} \notin H) \\ &+ \mathbb{I}(i=h)\mathbb{I}(\mathtt{s} \notin H) + \mathbb{I}(j=w)\mathbb{I}(\mathtt{e} \notin H), \end{aligned} \tag{3}$$

where $\mathbb{I}(\cdot)$ is the usual indicator function, *i.e.* $\mathbb{I}(p) = 1$ if p is true, 0 otherwise. Similarly, the *number of liberties of* (i, j) *in* S, denoted by $\ell_S(i, j)$, is defined by

$$\ell_S(i,j) = \max(\min(\ell^{\text{out}}_S(i,j), 2 - \deg_S(i,j)), 0). \tag{4}$$

It follows from the definition that $\ell_S(i, j) \in \{0, 1, 2\}$ if (i, j) is a boundary cell, while (i, j) is an internal cell implies that $\ell_S(i, j) = 0$. Intuitively, the number of liberties of a cell is the maximum number of its edges that can be connected to other cells without creating cells of degree greater than 2. Let C be a component of S and define the number of liberties of C as $\ell_S(C) = \sum_{(i,j)\in C} \ell_S(i, j)$. Then C is called *unconnectable (resp. terminal, crossing)* if $\ell_S(C) = 0$ (resp. $\ell_S(C) = 1$, $\ell_S(C) = 2$). Moreover, given $d \in D$, we say that C *is connectable on* d *in* S if there exists a d-boundary cell $(i, j) \in C$ such that $\ell_S(i, j) > 0$. The number of unconnectable, crossing, terminal components of S are respectively denoted $u(S)$. $x(S)$, $t(S)$. For each $d \in D$, the number of terminal components connectable on d is denoted by $t_d(S)$. The first main result of this extended abstract is a characterization of sided snake factors:

Theorem 1. *Let $S = (W, H)$ be a sided word. Then S is a sided snake factor if and only if S is a sided snake forest word, $t(S) \leq 2$, and exactly one of the following conditions holds:*

(i) $H = D$, S is inscribed and $c(S) = 1$.
(ii) $H = D - \{d\}$ for some $d \in D$, S is $\rho^2(d)$-inscribed and $u(S) = 0$.
(iii) $|H| \in \{0, 1\}$ or $H = \{d, \rho(d)\}$ for some d, and $u(S) = 0$.
(iv) $H = D - \{d, \rho^2(d)\}$ for some d, $u(S) = 0$, S has at least one crossing component connectable on both d and $\rho^2(d)$, and the condition $t(S) = 2$ implies $t_d(S), t_{\rho^2(d)}(S) \equiv_2 x(S)$.

In order to prove Theorem 1, we need the following lemmas:

Lemma 1. *Let $S = (W,H)$ be a sided snake factor and C a terminal component of S. Then exactly one of the following two conditions holds:*

(1) S has a unique internal cell $(i,j) \in C$ such that $\deg_S(i,j) = 1$.
(2) S has exactly one boundary cell $(i,j) \in C$ such that $\deg_S(i,j) = 0$ and $\ell_S^{\text{out}}(i,j) = 1$.

The unique cell satisfying Condition (1) or (2) is called the terminal *cell of C.*

The following observation is well-known in graph theory:

Lemma 2. *Let $S = (W,H)$ be a sided snake word and C the only component of S. Then for any pair of cells c and c' of C, there exists a unique chain between c and c'.*

We are now ready to prove theorem 1.

Proof of Theorem 1. ($\Rightarrow$) Assume that $S = (W,H)$ is a sided snake factor. Then there exists an inscribed sided snake word $S' = (W',D)$ such that S is a factor of S'. Let $G = G[W]$ and $G' = G[W']$. First, since G is a subgraph of G', we conclude that G is a snake forest, so that S is a sided snake forest word. Next, we cannot have $t(S) \geq 3$: if it was the case, by Lemma 1, G' would have at least 3 vertices of degree ≤ 1, contradicting the fact that G' is a chain. Hence, $t(S) \leq 2$. To conclude this part, we consider four cases.

Case $H = D$. Since S is a factor of S', by definition of H-factor, we conclude that $W = W'$, which means that $S = S'$ is a snake word. Hence, $c(S) = 1$.

Case $H = D - \{d\}$. Then S is $\rho^2(d)$-inscribed: if it was not the case, then S' would not be inscribed. Moreover, we cannot have $u(S) > 0$: if it was the case, then we would have $c(S') > 1$.

Case $|H| \in \{0,1\}$ or $H = \{d, \rho(d)\}$. As for the preceding case, the assumption $u(S) > 0$ implies $c(S') > 1$, a contradiction.

Case $H = D - \{d, \rho^2(d)\}$. The fact that $u(S) = 0$ follows from an argument similar to those of the two previous cases. Also, S must have at least one crossing component connectable on both d and $\rho^2(d)$, otherwise, S' would have at least two components, on each open sides of S. Finally, assume that $t(S) = 2$ and let $t = (t_d(S), t_{\rho^2(d)}(S))$, which implies $t \in \{(0,2),(1,1),(2,0)\}$. If $t = (1,1)$, then $x(S)$ must be odd. Indeed, by Lemma 2, there exists a unique chain between the two terminal cells of the only component of S'. Since the terminal components of S are connected on opposite sides of S , they must cross S an odd number of times. Using a similar argument, if $t \in \{(0,2),(2,0)\}$, then $x(S)$ must be even. Since all cases have been covered, the result follows.

($\Leftarrow$) Due to space restriction, we only provide an idea of the proof. We need to prove that there exists an inscribed sided word $S' = (W',D)$ such that S is a factor of S' and W is a snake factor of W'. There are four cases to consider, according to which condition between (i) and (iv) holds.

Case (i). If suffices to take $S' = S$.

Case (ii). If suffices to connect all components of S by extending it on its only open side, by making sure that the result is d-inscribed, for all $d \in D - \{\rho^2(d)\}$.

Case (iii). Similarly to case (ii), il suffices to connect all components of S and by making sure that the result is d-inscribed in all directions.

Case (iv). In that case, the inscription is easily obtained in all directions. The connectivity is guaranteed by the conditions on the components: the existence of at least one crossing component allows to connect each side, and if there are two terminal components, then they can be connected consistently with the crossing components, thanks to their equivalence modulo 2. □

Algorithm 1 Generation of sided snake factors

```
function BSF(h, w, a : positive integers, H : hull)
    if h > w then
        return τ(BSF(w, h, a, τ(H)))
    else if (h, w) = (1, 1) then
        if a = 1 then
            return {(■, H)}
        else if a = 0 and |H| ≠ 3 and H ∉ {ns, we} then
            return {(□, H)}
        else
            return ∅
    else
        (w1, w2) ← (⌊w/2⌋, ⌈w/2⌉)
        (H1, H2) ← (H − {e}, H − {w})
        S ← ∅
        for a1 ∈ {0, 1, ..., a} do
            a2 ← a − a1
            L ← BSF(h, w1, a1, H1)
            R ← BSF(h, w2, a2, H2)
            S ← S ∪ {ℓ ⊕ r | ℓ ∈ L, r ∈ R and ℓ ⊕ r is a snake factor}
        return S
```

4 Generation of Sided Snake Factors

Let $\mathbf{S}^{\mathrm{s-f}}_{h,w,a,H}$ be the set of all sided snake factors of dimensions $h \times w$, of area a and of hull H. We were able to use Algorithm 1 to generate $\mathbf{S}^{\mathrm{s-f}}_{h,w,a,H}$ for small dimensions $h \times w$.

We need additional definitions and notation. First, let $S = (W, H) \in \mathbf{S}^{\mathrm{s-f}}_{h,w,H}$. We denote by $|S|_{\square}$ the 4-tuple (n, w, e, s) indicating, for each side d, the number of empty cells of S located on the side d boundary of S if $d \in H$, and we write _ otherwise. Similarly, we denote by $|S|_{\deg=2}$ the 4-tuple (n, w, e, s) indicating, for each side d, the number of cells of S, whose degree in S is 2, located on the side d boundary of S if $d \in H$, and we write _, otherwise. For any $h, w \in \mathbb{N}^*$ and $H \subseteq D$, let $\mathbf{S}^{\mathrm{s-f}}_{h,w,H}$ be the set of all sided snake factors of dimensions $h \times w$ and of hull H, whatever their area, and let $a_{\max}(h, w, H)$ be the maximal area that can be realized by a sided snake factor of dimensions $h \times w$ with hull H, i.e. $a_{\max}(h, w, H) = \max\{|S|_{\blacksquare} : S \in \mathbf{S}^{\mathrm{s-f}}_{h,w,H}\}$. The following observation is immediate:

Lemma 3. *Let $h, w, w_1, w_2 \in \mathbb{N}^*$, with $w = w_1 + w_2$, and $H \subseteq D$. Then*

$$a_{\max}(h, w, H) \leq a_{\max}(h, w_1, H - \{e\}) + a_{\max}(h, w_2, H - \{w\}). \tag{5}$$

Algorithm 1 can be helpful in establishing exact values or upper bounds of $a_{\max}(h, w, H)$. More precisely:

Lemma 4. *Let $h, w, a \in \mathbb{N}^*$ and $H \subseteq D$, where $h + w - 1 \leq a \leq hw$. If $\mathbf{S}^{\mathrm{s-f}}_{h,w,a,H} \neq \emptyset$ and $\mathbf{S}^{\mathrm{s-f}}_{h,w,a+1,H} = \emptyset$, then $a_{\max}(h, w, H) = a$.*

As a consequence, we obtain the following observations for $h = 1, 2, 3$.

Lemma 5. *Let $w \in \mathbb{N}^*$ and $H \subseteq D$. Then the following identities hold:*

(i) $a_{\max}(1,1,H) = 1$;
(ii) $a_{\max}(2,1,H) = 2$, $a_{\max}(2,2,H) = 3$;
(iii) $a_{\max}(2,w,\texttt{ns}) \leq 3w/2$ *if* $w \bmod 2 = 0$;
(iv) $a_{\max}(3,1,H) = 3$, $a_{\max}(3,2,\texttt{nws}) = a_{\max}(3,2,\texttt{nes}) = 5$, $a_{\max}(3,2,\texttt{ns}) = 4$;
(v) $a_{\max}(3,w,\texttt{ns}) \leq 4w/2$ *if* $w \bmod 2 = 0$;

Proof. (i), (ii) and (iv) follow from direct observation, Algorithm 1 and Lemma 4.
(iii) follows from (ii), Lemma 3 and from induction on $w/2$.
(v) follows from (iv), Lemma 3 and from induction on $w/2$. □

The cases $h = 4$ and $h = 5$ show more complex periodicity patterns:

Lemma 6. *For any* $w \in \mathbb{N}^*$,

$$a_{\max}(4,w,\texttt{nws}) \leq \begin{cases} (8w+4)/3, & \text{if } w = 1; \\ (8w+3)/3, & \text{if } w \bmod 3 = 0; \\ (8w+1)/3, & \text{if } w \bmod 3 = 1; \\ (8w+2)/3, & \text{if } w \bmod 3 = 2. \end{cases} \quad a_{\max}(5,w,\texttt{nws}) \leq \begin{cases} (22w+13)/7, & \text{if } w \in \{1,8\}; \\ (22w+15)/7, & \text{if } w = 6; \\ (22w+14)/7, & \text{if } w = 7; \\ (22w+7)/7, & \text{if } w \bmod 7 = 0; \\ (22w+6)/7, & \text{if } w \bmod 7 = 1; \\ (22w+12)/7, & \text{if } w \bmod 7 = 2; \\ (22w+11)/7, & \text{if } w \bmod 7 = 3; \\ (22w+10)/7, & \text{if } w \bmod 7 = 4; \\ (22w+9)/7, & \text{if } w \bmod 7 = 5; \\ (22w+8)/7, & \text{if } w \bmod 7 = 6. \end{cases}$$

5 Main Result

Let $a_{\max}(h,w)$ the the maximal area that can be realized by a snake-like polyomino inscribed in a rectangle $h \times w$. Also, for $w \in \mathbb{N}^*$, let

$$\hat{a}_{\max}(1,w) = w$$

$$\hat{a}_{\max}(2,w) = \begin{cases} 3w/2, & \text{if } w \bmod 2 = 0; \\ (3w+1)/2, & \text{if } w \bmod 2 = 1; \end{cases}$$

$$\hat{a}_{\max}(3,w) = \begin{cases} 2w+1, & \text{if } 1 \leq w \leq 5; \\ 2w+2, & \text{if } w \geq 6. \end{cases}$$

$$\hat{a}_{\max}(4,w) = \begin{cases} (8w+1)/3, & \text{if } w = 4; \\ (8w+3)/3, & \text{if } w \bmod 3 = 0; \\ (8w+4)/3, & \text{if } w \bmod 3 = 1; \\ (8w+2)/3, & \text{if } w \bmod 3 = 2. \end{cases}$$

$$\hat{a}_{\max}(5,w) = \begin{cases} 3w+4, & \text{if } w \in \{10,11,12\}; \\ (22w+14)/7, & \text{if } w \bmod 7 = 0; \\ (22w+13)/7, & \text{if } w \bmod 7 = 1; \\ (22w+12)/7, & \text{if } w \bmod 7 = 2; \\ (22w+11)/7, & \text{if } w \bmod 7 = 3; \\ (22w+10)/7, & \text{if } w \bmod 7 = 4; \\ (22w+9)/7, & \text{if } w \bmod 7 = 5; \\ (22w+15)/7, & \text{if } w \bmod 7 = 6. \end{cases}$$

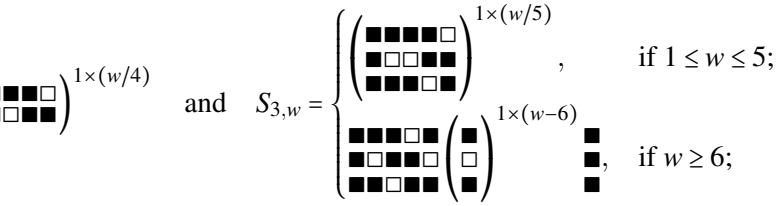
The main result of this extended abstract is the following.

Theorem 2. *For $1 \leq h \leq 5$ and $w \in \mathbb{N}^*$, $a_{\max}(h,w) = \hat{a}_{\max}(h,w)$.*

The proof of Theorem 2 is divided in two cases, that are addressed in the following two lemmas.

Lemma 7. *For $1 \leq h \leq 5$ and $w \in \mathbb{N}^*$, $a_{\max}(h,w) \geq \hat{a}_{\max}(h,w)$.*

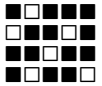
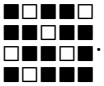
Proof. It suffices to exhibit families of snakes having area $\hat{a}_{\max}(h,w)$ for $1 \leq h \leq 5$ and for any $w \in \mathbb{N}^*$. Let

$S_{1,w} =$, $S_{2,w} =$ and $S_{3,w} =$ 

Moreover, let

$$S_{4,w} = \begin{cases} A_4^{1\times(w/2)}, & \text{if } 1 \leq w \leq 2; \\ A_4(B_4B_4^h)^{1\times((w-2)/6)}, & \text{if } 3 \leq w \leq 6 \text{ or } w \bmod 3 \neq 1; \\ A_4(B_4B_4^h)^{1\times((w-7)/6)}C_4, & \text{if } w \geq 7 \text{ and } w \bmod 6 = 1; \\ A_4(B_4B_4^h)^{1\times((w-7)/6)}B_4C_4^h, & \text{if } w \geq 7 \text{ and } w \bmod 6 = 4. \end{cases} \tag{6}$$

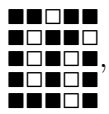
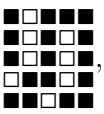
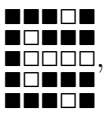
where

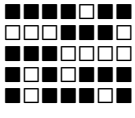
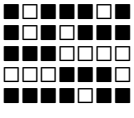
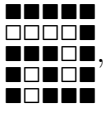
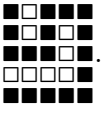
$A_4 =$ , $B_4 =$ , $B_4^h =$ , $C_4 =$  and $C_4^h =$ .

Finally, let

$$S_{5,w} = \begin{cases} A_5, & \text{if } w = 1; \\ B_5C_5^{1\times(w-2)}B_5^h, & \text{if } 2 \leq w \leq 5; \\ B_5C_5^{1\times(w-6)}D_5^r, & \text{if } 6 \leq w \leq 9; \\ D_5C_5^{1\times(w-10)}D_5^r, & \text{if } 10 \leq w \leq 19; \\ E_5C_5^{1\times m}F_5(G_5G_5^h)^{1\times n}H_5^h, & \text{if } w \geq 20 \text{ and } 0 \leq w \bmod 14 \leq 6; \\ E_5C_5^{1\times m}F_5(G_5G_5^h)^{1\times n}H_5, & \text{if } w \geq 20 \text{ and } 7 \leq w \bmod 14 \leq 13. \end{cases} \tag{7}$$

where $m = (w+1) \bmod 7 + 1$, $n = (w-m-12)/14$ and

$A_5 =$ , $B_5 =$ , $B_5^h =$ , $C_5 =$ , $D_5 =$ , $D_5^r =$ , $E_5 =$ ,

$F_5 =$ , $G_5 =$ , $G_5^h =$ , $H_5 =$ , $H_5^h =$ .

Then, for any $h = 1,2,3,4,5$ and $w \in \mathbb{N}^*$, the dimensions of $S_{h,w}$ are $h \times w$ and $|S_{h,w}|_{\blacksquare} = \hat{a}_{\max}(h,w)$. Moreover, by using induction on w, we have that $S_{h,w}$ is a snake, concluding the proof. □

There is one last result to prove.

Lemma 8. *For $1 \le h \le 5$ and $w \in \mathbb{N}$, $a_{\max}(h,w) \le \hat{a}_{\max}(h,w)$.*

Proof. Let $S \in \mathbf{S}^{\text{s-f}}_{h,w,D}$ with $|S|_\blacksquare = a_{\max}(h,w)$, i.e. S is a sided snake of maximal area inscribed in a rectangle of dimensions $h \times w$. We show that $|S|_\blacksquare \le \hat{a}_{\max}(h,w)$.

Case $h = 1$. It suffices to notice that $|S|_\blacksquare \le w = \hat{a}_{\max}(1,w)$.

Case $h = 2$. The cases $w = 1$ and $w = 2$ can be proved by exhaustive verification. Now, assume that $w \ge 3$. Let $w_3 = 1 + (w-1) \bmod 2$ and $w_2 = w - w_3 - 2$. Notice that $w_2 \bmod 2 = 0$. Then $S = S_1S_2S_3$, for some $S_1 \in \mathbf{S}^{\text{s-f}}_{2,2,\texttt{nws}}$, $S_2 \in \mathbf{S}^{\text{s-f}}_{2,w_2,\texttt{ns}}$ and $S_3 \in \mathbf{S}^{\text{s-f}}_{2,w_3,\texttt{nes}}$. By Lemma 5(ii)-(iii),

$$\begin{aligned}|S|_\blacksquare &= |S_1|_\blacksquare + |S_2|_\blacksquare + |S_3|_\blacksquare \\ &\le 3 + 3w_2/2 + (w_3+1) = (3w+2-w_3)/2 = \begin{cases} 3w/2, & \text{if } w \bmod 2 = 0; \\ (3w+1)/2, & \text{if } w \bmod 2 = 1. \end{cases} \\ &= a_{\max}(2,w)\end{aligned}$$

Case $h = 3$. This case follows from [1]. However, for sake of consistency, we provide an alternate proof based on the tools introduced in the previous pages. First, the cases $1 \le w \le 5$ can be proved by exhaustive verification. Now, assume that $w \ge 6$. Let $w_3 = 1 + (w-1) \bmod 2$ and $w_2 = w - w_3 - 1$. Notice that $w_2 \bmod 2 = 0$. Then $S = S_1S_2S_3$, for some $S_1 \in \mathbf{S}^{\text{s-f}}_{3,1,\texttt{nws}}$, $S_2 \in \mathbf{S}^{\text{s-f}}_{3,w_2,\texttt{ns}}$ and $S_3 \in \mathbf{S}^{\text{s-f}}_{3,w_3,\texttt{nes}}$. By Lemma 5(ii)-(iii),

$$|S|_\blacksquare = |S_1|_\blacksquare + |S_2|_\blacksquare + |S_3|_\blacksquare \le 3 + 4w_2/2 + (2w_3+1) = 2w+2 = a_{\max}(3,w)$$

Case $h = 4$. The cases $1 \le w \le 5$ can be proved by exhaustive verification. Now, assume that $w \ge 6$. Then $S = S_1S_2$, for some $S_1 \in \mathbf{S}^{\text{s-f}}_{4,w-4,\texttt{nws}}$ and $S_2 \in \mathbf{S}^{\text{s-f}}_{4,4,\texttt{nes}}$. By Lemma 6(iv),

$$\begin{aligned}|S|_\blacksquare = |S_1|_\blacksquare + |S_2|_\blacksquare &\le \left(\begin{cases} (8(w-4)+3)/3, & \text{if } (w-4) \bmod 3 = 0; \\ (8(w-4)+1)/3, & \text{if } (w-4) \bmod 3 = 1; \\ (8(w-4)+2)/3, & \text{if } (w-4) \bmod 3 = 2. \end{cases} \right) + 11 \\ &= \begin{cases} (8w+4)/3, & \text{if } w \bmod 3 = 1; \\ (8w+2)/3, & \text{if } w \bmod 3 = 2; \\ (8w+3)/3, & \text{if } w \bmod 3 = 0. \end{cases}\end{aligned}$$

Case $h = 5$. The structure of the proof is similar to the case $h = 4$, but requires to study more cases. In particular, one shows that all concatenations of maximal elements of $\mathbf{S}^{\text{s-f}}_{5,w,\texttt{nws}}$ and $\mathbf{S}^{\text{s-f}}_{5,7,\texttt{ns}}$ yield invalid concatenations for small values of w, and that maximal elements of $\mathbf{S}^{\text{s-f}}_{5,w,\texttt{nws}}$ for any w satisfy some boundary conditions. □

Proof of Theorem 2. Follows from Lemmas 7 and 8. □

6 Conclusion

Computations not reported here for larger values of h suggest that similar formulas might exist for any h. Hence, we believe that the results presented in the previous sections introduce arguments that can be extended for the area of snake-like polyominoes in any rectangle.

References

[1] Alexandre Blondin Massé, Alain Goupil, Raphael L'Heureux & Louis Marin (2025): *Maximal 2-Dimensional Binary Words of Bounded Degree*. In: *International Conference on Combinatorics on Words*, Springer, pp. 37–48, doi:10.1007/978-3-031-97548-6_4.

[2] Dora Giammarresi & Antonio Restivo (1997): *Two-dimensional languages*. In: *Handbook of formal languages: volume 3 beyond words*, Springer, pp. 215–267, doi:10.1007/978-3-642-59126-6_4.

[3] Kenichi Morita (2004): *Two-dimensional languages*. In: *Formal Languages and Applications*, Springer, pp. 427–437, doi:10.1007/978-3-540-39886-8_22.

Entropic Generation of Binary Words

Olivier Bodini
EREN
Université Sorbonne Paris-Nord
olivier.bodini@univ-paris13.fr

Francis Durand
EREN
Université Sorbonne Paris-Nord
LIP6
Sorbonne Université, France
francis.durand@ens-paris-saclay.fr

The uniform generation of k Hamming weight binary words, equivalent to sampling k-subsets from n elements, relies on random bits, which can be expensive. We introduce a novel paradigm–*random bit recycling*–and use it to generate such binary words in linear time while consuming as few random bits as possible. The resulting algorithm is nearly optimal in terms of random bit consumption, meaning that it closely matches the *Shannon entropic lower bound* coming from information theory.

1 Introduction

The uniform generation of combinatorial structures is a cornerstone of computer science. In particular, the generation of binary words of length n with exactly k ones (or equivalently, random subsets of size k from a set of n elements) is an important primitive. While the time complexity of these generators has been studied extensively, a secondary cost metric has gained prominence in recent years: the consumption of random bits.

In many computing environments, pseudo-random bits are cheap. However, in high-security applications, cryptography, and embedded systems, randomness is often derived from hardware true random number generators (TRNGs) [2]. In these contexts, random bits are an expensive resource and can be a bottleneck for random generation. When generating a combinatorial structure uniformly at random from a finite set of F possibilities, the information-theoretic entropy [13] of the generated object is $\log_2 F$. By Shannon's source coding theorem, this entropy establishes a lower bound: any algorithm relying on fair coin flips (unbiased random bits) must consume, on average, at least $\mathscr{H} = \log_2 F$ bits to produce a perfectly uniform sample. For the problem at hand of generating binary words of length n with exactly k ones, the size of the state space is $\binom{n}{k}$. Consequently, we strive to give algorithms whose expected random bit consumption is asymptotically close to $\log_2 \binom{n}{k}$.

It is important to emphasize that all algorithmic complexities presented in this work–both in terms of execution time and random bit consumption–are stated as *expected* (average-case) bounds. This is a necessity as generating a uniform random integer over a range whose size is not a power of 2 (for instance, a uniform integer in $\{1,2,3\}$) from a stream of fair coin flips cannot be achieved in bounded time [9]. We call an algorithm *first order entropic* (resp. near-entropic) if its random bit consumption is asymptotically in $\mathscr{H} + o(\mathscr{H})$ (resp. in $(1+\varepsilon)\mathscr{H} + o(\mathscr{H})$, for any $\varepsilon > 0$) where $\mathscr{H} = \log_2 \binom{n}{k}$ is the entropy of our structure.

Generating fixed Hamming weight binary words is a two-parameter problem defined by length n and Hamming weight k (where we can assume $k \leq n/2$ by symmetry). To analyze our algorithms asymptotically, we evaluate complexities as $n \to \infty$ along the trajectory $k \sim n^\beta$ for a fixed $\beta \in (0,1)$. We call this setting the *polynomially sparse regime*. Current generation methods generally bifurcate based on the number of 1s. For highly sparse inputs ($k \ll n$), the partial Fisher-Yates shuffle [7] is efficient.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 38–46, doi:10.4204/EPTCS.445.6

© O. Bodini & F. Durand
This work is licensed under the
Creative Commons Attribution License.

Conversely, for denser inputs ($k = \Omega(n^{2/3+\varepsilon})$ for any $\varepsilon > 0$), an adaptation of the MERGE algorithm [1] performs well by initially approximating the target word with Bernoulli(k/n) variables and subsequently correcting for bias.

To the best of our knowledge, no prior algorithm simultaneously achieves linear time complexity $O(n)$ and near-optimal entropic cost in the polynomially sparse regime with $\beta \in (0, 2/3]$.

Our Contribution

In this paper, we present a novel algorithm that bridges this gap. We propose a generator for uniform binary words that is both entropically near-optimal and computationally linear.

We observe that the standard Fisher-Yates procedure, while inserting the k ones among the $n-k$ zeros, implicitly constructs a uniform random permutation on these 1s. In order to capitalize on this implicit uniform random permutation, we introduce an algorithm for *deconstructing uniform random permutations*. The procedure is an emulation of a generative process in reverse, transforming the uniform random permutation into independent random bits. In a way, we "recycle" the randomness usually discarded during the insertion process. It is worth noting that the broader paradigm of entropy recycling in random generation is concurrently being explored in some different frameworks, see [5] for online sampling and [4, Alg. 5] for recycling bits while sampling batches of i.i.d variables.

The random bits obtained at the end of the generation can then be used to do something else, having an effective entropic random bit consumption (the number of random bit used in the process minus the number of random bits returned) that tightly matches the lower bound. This allows entropic generation of batches of binary words of size n with k ones, by using the random bits recycled for the previous words to generate the next one.

Alternatively, we can stop the insertions mid process, the smaller uniform random permutation can then be recycled mid-process, and the random bits reinjected into the generation procedure. Doing this multiple times allow a $1+\varepsilon$ entropic generation for any $\varepsilon > 0$, meaning that the generation algorithm is near-entropic. The algorithm remains linear.

The remainder of this paper is organized as follows. Section 2 presents two algorithms for sampling binary words that are we adapt in the subsequent sections. Section 3 details our "entropy recycling" mechanism for uniform random permutations. Section 4 presents the full sampling algorithm and its complexity analysis.

2 Existing Algorithm

2.1 The Fisher-Yates Insertion Algorithm

A standard algorithm for generating a uniform random permutation is the Fisher-Yates shuffle, also known as the Knuth shuffle [7]. While typically used to permute n distinct elements, the procedure can be adapted to generate random binary words of length n with fixed Hamming weight k.

The adaptation exploits the indistinguishability of the 0s. Instead of performing a full shuffle of n elements, we perform a *partial shuffle* restricted to the k ones. The array is initialized with $n-k$ zeros followed by k ones. The algorithm then iterates backwards from n down to $n-k+1$. In each step i, the element at position i (guaranteed to be a 1 prior to the swap) is exchanged with an element at a uniformly selected index $j \in \{1, \cdots, i\}$.

2.2 The Balanced Merge Procedure

The core of the MERGESHUFFLE algorithm from [1] is the BALANCEMERGE procedure, which combines two previously shuffled sub-arrays of size $n/2$ into a single randomly shuffled array of size n. When the input sub-arrays are of equal size (the "balanced" case), this procedure is highly efficient in terms of random bits. This corresponds to generating a balanced binary word (i.e. with $k = n/2$).

Algorithm 1: FY-Binary(n,k)

Input : $n,k \in \mathbb{N}, k \leq n$
Output : Binary word $c \in \{0,1\}^n$

Initialize $c \leftarrow 0^{n-k}1^k$
for $i \leftarrow n$ ***downto*** $n-k+1$ **do**
  $j \leftarrow \text{Uniform}(1,i)$
  $\text{Swap}(c_i, c_j)$
return c

The algorithm proceeds in two distinct phases:

1. **Bit-Driven Selection:** We iterate through the target array. At each step, a random bit is generated (a coin flip). If the bit is 0, we select the next element from the left sub-array; if 1, we select from the right. This loop continues until one of the two sub-arrays is completely exhausted.
2. **Fisher-Yates Completion:** Once one sub-array is empty, the remaining elements from the other sub-array are not simply appended. To ensure a uniform distribution, they are inserted into the final array using a partial Fisher-Yates shuffle. Specifically, each remaining element is swapped with a randomly selected position in the valid range processed so far.

When $k = n/2$ generating a uniform binary balanced word using this method consumes $n + \Theta(\sqrt{n}\log n)$ random bits on average. [1].

Lemma 1. *In the balanced merge procedure called on an even n, the number of elements M that remain in the non-depleted sub-array (which are inserted using the Fisher-Yates procedure) follows the Banach Matchbox [8, VI. 8. (a)] distribution, defined for $m \in [0, n/2]$ by:*

$$\mathbb{P}(M = m) = \binom{n-m}{n/2} \frac{1}{2^{n-m}+1}.$$

Proof. It is exactly the same problem than in [8, VI. 8. (a)]. □

Remark 1. *A consequence is that the expected number of elements that require Fisher-Yates insertion is $\Theta(\sqrt{n})$. Because these remaining elements require drawing discrete uniform integers of increasing ranges, the bit cost for this completion phase is $\Theta(\sqrt{n}\log n)$. This explains why the* BALANCEDMERGE *algorithm is entropically effective: it processes the vast majority of elements using one bit per element, and restricts the more expensive Fisher-Yates fallback to a vanishingly small $O(\sqrt{n})$ fraction of the input.*

Remark 2. *The* BALANCEDMERGE *algorithm can be modified to sample binary words that are not balanced by using Bernoulli variables of parameter k/n instead of coin flips. This leads to linear time, near-entropic algorithms for binary words, as long as k is not too small (as long as $k = \Omega(n^{2/3+\varepsilon})$, with some $\varepsilon > 0$ to be precise) and all the Bernoulli variables are generated in a near-entropic way. This last part can be achieved by generating the Bernoulli variables in fixed sized batch, say by using [12].*

3 Random Bit Recycling

3.1 Random Bit Extraction

In the standard paradigm of random generation, an algorithm consumes a stream of independent, uniform random bits to produce a combinatorial structure [9]. Here, we invert this process: in the case of a uniform random permutation, we map the uniform combinatorial structure back into a sequence of independent, uniform random bits. We call this *random bit extraction* or *random bit recycling* as we give back some of the bits used to generate the structure.

Consider a uniform balanced binary word $W \in \{0,1\}^n$, which consists of exactly $n/2$ zeros and $n/2$ ones. The entropy of this structure is $\log_2 \binom{n}{n/2}$, which by Stirling's approximation is $n - \frac{1}{2}\log_2(\pi n) + O(1)$. Our goal is to extract a sequence of independent random bits from W whose expected length is as close as possible to $\mathscr{H}$, the entropy of the structure, which, by Shannon's source coding theorem [13], is the maximum amount that one could extract.

We achieve this by conceptually reversing the generative BALANCEDMERGE procedure. To do so, we first sample the simulated depletion condition: the number of remaining elements M whose distribution is given by lemma 1, then emulate in reverse the M Fisher-Yates insertions. The resulting word is composed of independent unbiased bits.

The full extraction procedure is detailed in Algorithm 3.

Algorithm 2: BalancedMerge(n), n even

```
   // Generates a balanced word
 1 Let W ← ε, the empty word
 2 Let i_0 ← 0,  i_1 ← 0 // The numbers of
     1s and 0s in W
   // Phase 1:  Coin flipping
 3 Loop:
 4 if RandomBit() = 0 then
 5     if i_0 = n/2 then
 6         break, let b ← 0
 7     else
 8         i_0 ← i_0 + 1
 9         W ← W · 0
10 else
11     if i_1 = n/2 then
12         break, let b ← 1
13     else
14         i_1 ← i_1 + 1
15         W ← W · 1
   // Phase 2:  Fisher-Yates
      insertion for remaining
      elements
16 while i_0 + i_1 < n do
17     W ← W · b
18     j ← Uniform(1, i_0 + i_1)
19     Swap(W[j], W[i_0 + i_1])
20     i_b ← i_b + 1
```

Lemma 2. *When called on W a uniform balanced binary word of length n, algorithm 3 runs in linear time, consumes $O(\sqrt{n}\log n)$ random bits and returns $n - O(\sqrt{n})$ uniform*[1]*, independent bits.*

Proof of Lemma 2. Let W be distributed uniformly over all $\binom{n}{n/2}$ balanced words. We denote by $\tilde{W}$ the returned binary prefix of length $n-M$. We condition our analysis of the loop invariant on the event $(b=0)$ (the case $(b=1)$ is symmetric). Let $W^{(i)}$ be the prefix of length $L = n-i$ after i iterations of the extraction loop.

Claim: For any $i \in \{0,\dots,M\}$, conditional on $b=0$, the prefix $W^{(i)}$ is uniformly distributed among all binary words of length L containing exactly $n/2$ ones.

Proof of claim: We proceed by induction. The base case $W^{(0)} = W$ is trivial. Assume $W^{(i)}$ is uniformly distributed among all binary words of length L with exactly $n/2$ ones. The algorithm selects an index j uniformly from the $n/2 - i$ positions of '0', swaps $W_j^{(i)}$ with $W_L^{(i)}$, and truncates the word to

[1] Standard $O(\cdot)$ notation absorbs the sign, making $+O(f(n))$ and $-O(f(n))$ equivalent. However, we retain the sign to indicate the direction of the deviation.

length $L-1$. Let y be a specific target word of length $L-1$. To obtain y, the state before truncation must be $y\cdot 0$. There are exactly $L=n-i$ valid ways to choose an initial word x and a distinguished '0' that yield $y\cdot 0$ post-swap: $n/2-i$ ways by choosing $x=y\cdot 0$, and $n/2$ ways by choosing an x ending in '1'. Because the prior word and the index choice are uniform, each configuration has probability $((n/2-i)\binom{L}{n/2})^{-1}$. Therefore, the probability of obtaining y is exactly $L\times((n/2-i)\binom{L}{n/2})^{-1}=\frac{L}{n/2-i}\frac{(n/2)!(L-n/2)!}{L!}=\binom{L-1}{n/2}^{-1}$. This confirms the conditional distribution of $W^{(i+1)}$ remains uniform, proving the claim.

Net bit extraction: The variable M (following the Banach Matchbox distribution) and the uniform coin flip b are sampled independently, yielding the joint probability $\mathbb{P}(M=m\cap b=0)=\binom{n-m}{n/2}2^{-(n-m)}\times\frac{1}{2}=\binom{n-m}{n/2}2^{-(n-m+1)}$. Let w be a specific binary sequence of length $n-m$ containing exactly $n/2$ ones. By our previous claim, the conditional probability of obtaining w is exactly $\mathbb{P}(\tilde{W}=w\mid M=m\cap b=0)=\binom{n-m}{n/2}^{-1}$. Multiplying these gives the joint probability of extracting the exact prefix w alongside $b=0$, which cancels the binomial coefficient: $\mathbb{P}(\tilde{W}=w\cap b=0)=\binom{n-m}{n/2}^{-1}\times\binom{n-m}{n/2}2^{-(n-m+1)}=2^{-(n-m+1)}$. This proves that the extracted prefix $\tilde{W}$, when concatenated with the independent bit b, yields a sequence of exactly $n-M+1$ perfectly independent and uniform random bits.

The algorithm runs in $O(n)$ time. As the expected value of the Banach Matchbox distribution yields $\mathbb{E}[M]=O(\sqrt{n})$, this procedure successfully extracts $n-O(\sqrt{n})$ bits on average. Sampling M takes on average $O(\log n)$ bits to run using the method in [3], and each sample of Uniform(S_b) takes $O(\log n)$ bits and time, so the procedure consumes $O(\sqrt{n}\log n)$ random bits, for a net gain of $n-O(\sqrt{n}\log n)$. □

Algorithm 3: ExtractBits(W)

Input : A uniform balanced binary word $W\in\{0,1\}^n$ (exactly $n/2$ zeros and n ones)
Output : A sequence of independent uniform random bits

```
// 1. Sample the simulated depletion state
```
Sample $M\in\{0,\cdots,n/2\}$ according to the Matchbox distribution[2]: $\mathbb{P}(M=m)=\binom{n-m}{n/2}2^{-(n-m)}$
Sample a uniform bit $b\leftarrow$ Uniform$(\{0,1\})$

```
// 2. Reverse Fisher-Yates Extraction
```
for $i\leftarrow 0$ **to** $M-1$ **do**
```
    // Find all active occurrences of bit b
```
    Let $S_b\leftarrow\{k\in\{1,\cdots,n-i\}\mid W_k=b\}$
```
    // Select one occurrence uniformly at random
```
    $j\leftarrow$ Uniform(S_b)
```
    // Swap it to the current end of the active word
```
    Swap(W_j,W_{n-i})

```
// 3. The remaining prefix contains the unconditioned bits
```
return $W[1\cdots n-M]\cdot b$

3.2 Recursive Deconstruction of Uniform Permutations

Using the recycling mechanism for balanced binary words, we deconstruct uniform random permutations. The core idea is to split the permutation of size n into one balanced binary word and two uniform

[2]One can easily show that the matchbox distribution is log-concave, hence it is possible to sample M very efficiently, using the algorithm described in [3].

permutations of size $n/2$. This gives algorithm 4.

The entropy of such a uniform permutation is $\log_2(n!) = n\log_2 n - O(n)$. We demonstrate that our algorithm achieves this optimal bound up to a linear error term.

Theorem 1. *Algorithm 4 has a running time in $O(n\log n)$ and if we denote $B(n)$ the expected number of independent uniform random bits extracted from a uniform random permutation of size n :*

$$B(n) = n\log_2 n - O(n).$$

Proof. By the master theorem for divide-and-conquer algorithms (see [11, Thm. 2.3] for a formal statement), as the toll function is linear and the two sub-problems are of size $n/2$ the runtime of algorithm 4 is in $O(n\log(n))$. For the random bit extraction, the result is an application of the more complex [6, Thm. 1], with $B(n) = n + 2B(\lfloor n/2 \rfloor) - O(\sqrt{n}\log n)$. □

Algorithm 4: DeconstructPermutation(π)

Input : A uniform random permutation π of size $n \geq 2$
Output : A stream of independent uniform random bits

if $n = 2$ **then**
  Yield 0 if $\pi = (1,2)$, 1 if $\pi = (2,1)$
  `// Base case`

`// 1. Odd Size Reduction`
if *n is odd* **then**
  $\pi' \leftarrow \pi$ with the maximum element removed
  Yield DeconstructPermutation(π')

`// 2. Even Size Decomposition (n)`
Initialize an empty word $W \in \varepsilon$
Initialize empty sequences π_{top}, π_{bot}
for $i \leftarrow 1$ **to** n **do**
  if $\pi[i] > n/2$ **then**
    $W[i] \leftarrow 1$
    $\pi_{top} \leftarrow \pi_{top} \cdot (\pi[i] - n/2)$
  else
    $W[i] \leftarrow 0$
    $\pi_{bot} \leftarrow \pi_{bot} \cdot (\pi[i])$

`// 3. Bit Extraction and Recursion`
$Bits \leftarrow$ ExtractBits(W)
Yield $Bits$
DeconstructPermutation(π_{top})
DeconstructPermutation(π_{bot})

4 Entropic Sampling of Sparse Binary Words

4.1 Extracting Fisher-Yates Entropy via Distinguishable Insertions

Morally, the reason the partial Fisher-Yates shuffle fails to be entropic for generating combinations is that it implicitly generates an ordering on the 1s that are inserted, from their order of insertion. This wastes $\log_2(k!)$ random bits on an internal ordering that is ultimately ignored. We can recover this "wasted" entropy by making this ordering explicit, extracting it, and recycling it.

Instead of initializing the array with indistinguishable 1s, we initialize it with distinct identifiers $1, 2, \cdots, k$. We then run the partial Fisher-Yates shuffle as before. Once the k insertions are complete, the array indices containing non-zero elements dictate the generated binary word. Additionally, the sequence of these non-zero elements, when read from left to right, forms a uniform random permutation of the set $\{1, \cdots, k\}$.

By isolating this permutation and feeding it into the DeconstructPermutation algorithm, we "give back" the excess random bits consumed during the generation phase. This coupling of Fisher-Yates generation with recursive deconstruction yields an effective algorithm for the polynomially sparse regime.

Theorem 2. *We assume $k \sim n^\beta$ with a fixed $\beta \in (0,1)$. The generation of a binary word of length n and Hamming weight k via distinguishable Fisher-Yates insertion followed by* DeconstructPermutation

has an expected runtime of $O(n)$. Furthermore, the algorithm is first order entropic: the expected net bit consumption (bits consumed minus bits recycled) is $\log_2 \binom{n}{k} + O(k)$.

Proof. **Time Complexity:** Performing the k Fisher-Yates insertions takes $O(k \log n)$ time, while extracting the random bits from the resulting permutation of size k requires $O(k \log k)$ time. Because $k \sim n^\beta$ for $\beta \in (0,1)$, $k \log n \sim n^\beta \log n = O(n)$ and $k \log k = O(n)$, hence both steps are bounded by $O(n)$, so the overall runtime is in $O(n)$. One can note that this cost could be decreased to sub-linear time if the algorithm returned the sparse set of indices for the 1s rather than explicitly writing out the complete binary word.

Net Bit Complexity: The partial Fisher-Yates shuffle performs k independent draws from discrete uniform distributions of decreasing sizes $n, \ldots, n-k+1$. Using optimal generation algorithms [10, 12], sampling in $[i]$ requires $\log_2 i + O(1)$ bits. The expected total consumption is therefore $\sum_{i=n-k+1}^{n} \log_2 i + O(k) = \log_2(\frac{n!}{(n-k)!}) + O(k)$. The algorithm subsequently recycles a uniform permutation of size k, returning $k \log_2 k - O(k) = \log_2(k!) - O(k)$ random bits to the pool (Theorem 1). Overall, the net bit consumption is exactly $\log_2(\frac{n!}{(n-k)!}) - \log_2(k!) + O(k) = \log_2 \binom{n}{k} + O(k)$, which asymptotically matches the entropy of the structure since $O(k) = o(\log_2 \binom{n}{k})$ in our target regime.

□

4.2 Near Entropic Binary Word Sampling

We now present a linear-time, near-entropic algorithm that generates a uniform binary word of length n with $k \sim n^\beta$ 1s (where $\beta \in (0,1)$). Algorithm 5 simply consumes a small number of random bits, and it does not "give back" random bits. It works by chaining generation and deconstruction phases across $l+1$ discrete steps. The entropy "wasted" on the internal ordering of the 1s inserted during step i is extracted and recycled to power the insertions of step $i+1$. We define the constant $\alpha := \frac{1-\beta}{1-\beta^{l+1}}$. On the i-th step (for $i = 0, \cdots, l$), we insert $k_i = \alpha\beta^i k$ elements. The procedure is described in algorithm 5.

Theorem 3. *We assume $k \sim n^\beta$ with a fixed $\beta \in (0,1)$. For any constant integer $l \geq 0$, algorithm 5 generates a uniform binary word of length n and Hamming weight k in $O(n)$ time. The total cost in random bits is*

$$\frac{1-\beta}{1-\beta^{l+1}} k \log_2 n + O(k), \quad \textit{while} \quad \log_2 \binom{n}{k} = (1-\beta) k \log_2 n + O(k)$$

Proof. **Time Complexity:** The algorithm performs $l+1 = O(1)$ steps. Step i, runs in $O(k \log n)$ + $O(k \log k)$ as it inserts $O(k)$ elements and then recycles a permutation of size $O(k)$. Since $k = n^\beta$ for a constant $\beta < 1$, we have $k \log n = O(n)$, hence the overall expected runtime in $O(n)$.

Bit Complexity: To insert k_i elements into an array of size $\leq n$, the Fisher-Yates procedure requires $k_i \log_2 n + O(k_i)$ bits. Once inserted, these k_i elements form a uniform random permutation. Deconstructing this permutation yields $\log_2(k_i!) = k_i \log_2 k_i - O(k_i)$ independent random bits.

Since $k = n^\beta$, we have $k_i = \alpha\beta^i n^\beta$. Thus, the number of bits recycled from step i is $k_i \log_2(k_i) = \beta k_i \log_2 n - O(k_i)$. Crucially, because $k_{i+1} = \beta k_i$, the recycled bits amount to exactly $k_{i+1} \log_2 n - O(k_i)$. Therefore, summing across all stages gives a total of $k_0 \log_2 n + O(k_0) + \sum_{i=1}^{l} O(k_i) = \alpha k \log_2 n + O(k)$. With $\alpha = \frac{1-\beta}{1-\beta^{l+1}}$, this yields the stated bit complexity. □

Remark 3. *Theorem 3 implies that for any desired tolerance $\varepsilon > 0$, we can bound the multiplicative overconsumption of random bits to $1+\varepsilon$ by demanding: $\frac{1}{1-\beta^{l+1}} \leq 1+\varepsilon$. I.e. by taking $l = \left\lceil \frac{-\log(1+1/\varepsilon)}{\log \beta} - 1 \right\rceil$, l-*CHAINEDRECYCLINGFY *is an ε-entropic, linear time generation algorithm.*

Algorithm 5: l-ChainedRecyclingFY(n,k)

Input : Word length n, k such that $k \sim n^\beta$
Output : A uniform binary word $W \in \{0,1\}^n$ of Hamming weight k

$\alpha \leftarrow \frac{1-\beta}{1-\beta^{l+1}}$
Initialize $W \leftarrow 0^{n-k}$
for $i \leftarrow 0$ **to** l **do**
  $k_i \leftarrow \alpha\beta^i k$
  if $i = 0$ **then**
    $Bits \leftarrow \varepsilon$ the empty word
  else
    `// Recycle the permutation from the previous step`
    $Bits \leftarrow Bits \cdot \text{DeconstructPermutation}(\pi_{i-1})$
  `// Insert distinguishable elements into the remaining zeros`
  Insert elements $\{1,\dots,k_i\}$ into W via Fisher-Yates, consuming the random bits from $Bits$ and drawing fresh bits only when $Bits$ is exhausted
  `// Extract their spatial ordering as a uniform permutation`
  $\pi_i \leftarrow$ The sequence of the newly inserted k_i elements read from left to right
  `// Standardize the inserted elements`
  Replace all elements $\{1,\cdots,k_i\}$ in W with 1
return W

5 Conclusion

We presented a linear-time (in the length of the generated word) algorithm for the uniform generation of fixed Hamming weight binary words in the previously untreated polynomially sparse regime. Our method achieves a random bit consumption that matches the information-theoretic lower bound up to a multiplicative factor of $1+\varepsilon$ for a chosen $\varepsilon > 0$.

The algorithm is based on a random bit recycling procedure. We believe this paradigm of entropy deconstruction is generalizable to other combinatorial contexts. Most notably, we are currently working to adapt this mechanism to recycle the "wasted" entropy from the rejected structures that inherently occur during exact-size generation with Boltzmann-type samplers. We believe that there is practical relevance to this recycling paradigm: it allows for entropic generation when sampling structures in large batches (by pooling recovered bits across iterations), or almost-entropic generation for single instances when the recycling is chained recursively within the generation procedure itself like done in algorithm 5.

More broadly, we believe that evaluating randomized algorithms through the lens of random bit consumption provides a more precise analytical framework than standard runtime analysis alone. Treating random bits as a resource, can expose inefficiencies of standard methods and provides a guide for the conception of new algorithms.

Acknowledgments: The authors would like to express their gratitude to Étienne Objois for many valuable and careful comments. This research was supported by anr-fwf project PAnDAG ANR-23-CE48-0014-01.

References

[1] Axel Bacher, Olivier Bodini, Alexandros Hollender & Jérémie Lumbroso (2015): *Mergeshuffle: a very fast, parallel random permutation algorithm*. *arXiv preprint arXiv:1508.03167*, doi:10.48550/arXiv.1508.03167.

[2] Haozhe Chai, Qianqian Pan & Jun Wu (2025): *A survey of random number generator: Approaches, tests, novel applications in block-chain and AI driven industrial networks*. *Security and Safety* 4, p. 2025015, doi:10.1051/sands/2025015.

[3] Luc Devroye (1987): *A simple generator for discrete log-concave distributions*. *Computing* 39(1), pp. 87–91, doi:10.1007/BF02307716.

[4] Luc Devroye & Claude Gravel (2015): *Sampling with arbitrary precision*. *CoRR* abs/1502.02539, doi:10.48550/arXiv.1502.02539. arXiv:1502.02539.

[5] Thomas L Draper & Feras A Saad (2026): *Efficient Online Random Sampling via Randomness Recycling*. In: *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SIAM, pp. 2473–2511, doi:10.48550/arXiv.2505.18879.

[6] Michael Drmota & Wojciech Szpankowski (2013): *A master theorem for discrete divide and conquer recurrences*. *Journal of the ACM (JACM)* 60(3), pp. 1–49, doi:10.1145/2487241.2487242.

[7] Richard Durstenfeld (1964): *Algorithm 235: Random permutation*. *Commun. ACM* 7(7), p. 420, doi:10.1145/364520.364540.

[8] William Feller (1968): *An Introduction to Probability Theory and Its Applications*, 3rd edition. 1, John Wiley & Sons, New York.

[9] Donald Ervin Knuth & Andrew Chi-Chih Yao (1976): *The complexity of nonuniform random number generation*. Available at https://api.semanticscholar.org/CorpusID:115400979.

[10] Jérémie Lumbroso (2013): *Optimal discrete uniform generation from coin flips, and applications*. *arXiv preprint arXiv:1304.1916*, doi:10.48550/arXiv.1304.1916.

[11] Salvador Roura (2001): *Improved master theorems for divide-and-conquer recurrences*. *Journal of the ACM (JACM)* 48(2), pp. 170–205, doi:10.1145/375827.375837.

[12] Feras Saad, Cameron Freer, Martin Rinard & Vikash Mansinghka (2020): *The Fast Loaded Dice Roller: A Near-Optimal Exact Sampler for Discrete Probability Distributions*. In Silvia Chiappa & Roberto Calandra, editors: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research* 108, PMLR, pp. 1036–1046, doi:10.48550/arXiv.2003.03830. Available at https://proceedings.mlr.press/v108/saad20a.html.

[13] Claude Elwood Shannon (1948): *A mathematical theory of communication*. *The Bell system technical journal* 27(3), pp. 379–423, doi:10.1002/j.1538-7305.1948.tb01338.x.

Growing Binary Trees

Olivier Bodini
Sorbonne Paris Nord, CNRS, LAGA.
Olivier.Bodini@univ-paris13.fr

Antoine Genitrini
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France.
Antoine.Genitrini@lip6.fr

Khaydar Nurligareev
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France.
Khaydar.Nurligareev@lip6.fr

This paper introduces a new combinatorial framework for modeling the growth of binary trees through a discrete evolution process that incorporates a growing rule and an extinction rule. Building upon the theory of increasingly labeled structures and the analysis of polynomial iterates, we extend previous models of increasing trees with label repetitions by allowing growth branches to terminate. This mechanism enables a direct connection between dynamic evolutionary processes and classical unlabeled binary trees. We provide a combinatorial outlook for this model, linking our new approach to essential but traditionally complex parameters such as tree height, the maximum number of leaves at the deepest level (for a given tree size), and the overall tree profile. Our approach reveals structural links with Mandelbrot polynomials and coding theory. Furthermore, we leverage these structural insights to develop an efficient, iterative uniform random sampler for binary trees with a prescribed profile, achieving optimal complexity in both time and space and in random bit consumption.

1 Introduction

The combinatorial modeling of tree evolution serves as a fundamental framework for understanding the growth of discrete structures, bridging the gap between local recursive rules and global structural properties. Historically, a significant portion of the literature has been dedicated to labeled increasing trees, where nodes are integrated into a hierarchy according to specific temporal constraints. Early studies, such as the seminal work of Meir and Moon on recursive trees [18], focused on the typical shape of structures generated by sequential addition. The analysis of functional equations for trees was formalized within the framework of species theory by Bergeron *et al.* [3] and the references therein. Later the combinatorics of increasing trees has been extensively studied in the paper [2]. Those structures comprise for example recursive trees, plane-oriented recursive trees, or binary increasing trees and have been studied through the lens of symbolic functional equations and differential equations. More recently, this research has been extended to include more complex labeling rules and structural synchronizations, notably through the study of increasing Schröder trees and strict monotonic trees [7, 6]. These models are particularly relevant in phylogenetics, where node labels encode the chronology of evolutionary branching, and exhibit deep connections to random graphs, such as the one-to-one correspondence between certain increasing Schröder trees and classical labeled graphs [8].

The mathematical genesis of these evolutionary models lies in the study of recursive processes where each new node is attached to an existing one according to a predefined rule. A seminal contribution to this area is the work of Flajolet and Odlyzko on iterates of polynomials [15] itself extending the paper [1]. Their analysis provided a robust framework for understanding the height and size of trees generated by iterative growth. This approach was then significantly extended in our previous work, On the number of increasing trees with label repetitions [5], which explored discrete growth processes where multiple nodes can share the same label. These models, which involve synchronized branching events, typically lead to purely formal power series where the underlying increasing labeling is intrinsically tied to the structural growth.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 47–56, doi:10.4204/EPTCS.445.7

© O. Bodini, A. Genitrini & K. Nurligareev
This work is licensed under the
Creative Commons Attribution License.

In these previous frameworks, the growth process was essentially monotonic: the structures could only expand, and every leaf remained a potential site for further development. The present paper introduces a paradigm shift by adding an extinction (or death) rule to the evolution process. By allowing branches to terminate, we reconnect these dynamic growth models with classical unlabeled structures, specifically unlabeled binary trees. In this perspective, nodes are categorized into three types: internal nodes, active leaves (anchors), and dead leaves. This distinction allows us to consider essential but traditionally difficult-to-access parameters from a new perspective. By leveraging some symbolic substitution and the iterative dynamics of the resulting polynomials, we provide a combinatorial interpretation of the tree height, the maximum number of the deepest leaves (for a fixed tree size), and the tree profile (the number of leaves at each level). Finally, we leverage these structural insights to design a uniform random sampler for binary tree respecting a given profile, that is optimal in terms of time and space complexity, and random bits consumption.

The paper is organized as follows. In Section 2, we formally define the growth process under consideration and describe the specific tree families and parameters that arise from this framework. Section 3 focuses on the "bushy" trees (those with a maximum number of leaves at the maximum depth for a given tree size), analyzing their behavior as the overall tree size increases. In this context, we establish structural links to meta-Fibonacci sequences and coding theory. In Section 4, we provide a comprehensive outlook on the growth dynamics of binary trees with respect to their height. Finally, leveraging our study of leaf distribution, Section 5 explores the algorithmic implications of this model, specifically focusing on the uniform random sampling of binary trees satisfying a prescribed profile.

2 Combinatorial modeling of a growth process

2.1 Growth process

In our research, we focus on the specific family of binary trees whose nodes are of the following three types: *internal nodes* (•), active leaves or *anchors* (○), and dead leaves or simply *leaves* (□). Each tree is generated by the *growth process* proceeding as follows.

1. In the beginning (at the moment $t = 0$), the tree is reduced to an anchor (○).
2. At every moment $t \in \mathbb{Z}_{>0}$, we replace each anchor with a (dead) leaf (□) or a subtree consisting of an internal node with two anchors attached as children ().

We refer to a tree obtained after several steps of the growth process as a *growing binary tree*. Note, that the internal nodes always have two children in our model. Such trees are sometimes called *locally complete binary trees*.

From the construction, it is seen that the anchors appear exclusively on the last level. An example of a growing binary tree is shown in Figure 1 a) at moment $t = 5$.

2.2 The set of growing binary trees

Let us denote by $t_{n,m}$ the total number of growing binary trees with n internal nodes and m anchors. From the description of the growth process, it is clear that the number of anchors of a tree is even with the exception of the initial tree. Thus, $t_{n,2k-1} = 0$ for all positive integers n and k. The first values of $t_{n,2k}$ are shown in Table 1.

Note that there is no need to additionally count the number of (dead) leaves ℓ, since it is completely determined by two other parameters: if m and n are, respectively, the numbers of anchors and internal

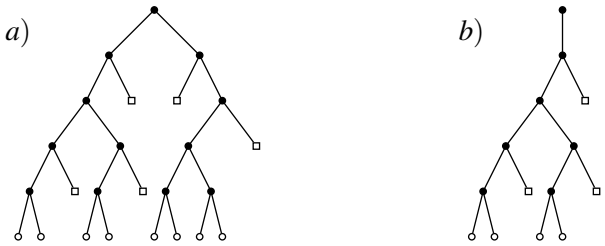


Figure 1: a) A growing binary tree of height $h = 5$ with $m = 8$ anchors. b) A growing binary tree with an additional internal node attached to its root.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$t_{n,2}$	1	2	4	12	32	104	328	1080	3648	12544	43600	153504	546272	1960368
$t_{n,4}$	0	0	1	2	10	24	92	308	1028	3584	12736	45160	161152	581632
$t_{n,6}$	0	0	0	0	0	4	8	40	176	584	2144	8192	30720	112496
$t_{n,8}$	0	0	0	0	0	0	1	2	10	84	282	1048	4368	18224
$t_{n,10}$	0	0	0	0	0	0	0	0	0	0	24	104	352	1616
$t_{n,12}$	0	0	0	0	0	0	0	0	0	0	0	4	36	96
$t_{n,14}$	0	0	0	0	0	0	0	0	0	0	0	0	0	8

Table 1: The first numbers $t_{n,2k}$ of growing binary trees with n internal nodes and $2k$ anchors.

nodes in a growing binary tree, then $\ell = n - m + 1$. Indeed, this relation holds for the initial tree, and the value $n - m - \ell$ is invariant under replacements of the growth process.

Let us further denote by

$$T(x,z) := x + \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} t_{n,2k} x^{2k} z^n$$

the generating function of *active* growing binary trees, that is, trees with at least one anchor. Here, the marking variables x and z count the anchors and internal nodes, respectively. The series $T(x,z)$ satisfies the following relation:

$$T(x,z) = x + T(1 + zx^2, z) - T(1,z)\,. \tag{1}$$

Indeed, replacing an anchor with a leaf corresponds symbolically to the substitution $x \mapsto 1$, while a replacement with a subtree  is represented by the substitution $x \mapsto zx^2$. Note that we have to deduce the term $T(1,z)$ in order to eliminate trees without anchors. By substituting x with the generating function $C(z)$ of Catalan numbers and taking into account that $C(z) = 1 + zC(z)^2$, we conclude that

$$T(1,z) = C(z)\,, \qquad \text{where} \quad C(z) = \frac{1 - \sqrt{1-4z}}{2z}\,.$$

Combinatorially, this fact confirms that the growing binary trees without anchors are indeed all binary trees counted by the series $C(z)$. In particular, the column sums in Table 1 are Catalan numbers:

$$\sum_{m=1}^{n+1} t_{n,m} = C_n\,.$$

Another immediate consequence of Relation (1) is the following recurrent formula for $t_{n,2k}$:

$$t_{1,2} = 1\,, \qquad t_{n,2k} = \sum_{m=1}^{\infty} \binom{2m}{k} t_{n-k,2m} \quad \text{for } n > k \geqslant 1\,. \tag{2}$$

Relation (2) also has a combinatorial explanation: to get a tree with n internal nodes and $2k$ anchors, we have to replace k anchors of the tree obtained in the previous step of the growth process with an internal node with two children; the other anchors become leaves. The binomial coefficient $\binom{2m}{k}$ then represents the number of choices of k anchors to be replaced.

2.3 Cumulative anchor counting

To enumerate the total number of leaves in active growing binary trees, we use the series $\tilde{T}(1,z)$, where

$$\tilde{T}(x,z) := \frac{\partial}{\partial x} T(x,z) = 1 + \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} 2k t_{n,2k} x^{2k-1} z^n .$$

By differentiating Relation (1) with respect to x, we obtain the following relation for $\tilde{T}(x,z)$:

$$\tilde{T}(x,z) = 1 + 2xz\tilde{T}(1+zx^2,z) . \tag{3}$$

This relation allows us to express the series $\tilde{T}(x,z)$ in terms of polynomials $p_h(x,z)$ that count all growing binary trees that can be obtained after the first h steps of the growing process. In fact, the polynomials $p_h(x,z)$ are determined by the following recurrence relation:

$$p_0(x,z) = x , \qquad p_{h+1}(x,z) = p_h(1+zx^2,z) \quad \text{for } h \geqslant 0 . \tag{4}$$

Therefore, taking into account that $1+zx^2 = p_1(x,z)$, we can iterate Relation (3) according to (4). By passing to the limit, this gives us the following expression:

$$\tilde{T}(x,z) = 1 + \sum_{i=1}^{\infty} (2z)^i \prod_{j=0}^{i-1} p_j(x,z) .$$

Note that the polynomials $p_h(x,z)$ are related to the so-called (shifted) *Mandelbrot polynomials* [9]:

$$M_h(z) = z\, q_h(1,z) .$$

From a combinatorial point of view, Mandelbrot polynomials count growing binary trees with an additional internal node attached to the root; see Figure 1 b). By substituting $x = 1$ into the relation for $\tilde{T}(x,z)$, we obtain an expression for $\tilde{T}(1,z)$ in terms of Mandelbrot polynomials:

$$\tilde{T}(1,z) = 1 + \sum_{i=1}^{\infty} 2^i \prod_{j=0}^{i-1} M_j(z) .$$

Remark 2.1. The formal link to Mandelbrot polynomials highlights a phase transition in the tree growth. The discrete dynamic of the iterates $p_h(1,z)$ has fixed points satisfying $x = 1+zx^2$, which results in a critical bifurcation at $z_c = 1/4$. For $|z| < 1/4$, the process contracts towards the Catalan generating function $C(z)$ (almost-sure extinction), whereas for $z > 1/4$, it diverges (explosive growth). Consequently, the analyticity domain of the process is intrinsically bounded by the main cardioid of the Mandelbrot set.

3 Maximal number of anchors for a given tree size

Let us turn our focus to the columns of Table 1. We can see that in every column the number of nonzero elements is finite. For the nth column, we denote this number a_n. In other words,

$$a_n = \max\{k\colon t_{n,2k} > 0\}\,.$$

The first several values of the sequence $(a_n)_{n\geqslant 1}$ are

$$(a_n)_{n\geqslant 1} = 1,1,2,2,2,3,4,4,4,4,5,6,6,7,8,8,8,8,8,9,10,10,11,12,12,12,13,\ldots$$

The purpose of this section is to provide a description of the sequence $(a_n)_{n\geqslant 1}$ and to show that this sequence is actually A006949 from [19]. First, we show that $(a_n)_{n\geqslant 1}$ can be defined independently of growing binary trees.

Lemma 3.1. The sequence $(a_n)_{n\geqslant 1}$ satisfies the following recurrent relation:

$$a_1 = 1\,, \qquad a_n = \max\{k\colon k \leqslant 2a_{n-k}\} \quad \text{for } n > 1\,.$$

Proof. Using the sequence $(a_n)_{n\geqslant 1}$, Relation (2) can be rewritten as

$$t_{n,2k} = \sum_{m=\lceil k/2\rceil}^{a_{n-k}} \binom{2m}{k} t_{n-k,2m}\,.$$

Hence, the condition $t_{n,2k} \neq 0$ means that there exists a positive integer m such that $k \leqslant 2m \leqslant 2a_{n-k}$, which is equivalent to the condition $k \leqslant 2a_{n-k}$. □

From Lemma 3.1, it follows by induction that the sequence $(a_n)_{n\geqslant 1}$ is increasing in the sense that, for any $n \in \mathbb{Z}_{>0}$, we have $a_{n+1} = a_n$ or $a_{n+1} = a_n + 1$. This fact, coupled with the initial condition $a_1 = 1$, means that the behavior of the sequence $(a_n)_{n\geqslant 1}$ is completely determined by the number of repetitions:

$$a_n = \max\left\{k\colon \sum_{i=1}^{k} b_k \geqslant n\right\},$$

where b_n is the number of elements in $(a_n)_{n\geqslant 1}$ that are equal to n:

$$b_n = \#\{k\colon a_k = n\}\,.$$

The first several values of the sequence $(b_n)_{n\geqslant 1}$ are

$$(b_n)_{n\geqslant 1} = 2,3,1,4,1,2,1,5,1,2,1,3,1,2,1,6,1,2,1,3,1,2,1,4,1,2,1,\ldots$$

Proposition 3.2. The sequence $(b_n)_{n\geqslant 1}$ satisfies the following relation:

$$b_n = \begin{cases} p+2 & \text{if } n = 2^p,\\ p+1 & \text{if } n = 2^p a,\ a \text{ is odd},\ a > 1\,. \end{cases}$$

In particular, we have $b_{2n} = b_n + 1$ for even indices, $b_{2n+1} = 1$ for odd indices greater than 1, and $b_1 = 2$.

Proposition 3.2 allows us to claim that $(a_n)_{n\geqslant 1}$, with an additional value $a_0 = 1$, coincides with the entry `A006949` from OEIS [19] and is known as a *meta-Fibonacci sequence* (for the sequence $(b_n)_{n\geqslant 1}$, see the entries `A135560` and `A241235`). In particular, due to [22] and [21], we obtain the following properties of this sequence.

Corollary 3.3. The sequence $(a_n)_{n\geqslant 1}$ satisfies the following recurrent relation:

$$a_0 = a_1 = a_2 = 1\,, \qquad a_n = a_{n-1-a_{n-1}} + a_{n-2-a_{n-2}} \quad \text{for } n > 2\,.$$

Its generating function and asymptotic behavior satisfy, respectively,

$$\sum_{n=0}^{\infty} a_n z^n = z \sum_{n=0}^{\infty} \prod_{i=1}^{n} (z + z^{2^i}) \qquad \text{and} \qquad \lim_{n\to\infty} \frac{a_n}{n} = \frac{1}{2}\,.$$

Note that the sequence $(a_n)_{n\geqslant 1}$ also admits another combinatorial interpretation in terms of infinite binary trees [21, 13] and is related to extremal compact codes [17].

4 Growing binary trees of fixed height

In this section, we study the behavior of growing binary trees with respect to their heights.

Lemma 4.1. For any active growing binary tree, the number n of its internal nodes, the number $2k$ of its anchors, and its height h satisfy

$$h \leqslant n - k + 1 \leqslant 2^{h-1}\,.$$

Both inequalities are sharp.

Let us denote by $t_{n,m,h}$ the number of active growing binary trees of height h with n internal nodes and m anchors. Clearly, apart from the case $h = 0$, the number of anchors in a tree of height h is even. For example, for $h = 4$, the values of $t_{n,2k,h}$ are shown in Figure 2.

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	n
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	
6	0	0	0	0	0	0	0	0	0	0	0	0	4	28	0	0	
5	0	0	0	0	0	0	0	0	0	0	0	24	56	0	0	0	
4	0	0	0	0	0	0	0	0	2	6	60	70	0	0	0	0	
3	0	0	0	0	0	0	0	8	24	80	56	0	0	0	0	0	
2	0	0	0	0	0	4	16	36	60	28	0	0	0	0	0	0	
1	0	0	0	0	8	16	24	24	8	0	0	0	0	0	0	0	

Figure 2: Numbers $t_{n,2k,h}$ of growing binary trees of height $h = 4$.

Given a positive integer h, we also introduce the domain S_h of nonzero values of $t_{n,2k,h}$:

$$S_h = \{(n,k)\colon t_{n,2k,h} \neq 0\}\,. \tag{5}$$

From Lemma 4.1, it follows that S_h is finite. The following results describe its behavior.

Proposition 4.2. The right boundary Γ_h of the nonzero domain S_h (blue cells in Figure 2) satisfies

$$\Gamma_h = \{(2^{h-1} - 1 + i, i)\colon 1 \leqslant i \leqslant 2^{h-1}\}\,. \tag{6}$$

In particular, all elements of Γ_h belong to the line $n - k = 2^{h-1} - 1$.

Proposition 4.3. The upper boundary Λ_h of the nonzero domain S_h (red cells in Figure 2) satisfies

$$\Lambda_h = \{(h,\hat{a}_2),(h+1,\hat{a}_3),\dots,(2^h-2,\hat{a}_{2^h-h}),(2^h-1,\hat{a}_{2^h-h+1})\}, \tag{7}$$

where

$$\hat{a}_n = \max\left\{k\colon \sum_{i=1}^{k}\hat{b}_k \geqslant n\right\},$$

and $\hat{b}_n$ is the *ruler function*, that is, the number of times $2n$ can be divided by two.

Corollary 4.4. The area of the nonzero domain S_h is equal to $2^{h-2}(2^{h-1}-h+2)$.

Proposition 4.5. The scaling limit of the nonzero domain S_h (normalized by 2^{h-1}), as $h\to\infty$, is the triangle with vertices $(0,0)$, $(1,0)$, and $(2,1)$.

5 Random sampling

5.1 Combinatorial context

We are focusing on binary trees, where the atoms under consideration are the leaves. An analogous approach can be applied to binary trees according to their number of internal nodes.

Definition 5.1. The *level* of a node in a binary tree is the distance (in terms of traversed edges) from the root to that node. The root is at level 0.

Definition 5.2. The profile of a binary tree is the sequence $(\ell_0,\ell_1,\dots,\ell_h)$ of the nonnegative numbers of leaves at each level. The index h corresponds to the *height* of the tree. The *size* of the tree (*i.e.* the total number of leaves it contains) is equal to the sum $\sum_i \ell_i$.

For a tree reduced to a single leaf (the root), we get the profile (1). For any other binary tree with height $h>0$, we have $\ell_0=0$ (the root is not a leaf).

Fact 5.3 (Kraft-McMillan equality)**.** There exists a binary tree with profile $(\ell_0,\ell_1,\dots,\ell_h)$ if and only if

$$\sum_{i=0}^{h}\frac{\ell_i}{2^i}=1.$$

This is a classical result in the context of code theory; see, for instance, [12].

Proposition 5.4. Let h be a positive integer and T be a binary tree with profile $(\ell_0,\ell_1,\dots,\ell_h)$. Its internal node profile (*i.e.* the sequence of the number of internal nodes at each level) is given by $(i_0,i_1,\dots,i_{h-1})$ satisfying

$$\begin{cases} i_0 = 1 \\ i_k = 2i_{k-1}-\ell_k, \quad \text{for } 1\leqslant k\leqslant h-1. \end{cases}$$

Definition 5.5. We say that a profile $(\ell_0,\ell_1,\dots,\ell_h)$ is *valid* if there exists at least one binary tree having this profile.

Proposition 5.6. There exists a unique binary tree, which is reduced to a single leaf, having the profile (1). Let $(\ell_0,\ell_1,\dots,\ell_h)$, with $h\geqslant 1$, be a valid profile. The number of binary trees having this profile is

$$\prod_{k=0}^{h-1}\binom{2i_k}{\ell_{k+1}},$$

where the sequence $(i_k)_{k=0\dots h-1}$ is defined in Proposition 5.4.

5.2 Uniform random sampling of trees with a given profile

A natural way to sample binary trees with a given profile is the following. Start with the root of the tree; if the latter is a leaf, then stop; otherwise, we have to decide what are the valid profiles of the two children. In the classical recursive method [20, 16], even for sampling trees of a given size, the necessary computations are heavy. In our case, the combinatorial complexity is much higher, since we have to consider the profiles of the two subtrees, which are not independent (like in the case of size) but are also constrained to be valid profiles. In the context of binary decision diagram sampling [10], such a top-down approach has been addressed to generate structures with a given profile, but it has finally been overtaken by an iterative approach [11], level by level, only focusing on a given level at a time without the need to consider the rest of the profile.

Here in the context of binary trees, such an iterative approach can be developed as well (and will be more efficient than the one for decision diagrams), but an even more efficient approach is possible, looking to the tree from the bottom to the top (and for a given level, from left to right).

We can sample a binary tree with a given profile by starting from the last level, and then iteratively going up to the root, level by level, and for each level. In fact, in Proposition 5.6, we can look to the product formula counting the trees with a given profile from the last factor to the first ones, *i.e.* from the deepest levels of the trees to the root level.

Algorithm 1 Uniform random sampling of a tree with profile $(\ell_0, \ell_1, \ldots, \ell_h)$

Input: $L = (\ell_0, \ell_1, \ldots, \ell_h)$
Output: a uniform binary tree with profile L

1: **function** UNIFORM_TREE(L)
2: T is a sequence of $\ell_h/2$ nodes, each one pointing to 2 leaves
3: **for** i from $h-1$ to 0 with step -1 **do**
4: T' = shuffling T with a sequence of ℓ_i leaves
5: T is a sequence of $len(T')/2$ nodes with the i-th node pointing to nodes $2i$ and $2i+1$ of T'
6: **return** first (and single) node of T, corresponding to the root of the tree

Function *len* returns the number of elements of a sequence.

The shuffling operation is presented in detail in Algorithm 3 from [4].

Proposition 5.7. Algorithm 1 is correct and optimal in time, space, and random bit consumption.

Optimality stands for the time complexity, for the space complexity, and for the consumption of random bits (whose derivation comes from Proposition 5.6 and the shuffling operation from [4]).

6 Perspectives

As a primary perspective, the connection between Mandelbrot polynomials and the framework of Flajolet and Odlyzko [14] allows for a precise characterization of the limit distribution of the deepest leaves. This approach provides a new way to analyze the tree's boundary and could be generalized to other substitution rules, offering a unified combinatorial view of extinction-based growth processes.

Algorithmically, the iterative bottom-up sampling introduced in Section 5 can be broadly extended. Natural next steps include the uniform generation of binary forests and the sampling of trees with partial profiles (e.g., profiles with "holes" representing intermediate levels of arbitrary sizes).

Acknowledgments This work is partially funded by ANR-FWF project PANDAG ANR-23-CE48-0014 and ANR project COMETA-GAE ANR-25-CE48-0602.

References

[1] A.V. Aho & N.J.A. Sloane (1973): *Some doubly exponential sequences*. *Fibonacci Q.* 11, pp. 429–437, doi:10.1080/00150517.1973.12430815.

[2] F. Bergeron, P. Flajolet & B. Salvy (1992): *Varieties of Increasing Trees*. In: *CAAP*, pp. 24–48, doi:10.1007/3-540-55251-0_2.

[3] F. Bergeron, G. Labelle & P. Leroux (1988): *Functional Equations for Data Structures*. In: *Proceedings of the 5th Annual Symposium on Theoretical Aspects of Computer Science*, STACS '88, Springer-Verlag, Berlin, Heidelberg, p. 73–80.

[4] O. Bodini, M. Dien, A. Genitrini & F. Peschanski (2017): *Entropic Uniform Sampling of Linear Extensions in Series-Parallel Posets*. In P.l Weil, editor: *12th International Computer Science Symposium in Russia, CSR*, *Lecture Notes in Computer Science* 10304, Springer, pp. 71–84, doi:10.1007/978-3-319-58747-9_9.

[5] O. Bodini, A. Genitrini, B. Gittenberger & S. Wagner (2020): *On the number of increasing trees with label repetitions*. *Discrete Mathematics* 343(8), p. 111722, doi:10.1016/j.disc.2019.111722.

[6] O. Bodini, A. Genitrini, C. Mailler & M. Naima (2022): *Strict monotonic trees arising from evolutionary processes: Combinatorial and probabilistic study*. *Advances in Applied Math.* 133, p. 102284, doi:10.1016/j.aam.2021.102284.

[7] O. Bodini, A. Genitrini & M. Naima (2019): *Ranked Schröder Trees*. In: *Proceedings of the Sixteenth Workshop on Analytic Algorithmics and Combinatorics, ANALCO 2019*, pp. 13–26, doi:10.1137/1.9781611975505.2.

[8] O. Bodini, A. Genitrini & M. Naima (2022): *A Combinatorial Link Between Labelled Graphs and Increasingly Labelled Schröder Trees*. In: *LATIN 2022: Theoretical Informatics - 15th Latin American Symposium*, *Lecture Notes in Computer Science* 13568, Springer, pp. 493–509, doi:10.1007/978-3-031-20624-5_30.

[9] N. Calkin, E. Chan & R.M. Corless (2021): *Some facts and conjectures about Mandelbrot polynomials*. *Maple Transactions* 1(1), p. 13, doi:10.5206/mt.v1i1.14037. 14037.

[10] J. Clément & A. Genitrini (2020): *Binary Decision Diagrams: From Tree Compaction to Sampling*. In: *LATIN 2020: Theoretical Informatics - 14th Latin American Symposium*, *Lecture Notes in Computer Science* 12118, Springer, pp. 571–583, doi:10.1007/978-3-030-61792-9_45.

[11] J. Clément & A. Genitrini (2023): *An Iterative Approach for Counting Reduced Ordered Binary Decision Diagrams*. In: *48th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, *Leibniz International Proceedings in Informatics* 272, pp. 36:1–36:15, doi:10.4230/LIPIcs.MFCS.2023.36.

[12] T.M. Cover & J.A. Thomas (2006): *Elements of information theory*, 2nd ed. edition. Hoboken, NJ: John Wiley & Sons, doi:10.1002/047174882X.

[13] A. Das (2011): *A combinatorial approach to Tanny sequence*. *Discrete Math. Theor. Comput. Sci.* 13(2), pp. 97–108, doi:10.46298/dmtcs.547.

[14] P. Flajolet & A. Odlyzko (1982): *The average height of binary trees and other simple trees*. *J. Comput. Syst. Sci.* 25(2), pp. 171–213, doi:10.1016/0022-0000(82)90004-6.

[15] P. Flajolet & A. M. Odlyzko (1984): *Limit distributions for coefficients of iterates of polynomials with applications to combinatorial enumerations*. *Math. Proc. Camb. Philos. Soc.* 96(2), pp. 237–253, doi:10.1017/S0305004100062149.

[16] P. Flajolet, P. Zimmerman & B. van Cutsem (1994): *A calculus for the random generation of labelled combinatorial structures*. *Theor. Comput. Sci.* 132(1-2), pp. 1–35, doi:10.1016/0304-3975(94)90226-7.

[17] B. Jackson & F. Ruskey (2006): *Meta-Fibonacci sequences, binary trees and extremal compact codes*. *Electron. J. Comb.* 13(1), pp. research paper r26, 13, doi:10.37236/1052.

[18] A. Meir & J. W. Moon (1978): *On the altitude of nodes in random trees*. *Can. J. Math.* 30(5), pp. 997–1015, doi:10.4153/CJM-1978-085-0.

[19] N. J. A. Sloane and The OEIS Foundation Inc. (2024): *The On-Line Encyclopedia of Integer Sequences*. Published electronically at https://oeis.org.

[20] A. Nijenhuis & H.S. Wilf (1975): *Combinatorial algorithms*. Computer science and applied mathematics, Academic Press, New York, NY.

[21] F. Ruskey & C. Deugau (2009): *The combinatorics of certain k-ary meta-Fibonacci sequences*. *J. Integer Seq.* 12(4), p. 36.

[22] S.M. Tanny (1992): *A well-behaved cousin of the Hofstadter sequence*. *Discrete Math.* 105(1-3), pp. 227–239, doi:10.1016/0012-365X(92)90145-6.

On the Counting Sequence of Z-convex Polyominoes

Luca Castelli Paolo Massazza
Department of Theoretical and Applied Sciences
University of Insubria
Varese, Italy
lcastelli@studenti.uninsubria.it paolo.massazza@uninsubria.it

The degree of convexity of a convex polyomino P is the smallest integer k such that any two cells of P can be joined by a monotone path inside P with at most k changes of direction. In this paper we present a set of formulas and equations that are the basis of a C++ program that allows you to compute the longest counting sequence known to date (with respect to the area) of convex polyominoes of degree of convexity at most 2.

1 Introduction

A polyomino is a geometrical figure consisting of a finite set of connected unitary squares (called cells) in the plane $\mathbb{Z} \times \mathbb{Z}$, considered up to translations. The problem of counting polyominoes with n cells (*i.e.* of area n) is probably one of the fundamental open problems in combinatorial geometry (see problem 37 in [9]). Recently, the problem has been solved up to $n \leq 70$ [1]. Due to the difficulty of the problem, simpler classes of polyominoes have been introduced and studied. In particular, the class of convex polyominoes (polyominoes where the intersection with an infinite horizontal or vertical stripe is a finite segment) and some of its subclasses have been thoroughly investigated [2, 3, 14, 5]. Sometimes the generating function with respect to the area $\phi_C(x) = \sum_{n \geq 0} c_n x^n$ of a class C of polyominoes is known, see for instance [13]. This often allows one to get an estimate of the asymptotic growth of the number of polyominoes of area n in C using standard analytical methods.

When $\phi_C(x)$ is not known (neither explicitly nor implicitly) an algorithm for the exhaustive generation of C may produce c_n for small (but still significant) values of n. For instance, Constant Amortized Time (CAT) algorithms for generating several classes of polyominoes have been developed, where the exhaustive generation is done by semiperimeter [7, 6] or by area [10, 11].

In this paper we consider the class ZConv of Z-convex polyominoes. This class has been introduced in [8] and studied in [16]. It contains all convex polyominoes P with the property that any two cells of P can be joined by a path in P with at most two changes of direction. Recently, a combinatorial approach [12] has been used to prove that Z-convex polyominoes can be counted with respect to the area in polynomial time. By slightly changing some of the combinatorial decompositions in [12], we have obtained a new set of recurrence equations and formulas that can be implemented more efficiently. We developed a C++ program that allowed to compute $|\mathsf{ZConv}(n)|$ up to $n = 75$.

2 Basic Notions

Let P be a polyomino with an $r \times c$ minimal bounding rectangle. We number the columns (resp., rows) of P from left to right (resp., from bottom to top). The first (resp., last) column of P is denoted by $\textsc{first}(P)$ (resp., $\textsc{last}(P)$). Lastly, we indicate by $\mathrm{A}(P)$ the *area* of P, that is, the number of its cells.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 57–65, doi:10.4204/EPTCS.445.8

© L. Castelli & P. Massazza
This work is licensed under the
Creative Commons Attribution License.

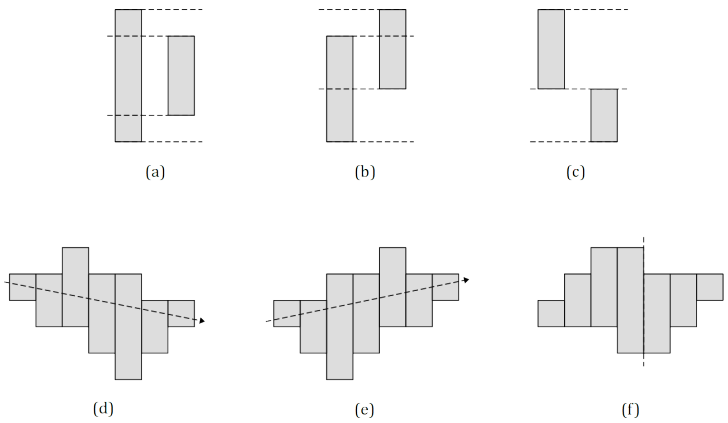


Figure 1: An included column (a), two overlapping columns (b), two disjoint columns (c), a descending polyomino (d), an ascending polyomino (e) and a polyomino in LR (f).

We consider a Z-convex polyomino as the result of the concatenation of polyominoes belonging to well-known subclasses of convex polyominoes (stacks S, rectangles T, Ferrers diagrams F, parallelograms C). We denote by $\mathrm{low}(j)$ (resp., $\mathrm{high}(j)$) the row index of the bottom cell (resp., top cell) of column j. We introduce some binary relations on the set of columns of a convex polyomino, which play a special role in the decomposition of a Z-convex polyomino.

Definition 1 *Let i and j be two columns of a convex polyomino P. We say that*

- i includes j*, denoted by $j \subseteq i$, if and only if $\mathrm{low}(i) \leq \mathrm{low}(j)$ and $\mathrm{high}(i) \geq \mathrm{high}(j)$, see Fig. 1 (a);*
- *i and j are* overlapping*, denoted by $i \uparrow\downarrow j$, if and only if $\mathrm{low}(j) < \mathrm{low}(i) \leq \mathrm{high}(j) < \mathrm{high}(i)$ or $\mathrm{low}(i) < \mathrm{low}(j) \leq \mathrm{high}(i) < \mathrm{high}(j)$, see Fig. 1 (b);*
- *i and j are* disjoint*, denoted by $i \asymp j$, if and only if $\mathrm{low}(i) > \mathrm{high}(j)$ or $\mathrm{low}(j) > \mathrm{high}(i)$, see Fig. 1 (c).*

In the following, an index $^\bullet$ (resp., $^\circ$) to the right of a class indicates that all polyominoes in the class have (resp., do not have) disjoint columns. Thus, for any class A we have the partition $\mathsf{A} = \mathsf{A}^\bullet \cup \mathsf{A}^\circ$.

We define the *center* $\mathscr{C}(P)$ of a convex polyomino P as the rightmost column e of P such that $c \subseteq e$ for all columns c to the left of e. We also define the *pivot* $\mathscr{P}(P)$ of P as the leftmost column j of P such that $j \uparrow\downarrow \mathscr{C}(P)$ (possibly $\mathscr{P}(P)$ does not exist, that is, $\mathscr{P}(P) = \varepsilon$). If $\mathscr{P}(P) \neq \varepsilon$ we say that P is *descending* (resp., *ascending*) if $\mathrm{low}(\mathscr{C}(P)) > \mathrm{low}(\mathscr{P}(P))$ (resp., $\mathrm{low}(\mathscr{C}(P)) < \mathrm{low}(\mathscr{P}(P))$), see Fig. 1 (d) (resp., (e)). The set of descending (resp., ascending) convex polyominoes is indicated by DConv (resp., AConv). If $\mathscr{P}(P) = \varepsilon$ then $\mathscr{C}(P)$ includes all columns of P. This means that P is in $\mathsf{T} \cup \mathsf{F} \cup \mathsf{S} \cup \mathsf{LR}$ where LR is the class containing all convex polyominoes that are the concatenation of two polyominoes, $P = P_1 \cdot P_2$, with $P_1 \in \mathsf{S} \cup \mathsf{F}$, $P_2 \in \mathsf{T} \cup \mathsf{S} \cup \mathsf{F}$ and $\textsc{first}(P_2) \subsetneq \textsc{last}(P_1)$. We indicate by DConv_2 (resp., AConv_2) the set of descending (resp., ascending) polyominoes of degree of convexity 2. Thus, one has

$$\mathsf{ZConv} = \mathsf{T} \cup \mathsf{S} \cup \mathsf{F} \cup \mathsf{LR} \cup \mathsf{AConv}_2 \cup \mathsf{DConv}_2$$

where the unions are disjoint. The counting problem for ZConv is therefore reduced to computing $|\mathsf{DConv}_2(n)|$ (by symmetry one has $|\mathsf{DConv}_2(n)| = |\mathsf{AConv}_2(n)|$), since the counting problem for $\mathsf{T} \cup \mathsf{S} \cup \mathsf{F} \cup \mathsf{LR}$ is easily solved in polynomial time (see [12, Sect. 4]). Lastly, we recall [4, Thm. 1] that $P \in \mathsf{ZConv}$ if and only if for any two disjoint columns i and j of P there exists a column k, with $i < k < j$, such that $i \subsetneq k$ and $j \subsetneq k$. This characterization is the basis of the combinatorial decompositions that lead to the formulas for computing $|\mathsf{DConv}_2(n)|$.

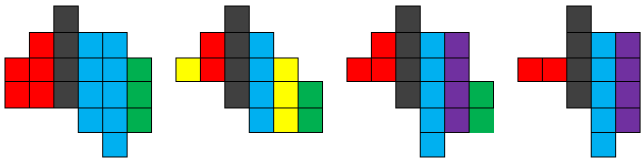


Figure 2: From left to right: $P \in \mathsf{Z}_2^\circ \setminus \text{u-}\mathsf{Z}_2^\circ$, $P \in \mathsf{Z}_2^\bullet \setminus \text{t-}\mathsf{Z}_2^\bullet$, $P \in \text{t-}\mathsf{Z}_2^\bullet$ and $P \in \text{u-}\mathsf{Z}_2^\circ$.

3 Polyominoes decomposition

We consider the *standard decomposition* of $P \in \mathsf{DConv}$ as defined in [12]:

Definition 2 *[standard decomposition] A polyomino $P \in \mathsf{DConv}$ is uniquely decomposed as $P = L \cdot F \cdot C \cdot R$ for suitable polyominoes $L \in \mathsf{S} \cup \mathsf{T} \cup \mathsf{F}$, $F \in \mathsf{F} \cup \mathsf{T} \cup \{\varepsilon\}$, $C \in \mathsf{C} \cup \mathsf{T} \cup \mathsf{F}$, and $R \in \mathsf{S} \cup \mathsf{T} \cup \mathsf{F} \cup \{\varepsilon\}$ with $\textsc{last}(L) = \mathscr{C}(P)$, $\textsc{first}(C) = \mathscr{P}(P)$ and $low(\textsc{last}(C)) < low(\textsc{first}(R))$.*

Note that any column e of F (if $F \neq \varepsilon$) satisfies $\text{low}(\textsc{last}(L)) = \text{low}(e)$ and that $\textsc{last}(F) \uparrow\downarrow \textsc{first}(C)$ or $\textsc{last}(F) \subsetneq \textsc{first}(C)$. Furthermore, if $R \neq \varepsilon$ one has $\textsc{first}(R) \subsetneq \textsc{last}(C)$. We point out that by [4, Thm. 1], given $P \in \mathsf{DConv}_2$ and its standard decomposition $P = L \cdot F \cdot C \cdot R$, each column c of P to the right of $\textsc{last}(L)$ satisfies the relation $c \uparrow\downarrow \textsc{last}(L)$ or $c \subsetneq \textsc{last}(L)$.

We indicate by LCR_2 (resp., LC_2, LFCR_2, LFC_2) the subset of DConv_2 containing polyominoes whose standard decomposition is $L \cdot C \cdot R$ (resp., $L \cdot C$, $L \cdot F \cdot C \cdot R$, $L \cdot F \cdot C$). Polyominoes in DConv will be recursively decomposed into polyominoes belonging to three particular subclasses, see Fig. 2. The first one is the class Z_2 (slightly modified compared to that defined in [12]).

Definition 3 (Z_2) *Z_2 is the set of all P in $\mathsf{LCR}_2 \cup \mathsf{LC}_2$ such that: 1) for all columns l to the left of $\mathscr{C}(P)$, one has $A(l) < A(\mathscr{C}(P))$ and $l \subsetneq \mathscr{P}(P)$; 2) $c \uparrow\downarrow \mathscr{C}(P)$ for all columns c of R (if $R \neq \varepsilon$).*

The following two classes are new with respect to [12] and are introduced to obtain simpler equations.

Definition 4 (t-$\mathsf{Z}_2^\bullet$) *The set t-$\mathsf{Z}_2^\bullet$ contains all $P \in \mathsf{Z}_2^\bullet$ that can be written as $P = L \cdot C \cdot E \cdot R$ for suitable $L \in \mathsf{S} \cup \mathsf{F}$, $C \in \mathsf{C} \cup \mathsf{T} \cup \mathsf{F}$, $E \in \mathsf{S} \cup \mathsf{T} \cup \mathsf{F} \cup \{\varepsilon\}$, $R \in \mathsf{S} \cup \mathsf{T} \cup \mathsf{F}$ such that:*

- $\textsc{last}(L) = \mathscr{C}(P)$, $\textsc{first}(C) = \mathscr{P}(P)$ *and* $low(\textsc{last}(C)) < low(\textsc{first}(E))$ *(if $E \neq \varepsilon$);*
- *L has at least two columns and $l \subsetneq \textsc{last}(C \cdot E)$ for all columns l to the left of $\mathscr{C}(P)$;*
- $low(\textsc{last}(C \cdot E)) \leq low(\textsc{first}(R))$ *and* $\textsc{first}(L) \asymp \textsc{first}(R)$.

Definition 5 (u-Z_2°) *The set u-Z_2° contains all $P \in \mathsf{Z}_2^\circ$ that can be written as $P = L \cdot C \cdot E$ for suitable $L \in \mathsf{S} \cup \mathsf{F}$, $C \in \mathsf{C} \cup \mathsf{T} \cup \mathsf{F}$ and $E \in \mathsf{S} \cup \mathsf{T} \cup \mathsf{F} \cup \{\varepsilon\}$ such that:*

- $\textsc{last}(L) = \mathscr{C}(P)$, $\textsc{first}(C) = \mathscr{P}(P)$ *and* $low(\textsc{last}(C)) < low(\textsc{first}(E))$ *(if $E \neq \varepsilon$);*
- *L has at least two columns and $l \subsetneq \textsc{last}(C \cdot E)$ for all columns l to the left of $\mathscr{C}(P)$.*

3.1 Computing $|\mathsf{DConv}_2(n)|$

The first step is to refine the standard decomposition of $P \in \mathsf{DConv}_2$ in order to obtain $P' \in \mathsf{Z}_2$ and some other simpler polyominoes. Let $P = L \cdot F \cdot \mathscr{P}(P) \cdot C \cdot R$. We split L into three parts, $L = L_1 \cdot L_2 \cdot \mathscr{C}(P)$, where $\textsc{last}(L_1) \subsetneq \mathscr{P}(P)$ and $\textsc{first}(L_2) \uparrow\downarrow \mathscr{P}(P)$. Then, we split R into two parts, $R = R_1 \cdot R_2$, with $\textsc{last}(R_1) \uparrow\downarrow \mathscr{C}(P)$ and $\textsc{first}(R_2) \subsetneq \mathscr{C}(P)$. By construction it follows that $L_1 \cdot \mathscr{C}(P) \cdot \mathscr{P}(P) \cdot C \cdot R_1$

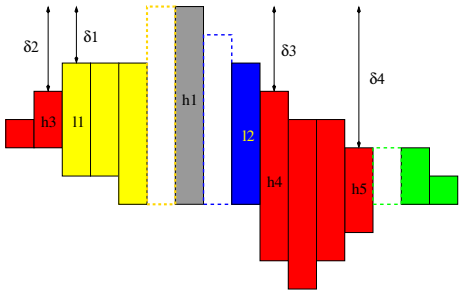


Figure 3: Computing $|\mathsf{DConv}_2(n)|$: dashed columns are (artificial) extra-columns, see formula (1). The grey column and the red columns form the polyomino counted by $Z_2(a_1,h_1,h_3,h_4,h_5,\delta_2,\delta_3,\delta_4)$, blue columns form a polyomino in $\mathsf{F}\cup\mathsf{T}$ (counted by $S(a_3+h_1-1,h_1-1,l_2,0)$), green columns form the polyomino in $\mathsf{S}\cup\mathsf{F}\cup\mathsf{T}$ (counted by $S(n-a_1-a_2-a_3+h_1-\delta_4,h_1-\delta_4)$) and the yellow columns form the polyomino counted by $S(a_2+h_1,h_1,l_1,\delta_1)$.

belongs to Z_2 (see the red columns and the grey column in Fig. 3). This leads to a formula for $|\mathsf{DConv}_2(n)|$,

$$\begin{aligned}|\mathsf{DConv}_2(n)| = \sum_{\substack{h_1,a_1,h_3,\delta_2\\ h_4,\delta_3,h_5,\delta_4}} & Z_2(a_1,h_1,h_3,h_4,h_5,\delta_2,\delta_3,\delta_4)\cdot\\ & \sum_{a_2,l_1,\delta_1} S(a_2+h_1,h_1,l_1,\delta_1)\cdot\sum_{a_3,l_2} S(a_3+h_1-1,h_1-1,l_2,0)\cdot\\ & \cdot S(n-a_1-a_2-a_3+h_1-\delta_4,h_1-\delta_4), \end{aligned} \tag{1}$$

where:

- $Z_2(n,h_1,h_2,h_3,h_4,\delta_1,\delta_2,\delta_3)$ is the number of polyominoes P in $\mathsf{Z}_2(n)$ whose standard decomposition $P=L\cdot C$ or $P=L\cdot C\cdot R$ satisfies the following conditions (see Fig. 4):
  - $L=L_1\cdot\mathscr{C}(P)$, $\mathrm{A}(\mathscr{C}(P))=h_1$, $\mathrm{A}(\textsc{last}(L_1))=h_2$ (remark: $h_2<h_1$), $\mathrm{A}(\mathscr{P}(P))=h_3$ and $\mathrm{A}(\textsc{last}(P))=h_4$;
  - $\mathrm{high}(\mathscr{C}(P))-\mathrm{high}(\textsc{last}(L_1))=\delta_1$, $\mathrm{high}(\mathscr{C}(P))-\mathrm{high}(\mathscr{P}(P))=\delta_2$ and $\mathrm{high}(\mathscr{C}(P))-\mathrm{high}(\textsc{last}(P))=\delta_3$.
- $S(n,p,q,i)$ is the number of $P\in\mathsf{T}\cup\mathsf{S}\cup\mathsf{F}$ of area n, with largest (resp., smallest) column of area p (resp., q) and such that $|\mathrm{low}(\textsc{first}(P))-\mathrm{low}(\textsc{last}(P))|=i$;
- $S(n,p)$ is the number of $P\in\mathsf{T}\cup\mathsf{S}\cup\mathsf{F}$ of area n, with largest column of area p.

Notice the trick of adding (artificial) extra-columns of area h_1, h_1-1 and $h_1-\delta_4$ in formula (1), to deal with the cases $L_2=\varepsilon$ or $F=\varepsilon$ or $R_2=\varepsilon$. In the sequel we indicate by $C(n,h_1,h_2,i)$ the number of polyominoes P of area n in $\mathsf{C}\cup\mathsf{T}\cup\mathsf{F}$ with $\mathrm{A}(\textsc{first}(P))=h_1$, $\mathrm{A}(\textsc{last}(P))=h_2$ and $\mathrm{low}(\textsc{first}(P))-\mathrm{low}(\textsc{last}(P))=i$. In [12] it is shown how to compute $C(n,h_1,h_2,i)$ efficiently.

3.1.1 Computing $Z_2^\bullet(\alpha)$

We introduce a refinement (see Fig. 5) of the standard decomposition $L\cdot C\cdot R$ of a given $P\in\mathsf{Z}_2^\bullet\setminus\text{t-}\mathsf{Z}_2^\bullet$.

Write L as $L=L'\cdot D$, with $D=\mathscr{C}(P)$, and let e be the rightmost column of P such that $\textsc{last}(L')\subsetneq e$ (e exists by Def. 3). Then, let c be the leftmost column in L' that is included in e but not in column $e+1$.

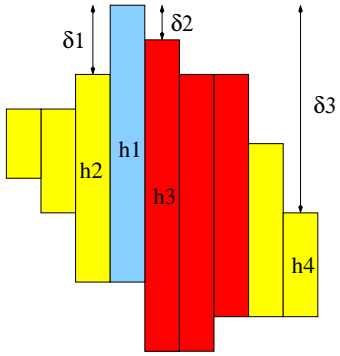


Figure 4: A polyomino counted by $Z_2(n,h_1,h_2,h_3,h_4,\delta_1,\delta_2,\delta_3)$.

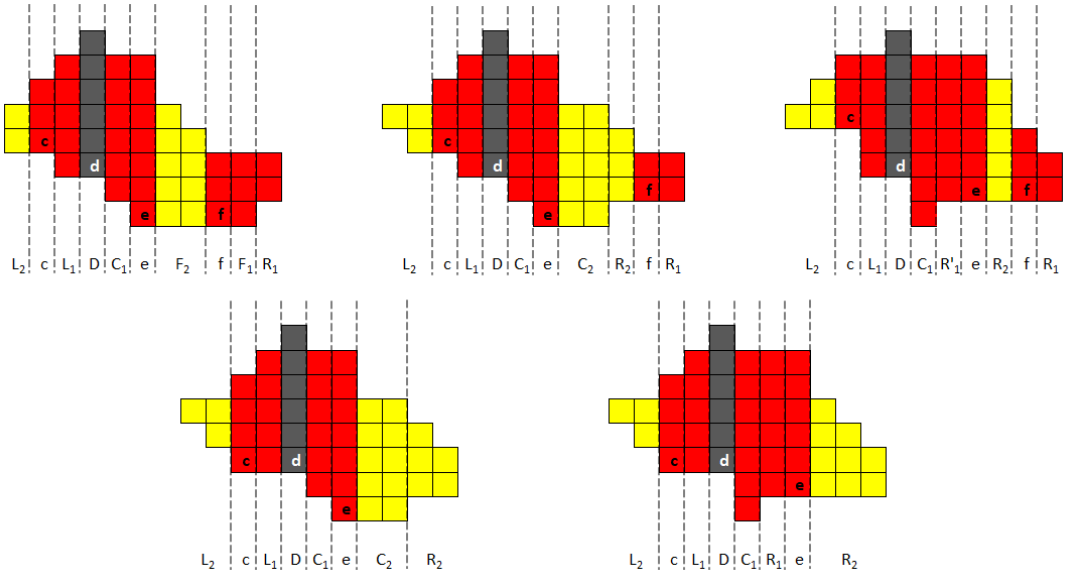


Figure 5: Refining the standard decomposition. Cases 1–5 (from top to bottom and from left to right).

Lastly, consider the leftmost column f in $C\cdot R$ such that $f \asymp c$ (possibly $f=\varepsilon$). We stress that if f belongs to C one has $\mathrm{low}(f')=\mathrm{low}(e)$ for any column f' of C to the right of e. Indeed, if $\mathrm{low}(f)<\mathrm{low}(e)$ no column of P includes both f and c, and so the degree of convexity of P is greater than 2 by [4, Thm. 1]. So, we have five different cases (see Fig. 5), depending on the component to which the columns e and f belong:

1. $P=L_2\cdot c\cdot L_1\cdot D\cdot C_1\cdot e\cdot F_2\cdot f\cdot F_1\cdot R_1$ (if $e,f\in C$),
2. $P=L_2\cdot c\cdot L_1\cdot D\cdot C_1\cdot e\cdot C_2\cdot R_2\cdot f\cdot R_1$ (if $f\in R$ and $e\in C$),
3. $P=L_2\cdot c\cdot L_1\cdot D\cdot C_1\cdot R'_1\cdot e\cdot R_2\cdot f\cdot R_1$ (if $e,f\in R$),
4. $P=L_2\cdot c\cdot L_1\cdot D\cdot C_1\cdot e\cdot C_2\cdot R_2$ (if $f=\varepsilon$ and $e\in C$).
5. $P=L_2\cdot c\cdot L_1\cdot D\cdot C_1\cdot R_1\cdot e\cdot R_2$ (if $f=\varepsilon$ and $e\in R$).

With respect to Fig. 5, the polyominoes obtained by joining the black column (the center) and the red columns belong to t-$\mathsf{Z}_2^{\bullet}$ (cases 1–3) or to u-Z_2° (cases 4–5), whereas the polyominoes obtained by

joining the black column and the yellow columns belong to Z_2 (cases 1–3) or to $\mathsf{Z}_2^\bullet$ (cases 4–5). Let $\alpha = n, h_1, h_2, h_3, h_4, \delta_1, \delta_2, \delta_e$. The idea we exploit is that a polyomino P counted by $Z_2^\bullet(\alpha)$ is uniquely obtained by piecewise combining two polyominoes A and B that are determined by the decomposition of P seen above. In other words, one has

$$Z_2^\bullet(\alpha) = \text{t-}Z_2^\bullet(\alpha) + \mathscr{A}(\alpha) + \mathscr{B}(\alpha) + \mathscr{C}(\alpha) + \mathscr{D}(\alpha),$$

where:

$\mathscr{A}(\alpha)$ counts all polyominoes with decomposition (3), or decomposition (2) and $C_2 = \varepsilon$;

$\mathscr{B}(\alpha)$ counts all polyominoes with decomposition (1), or decomposition (2) and $C_2 \neq \varepsilon$;

$\mathscr{C}(\alpha)$ counts all polyominoes with decomposition (5), or decomposition (4) and $C_2 = \varepsilon$;

$\mathscr{D}(\alpha)$ counts all polyominoes with decomposition (4) and $C_2 \neq \varepsilon$.

By considering Fig. 6 (a) and using (artificial) extra columns of area $x_2 - 1$ (for the magenta component) and $\kappa = i_1 + j_1 + x_1 - e$ (for the yellow component to the left), one has

$$\begin{aligned}\mathscr{A}(\alpha) = &\sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1) \cdot \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2) \cdot \sum_{a_3,x_3,j_3} S(a_3+x_2-1,x_2-1,x_3,j_3) \cdot \\ &\cdot \sum_{a_4,k_1,e} S(a_4,k_1,h_4,i_3-e) \cdot S(n-h_1-a_1-a_2-a_3-a_4+\kappa,\kappa).\end{aligned}$$

Similarly, by considering Fig. 6 (b) one has

$$\mathscr{B}(\alpha) = \sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1) \cdot \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2) \cdot \sum_{a_4,x_4,j_4} S(a_4,x_4,h_4,j_4) \cdot \sum_{\substack{k_3,e_2,k_4 \\ e_3,k_2,e_1}} (Z_2^\bullet(\beta) + Z_2^\circ(\beta)),$$

where $\beta = n - a_1 - a_2 - a_4, h_1, k_2, k_3, k_4, e_1, e_2, e_3$. With respect to Fig. 6 (c), one has

$$\begin{aligned}\mathscr{C}(\alpha) = &\sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1) \cdot \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2) \cdot \sum_{a_3,x_3,j_3} S(a_3+x_2-1,x_2-1,x_3,j_3) \cdot \\ &\cdot \sum_{a_4,k_1,e_1} S(a_4,k_1,h_4,i_3-e_1) \cdot \sum_{e_2,k_2} S(n-h_1-a_1-a_2-a_3-a_4+\kappa,\kappa,k_2,e_2-e_1),\end{aligned} \tag{2}$$

where $\kappa = i_1 + j_1 + x_1 - e_1$. Lastly, from Fig. 6 (d) we obtain

$$\mathscr{D}(\alpha) = \sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1) \cdot \sum_{a_2,j_2,x_2} C(a_2,x_2,h_3,j_2) \cdot \sum_{k_3,e_2,e_1,k_2} Z_2^\bullet(\beta),$$

where $\beta = n - a_1 - a_2, h_1, k_2, k_3, h_4, e_1, e_2, i_3$.

3.1.2 Computing t-$Z_2^\bullet(\alpha)$ and $Z_2^\circ(\alpha)$

Formulas for t-$Z_2^\bullet(\alpha)$ and $Z_2^\circ(\alpha)$ follow directly from their definitions. Indeed, one has (see Fig. 6 (e))

$$\begin{aligned}\text{t-}Z_2^\bullet(\alpha) = &\sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1) \cdot \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2) \cdot \sum_{a_3,x_3,j_3} S(a_3+x_2-1,x_2-1,x_3,j_3) \cdot \\ &\cdot S(n-h_1-a_1-a_2-a_3+\gamma,\gamma,h_4,i_3-i_1-j_1-x_1),\end{aligned}$$

where we use (artificial) extra-columns of area $x_2 - 1$ and γ, with $\gamma = i_2 + j_2 + j_3 + x_3 - i_1 - j_1 - x_1$ (if $a_3 = 0$) or $\gamma = i_2 + j_2 + x_2 - i_1 - j_1 - x_1$ (if $a_3 > 0$). Lastly, consider the polyomino in Z_2° of Fig. 6 (f).

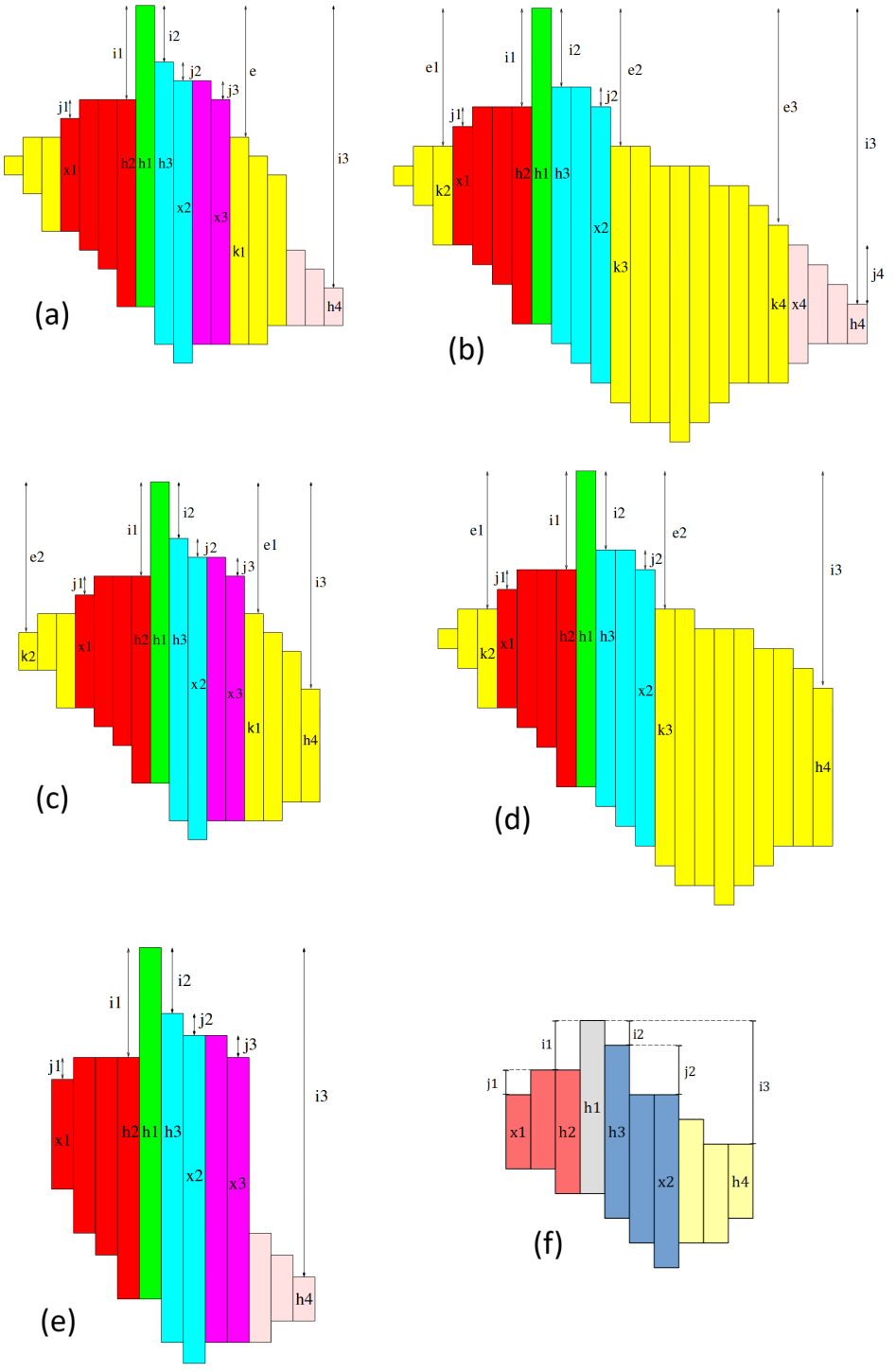


Figure 6: From top to bottom and from left to right: a polyomino with decomposition 3 (counted by $\mathscr{A}(\alpha)$), a polyomino with decomposition 2 and $C_2 \neq \varepsilon$ (counted by $\mathscr{B}(\alpha)$ – the green column and the yellow columns form P counted by $Z_2(\beta)$), a polyomino with decomposition 5 (counted by $\mathscr{C}(\alpha)$), a polyomino with decomposition 4 and $C_2 \neq \varepsilon$ (counted by $\mathscr{D}(\alpha)$), $P \in$ t-$Z_2^{\bullet}$ and $P \in Z_2^{\circ}$ (counted by 3).

The center $\mathscr{C}(P)$ (of area h_1) and the pivot $\mathscr{P}(P)$ (of area h_3) are always present, while in general the red component and the yellow component are optional. Thus, one has

$$\begin{aligned} Z_2^{\circ}(\alpha) = {} & C(n-h_1,h_3,h_4,i_3+h_4-i_2-h_3) + \\ & + \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2)\cdot S(n-h_1-a_2+x_2-1,x_2-1,h_4,i_3-i_2-j_2) + \\ & + \sum_{a_1,x_1,j_1} S(a_1,h_2,x_1,j_1)\cdot \Big(C(n-h_1-a_1,h_4,h_3,i_3-i_2) + \\ & + \sum_{a_2,x_2,j_2} C(a_2,x_2,h_3,j_2)\cdot S(n-h_1-a_1-a_2+x_2-1,x_2-1,h_4,i_3-i_2-j_2)\Big). \end{aligned} \tag{3}$$

4 Implementation

We have developed a C++ program that exploits the formulas and the equations presented in the previous section to compute tables for $Z_2^{\bullet}(\alpha)$, t-$Z_2^{\bullet}(\alpha)$ and $Z_2^{\circ}(\alpha)$. More precisely, we used a dynamic hash table for $Z_2^{\bullet}(\alpha)$ (resp., t-$Z_2^{\bullet}(\alpha)$, $Z_2^{\circ}(\alpha)$), with 64 bits keys encoding the eight parameters $n,h_1,h_2,h_3,h_4,\delta_1,\delta_2$ and δ_3. Each parameter takes up 8 bits, hence the program only works for $n < 256$. Furthermore, each table contains entries strictly greater than zero. The construction of the table exploits dynamic programming. This means that when you search for a key α in the table and you do not found it then $Z_2^{\bullet}(\alpha) = 0$. The size of the hash table for $n = 65$ is approximately 2^{32}, compared to the size 2^{48} of a table implemented via an 8-dimensional array (with $O(n^8)$ space complexity). Each entry in a table is computed in polynomial time using one of the formulas in the previous section. The worst case is represented by formula (2), which leads to a time complexity of $O(n^{14})$ (due to 14 nested summations). Actually, special care has been taken to identify necessary conditions on α for which $Z_2^{\bullet}(\alpha) \neq 0$ (so avoiding unnecessary computations), as well as finding tight limits for summation indices. This allowed to obtain the counting sequence of ZConv up to $n = 75$ in a couple of days of computation. We point out that the sequence indicated in [12] turned out to be wrong due to a bug in the prototype program used to generate it. By using the new sequence, the conjecture for $|\mathsf{ZConv}(n)|$ [12] is corrected in $C\cdot\exp(\pi\sqrt{17n/6})/n^{3/2})$, for a suitable constant C. Lastly, the sequence in Table 1 does not currently appear in OEIS [15].

Table 1: $|\mathsf{ZConv}(n)|$ for $1 \leq n \leq 75$.

1, 2, 6, 19, 55, 148, 370, 874, 1966, 4242, 8838, 17851, 35098, 67356, 126518, 233033, 421696, 750780, 1316916, 2278259, 3891347, 6567788, 10962524, 18108061, 29619788, 48004616, 77126190, 122896541, 94304762, 304931206, 475173306, 735490162, 1131122763, 1728912988, 2627129510, 3969544022, 5965539010, 8918685922, 13267244448, 19641297340, 28943118312, 42459768630, 62020388694, 90215127592, 130699008236, 188612271962, 271160267498, 388410979355, 554388995796, 788571802426, 1117931208744, 1579711035909, 2225197472424, 3124823947072, 4375066863050, 6107738271355, 8502480896054, 11803531168232, 16342160359758, 22566652780440, 31082270916640, 42704425846610, 58529278188804, 80027301158729, 109167041705331, 148578521572114, 201768586748134, 273404214622133, 369684580110248, 498828852782858, 671714659904174, 902712378703847, 1210773577713870, 1620848784573584, 2165731375716902

Acknowledgements

We would like to thank Andrew R. Conway who provided us with the drawings of the Z-convex polyominoes (generated by a smart program and grouped by area). They were extremely useful for debugging our program. We also thank Anthony J. Guttmann for updating the conjecture on $|\mathsf{ZConv}(n)|$ based on the new sequence.

References

[1] G. Barequet & G. Ben-Shachar (2024): *Counting Polyominoes, Revisited*. *2024 Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, pp. 133–143, doi:10.1137/1.9781611977929.10.

[2] M. Bousquet-Mélou (1992): *Convex polyominoes and heaps of segments*. *Journal of Physics A: Mathematical and General* 25(7), pp. 1925–1934, doi:10.1088/0305-4470/25/7/031.

[3] M. Bousquet-Mélou (1996): *A method for the enumeration of various classes of column-convex polygons*. *Discrete Math.* 154(1-3), pp. 1–25, doi:10.1016/0012-365X(95)00003-F.

[4] G. Castiglione & P. Massazza (2014): *An efficient algorithm for the generation of Z-convex polyominoes*. In: *IWCIA 2014*, *Lecture Notes in Comput. Sci.* 8466, Springer, pp. 51–61, doi:10.1007/978-3-319-07148-0_6.

[5] G. Castiglione & A. Restivo (2005): *Ordering and Convex Polyominoes*. In: *MCU 2004*, *Lecture Notes in Comput. Sci.* 3354, Springer, pp. 128–139, doi:10.1007/978-3-540-31834-7_10.

[6] A. Del Lungo, E. Duchi, A. Frosini & S. Rinaldi (2004): *On the Generation and Enumeration of some Classes of Convex Polyominoes*. *Electron. J. Comb.* 11(1), doi:10.37236/1813.

[7] A. Del Lungo, A. Frosini & S. Rinaldi (2003): *ECO Method and the Exhaustive Generation of Convex Polyominoes*. In: *DMTC 2003*, *Lecture Notes in Comput. Sci.* 6795, Springer, pp. 129–140, doi:10.1007/3-540-45066-1_10.

[8] E. Duchi, S. Rinaldi & G. Schaeffer (2008): *The number of Z-convex polyominoes*. *Adv. Appl. Math.* 40(1), pp. 54 – 72, doi:10.1016/j.aam.2006.07.004.

[9] J. O'Rourke E. D. Demaine, J. S. B. Mitchell: *The Open Problems Project*. Available at `http://cs.smith.edu/~jorourke/TOPP`.

[10] E. Formenti & P. Massazza (2017): *From Tetris to polyominoes generation*. *Electronic Notes in Discrete Mathematics* 59, pp. 79 – 98, doi:10.1016/j.endm.2017.05.007.

[11] E. Formenti & P. Massazza (2018): *On the Generation of 2-Polyominoes*. In: *DCFS 2018*, *Lecture Notes in Comput. Sci.* 10952, Springer, pp. 101–113, doi:10.1007/978-3-319-94631-3_9.

[12] Guttmann, Anthony J. & Massazza, Paolo (2024): *Asymptotics of Z-convex polyominoes*. *RAIRO-Theor. Inf. Appl.* 58, p. 26, doi:10.1051/ita/2024009.

[13] D. A. Klarner, S.W. Golomb & G. Barequet (2017): *Polyominoes*. In J.E. Goodman, J. O'Rourke & C. D. Tóth, editors: *Handbook of Discrete and Computational Geometry*, CRC Press, Boca Raton, FL, pp. 359–380. Available at `https://www.csun.edu/~ctoth/Handbook/chap14.pdf`.

[14] A. Del Lungo, M. Nivat, R. Pinzani & S. Rinaldi (2004): *A bijection for the total area of parallelogram polyominoes*. *Discret. Appl. Math.* 144(3), pp. 291–302, doi:10.1016/j.dam.2003.11.007.

[15] OEIS Foundation Inc. (2026): *The On-Line Encyclopedia of Integer Sequences*. Published electronically at `http://oeis.org`.

[16] K. Tawbe & L. Vuillon (2011): *2L-convex polyominoes: Geometrical aspects*. *Contributions Discret. Math.* 6(1), doi:10.55016/ojs/cdm.v6i1.62040.

Penrose P2 Tilings: A Study of Fully Leafed Induced Subtrees

Mathieu Cloutier*
Université du Québec à Trois-Rivières
Trois-Rivières, Québec
mathieu.cloutier4@uqtr.ca

Alain Goupil†
LACIM, Montréal, Québec
Université du Québec à Trois-Rivières
Trois-Rivières, Québec
alain.goupil@uqtr.ca

Alexandre Blondin Massé‡
LACIM, Montréal, Québec
Université du Québec à Montréal
Montréal, Québec
blondin_masse.alexandre@uqam.ca

We present new results about fully leafed induced subtrees in Penrose P2 tilings, also known as kites and darts tilings. We first determine the graph structure of these subtrees and show that they are caterpillars up to an appendix of at most six tiles. In other words, if we remove their degree one vertices, then they are path graphs up to an additional connected path of at most two tiles. We then study bi-infinite fully leafed induced caterpillars in P2 tilings and their geometric properties. In particular, we refute the conjecture proposed by C. Porrier, A. Goupil and A. Blondin Massé that there is a unique bi-infinite fully leafed caterpillar in Penrose P2 tilings.

1 Introduction

Let us play a mathematical game. Consider a tiling $\mathscr{T}$ and a positive integer n. For a patch of tiles P in $\mathscr{T}$, a tile of P is said to be a degree one tile for P if it is connected to only one other tile of P. These degree one tiles can be viewed as extremities of P. The goal of the game is to find a connected and acyclic patch of n tiles in $\mathscr{T}$ that maximizes the number of degree one tiles. These winning patches correspond to what we call fully leafed induced subtrees. These structures have been studied in periodic tilings [1, 2]. In this paper, we are interested in describing these structures in a family of aperiodic tilings. More precisely, the tilings considered in this paper are the Penrose kites and darts tilings, which we denote by the P2 tilings. Interestingly, fully leafed induced subtrees can be used in chemistry to study adsorption on surfaces with a maximal number of boundary sites [7]. Adsorption is the process by which particles adhere to a surface. Since Penrose tilings model quasicrystals, the results presented here may be of interest in the study of adsorption on quasicrystal surfaces [6, 8, 12].

Section 2 presents the definitions needed for this work. In Section 3, we establish the graph structure of fully leafed induced subtrees in Penrose P2 graphs. Section 4 describes the construction of fully leafed induced subtrees, taking into account the geometry of the tilings in which they are embedded. Section 5 studies the construction of bi-infinite fully leafed induced caterpillars.

Complete proofs of the new results presented here, as well as additional results on the same topic, can be found in the extended version of this paper [3].

*Supported by NSERC and FRQNT.
†Supported by NSERC.
‡Supported by NSERC.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 66–73, doi:10.4204/EPTCS.445.9

© M. Cloutier, A. Goupil & A. Blondin Massé
This work is licensed under the
Creative Commons Attribution License.

2 Definitions

Basic definitions from graph theory and tilings can be found in [4] and [5]. The vertices of degree one in a tree T are called the **leaves** of T. For a graph G, the graph G' obtained by removing the leaves from G is called the **derived** graph of G. A **caterpillar** in G is a subgraph $\mathscr{C}$ of G such that $\mathscr{C}'$ is a chain.

Definition 1. *([2]) A **fully leafed induced subtree** of order n of a graph G is an induced subtree of G that maximizes the number of leaves among all induced subtrees of order n in G.*

A Penrose P2 tiling, shortly a P2 tiling, is a tiling of the plane constructed by two sorts of tiles, called **prototiles**, described in Figure 1. We know that P2 tilings are aperiodic [5]. A **patch** is a finite connected set of tiles that respects the matching rules. We say a patch P is **valid** if, when P is extended by the tiles that are forced to be placed according to the matching rules, the resulting set of tiles is still a patch.

We call **P2-graph** the dual graph of a P2 tiling. So we often use the term *tile* to denote a vertex of a P2-graph. All graphs considered in this paper are induced subgraphs of P2-graphs, so we may not mention it every time. In this paper, we want to describe the graph structure of the fully leafed induced subtrees in P2-graphs. We know that P2 tilings have the local isomorphism property, which implies that every valid P2 patch appears in any P2 tiling [5]. Then, by the local isomorphism property, the fully leafed induced subtrees do not depend on a specific P2 tiling in which they would be embedded.



(a) Kite (b) Dart

Figure 1: Prototiles of P2 tilings. Tiles are edge-adjacent only when black (resp. white) vertices coincide. The golden ratio $\varphi = \frac{1+\sqrt{5}}{2}$ is the ratio of long to short edges in both types of tile.

Two important patches of P2 tilings will be used throughout this paper: the star and the sun. They are presented in Figure 2.



(a) Star (b) Sun

Figure 2: Two P2 patches

An important concept of Penrose tilings is inflation [5]. Consider a kite (resp. a dart), that we have outlined by a blue dashed contour in Figure 3a (resp. 3b). The **inflation** of this kite (resp. this dart) is the patch of 4 tiles (resp. 3 tiles) that are φ times smaller than the original tiles, represented in brown and orange in Figure 3a (resp. 3b). The inflation of a P2 patch P is defined as the patch obtained by inflating all the P2 tiles of P. By doing an infinite number of inflations from a given patch and by rescaling the dimension at each inflation, we can construct a P2 tiling.



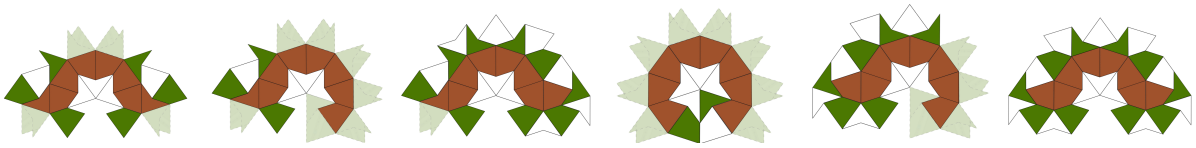
(a) Inflation of a kite (b) Inflation of a dart

Figure 3: Inflation of P2 tiles

3 Graph structure

It has been shown that every fully leafed induced subtree of a P2-graph that has 8 internal tiles (i.e., tiles of degree 2 or more) or fewer is a subcaterpillar of a caterpillar shown in Figure 4 [10]. In this paper, we call a **prime caterpillar** such a fully leafed induced subtree of 8 internal tiles. We notice that internal tiles of prime caterpillars are all of degree 3. It has been verified that a fully leafed induced subtree of 9 internal tiles or more must have at least one tile of degree 2 [10]. We know that an induced subtree of a P2-graph is fully leafed when its number of degree three tiles is maximal, or equivalently, when its number of degree two tiles is minimal, because a P2 tile of a tree has degree 3 or less [3, 9, 10]. Since prime caterpillars are the largest fully leafed induced subtrees with all internal tiles of degree 3, in order to construct a fully leafed induced subtree, we want to maximize the number of prime caterpillars in it. Thus, prime caterpillars are primitive structures that we want to join together to form fully leafed induced subtrees.

Definition 2 ([2, 10]). *Let I_1, I_2 be two fully leafed induced subtrees and t a tile such that t is a leaf of I_1 and I_2, $I_1 \cup I_2$ is a fully leafed induced subtree and $I_1 \cap I_2 = \{t\}$. Then we say that $I = I_1 \cup I_2$ is a **grafting** of I_1 and I_2 at t and we write $I = I_1 \diamond_t I_2$, or just $I = I_1 \diamond I_2$.*



(a) PC_1 (b) PC_2 (c) PC_3 (d) PC_4 (e) PC_5 (f) PC_6

Figure 4: The six possible prime caterpillars up to isometry and choice of leaves. Light green tiles are choices for leaves.

There are two possibilities for extending a prime caterpillar PC_k into a bigger fully leafed induced subtree. We can extend it into a bigger caterpillar or into a tree that is not a caterpillar. In the second case, the fully leafed induced subtree grafted to PC_k is called an **appendix** of PC_k. We have proved the following results.

Lemma 1 ([3]). *Let $\mathscr{C} = A \diamond PC_k$ be a fully leafed induced subtree obtained by grafting the appendix A to the prime caterpillar PC_k. Then A has at most 2 internal tiles.*

Theorem 1 ([3]). *Every fully leafed induced subtree of a P2-graph belongs to one of the following structures:*

1. *A subcaterpillar of a prime caterpillar;*

2. *A caterpillar obtained by grafting several prime caterpillars with at most one proper subcaterpillar of a prime caterpillar structure;*

3. *A caterpillar obtained by grafting several prime caterpillars, to which an appendix of at most 2 internal tiles and 4 leaves is grafted.*

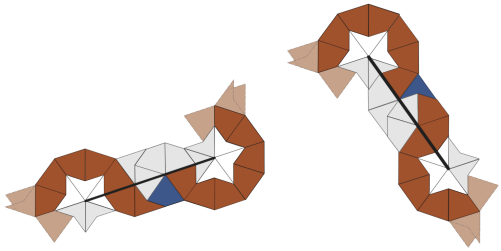
Thus, in order to study fully leafed induced subtrees in P2 tilings, it is sufficient to study caterpillars obtained by grafting prime caterpillars, since any fully leafed induced subtree can be obtained from such a caterpillar by adding a small number of tiles. Caterpillars obtained by grafting prime caterpillars are precisely the fully leafed induced subtrees that have the property of saturation [3]. Let us explain what it means. The leaf function $L_{P2}(n)$ gives the number of leaves of a fully leafed induced subtree of order n. We give here an exact expression for $L_{P2}(n)$ that corrects the one given in [10].

Proposition 1 ([3]). $L_{P2}(n) = \begin{cases} 0 & \text{if } 0 \leq n \leq 1, \\ \left\lfloor \frac{n}{2} \right\rfloor + 1 & \text{if } 2 \leq n \leq 18, \\ 8\left\lfloor \frac{n}{17} \right\rfloor + \left\lfloor \frac{n \bmod 17}{2} \right\rfloor + 1 + \mathbb{1}(n \bmod 17 = 1) & \text{if } n \geq 19. \end{cases}$

Let $\overline{L_{P2}}$ be the lowest linear function that upper bounds L_{P2}. We say that a fully leafed induced subtree of order n is **saturated** if $L_{P2}(n) = \overline{L_{P2}}(n)$ [1].

4 Prime caterpillar graftings

We study more precisely here how prime caterpillars can be grafted together. The only two possibilities for the grafting of two prime caterpillars are illustrated in Figure 5 [3].



(a) (b)

Figure 5: The only two derived paths of graftings of two prime caterpillars. Light brown tiles are possibilities for degree three tiles. Blue tiles have degree 2. We add a segment that joins the centers of the stars adjacent to the prime caterpillars.

For a given P2 tiling T, if we connect the centers of the stars in T with edges as in Figure 5, we obtain a new graph that we call a **star-graph**. The tilings with vertices and edges of the star-graphs are precisely the HBS tilings [11]. We illustrate a region of an HBS tiling in Figure 6. A vertex of a star-graph is colored red, green or blue if it is the vertex in the center of a P2 star adjacent to 0, 1 or 2 suns, respectively. Then, the problem of finding fully leafed caterpillars in P2-graphs becomes the problem of finding paths in star-graphs that determine fully leafed caterpillars.

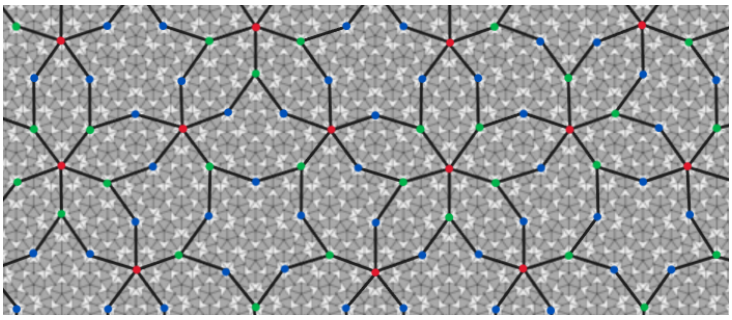


Figure 6: A region of an HBS tiling superimposed on a P2 tiling. Vertices at the center of a P2 star are color-coded red, green and blue depending on whether the star is adjacent to 0, 1, or 2 suns, respectively.

Let $\mathscr{C}$ be a fully leafed caterpillar with star-graph path $p_{\mathscr{C}}$ that contains at least two prime caterpillars C_1 and C_2. Let $p'_{\mathscr{C}}$ be the extension of $p_{\mathscr{C}}$ obtained by extending the two end segments of $p_{\mathscr{C}}$ into two Euclidean half lines, so $p'_{\mathscr{C}}$ splits the Euclidean plane into two disjoint regions. We say that C_1 and C_2 **lie on the same side** (resp. **lie on opposite sides**) of $p_{\mathscr{C}}$ if the two internal tiles of C'''_1 and C'''_2 all lie (resp. do not lie) on a same side of $p'_{\mathscr{C}}$. The two internal tiles of C'''_1 and C'''_2 are the tiles that are at the center of C_1 and C_2 respectively. An example that illustrates this definition can be found in [3].

Proposition 2 ([3])**.** *If $\mathscr{C}$ is a fully leafed caterpillar, for C_j and C_{j+1} two grafted prime caterpillars in $\mathscr{C}$, C_j and C_{j+1} are on opposite sides of the star-graph path of $\mathscr{C}$.*

From Figure 5, we observe that every prime caterpillar C determines a path p_C of two connected edges in a star-graph. We define the **angle** α of a prime caterpillar C as the Euclidean angle determined by p_C and measured on the side of p_C where the two internal tiles of C''' lie. According to this definition, the angle measured in radians of the prime caterpillars PC_1, PC_3 and PC_6 is $\frac{4\pi}{5}$, the angle of PC_2 and PC_5 is $\frac{6\pi}{5}$ and the angle of PC_4 is $\frac{8\pi}{5}$ [3].

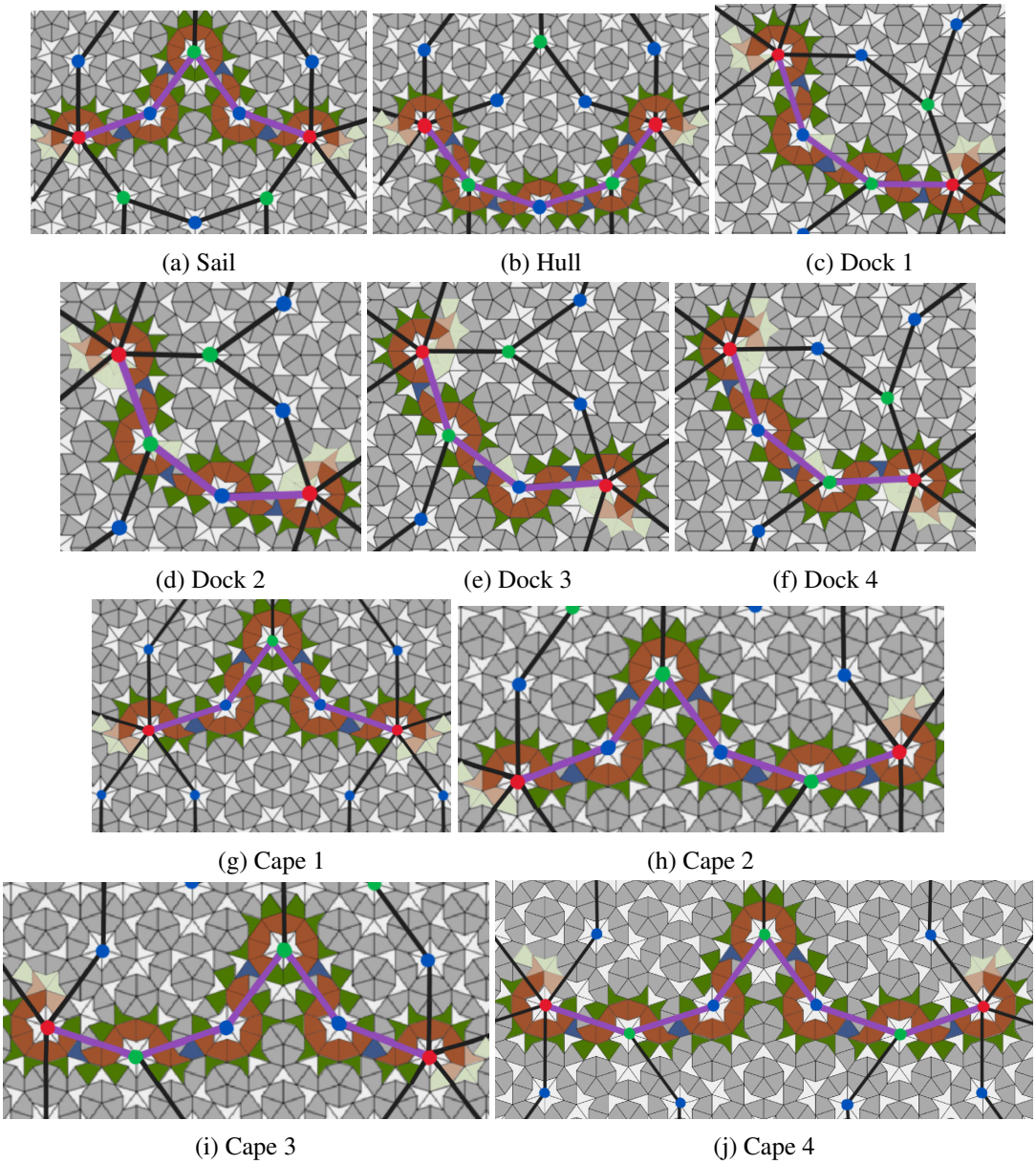
We claim that if the star-graph path of a fully leafed saturated caterpillar $\mathscr{C}$ contains at least one pair of edges forming an angle of $\frac{8\pi}{5}$, then this star-graph path uniquely determines the saturated caterpillar $\mathscr{C}$. Indeed, an angle of measure $\frac{8\pi}{5}$ has the conjugate angle $\frac{2\pi}{5}$. Since there exists no prime caterpillar with an angle of $\frac{2\pi}{5}$, an angle measuring $\frac{8\pi}{5}$ on one side and $\frac{2\pi}{5}$ on the other side can correspond only to a prime caterpillar forming an angle of $\frac{8\pi}{5}$. The only possible prime caterpillar for such an angle is PC_4. By Proposition 2, the rest of the star-graph path of $\mathscr{C}$ completely determines the saturated caterpillar $\mathscr{C}$. Hence, we can use a star-graph path containing at least one angle that measures $\frac{8\pi}{5}$ on one side to represent a fully leafed saturated caterpillar.

Just as we did with prime caterpillars, we find paths in the star-graphs that are primitive blocks to construct bigger fully leafed caterpillars. We call the caterpillars corresponding to these paths **sea caterpillars**, and we present them in Figure 7.

5 Bi-infinite fully leafed caterpillars

Since we now know how to graft caterpillars to construct large fully leafed caterpillars, the natural question that comes to mind is to ask if there are such caterpillars that can be extended infinitely. An infinite induced caterpillar $\mathscr{C}$ is said to be fully leafed if there exists a sequence $(\mathscr{C}_n)_{n\in\mathbb{N}}$ of finite fully leafed induced caterpillars such that for every $n \in \mathbb{N}$, $\mathscr{C}_n \subseteq \mathscr{C}_{n+1}$ and $\mathscr{C} = \lim_{n\to\infty} \mathscr{C}_n$. Since a caterpillar can be extended in two directions, we are interested in finding bi-infinite fully leafed caterpillars. A

first one was found in [10], and we formalize its construction in [3]. This bi-infinite caterpillar does not contain any cape 4. Conjecture 2 of [10] suggests that this bi-infinite caterpillar is the unique bi-infinite fully leafed caterpillar of P2-graphs. It is not a trivial problem, because any finite fully leafed induced caterpillar cannot necessarily be extended into a bi-infinite fully leafed caterpillar. To approach this conjecture, we focus on results that allow us to identify caterpillars that cannot be extended to bi-infinite fully leafed caterpillars.



(a) Sail (b) Hull (c) Dock 1

(d) Dock 2 (e) Dock 3 (f) Dock 4

(g) Cape 1 (h) Cape 2

(i) Cape 3 (j) Cape 4

Figure 7: Sea caterpillars (up to rotation). We keep the color code of the previous figures.

Lemma 2 ([3])**.** *There exists no bi-infinite fully leafed caterpillar that contains two consecutive* $\frac{4\pi}{5}$ *angles.*

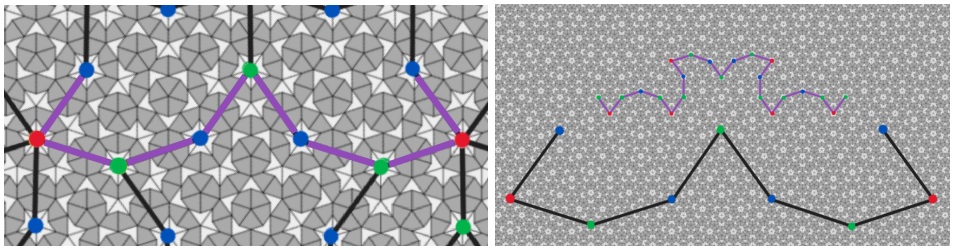
Proposition 3 ([3]). *The prime caterpillar PC_1 cannot belong to any bi-infinite fully leafed caterpillar.*

Proposition 4 ([3]). *Neither a cape 2 nor a cape 3 sea caterpillar can belong to a bi-infinite fully leafed caterpillar.*

Other similar results, in particular about graftings of sea caterpillars, can be found in [3]. These results allow us to construct a new bi-infinite fully leafed caterpillar.

Theorem 2 ([3]). *There exists a bi-infinite fully leafed caterpillar that contains a cape 4.*

Proof. The proof sketch is as follows. We consider an extension of a cape 4, that we call a **Super cape 4**, which is illustrated in Figure 8a. By inflating the region in which it is embedded, we obtain a new region of a P2 tiling where we can construct a larger fully leafed caterpillar that contains a Super cape 4, which we denote by $\psi(\text{Super cape 4})$ and show in Figure 8b. We then proceed to find similar constructions for the new caterpillars in $\psi(\text{Super cape 4})$, and associate them with larger caterpillars in the third inflation of the tiling region where they are embedded. There are 8 such constructions to find, and they must be defined so that for all $n \in \mathbb{N}$, $\psi^n(\text{Super cape 4})$ remains connected and acyclic and is embedded into a valid patch.



(a) Super cape 4 (b) $\psi(\text{Super cape 4})$

Figure 8: Super cape 4 and $\psi(\text{Super cape 4})$. In Figure (b), the black path is the star-graph chain of a Super cape 4 before applying three inflations and is used as a reference in the tiling and the purple path is the star-graph chain of $\psi(\text{Super cape 4})$.

6 Conclusion

We now have a better understanding of the structure of fully leafed induced subtrees in P2 tilings. By Theorem 1, they are caterpillars up to an appendix of at most six tiles, and the saturated ones are caterpillars. We disproved Conjecture 2 from [10] that there is a unique bi-infinite fully leafed caterpillar in P2 tilings.

Since we now know that there exists more than one bi-infinite fully leafed caterpillar in Penrose P2 tilings, we aim to find them all. We have already eliminated several patterns. It is also possible to consider the star-graph chains of these bi-infinite structures as words over a 3-letter alphabet whose letters are either the colors of the vertices in these chains (red, green, and blue) or the angles of the prime caterpillars, and to study the properties of these words. We have begun this work in [3]. Of course, it would also be interesting to describe the fully leafed induced subtrees in other aperiodic tilings.

Acknowledgements: We thank Kevin Bertman for developing the website `https://www.mrbertman.com/penroseTilings.html`, which we used to construct patches of P2 tilings.

References

[1] Alexandre Blondin Massé, Julien de Carufel, Alain Goupil, Mélodie Lapointe, Émile Nadeau & Élise Vandomme (2018): *Fully Leafed Induced Subtrees*. In Costas Iliopoulos, Hon Wai Leong & Wing-Kin Sung, editors: *Combinatorial Algorithms*, Springer International Publishing, Cham, pp. 90–101, doi:10.1007/978-3-319-94667-2_8.

[2] Alexandre Blondin Massé, Julien de Carufel & Alain Goupil (2018): *Saturated fully leafed tree-like polyforms and polycubes*. *Journal of Discrete Algorithms* 52-53, pp. 38–54, doi:10.1016/j.jda.2018.11.004. Combinatorial Algorithms – Special Issue Devoted to Life and Work of Mirka Miller.

[3] Mathieu Cloutier, Alain Goupil & Alexandre Blondin Massé (2026): *Fully Leafed Induced Subtrees in Penrose P2 Tilings*. doi:10.48550/arXiv.2602.13798.

[4] Reinhard Diestel (2025): *Graph theory*. 173, Springer Nature.

[5] Branko Grünbaum & Geoffrey Colin Shephard (1987): *Tilings and patterns*. Courier Dover Publications.

[6] AE Madison (2013): *Symmetry of quasicrystals*. *Physics of the solid state* 55(4), pp. 855–867, doi:10.1134/S1063783413040185.

[7] Neal Madras (2017): *Location of the adsorption transition for lattice polymers*. *Journal of Physics A: Mathematical and Theoretical* 50(6), p. 064003, doi:10.1088/1751-8121/aa53b7.

[8] R McGrath, JA Smerdon, HR Sharma, Wolfgang Theis & J Ledieu (2010): *The surface science of quasicrystals*. *Journal of Physics: Condensed Matter* 22(8), p. 084022, doi:10.1088/0953-8984/22/8/084022.

[9] Carole Porrier & Alexandre Blondin Massé (2019): *The Leaf Function for Graphs Associated with Penrose Tilings*. In: *2019 First International Conference on Graph Computing (GC)*, pp. 37–44, doi:10.1109/GC46384.2019.00014.

[10] Carole Porrier, Alexandre Blondin Massé & Alain Goupil (2025): *The Leaf Function of Penrose P2 Graphs*. *Discrete Mathematics & Theoretical Computer Science* vol. 27:3:9, doi:10.46298/dmtcs.13662.

[11] Porrier, Carole (2025): *HBS and star tilings: State of the art and novel observations*. *RAIRO-Theor. Inf. Appl.* 59, p. 10, doi:10.1051/ita/2025012.

[12] Marjorie Senechal (1996): *Quasicrystals and Geometry*. Cambridge University Press, Cambridge.

k-Convex Polyominoes by Semi-perimeter.

Andrew R. Conway
Fairfield, Vic. 3078, Australia
andrewcombinatorics@greatcactus.org

Anthony J. Guttmann
School of Mathematics and Statistics
The University of Melbourne
Vic. 3010, Australia
tony.guttmann@gmail.com

We give the conjectured solution for the generating function of k-convex polyominoes, enumerated by semi-perimeter. The solution was obtained from the analysis of enumeration data that we generated.

1 Introduction

Convex polyominoes refers to polyominoes on the square lattice with the property that their perimeter is equal in length to the perimeter of their minimal bounding rectangle. Alternatively, any line through the polyomino, either horizontal or vertical, will cut exactly two bonds. Loosely speaking, this means that there are no indentations in the perimeter.

A *k-convex polyomino* [5] is a convex polyomino with the additional constraint that any two cells of the polyomino can be joined by a directed *internal* NE, NW, SE or SW path with at most k right-angle bends. The path must be parallel to the axes of the polyomino.[1] See Figure 1 for an example.

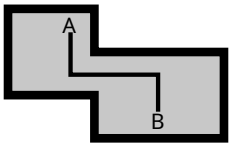


Figure 1: Example 2-convex polyonimo, showing the fewest-corner path between two points A and B.

We have developed an algorithm to enumerate k-convex polyominoes by both semi-perimeter and area. As the perimeter must be even, it makes sense to refer to the semi-perimeter, otherwise every second coefficient, when enumerating by perimeter, would be zero. Using the perimeter data, we are able to conjecture the exact solution for all k. The description of the enumeration algorithm in appendix A is quite general, and includes enumeration by both perimeter and area. However, in this article we are only subsequently discussing the analysis of the semi-perimeter data.

Our analysis is based on the observation that the two known cases, that of 2-convex [3] and fully convex polyominoes [2] have generating functions of identical structure,

$$Q(x) = Q_1(x) - Q_2(x),$$

[1] There is an alternative definition of m-convex polyominoes in the literature [4], which refers to a generalisation of convex polyominoes. In that definition, such a polyomino has perimeter equal to the perimeter of its minimal bounding rectangle $+2m$. We are not discussing this class of polyomino here.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 74–86, doi:10.4204/EPTCS.445.10

where

$$Q_1(x) = \frac{x^2 p(x)}{(1-4x)^2 d(x)}, \qquad Q_2(x) = \frac{x^4 q(x)}{(1-4x)^{3/2} d(x)},$$

where $p(x)$, $q(x)$ and $d(x)$ are polynomials. We conjecture that this is true for all k, and use our enumerations to identify the polynomials. This cannot be done directly, but requires a non-linear transformation to eliminate the implicit $\sqrt{1-4x}$ in $Q_2(x)$, and transform the result into a rational function. Then a simple Padé approximant suffices to identify that rational function, and, transforming back to the original variable, one can confirm the conjectured form from the known series coefficients that are not implicit in the solution. That is to say, if we have polynomials of degrees that require say, 15 terms for their specification, and we have 25 terms, the extra 10 terms provide resounding confirmation of our conjectured results.

In this way we conjectured the results for $3 \le k \le 7$, and from the observed properties of the numerator and denominators, we were able to conjecture simple recurrences, which we were then able to solve.

Our conjectured solution for the semi-perimeter generating function of k-convex polyominoes is:

$$Q_k(x) = Q_{1,k}(x) - Q_{2,k}(x),$$

where

$$Q_{1,k} = \frac{-2x^4 + x^2(1-6x+7x^2+4x^3)U_k\left(\frac{1-2x}{2x}\right) + x^3(1-4x+8x^2)U_{k-1}\left(\frac{1-2x}{2x}\right)}{(1-4x)^2 U_k\left(\frac{1-2x}{2x}\right)},$$

and

$$Q_{2,k} = \frac{4x^2 T_k^2\left(\frac{1}{2\sqrt{x}}\right)}{(1-4x)^{3/2} U_k\left(\frac{1-2x}{2x}\right)}.$$

Here $T_k(t)$ and $U_k(t)$ are Chebyshev polynomials of the 1st and 2nd kind respectively.

2 Known results.

For the full class of convex polyominoes, the generating function is [2]

$$C(x) = \frac{x^2\left(1-6x+11x^2-4x^3\right)}{(1-4x)^2} - \frac{4x^4}{(1-4x)^{3/2}}.$$

The number of convex polyominoes of size $n+2$ grows asymptotically as $\frac{n}{8}4^n$. For 1-convex (otherwise known as L-convex) polyominoes, the generating function is [1]

$$L(x) = \sum l_n x^n = \frac{x^2\left(x^2-2x+1\right)}{2x^2-4x+1},$$

and $l_n \sim \left(\frac{1+\sqrt{2}}{4}\right)\left(2+\sqrt{2}\right)^n$.

For 2-convex (otherwise known as Z-convex polyominoes) the generating function is [3]

$$Z(x) = \frac{x^2\left(4x^5-19x^4+48x^3-35x^2+10x-1\right)}{(1-4x)^2(1-3x)(1-x)} - \frac{x^4(1-2x)^2}{(1-4x)^{\frac{3}{2}}(1-3x)(1-x)}.$$

The number of Z-convex polyominoes of size $n+2$ grows asymptotically as $\frac{n}{24}4^n$.

Prior to this work, the results for $k > 2$ were not known. Since the number of k-convex polyominoes with $k > 2$ is bounded above by the number of convex polyominoes, and bounded below by the number of $(k-1)$-convex polyominoes, it follows that the number of k-convex polyominoes ($k > 2$) of semi-perimeter $n+2$ grows as $\lambda(k)n4^n$, where $\lambda(k)$ is a monotone non-decreasing function growing from $1/24$ to $1/8$ as k goes from 2 to ∞.

Note that both for 2-convex and fully convex polyominoes, the structure of the solution is the same:

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2 p(x)}{(1-4x)^2 d(x)}, \qquad Q_2(x) = \frac{x^4 q(x)}{(1-4x)^{3/2} d(x)}$$

and $p(x)$, $q(x)$ and $d(x)$ are polynomials. Accordingly, as a first guess, we suggest that the generating function for k-convex polyominoes, enumerated by semi-perimeter is:

$$Q_k(x) = Q_{1,k}(x) - Q_{2,k}(x),$$

where

$$Q_{1,k}(x) = \frac{x^2 p_k(x)}{(1-4x)^2 d_k(x)}, \qquad Q_2(x) = \frac{x^4 q_k(x)}{(1-4x)^{3/2} d_k(x)}.$$

3 Conjectured results for $k > 2$.

To identify this structure from our data is not straightforward. For example, if one asks *gfun* to identify the algebraic equation satisfied by the series for 2-convex polyominoes, one requires 30 coefficients to find the equation. As we only have 25 coefficients for $k > 2$, such an approach is clearly not going to be successful.

Instead, we make a change of variable, and write $y = 1 - \sqrt{1-4x}$, so that $x = \frac{y}{2} - \frac{y^2}{4}$. Then, for example, the solution for 2-convex polyominoes is

$$\frac{y^{10} - 14y^9 + 95y^8 - 376y^7 + 1064y^6 - 2288y^5 + 3632y^4 - 4032y^3 + 2944y^2 - 1280y + 256)(y-2)^2 y^2}{768(y-1)^4(y^2 - 2y + 4/3)(y^2 - 2y + 4)}$$

Extracting the factors $\frac{y^2}{(y-1)^4}$ which we know must be present if our assumed form is correct, we are left with a rational function with numerator of degree 12 and denominator of degree 4. This requires 18 series coefficients to identify, simply by constructing a Padé approximant, rather than 30 searching for an algebraic equation from the original series.

3.1 3-convex polyominoes

We now apply this approach to our series for 3-convex polyominoes, and readily find the generating function:

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2(8x^6 - 34x^5 + 97x^4 - 110x^3 + 54x^2 - 12x + 1)}{(1-4x)^2(1-2x)(2x^2 - 4x + 1)}, \; Q_2(x) = \frac{x^4(1-3x)^2}{(1-4x)^{3/2}(1-2x)(2x^2 - 4x + 1)}.$$

Note that, in the original variable, the numerator of $Q_1(x)$ is of degree 6, the denominator is of degree 3 (we are not worrying about the known terms $\frac{x^2}{(1-4x)^2}$), and $Q_2(x)$ has denominator of degree 2, (and the same numerator). So 14 terms are needed to verify this equation, and we have 25.

Here are the first few terms of the series:

$$(426783233690940x^{24}+102833217472008x^{23}+24745444446676x^{22}+5946372452072x^{21}+1426784930290x^{20}+341793274268x^{19}+81735301562x^{18}+19508859728x^{17}+4646820820x^{16}+1104324184x^{15}+261791420x^{14}+61889640x^{13}+14586482x^{12}+3426076x^{11}+801634x^{10}+186760x^9+43302x^8+9988x^7+2292x^6+524x^5+120x^4+28x^3+7x^2+2x+1)x^2.$$

The asymptotics are

$$a_{n+2} \sim 4^n \left(\frac{n}{16} - \frac{\sqrt{n}}{8\sqrt{\pi}} + \frac{11}{32} + \frac{9}{64\sqrt{n\pi}} \right).$$

Proceeding in this way, we find the following:

3.2 4-convex by semi-perimeter

For 4-convex, we find

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2(-12x^7+49x^6-178x^5+282x^4-208x^3+77x^2-14x+1)}{(x^2-3x+1)(5x^2-5x+1)(1-4x)^2},$$

and

$$Q_2(x) = \frac{x^4\left(2x^2-4x+1\right)^2}{(x^2-3x+1)(5x^2-5x+1)(1-4x)^{3/2}}.$$

Here are the first few terms of the series:

$$(2095409493265028x^{25}+504284513809564x^{24}+121184348439516x^{23}+29075705815146x^{22}+6964164029736x^{21}+1664934912802x^{20}+397228761916x^{19}+94561501504x^{18}+22455408404x^{17}+5318002714x^{16}+1255648644x^{15}+295481944x^{14}+69273336x^{13}+16172574x^{12}+3757972x^{11}+868678x^{10}+199652x^9+45608x^8+10356x^7+2340x^6+528x^5+120x^4+28x^3+7x^2+2x+1)x^2.$$

The asymptotics are

$$a_{n+2} \sim 4^n \left(\frac{3n}{40} - \frac{\sqrt{n}}{10\sqrt{\pi}} + \frac{21}{80} + \frac{1}{16\sqrt{n\pi}} \right).$$

3.3 5-convex by semi-perimeter

For 5-convex, we find

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2(16x^8 - 84x^7 + 310x^6 - 632x^5 + 644x^4 - 350x^3 + 104x^2 - 16x + 1)}{(1-x)(1-2x)(1-3x)(1-4x+x^2)(1-4x)^2},$$

and

$$Q_2(x) = \frac{x^4\left(1 - 5x + 5x^2\right)^2}{(1-x)(1-2x)(1-3x)(1-4x+x^2)(1-4x)^{3/2}}.$$

Here are the first few terms of the series:

$$9455884509981378x^{26} + 2274678676735172x^{25} + 546358280976036x^{24} + 131014913237644x^{23} + 31360954319330x^{22} + 7492300767612x^{21} + 1786166443572x^{20} + 424837527020x^{19} + 100790612376x^{18} + 23845293592x^{17} + 5623999514x^{16} + 1321924560x^{15} + 309548722x^{14} + 72183568x^{13} + 16755130x^{12} + 3869600x^{11} + 888824x^{10} + 202988x^9 + 46092x^8 + 10412x^7 + 2344x^6 + 528x^5 + 120x^4 + 28x^3 + 7x^2 + 2x + 1)x^2.$$

The asymptotics are

$$a_{n+2} \sim 4^n \left(\frac{n}{12} - \frac{\sqrt{n}}{12\sqrt{\pi}} + \frac{13}{72} + \frac{5}{288\sqrt{n\pi}} \right).$$

3.4 6-convex by semi-perimeter

For 6-convex, we find

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2(-20x^9 + 127x^8 - 524x^7 + 1292x^6 - 1712x^5 + 1267x^4 - 544x^3 + 135x^2 - 18x + 1)}{(1 - 5x + 6x^2 - x^3)(1 - 7x + 14x^2 - 7x^3)(1-4x)^2},$$

and

$$Q_2(x) = \frac{x^4\left(1 - 4x + x^2\right)^2(1-2x)^2}{(1 - 5x + 6x^2 - x^3)(1 - 7x + 14x^2 - 7x^3)(1-4x)^{3/2}}.$$

Here are the first few terms of the series:

$$(9875153194401768x^{26} + 2371897317966784x^{25} + 568772899988186x^{24} + 136149363839848x^{23} + 32528371648986x^{22} + 7755467989812x^{21} + 1844902796266x^{20} + 437794641248x^{19} + 103609705944x^{18} + 24448602992x^{17} + 5750559506x^{16} + 1347831084x^{15} + 314691982x^{14} + 73165644x^{13} + 16933340x^{12} + 3899788x^{11} + 893464x^{10} + 203604x^9 + 46156x^8 + 10416x^7 + 2344x^6 + 528x^5 + 120x^4 + 28x^3 + 7x^2 + 2x + 1)x^2.$$

The asymptotics are

$$a_{n+2} \sim 4^n \left(\frac{5n}{56} - \frac{\sqrt{n}}{14\sqrt{\pi}} + \frac{11}{112} - \frac{11}{112\sqrt{n\pi}} \right).$$

3.5 7-convex by semi-perimeter

For 7-convex, we find

$$Q(x) = Q_1(x) - Q_2(x),$$

where

$$Q_1(x) = \frac{x^2(24x^{10} - 198x^9 + 867x^8 - 2476x^7 + 4072x^6 - 3896x^5 + 2251x^4 - 798x^3 + 170x^2 - 20x + 1)}{(1 - 8x + 20x^2 - 16x^3 + 2x^4)(1 - 4x + 2x^2)(1 - 2x)(1 - 4x)^2},$$

and

$$Q_2(x) = \frac{x^4\left(7x^3 - 14x^2 + 7x - 1\right)^2}{(1 - 8x + 20x^2 - 16x^3 + 2x^4)(1 - 4x + 2x^2)(1 - 2x)(1 - 4x)^{3/2}}.$$

Here are the first few terms of the series:

$$10095807143173876x^{26} + 2421965007900888x^{25} + 580044174368274x^{24} + 138664016164056x^{23} + 33083632385428x^{22} + 7876620943924x^{21} + 1870972607370x^{20} + 443313571324x^{19} + 104755646898x^{18} + 24681075816x^{17} + 5796404302x^{16} + 1356560728x^{15} + 316282360x^{14} + 73439268x^{13} + 16976952x^{12} + 3906036x^{11} + 894228x^{10} + 203676x^9 + 46160x^8 + 10416x^7 + 2344x^6 + 528x^5 + 120x^4 + 28x^3 + 7x^2 + 2x + 1)x^2.$$

The asymptotics are

$$a_{n+2} \sim 4^n \left(\frac{3n}{32} - \frac{\sqrt{n}}{16\sqrt{\pi}} + \frac{1}{64} - \frac{23}{128\sqrt{n\pi}}\right).$$

4 General solution

4.1 Observations from the above results

From the above cases, it appears that the general solution of k-convex polyominoes by semi-perimeter to be of the same form as the case for convex polyominoes and 2-convex polyominoes.

$$Q_k(x) = Q_{1,k}(x) - Q_{2,k}(x),$$

where

$$Q_{1,k}(x) = \frac{x^2 p_k(x)}{(1-4x)^2 d_k(x)}, \qquad Q_2(x) = \frac{x^4 q_k(x)}{(1-4x)^{3/2} d_k(x)}.$$

Here $p_k(x)$ appears to be a polynomial of degree $k+3$, and $q_k(x)$ similarly appears to be a polynomial of degree $2\lfloor \frac{k}{2} \rfloor$. Furthermore, the denominator displays so much regularity that it can be calculated explicitly. It appears to be:

$$d_k = 1 + \sum_{j=1}^{k} (-1)^j \binom{2k-j+1}{j} x^j.$$

The polynomials $d_k(x)$ satisfy the linear recurrence[2]

$$d_k(x) = (1-2x)\, d_{k-1}(x) - x^2\, d_{k-2}(x), \qquad k \geq 2,$$

[2]We thank Simone Rinaldi for this observation

with initial conditions $d_0(x) = 1,\ d_1(x) = 1-2x$. And this can be readily solved to yield

$$d_k(x) = \frac{1}{\sqrt{1-4x}}\left(\frac{1-2x+\sqrt{1-4x}}{2}\right)^{k+1} - \frac{1}{\sqrt{1-4x}}\left(\frac{1-2x-\sqrt{1-4x}}{2}\right)^{k+1}.$$

We can also write this solution more compactly in terms of Chebyshev polynomials of the second kind, $U_k(t)$. That is to say:

$$d_k(x) = x^k U_k\left(\frac{1-2x}{2x}\right).$$

Next, we note that $q_k(x)$ is a perfect square. That is, $q_k(x) = r_k(x)^2$, where $r_k(x)$ is obviously of degree $\lfloor \frac{k}{2} \rfloor$. And we observe from the data that $r_k(x)$ also satisfies a simple recurrence:

$$r_k(x) = r_{k-1}(x) - x \cdot r_{k-2}(x).$$

The initial conditions are $r_0(x) = 2$ and $r_1(x) = 1$. This recurrence can also be solved, and the solution is

$$r_k(x) = \left(\frac{1+\sqrt{1-4x}}{2}\right)^k + \left(\frac{1-\sqrt{1-4x}}{2}\right)^k.$$

This solution can be expressed in terms of Chebyshev polynomials of the first kind, $T_n(t)$. We find that:

$$r_k(x) = 2x^{k/2} T_k\left(\frac{1}{2\sqrt{x}}\right).$$

The numerator of $Q_{1,k}(x)$, that is, $p_k(x)$, caused us some trouble, but we eventually found the recurrence: It is

$$p_k(x) = (1-x)p_{k-1}(x) + (x^2-x)p_{k-2}(x) + x^3 p_{k-3}(x),$$

with initial conditions $p_0(x) = 4x^3+5x^2-6x+1$, $p_1(x) = -16x^3+20x^2-8x+1$, and $p_2(x) = -4x^5 + 19x^4 - 48x^3 + 35x^2 - 10x + 1$.
This too can be solved, with $Q = \sqrt{1-4x}$, to give:

$$\begin{aligned} p_k(x) = -2x^{k+2} &+ \left(\frac{1-2x+Q}{2}\right)^k \left[\frac{1-8x+21x^2-18x^3+8x^4+Q(1-6x+7x^2+4x^3)}{2Q}\right] \\ &+ \left(\frac{1-2x-Q}{2}\right)^k \left[\frac{-(1-8x+21x^2-18x^3+8x^4)+Q(1-6x+7x^2+4x^3)}{2Q}\right]. \end{aligned}$$

This looks a lot like the solution above for $d_k(x)$, and indeed we can write:

$$p_k(x) = -2x^{k+2} + x^k\left[(1-6x+7x^2+4x^3)U_k\left(\frac{1-2x}{2x}\right) + (x-4x^2+8x^3)U_{k-1}\left(\frac{1-2x}{2x}\right)\right]$$

4.2 Solution

Putting all this together, we find the solution for k-convex polyominoes by semi-perimeter is conjectured to be:

$$Q_k(x) = Q_{1,k}(x) - Q_{2,k}(x),$$

where

$$Q_{1,k} = \frac{-2x^4 + x^2(1-6x+7x^2+4x^3)U_k\left(\frac{1-2x}{2x}\right) + x^3(1-4x+8x^2)U_{k-1}\left(\frac{1-2x}{2x}\right)}{(1-4x)^2 U_k\left(\frac{1-2x}{2x}\right)},$$

and

$$Q_{2,k} = \frac{4x^2 T_k^2\left(\frac{1}{2\sqrt{x}}\right)}{(1-4x)^{3/2} U_k\left(\frac{1-2x}{2x}\right)}.$$

Simone Rinaldi [6] has shown that, in the limit $k \to \infty$, the known result for convex polyominoes is recovered. This is further evidence for the correctness of our conjectured result.

4.3 Asymptotics

We have calculated the first four terms in the asymptotic expansion of the coefficients for $k < 8$. The general form at leading order and sub-leading order is

$$a_{n+2,k} \sim 4^n \left(\frac{n(k-1)}{8(k+1)} - \frac{\sqrt{n}}{2\sqrt{\pi}(k+1)} \right).$$

Note that in the limit as $k \to \infty$ this gives the correct result, to leading order, for convex polyominoes, viz. $a_{n+2} \sim 4^n \left(\frac{n}{8}\right)$, though the sub-leading term vanishes in that limit, which it doesn't do in the actual case. That is to say, for convex polyominoes one has $a_{n+2} \sim 4^n \left(\frac{n}{8} - \frac{\sqrt{n}}{2\sqrt{\pi}}\right)$.

Acknowledgements: We would like to thank Simone Rinaldi and Paolo Massazza for suggesting this problem and for several illuminating discussions. We would also like to thank the University of Melbourne for making available the computing resources enabling these computations to be made.

References

[1] G. Castiglione, A. Frosini, A. Restivo & S. Rinaldi (2005): *Enumeration of L-convex polyominoes by rows and columns*. Theoretical Computer Science 347(1), pp. 336–352, doi:10.1016/j.tcs.2005.06.031.

[2] Marie-Pierre Delest & Gérard Viennot (1984): *Algebraic languages and polyominoes enumeration*. Theoretical Computer Science 34(1), pp. 169–206, doi:10.1016/0304-3975(84)90116-6.

[3] Enrica Duchi, Simone Rinaldi & Gilles Schaeffer (2008): *The number of Z-convex polyominoes*. Advances in Applied Mathematics 40(1), pp. 54–72, doi:10.1016/j.aam.2006.07.004.

[4] I. G. Enting, A. J. Guttmann, L. B. Richmond & N. C. Wormald (1992): *Enumeration of Almost-Convex Polygons on the Square Lattice*. Random Structures & Algorithms 3(4), pp. 445–461, doi:10.1002/rsa.3240030407.

[5] P. Massazza (2022): *On counting k-convex polyominoes*. In Ugo Dal Lago & Daniele Gorla, editors: Proc. of the 23rd Italian Conference on Theoretical Computer Science, CEUR Workshop Proceedings 3284, CEUR-WS, Aachen, pp. 116–121.

[6] Rinaldi S.: *Private communication*.

A Description of the algorithm for counting k-convex polyominoes.

A fairly standard dynamic programming/transfer matrix algorithm is used. For explanatory purposes we will start with a description of a well known algorithm to enumerate column-convex polyominoes. This

will then be modified to a slightly more complex algorithm to enumerate convex polyominoes, and then modified again to keep track of k for k-convexity.

The first two simple algorithms are not necessarily the most efficient known methods of solving said problems, but are ways that can be extended to the k-convex problem in a relatively straightforward manner.

A.1 Simple algorithm 1: Counting column-convex polyominoes.

A column-convex polyomino on the square lattice has a convexity constraint that only applies to columns. That is, each column of the polyomino contains a consecutive range of cells with no holes. The polyomino can be constructed as a sequence of columns. To ensure connectivity, each column must have non-zero overlap with the next column. Suppose one is enumerating by area up to a maximum area A. We will generate a results vector R indexed by the area used, by the following steps

1. **Initialization.** Start with the leftmost column and consider all columns of length 1 to A. Make a list L of each of these, containing three values:
   - *Signature*: The height of the column (1 to A).
   - *Variable Use* : The total area used so far (which will be the same as the height at this point).
   - *Multiplicity* : The number of ways of getting to this point (which will be 1 at this point).
2. **Next column.** Next take each element $e \in L$ of the list produced by the previous step and consider what could go in the next column.

   The next column may be empty - the polyomino is finished. In this case add the multiplicity of e to the results vector in the position indicated by the variable use of e.

   The next column may be non-empty. Consider all possible column lengths and relative positions of the column relative to the prior column. This will be constrained by the maximum area desired and the connectivity constraint. For each of these possibilities, add a new element to a list LL containing
   - *Signature*: The height of the new column being added (1 to A less prior area use of e).
   - *Variable Use*: The total area used so far (which will be the height of this column plus the prior area use of e)
   - *Multiplicity*: The number of ways to get to this point (which will be the multiplicity of e).

   If multiple elements end up in LL with the same signature and variable use, then merge them into one element of LL with multiplicity equal to the sum of the multiplicities of the merged elements. If the variable use gets above the maximum area A, discard the element.

   A picture of an example of one of the possibilities in this step is shown in Figure 2. The past is not relevant to the way columns can connect, other than in the number of different pasts. The future is not determined. The 5 high new column could be placed with the top starting anywhere from four units above the top of the prior column to three units below; in this example it is two above. The area variable usage in L is 4 plus the area of the past. The area variable usage in LL is 5 plus 4 plus the area of the (now more distant) past. If counting by perimeter, the perimeter usage in L would be 2 plus the perimeter of the past. The perimeter usage in LL would be 5 plus 2 plus the perimeter of the (now more distant) past [3].

[3]Note: there is a very simple optimization of this algorithm when enumerating just by area — instead of treating the different relative positions of the current and prior column separately, just multiply the multiplicity by this number — that makes the algorithm $O(A)$ faster. However it will not be usable for the more complex scenarios described later so is not relevant for this paper.

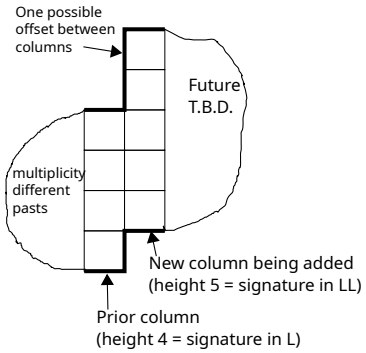


Figure 2: Example addition of a new column.

3. **Completing the algorithm.** After executing step 2, replace L by LL and repeat step 2. Iterate until the list of elements is empty. As the variable use will be increased by at least one for each application of step 2, eventually all elements will exceed the maximum area constraint and will be discarded resulting in an empty list.

   After one iteration of step 2, R will contain all the polyominoes of width 1 (up to the required area), and LL will contain all polyominoes of width exactly 2 (up to the required area), separated by the second column's height. After two iterations of step 2, R will contain all the polyominoes of width 1 or 2 (up to the required area), and LL will contain all polyominoes of width exactly 3 (up to the required area), separated by the third column's height. After a sufficient number of iterations, the area (or other) constraint will force LL to be empty. At this point the results vector contains the number of column-convex polyominoes for each area up to A.

4. **Enumerating by other variables.** The same algorithm can be used to enumerate by other variables - just replace the variable use computation appropriately. For instance, if counting by perimeter, the step 1 variable used for a column of height h would be $2h+2$ (including the minimum closing path), and the addition at step 2 would be 2 horizontal steps and a number of vertical steps depending upon how the columns overlap. Similar modifications allow for counting by many other variables, such as the number of columns, separate vertical and horizontal perimeter, or multiple simultaneous variables.

5. **Complexity of the algorithm.** Step 2 is the main complexity. Suppose there are S possible signatures and V possible variable uses. For enumerating convex poly- ominoes by area both V and S are O(A). For step 2, there are O(A) possible next columns and O(A) possible relative positions for each, leading to O(A2) work for each element of the list There are O(A2) elements of L for a total of O(A4) work for each iteration of step 2. There are O(A) possible iterations of step 2, for a total of O(A5) time complexity of the algorithm. Note that if enumerating by area, as mentioned before the O(A2) work for each element of L can be optimized to O(A) work by bunching together all the different offsets as they don?t affect anything needing recording. This reduces the total work from O(A5) to O(A4) in this special case. This does not work in the more complex scenarios described later, nor when enumerating by perimeter. Memory use is primarily due to the lists L and LL which are O(A2).

A.2 Simple problem 2: Counting convex polyominoes.

Convex polyominoes are a restriction of column-convex polyominoes with the additional constraint that each row of the polyomino is a single contiguous range. This means that adding a new column is more

complex than in the previous algorithm as now columns before the last column matter. However this can be constrained by adding an extra two Boolean flags to the signature. One represents whether a column has ever had a lower top than the prior column, the other indicates whether a column has ever had a higher bottom than the prior column.

In step two, if dealing with an element that has ever had a lower top than the prior column, then the subsequent column may not have a higher top than the current column. Similarly, if the element has ever had a higher bottom than the prior column, then the subsequent column may not have a lower bottom than the current column. This increases the number of signatures by a maximum of 4, increasing the time and memory use by a factor of 4 relative to column-convex polyominoes.

A.3 Full problem: k-convex polyominoes.

The same algorithm as for counting convex polyominoes will be used, except the signature will be extended to include some additional information that can be used to determine the value of k for each polyomino. When a polyomino is added to the results vector it will also be indexed by this k.

Consider a path P between two points in the polyomino such that P is within the polyomino, and P has the minimum number of bends to go between the two points. Due to the convexity constraint of the polyomino, there will never be any reason for P to double back on itself - that is, go left after going right, or go up after going down, etc. Assume without loss of generality that P starts from the leftmost of the two points (if they are in the same column then P is a simple line). This means we only need to consider paths alternating right and up, and paths alternating right and down.

The following discussion will focus on tracking the paths going right and up; the paths going right and down should be treated in an almost identical manner. The maximum path length (number of corners) will then be the maximum of the two sub-problems. Each time we add a column, we will keep track of

- The maximum number of lines MN in the minimum-corner route needed to get between any two points so far in the polyomino going up and right.
- For each point in the polyomino, the best route to other sites in the polyomino ending in a horizontal line going through the vertical border line to the right of the current column. These will be used to update MN when the next column is added.

In the following discussion, B is the vertical border line to the right of the current column.

1. **What the best route means.** What information do we need for each point in the polyomino? The best route to a point could potentially start with either a vertical bond or a horizontal bond. Each vertical bond should then go as far as possible up, and each horizontal bond should go as far as possible to the right. If you have gone beyond the destination point, that's fine - going beyond counts as reaching it. So for each point we need
   - If we are starting with a vertical bond, then:
     VN: Number of lines in the path to go through B. (This is even)
     VH: Height at which it goes through B.
   - If we are starting with a horizontal bond, then:
     HN: Number of lines in the path to go through B. (This is odd)
     HH: Height at which it goes through B.

   Note that sometimes it will be impossible to start with a vertical or horizontal bond. In this case, assume such a bond is of zero length, and the other option will always be used in practice.

   This means that each site will be associated with a tuple of 4 numbers (VN, VH, HN, HH).

2. **Eliminating dominated information.** There are some sites whose best route information can be discarded as some other site S dominates them - that is, it is always harder to get from S to somewhere than from the discardable site. In this case there is no point keeping track of the discardable site. S is said to dominate the other sites. In particular, when adding a column, the bottom cell will dominate all higher cells. That is we can ignore all higher cells. Similarly, if there is a cell to the left of said cell, that will dominate it. The only time adding a column will necessitate adding a cell to keep track of is when the column starts lower than the prior column, and even then only the lowest cell needs tracking. More formally, suppose we have two sites, *S1* with tuple (*VN1*, *VH1*, *HN1*, *HH1*) and *S2* with tuple (*VN2*, *VH2*, *HN2*, *HH2*). Then *S1* dominates *S2* if $VN1 \geq VN2$ and $VH1 \leq VH2$ and $HN1 \geq HN2$ and $HH1 \leq HH2$. This potentially allows removal of other sites, particularly if the polyomino goes through a pinch point. Note that removing this redundant information does not change the answer at all, but it makes the algorithm much faster as it vastly reduces the number of different signatures.

3. **Start state.** For a start column of height H, the initial state will be a single cell (the bottom one) with $(VN = 2,\ VH = H-1\ , HN = 1,\ HH = 0)$.

   Note that the $VN = 2$ limit start means the algorithm cannot distinguish between L convex polyominoes (1 bend) and the trivial case of polyominoes with zero bends (polyominoes that are a horizontal or vertical line). If these are wanted separately they are trivial to enumerate analytically.

4. **Update.** Consider a new column of height H' starting at a height of δ above the prior column of height H. If the new column starts below the old column ($\delta < 0$), a new cell must be added with the same values as the start state $(VN = 2,\ VH = H-1\ , HN = 1,\ HH = 0)$.

   All prior cells' information must be updated. Both the vertical and horizontal pairs are updated in the same way. Let (xN, xH) be either (VN, VH) or (HN, HH), then the following process makes the new state (xN', xH') :

   - If one can continue horizontally, that is $xH \geq \delta$, then $xN' = xN$ and $xH' = \min(xH - \delta, H' - 1)$ as the base has changed, and one can't go above the max height.
   - Otherwise one will need to add a vertical step in the last column (there can't have already been one, as otherwise xH would be $H-1$ and if $H - 1 < \delta$ then the new column is disconnected from the current column). Change xN' to $xN + 2$ and change xH' to $\min(H - 1 - \delta, H' - 1)$.

   One should check that none of the new states are dominated. Doing an imperfect job of this will increase time and memory usage but will not produce incorrect results. The program used does not do a perfect job of this as the check is fiddly and error-prone. Now we update *NL*, the number of lines needed to get anywhere in the polyomino so far. This will be the maximum of the existing *NL* value and the best route to everywhere in the new column.

   The maximum number of lines needed to get to anywhere in the new column is the maximum of the current value, and, for each of the prior cells C, the minimum number of lines needed to get to the top cell of the new line (everywhere else is easy). So for each of the cells being tracked, we compute the best route to get there being the minimum of the best route starting vertically or the best route starting horizontally. The best route for a pair (xN, xH) that has already been updated as above will be:

   - If $xH' = H' - 1$, then one can just get there with xN'.
   - Otherwise one will need to add a vertical step $xN' + 1$.

5. **Complexity.** The number of possible cells that are not dominated is hard to characterise. In practice the number of states (and thus the time and memory use) seem to grow exponentially with the size of the enumeration.

In practice, to count all k-convex polyominoes of semi-perimeter up to 25 took about 60 hours of CPU time on a single processor, and some 800GB of physical memory. To count all k-convex polyominoes of area up to 65 cells took about 80 hours on a single processor, and used 2TB of physical memory.

The program can be found at `https://gitlab.com/andrewconwaymaths/k_convex_polyominoes`.

The Genesis Sequence, Tree Records and Endofunctions.

Enrica Duchi
Université de Paris Cité, IRIF, Paris, France

Adrián Lillo
Universidad de Sevila, Sevilla, España

Pablo Puerto
Universidad de Sevila, Sevilla, España

Mercedes Rosas
Universidad de Sevila, Sevilla, España

Stefan Trandafir
Simon Fraser University, Vancouver, Canada

We present bijections connecting tree records, the girth of a connected endofunction, and the genesis sequence (the first sequence in OEIS). Using these, we derive generating functions for tree and forest record numbers in terms of Cayley's tree function and give a new proof of Cayley's forest formula.

This work is dedicated to Neil J. A. Sloane, renowned number-crusher, for his creation and tireless curation of the On-Line Encyclopedia of Integer Sequences.

The present work is an extended abstract of [DLP+26], building on earlier studies of records in rooted trees and forests [LRT25a, LRT25b]. Trees and forests are fundamental structures in discrete mathematics, and our work highlights an intriguing link between tree records, node heights, and endofunctions first suggested by the genesis sequence, the inaugural entry of the Online Encyclopedia of Integer Sequences. The observation that this sequence counts all records in rooted trees with n nodes motivated combinatorial proofs of several identities relating tree records to the Cayley tree function. Space constraints prevent us from including results on the record code and a new proof of Cayley's forest formula, these appear in [DLP+26].

In the Numberphile podcast episode "The Number Collector" [Bra19], Neil Sloane recounts how he encountered the genesis sequence of the OEIS, the sequence "Normalized total height of all nodes in all rooted trees with n labeled nodes", during his PhD research, and how it inspired the creation of the OEIS. A short version of this story appears at the encyclopedia's wiki that we quote: "The sequence database was begun by Neil J. A. Sloane in early 1964 when he was a graduate student at Cornell University in Ithaca, NY. He had encountered a sequence of numbers while working on his dissertation, namely 1, 8, 78, 944, ... (now entry A000435 in the OEIS [OEI26]), and was looking for a formula for the n-th term, in order to determine the rate of growth of the terms. He noticed that although several books in the Cornell library contained sequences somewhat similar to this, this particular sequence was not mentioned. In order to keep track of the sequences in these books, NJAS started recording them on file cards, which he sorted into lexicographic order." (From Brief History of the OEIS.)

As all structures we consider are labeled, we simply use the terms tree and forest to refer to labeled trees and labeled forests. Let T be a rooted tree. The number of nodes in a tree or a forest (not counting $\circ$ whenever it happens to appear as one of the labels) defines its *order*. A node v of T is a *record* if it has the largest label along the unique path from v to the root. Otherwise we say that v is a non-record. The *tree record number* $R_\bullet(n,k)$ is defined as the number of rooted trees of order n with precisely k records. The tree record numbers are already present in the literature in the context of queues and traffic lights in

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 87–94, doi:10.4204/EPTCS.445.11

© E. Duchi, A. Lillo, P. Puerto, M. Rosas & S. Trandafir
This work is licensed under the Creative Commons Attribution-Noncommercial-No Derivative Works License.

the work of Haight [Hai59], A259334. The *height* of a node of T is the length of the path (number of edges) from it to the root. The *total height* of T is the sum of the heights of the nodes of T.

Write $[n]$ for $\{1,2,\ldots,n\}$ and $[n]_0$ for $\{0,1,\ldots,n\}$. An *endofunction* on $[n]$ is a function $f:[n]\to[n]$ for some n. Its *functional digraph* is the directed graph with nodes $[n]$, and one directed edge $(i,f(i))$ for each i in $[n]$. Given a rooted tree T, we define the *parent map* of T as the map that sends each non-root vertex to its parent, and the root to itself. Observe that this is a natural way of embedding the set of rooted trees in the set of endofunctions. An endofunction is said to be *connected* if its functional graph is connected after ignoring orientations. The *girth* of a connected endofunction is the number of edges in the unique cycle of its functional graph.

The height of a record and the girth of an endofunction

Through the lens of Joyal's bijection [Joy81], we reveal an elegant relationship between record numbers and connected endofunctions that constitutes the starting point of our journey. Fix two nonnegative integers $k\le n$.

Theorem 1. *The number of rooted trees of order n with a distinguished record at height $k-1$ coincides with the number of connected endofunctions on $[n]$ of girth k.*

Bijective proof. Let T be a rooted tree with a distinguished record v, root r. Let p be its parent map. We send the pair (T,v) to the endofunction that sends the root r to the distinguished record v, and any other vertex to its parent in T. See Figure 1. Note that f is a connected endofunction, as T is a spanning tree of the graph of f.

The inverse map of $(v,T)\mapsto f$ can be described as follows. Let f be a connected endofunction of $[n]$, and let $\sigma=(x_1\ x_2\ \cdots\ x_m)$ be the only cycle of f. First, we need to determine the vertices v and r. These are given by $v=\max_i x_i$, and $r=\sigma^{-1}(v)$. We reconstruct the tree T by first removing the edge that connects v with r in the graph of f, and then rooting the resulting tree at r. Vertex v is a record of T as the vertices in the path from v towards the root are the elements of σ, and we have defined v as the maximum of these vertices. Consequently, the height of v is exactly one less than the girth of f. □

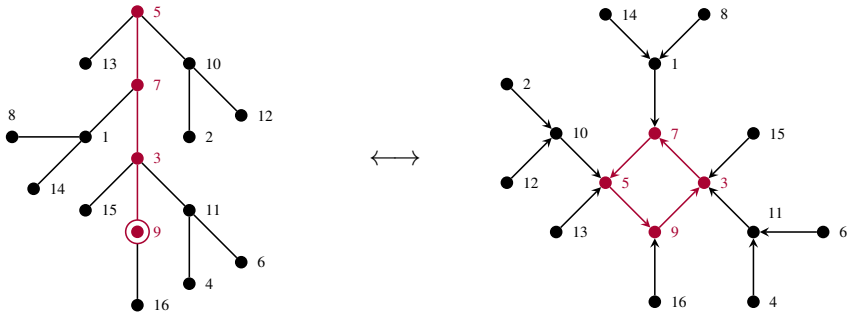


Figure 1: The bijection of Theorem 1. The distinguished record $v=9$ is circled. The path from v to the root (on the left) and the only cycle in the resulting endofunction (on the right) are highlighted.

Corollary 2. *The total number of records in all rooted trees with n nodes is equal to the number of connected endofunctions on $[n]$.*

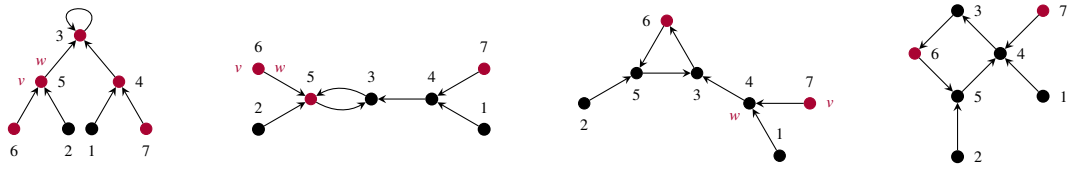


Figure 2: The repeated application of the bijection in Theorem 3. The records are highlighted. Note that the leftmost endofunction has at least 4 records and girth 1, and the rightmost endofunction has at least 1 record and girth 4.

The sequence counting connected endofunctions has been studied in depth, see [FS09], or the entries of the OEIS A001865 and A20168.

Some consequences

Let f be an endofunction on $[n]$. The set of elements of the form $f^k(i)$ for some $k \geq 0$ is referred to as the *orbit* of i, for any fixed element i of $[n]$. If the endofunction f is connected we refer to the *cycle* of f as the cyclic permutation obtained by restricting f to the elements of its unique cycle. Finally, an element $i \in [n]$ is said to be a *record* if i is larger than or equal to all the elements of its orbit. Fix three nonnegative integers $k, m \leq n$.

Theorem 3. *The number of connected endofunctions on $[n]$ of girth m and at least $k+1$ records coincides with the number of connected endofunctions on $[n]$ of girth $m+1$ and at least k records.*

Bijective proof. Let f be a connected endofunction of girth m and at least $k+1$ records. Let v be the k-th greatest record of f. Observe that there is exactly one record on the cycle σ of f: the smallest record of the endofunction, and that it is smaller than the k-th greatest record. Therefore, v does not belong to σ.

Let w be the first element in the orbit of v under f such that $f(w)$ belongs to σ. Denote $x = \sigma^{-1}(f(w))$. We define an endofunction g by $g(z) = f(z)$ if $z \neq x$, and $g(x) = w$. It is not hard to verify that g is a connected endofunction of girth $m+1$ and its k greatest records coincide with those of f. Reversing the bijection is straightforward by first identifying v and w. □

Corollary 4. *The number of connected endofunctions on $[n]$ of girth m and at least k records only depends on $m+k$. In particular, the number of rooted trees with n nodes and at least k records coincides with the number of connected endofunctions on $[n]$ of girth k.*

Bijective proof. Fix two positive integers k and m and let $E_{m,k}$ denote the set of endofunctions of $[n]$ with girth m and at least k records. Applying repeatedly the bijection of Theorem 3, we obtain the following bijections:

$$E_{m,k} \longleftrightarrow E_{m+1,k-1} \longleftrightarrow E_{m+2,k-2} \longleftrightarrow \cdots \longleftrightarrow E_{m+k-1,1}.$$

Therefore, the cardinality of $E_{m,k}$ only depends on $m+k$. Thus, with $m = 1$, $|E_{1,k}| = |E_{k,1}|$. The result follows by noticing that endofunctions with girth 1 are parent maps of trees, and that any endofunction has at least one record. □

We conclude that the number of rooted trees of order n with at least k records coincides with the number of tree cycles of order n of k rooted trees. The number of trees of order n with k records such that the subgraph induced by its records is connected coincides with the number of cycles of order n

of k unrooted trees. The set of rooted trees of order n with a distinguished record at height $k-1$ is in bijection with the set of trees of order n with at least k records. The set of rooted trees of order n and with a distinguished record v at height m and with at least k records greater than v is in bijection with the set of endofunctions on $[n]$ of girth $m+k+1$.

The tree record formula

We provide a combinatorial proof for the elegant closed formula for the tree record numbers of [LRT25a].

Corollary 5 (Theorem 4.1 of [LRT25a])**.** *The number of rooted trees labeled with $[n]$ and with k records obeys the equation*

$$R_\bullet(n,k) = k\,(n-1)\cdots(n-k+1)\,n^{n-k-1}.$$

Proof. The number of endofunctions of girth k is $n(n-1)\dots(n-k+1)\ n^{n-k-1}$ as kn^{n-k-1} counts the number of labeled rooted forests with a fixed k-set of roots (Cayley's forest formula), and $n(n-1)\dots(n-k+1)/k$ counts the number of ways to choose the k-set of roots of the trees and order it cyclically. Finally, Corollary 4 allows us to deduce the formula for the record numbers by subtracting the number of endofunctions of girth $k+1$ from those of girth k. □

The genesis sequence of the On-Line Encyclopedia of Integer Sequences

The *genesis sequence* $\Gamma = (\gamma_k)_k$ of the OEIS is defined by setting γ_k to be the sum of the height of all nodes of all rooted trees of order k, divided by k, OEIS A000435. Let T be a rooted tree. A *catalyst* for T to be a pair (u,v) of distinct vertices of T such that v is an ancestor of u, and denote by $C(T)$ the set of catalysts for T. A closely related, but different, notion of catalyst is introduced in [BEQG+25]. The number of catalysts for T is equal to the sum of the heights of all vertices in T.

Theorem 6. *The total number of non-root records in all rooted trees labeled with $[n]$ is equal to γ_n, the normalized total height of all nodes in all rooted trees with n labeled nodes.*

Bijective proof. Let $P(n)$ denote the set of pairs (T,r) where T is a rooted tree labeled with $[n]$ and r is a record of T different from the root. We exhibit an explicit bijection between the set $[n]\times P(n)$ and the set $C(n) = \bigsqcup_T C(T)$. Note that the existence of such a bijection implies the result. Let (v,T,r) be an element of $[n]\times P(n)$. We shall construct a tree T' and a vertex x such that (x,v) is a catalyst for T'. First, we compute the endofunction f corresponding with the pair (T,r) under the bijection of Theorem 1. Let σ denote the cycle of f, and let x be the first element in the orbit of v under f that belongs to the cycle σ. Define an endofunction g by $g(z) = f(z)$ if $z \neq \sigma^{-1}(x)$, and $g(\sigma^{-1}(x)) = v$, otherwise. Let π denote the cycle of g and let $w = \pi^{-1}(x)$. Lastly, define T' to be the tree with parent map p, where p is given by $p(z) = g(z)$ if $z \neq w, p(w) = w$. Then root T' at w. It is simple to check that $(x,v) \in C(T')$, and that all steps are reversible. □

Tree record generating functions

Fix a nonnegative integer k. Let $\mathscr{R}_k(z)$ be the generating function for rooted trees with precisely k records, and let $\mathscr{R}_{\geq k}(z)$ be the generating function for rooted trees at least k records:

$$\mathscr{R}_k(z) = \sum_{n\geq 0} R_\bullet(n,k)\ z^n, \qquad\qquad \mathscr{R}_{\geq k}(z) = \sum_{l\geq k} \mathscr{R}_l(z).$$

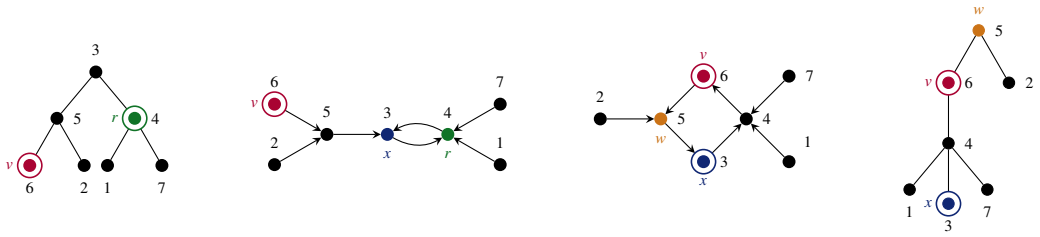


Figure 3: The bijection $[n] \times R(n) \to C(n)$ from Theorem 6. The left-most tree is rooted at 3, the selected vertex is $v = 6$ (crimson), and the selected nonroot record is $r = 4$ (green). The second graph is obtained via Theorem 1. Finally, $w = \pi^{-1}(x)$ (yellow) is the preimage of x on the unique cycle of the third graph and the root of the tree on the right.

Corollary 7. *The generating function for trees with at least k records is expressed in terms of the Cayley tree function as*

$$\mathscr{R}_{\geq k}(z) = \frac{\mathscr{T}^k(z)}{k}.$$

In Proposition 3.8 of [LRT25a], it was shown that the generating function for rooted trees with precisely k records obeys the elegant equation $\mathscr{R}_k = \frac{\mathscr{T}^k(z)}{k} - \frac{\mathscr{T}^{k+1}(z)}{k+1}$ Indeed, for $k = 1$, the equation becomes the familiar expression for the generating function $\mathscr{U}(z)$ for unrooted trees: $\mathscr{U}(z) = \mathscr{T}(z) - \frac{\mathscr{T}^2(z)}{2}$ (Observe that $\mathscr{U}(z) = \mathscr{R}_1(z)$ because trees rooted at their largest nodes are equinumerous with unrooted trees). A beautiful proof of this equation is given by Pierre Leroux's dissymmetry lemma [Ler88, LM92, Ges23].

Lemma 8 (A generalization of Pierre Leroux's dissimmetry lemma)**.**

$$\frac{\mathscr{T}^k(z)}{k} = \mathscr{R}_k + \frac{\mathscr{T}^{k+1}(z)}{k+1}. \tag{1}$$

Proof. There is a bijection between the set of endofunctions on $[n]$ of girth k and the union of the set of trees of order n with k records and the set of endofunctions on $[n]$ of girth $k+1$. Given an endofunction of girth k, the bijection of Corollary 4 gives a tree T with at least k records. Either T is in the set of trees of order n with k records, or it has at least $k+1$ records. In the latter case, the inverse of the same bijection gives an endofunction of girth $k+1$.

□

Differential equations and the record generating function

We provide a combinatorial interpretation for the derivative $\mathscr{R}'_k(z)$ of the series $\mathscr{R}_k(z)$. This series satisfies $\mathscr{R}'_k(z) = \frac{\mathscr{T}^k(z)}{z}$. Indeed, $\mathscr{R}'_1(z) = \frac{\mathscr{T}(z)}{z}$, as can be checked directly by taking the derivative with respect to z of $\mathscr{R}_1(z)$, the generating function for unrooted trees, and the expression of $\mathscr{R}_k(z)$ follows directly from Lemma 8 by differentiating with respect to z: $\mathscr{R}'_k(z) = \mathscr{T}(z)\mathscr{R}'_{k-1}(z)$.

Our aim is thus to give a combinatorial proof this equation. Let us consider the class of rooted Cayley trees $\mathscr{R}^*_k$ with k records, one of which is a marked virtual node, i.e. a node with no label that does not contribute to the size, and such that the subtree of the virtual node does not contain any record or, in other terms, the path going from each record to the root does not cross the virtual node (except for the record corresponding to the virtual node), then we have the following lemma.

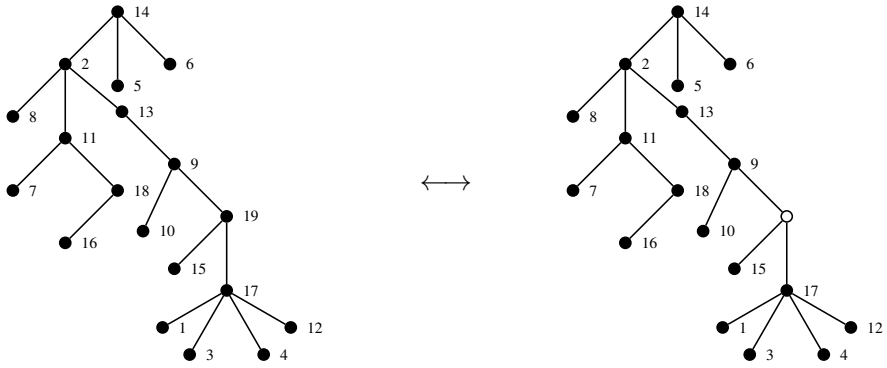


Figure 4: An example of the bijection between Cayley trees with n labels and Cayley trees with $n-1$ labels and a virtual node. The virtual node in white is obtained by blanking out the node with largest label, 19.

Lemma 9. *The series $\mathscr{R}'_k(z)$ is the generating function of the class $\mathscr{R}^*_k$.*

Proof. There is a simple bijection between the trees of size $n-1$ in $\mathscr{R}^*_k$ and the trees of size n in $\mathscr{R}_k$: this bijection consists in giving the label n to the virtual node of a tree of size $n-1$ from $\mathscr{R}^*_k$ and, conversely, in removing the label n of a tree of size n of $\mathscr{R}_k$ (see Figure 4). Since the virtual node is considered as a record and since it has no records in its sub-tree, the number of records does not change. Then, $\sum_{T\in\mathscr{R}^*_k}\frac{z^{|T|}}{|T|!}=\sum_{T'\in\mathscr{R}_k}\frac{z^{|T'|-1}}{(|T'|-1)!}=\mathscr{R}'_k(z)$. □

Let us now define an application ϕ mapping each element of $\mathscr{R}^*_k$ to a pair consisting after standardization in a Cayley tree and an element of $\mathscr{R}^*_{k-1}$: Given a tree T belonging to $\mathscr{R}^*_k$, we mark the parent of the virtual node and cut the tree into two sub-trees at the level of the virtual node. We denote by S' the sub-tree of T whose root is the virtual node, and by S the sub-tree obtained from T by stripping the sub-tree S' and marking the node where it was attached (see Figure 5, left-hand side). Then we transfer the label ℓ of the marked node of S to the root of S' to form a labeled tree $\bar{S}'$. Finally we shift all the labels $\ell_1<\ldots<\ell_p$ of S that are greater than ℓ and remove the mark to get a tree $\bar{S}$: the marked node of S gets label ℓ_i in $\bar{S}$, the node with label ℓ_i gets label ℓ_{i+1}, and the node with maximal label ℓ_p in S becomes a virtual (unlabeled) node in $\bar{S}$.

Proposition 10. *The mapping ϕ that sends T to $(\bar{S},\bar{S}')$ is a bijection between trees of $\mathscr{R}^*_k$ of size n, and pairs of trees $(\bar{S},\bar{S}')$ with total label set $\{1,\ldots,n\}$ such that the standardization of $\bar{S}$ belongs to $\mathscr{R}^*_k$ and the standardization of $\bar{S}'$ is a Cayley tree.*

Proof. Observe that the tree S has $k-1$ records since by definition of the class $\mathscr{R}^*_k$ there are no records in the sub-tree starting at the virtual node, except for the virtual node itself, and the shift of labels preserves the records: in particular the virtual node of $\bar{S}$ remains a record since it takes the place of the maximum of S. Clearly S and S' can be recovered from $\bar{S}$ and $\bar{S}'$ upon removing label ℓ at the root of $\bar{S}'$ and inserting it at the first larger label $\ell_1>\ell$ in $\bar{S}$ and shifting back the larger labels $\ell_1<\ldots<\ell_p$ until ℓ_p is put back on the virtual node of $\bar{S}$. Then S and S' are glued together to recover T. This proves that the mapping ϕ is injective.

Conversely, given a pair of trees $(\bar{S}, \bar{S}')$ as in the proposition the inverse construction yields a tree T with a virtual node and k records that are not in the sub-tree of the virtual node, that is a tree of $\mathscr{R}_k^*$, and $\phi(T) = (\bar{S}, \bar{S}')$. □

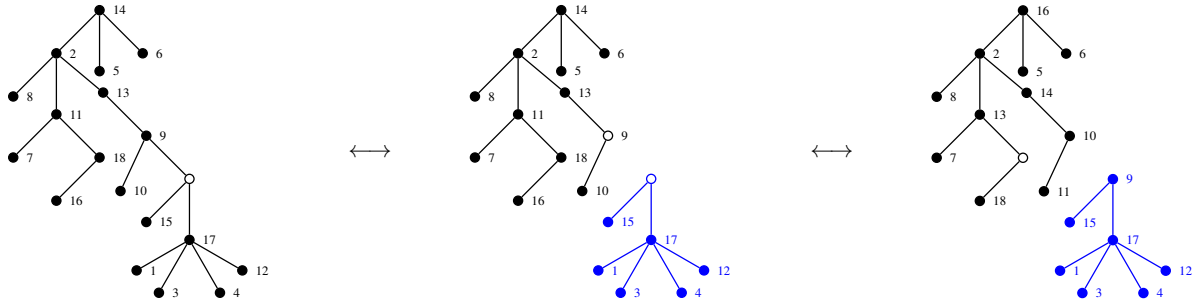


Figure 5: The decomposition ϕ of a Cayley tree with a virtual node. The trees obtained are $\bar{S}$, in black, and $\bar{S}'$, in blue. In the center there are the intermediary trees S and S'.

Corollary 11 (Proposition 3.4 of [LRT25a])**.** *The generating function for trees with precisely k records can be expressed in terms of the Cayley tree function as*

$$\mathscr{R}_k = \int_0^z \frac{1}{s} \mathscr{T}^k(s)\, ds.$$

Acknowledgements

We are grateful to Ira Gessel encouraging remarks and valuable suggestions, to Cyril Bandelier for engaging discussions, and to Andrea Sportiello for his insightful remarks.

This work has been partially supported by Grants PID2020-117843GB- I00 and PID2024-157173NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by FEDER, UE. Author ST is supported by FoQaCiA which is funded by the European Union and NSERC.

References

[BEQG+25] Emmanuel Briand, Luis Esquivias-Quintero, Álvaro Gutiérrez, Adrián Lillo & Mercedes Rosas (2025): *Non-intersecting paths and the determinant of the distance matrix of a tree*. doi:10.48550/arXiv.2407.01227. arXiv:2407.01227.

[Bra19] Haran Brady (2019): *The Number Collector, with Neil Sloane, Numberphile Podcast*. YouTube. Available at https://www.youtube.com/watch?v=mNk_MfFKnuY.

[DLP+26] Enrica Duchi, Adrián Lillo, Pablo Puerto, Mercedes Rosas & Stefan Trandafir (2026): *The genesis sequence, tree records and endofunctions*. *https://arxiv.org/abs/2601.07938*, doi:10.48550/arXiv.2601.07938. arXiv:2601.07938.

[FS09] Philippe Flajolet & Robert Sedgewick (2009): *Analytic Combinatorics*. Cambridge University Press, Cambridge, doi:10.1017/CBO9780511801655.

[Ges23] Ira M. Gessel (2023): *Good Will Hunting's Problem: Counting Homeomorphically Irreducible Trees*. doi:10.48550/arXiv.2305.03157. arXiv:2305.03157.

[Hai59] Frank A. Haight (1959): *Overflow at a traffic light*. *Biometrika* 46, pp. 420–424, doi:10.2307/2333538.

[Joy81] André Joyal (1981): *Une théorie combinatoire des séries formelles*. *Adv. in Math.* 42(1), pp. 1–82, doi:10.1016/0001-8708(81)90052-9.

[Ler88] Pierre Leroux (1988): *Methoden der Anzahlbestimmung für einige Klassen von Graphen*. *Bayreuth. Math. Schr.* (26), pp. 1–36.

[LM92] Pierre Leroux & Brahim Miloudi (1992): *Généralisations de la formule d'Otter*. *Ann. Sci. Math. Québec* 16(1), pp. 53–80. Available at https://www.labmath.uqam.ca/~annales/volumes/16-1/PDF/053-080.pdf.

[LRT25a] Adrián Lillo, Mercedes Rosas & Stefan Trandafir (2025): *On the enumeration of records of rooted trees and rooted forests*. doi:10.48550/arXiv.2510.12536. arXiv:2510.12536.

[LRT25b] Adrián Lillo, Mercedes Rosas & Stefan Trandafir (2025): *On Weary Drivers, Records of Trees, and Parking Functions*. doi:10.48550/arXiv.2506.22145. arXiv:2506.22145.

[OEI26] OEIS Foundation Inc. (2026): *The On-Line Encyclopedia of Integer Sequences*. Published electronically at http://oeis.org.

The Curious Case of Reversible Elementary Second Order Cellular Automaton 115

Enrico Formenti
Université Côte d'Azur, CNRS, i3s
Nice, France
enrico.formenti@univ-cotedazur.fr

Supreeti Kamilya
Department of Computer Science and Engineering,
Birla Institute of Technology, Mesra, Ranchi, India
supreeti.kamilya17@bitmesra.ac.in

We prove that the reversible elementary second order cellular automaton rule 115 is periodic when started on finite initial configurations. We also study some families of finite configurations that have interesting period functions.

1 Introduction

Cellular automata (CA) are a formal model widely used in the simulation of complex systems with application domains ranging from Biology to Mathematics and Computer Science.

CA can be seen as a collection of finite automata arranged on a regular grid. Each automaton updates its state on the basis of its own state and of the one of a fixed finite set of neighboring automata. All automata have the same transition function and update their state in parallel.

Higher order Cellular Automata (HOCA) have been introduced to include in the local rule time dependencies for modeling time dependent phenomena. The dynamical behavior of the class of CA is mostly still unexplored. Only a few results are known and they concern injectivity and surjectivity properties [2, 8]; or a specific subclass [5]. More recently, a systematic exploration of HOCA in which memory has been fixed to 2 (*i.e.* they remember their previous state) has started in [6]. From one end, one may remark that these systems can be seen as standard CA over a larger alphabet (see [5]) but adopting the larger alphabet hides all details about the micro-behavior which might be quite interesting in applications. From that preliminary explorations the authors came up with the class of reversible elementary second order CA (RESOCA) and more specifically on the one based on the elementary rule 115. This has absolutely puzzling behavior which might result useful in practical applications. The main contribution of this paper is that all finite configurations have periodic behavior. However, the function which relates an initial (finite) configuration with its period seems to be quite complex and it is not fully understood yet. We show some families of initial configurations that have simple periodic functions and one which seems to have a quite complex periodic function with exponential growth with respect to the size of the initial configuration.

The paper is structured as follows. The next section introduces all the main notions used in this paper. Section 3 discusses the behavior of RESOCA 115 and the many questions issued by it. The following section contains the proof of results in Section 3. The *merge construction*, one of the main tools used to generate new periods from finite configurations is briefly discussed in Section 5. Section 6 studies some families of initial (finite) configurations and their period. In the last section, we draw our conclusions and propose some perspectives.

Due to space constraints, most proofs and tables have been omitted. They will be included in the journal version of this work.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 95–103, doi:10.4204/EPTCS.445.12

© E. Formenti & S. Kamilya
This work is licensed under the
Creative Commons Attribution License.

2 Preliminaries

2.1 Elementary Higher Order CA

An **elementary higher order CA** (EHOCA) consists of a tuple $\langle S, m, r, \delta \rangle$ where $S = \{0,1\}$ is the **set of states** of the CA; $m \in \mathbb{N}$ is the **memory** (also called the **order**); $r \in \mathbb{N}$ is the **radius** and accounts for how many left and right neighbors are taken into consideration for updating the state of the current cell. Finally, $\delta : \left(\{0,1\}^{2r+1}\right)^m \to (\{0,1\})^m$ is the transition function (also called **local rule**) that updates the state of the current cell according to the content of the neighborhood and of the memory.

A local function δ induces a **global function** $G_\delta : \left(\{0,1\}^{\mathbb{Z}}\right)^m \to \left(\{0,1\}^{\mathbb{Z}}\right)^m$ that demonstrates the evolution of the current configuration in the next time step. An elementary second order CA sets the value of m to 2. More precisely, for elementary second order CA, G_δ associates the vector $\vec{e} = (e^1, e^2) \in \left(\{0,1\}^{\mathbb{Z}}\right)^2$ with the vector $G_\delta(\vec{e}) \in \left(\{0,1\}^{\mathbb{Z}}\right)^2$ where $G_\delta(\vec{e})^1_i = e^2_i$ and

$$G_\delta(\vec{e})^2_i = \delta\left(\begin{bmatrix}(e^1_{i-1}, e^1_i, e^1_{i+1})\\ (e^2_{i-1}, e^2_i, e^2_{i+1})\end{bmatrix}\right)$$

for all $i \in \mathbb{Z}$. In other words, the new vector at position i is obtained by putting the old value of component 2 to component one, and computing the updated value of the new component 2 using the local rule δ.

An EHOCA is **injective** if its global transition function G_δ is one-to-one. If the function is onto, the EHOCA is **surjective**. An EHOCA is **bijective** if G_δ is both one-one and onto. We recall that for any EHOCA, there exists a standard CA which is topologically conjugated to it [5]. Hence, we call a EHOCA **reversible** if and only if its conjugated CA is reversible. By a result in [7], the class of reversible EHOCA coincides with the class of bijective EHOCA.

The number of local rules of second order CA is quite huge. It goes beyond the scopes of this paper the dynamics all of them in a single paper. In this work, we propose to focus our attention on a special subclass of EHOCA rules δ'_n that are **induced by an elementary cellular automata rule** δ_n, more precisely :

$$\delta'_n\left(\begin{bmatrix}a,b,c\\ d,e,f\end{bmatrix}\right) = \begin{bmatrix}e\\ b+\delta_n(d,e,f) \mod 2\end{bmatrix} \tag{1}$$

for all $a,b,c,d,e,f \in \{0,1\}$. We call this special class of second order elementary CA as **RESOCA** [6] defined by Equation (1).

A **configuration** is a function from $\mathbb{Z}$ to $\{0,1\}$. A configuration is **finite** iff only a finite number of cells have a state different from 0. For a finite configuration c, define $\min c = \min\{i \in \mathbb{Z} \mid c_i = 1\}$ and $\max c = \max\{i \in \mathbb{Z} \mid c_i = 1\}$. The **length** of c is denoted $|c|$ and is given by $|c| = |\max c - \min c| + 1$. A configuration $\vec{c} = \begin{bmatrix}b\\ c\end{bmatrix}$ is **finite** if both b and c are finite. For $\vec{c} = \begin{bmatrix}b\\ c\end{bmatrix}$ define $\min \vec{c} = \min(\min b, \min c)$ and $\max \vec{c} = \max(\max b, \max c)$, the length of $\vec{c}$ is given by $|\vec{c}| = |\max \vec{c} - \min \vec{c}| + 1$.

In this paper, we are going to make a large usage of some specific finite configurations for which we are going to introduce a specific notation. We denote $\underline{0}$ a configuration in which all cells are in state 0; $\underline{u}$ a configuration in which all states are in state 0 except for the cell at position 0 that is in state 1.

2.2 Elementary Cellular Automata

An EHOCA is an **elementary CA** (or ECA) if memory $m = 1$. A cell of an elementary CA changes its state based on the states of itself, its left neighbor and its right neighbor. Hence, the local rule δ can be

	111	110	101	100	011	010	001	000
f	0	1	1	1	0	0	1	1

Table 1: The tabular representation of the ECA rule 115.

conveniently represented in tabular form (see for example Table 1).

Interpreting the output row of the tabular representation of a local rule as a decimal number written in binary gives the well-known **Wolfram's number**. Indeed, in the sequel of this paper we will refer to a specific ECA by its Wolfram's number n and refer to its local rule by δ_n.

There are total 256 distinct ECA rules. The rules exhibit different dynamical behaviour, from simple to chaotic. The ECA rules has been intensively investigated and several classifications of their dynamics have appeared over the years [1, 9, 4, 11, 3].

These 256 rules are the basis for forming the second order CA rules studied in [6]. During the previously mentioned study we came across the rule 115, which has a very curious behavior that motivated the present work.

3 The RESOCA 115

Given a RESOCA f, a configuration $\begin{bmatrix} m \\ c \end{bmatrix}$ is **periodic** iff there exists an integer p such that $f^p\left(\begin{bmatrix} m \\ c \end{bmatrix}\right) = \begin{bmatrix} m \\ c \end{bmatrix}$; the smallest p satisfying the previous property is the **period** of $\begin{bmatrix} m \\ c \end{bmatrix}$. We say that p is a **period for** RESOCA 115 if there exists a finite configuration which has period p for RESOCA 115.

Theorem 1. *Any finite configuration is periodic for RESOCA 115.*

The proof of the previous theorem is long and a bit tedious, and we leave it for the next section. The fact that RESOCA 115 has a periodic behavior on finite configurations should not mislead us into thinking that the dynamics is very simple. None of this is true. We will see that there exist classes of configurations which present very complex sequences of periods, most of which are not yet fully understood. Let us start exploring some conditions that periods should satisfy.

Lemma 1. *For RESOCA 115, every periodic finite configuration has a period p such that $p \geq 3$ and p is a multiple of* 3.

We call **background** the particular sequence of states that we can see in a column of the space time diagram of the configuration $\begin{bmatrix} \underline{0} \\ \underline{0} \end{bmatrix}$ used in the proof of the previous lemma: $[001001001001\ldots]^T$ (remark that all the columns are the same for this particular space-time diagram). Therefore, what the previous lemma says is that any finite configuration in a sense synchronizes his period according to the background, forcing it to be a multiple of 3. Hence, the following is a natural question.

Question 1. *For RESOCA* 115*, in the context of finite configurations, are all multiple of* 3 *periods?*

Remark that the product of two integers that are multiple of 3 is a multiple of 3. This means that if p is a period, then p^2 respects the necessary conditions given in Lemma 1. Hence, the question follows:

Question 2. *If p is a period for RESOCA* 115*, is p^2 also a period?*

We can not answer the previous question yet, but we can answer the following close version of it.

Proposition 1. *Consider two finite configurations of period p and q, respectively. Then* $\mathrm{lcm}(p,q)$ *is also a period.*

The proof of the previous result is obtained via the *merge construction* that is explained in Section 5. In order to find arguments for answering to Question 1 we made several experiments which are reported in the following sections. All experiments were based on configurations of the type $\begin{bmatrix} u \\ c \end{bmatrix}$ where c is a finite configuration. In this way, many periods were discovered : $3, 6, 12, 15, 27, 30, 54, 60, 63, 123, \ldots$ Two periods p and q are **twins** if $q = p+3$. For example, in the previous sequence the periods 3 and 6 are twins as well as 12 and 15. We do not know whether all periods are twins, but twins seem to satisfy the following.

Conjecture 1. *Assume that p and* $p+3$ *are twins periods. Then, the next twins pair is* p' *and* $p'+3$*, where* p' *is given by*

$$p' = 2p + \frac{9-3(-1)^p}{2} \quad .$$

We cannot prove the conjecture but a weaker statement holds.

Proposition 2. *If p and* $p+3$ *are twin periods, then*

$$p' = \begin{cases} 2p & \text{if } p \text{ is odd} \\ 2p+6 & \text{otherwise.} \end{cases}$$

is also a period.

We remark that the previous proposition allows to produce a new period but not a pair of twin periods.

Finally, we observe that since all periods of RESOCA 115 are multiples of 3 then, we can write $\Pi = 3H$, where Π is the set of periods and H is defined as follows

$$H = \{k \in \mathbb{N} \mid 3k \text{ is a period of RESOCA 115}\}\,.$$

Hence, the initial question can be reformulated as follows.

Question 1*. *Is* $H = \mathbb{N}$*?*

If the answer to the previous question is negative then, more questions arise.

Question 3. *Is H syndetic (*i.e. *does it have bounded gaps) or thick (*i.e. *does it contain arbitrary large intervals)?*

Answering to these question is important not only for the theoretical aspects but also for the practical aspects like the application to cryptography that we propose at the end of this work, for example.

4 Periodicity of finite configurations

We are going to decompose the proof of Theorem 1 into a series of Lemmata to make it easier to follow.

Lemma 2. *Consider a finite configuration b such that* $b_i = 0$ *for all* $i \leq 0$ *and another finite configuration c such that* $c_i = 0$ *for all* $i < 0$ *and* $c_0 = 1$*. Let f be the global rule of RESOCA* 115 *and* $d^t = f^t\left(\begin{bmatrix} b \\ c \end{bmatrix}\right)_2$*. Then we have for all time* $t \in \mathbb{N}$

$$\forall i \in \mathbb{N} \setminus \{0,1\} \quad d^t_{-i} = \begin{cases} 1 & \text{if } t \equiv 1 \mod 3 \\ 0 & \text{otherwise.} \end{cases}$$

and for all times $t \geq 2$

$$d^t_{-1} = \begin{cases} d^{t-1}_{-1} & \text{if } t \equiv 0 \mod 3 \\ 1 & \text{if } t \equiv 1 \mod 3 \\ 1-(d^{t-1}_0 + d^{t-2}_{-1}) & \text{if } t \equiv 2 \mod 3 \end{cases}$$

where addition is taken modulo 2*.*

Lemma 3. *Consider two finite configurations* b *and* c *such that, for both of them, all cells with an index strictly larger than* 0 *are in state* 0*. Let* f *be the global rule of RESOCA* 115 *and* $d^t = f^t\left(\begin{bmatrix} b \\ c \end{bmatrix}\right)_2$*. Then we have, for all times* $t \in \mathbb{N}$

$$d^t_1 = \begin{cases} 1 & \text{if } t \equiv 1 \mod 3 \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the previous lemma says that if we have an initial condition as described, then the column with index 1 will show only the background.

Remark 1. *Assume that* b *and* c *are two finite configurations satisfying the hypothesis of the previous Lemma 3. Then,* $\sigma(b)$ *and* $\sigma(c)$ *also satisfy the hypothesis. Hence, we can conclude that* $\begin{bmatrix} b \\ c \end{bmatrix}$ *will show background for all columns with index bigger than* 1*.*

Lemma 4. *Consider two finite configurations* b *and* c *such that* $b_i = c_i = 0$ *for all* $i < 0$ *and* $b_0 = c_0 = 1$*. Let* f *be the global rule of RESOCA* 115 *and* $d^t = f^t\left(\begin{bmatrix} b \\ c \end{bmatrix}\right)_2$*. Then we have, for all time* $t \in \mathbb{N}$

$$\forall i \in \mathbb{N} \setminus \{0\} \quad d^t_{-i} = \begin{cases} 1 & \text{if } t \equiv 1 \mod 3 \\ 0 & \text{otherwise.} \end{cases}$$

and for all times $t \geq 2$

$$d^t_0 = \begin{cases} d^{t-1}_0 & \text{if } t \equiv 0 \mod 3 \\ 1 & \text{if } t \equiv 1 \mod 3 \\ 1-(d^{t-1}_1 + d^{t-2}_0) & \text{if } t \equiv 2 \mod 3 \end{cases}$$

where addition is taken modulo 2*.*

Lemma 5. *Consider two finite configurations* b *and* c *such that* $b_i = c_i = 0$ *for all* $i < 0$ *and* $b_0 = c_0 = 1$*. Let* f *be the global rule of RESOCA* 115 *and* $d^t = f^t\left(\begin{bmatrix} b \\ c \end{bmatrix}\right)_2$*. Then we have, for all times* $t \geq 3$

$$\forall i \in \mathbb{N} \setminus \{0,1\} \quad d^t_{-i} = \begin{cases} 1 & \text{if } t \equiv 1 \mod 3 \\ 0 & \text{otherwise.} \end{cases}$$

and

$$d^t_{-1} = \begin{cases} d^{t-1}_0 & \text{if } t \equiv 0 \mod 3 \\ 1 & \text{if } t \equiv 1 \mod 3 \\ 1-(d^{t-1}_1 + d^{t-2}_0) & \text{if } t \equiv 2 \mod 3 \end{cases}$$

where addition is taken modulo 2*.*

Proof of Theorem 1. Consider two finite configurations b and c. If $b = c = \underline{0}$, then it is not difficult to see that the space-time diagram of $\begin{bmatrix}\underline{0}\\ \underline{0}\end{bmatrix}$ contains only background columns and hence has period 3. Now assume that at least one of b or c is different from $\underline{0}$. Without loss of generality we can assume that for all $i < 0$, $c_i = b_i = 0$ (otherwise we can choose $c' = \sigma^u(c)$ and $b' = \sigma^u(b)$, where $u = \min \begin{bmatrix}b\\ c\end{bmatrix}$). By Lemma 3, all columns with index larger than $\max \begin{bmatrix}b\\ c\end{bmatrix}$ are background. We will distinguish three disjoint cases. For each of them, we will prove that for all $i < -1$, all columns in the space-time diagram of $\begin{bmatrix}b\\ c\end{bmatrix}$ contain only background. This and the previous remark allow to deduce that the sequence of states that we can see in the space-time diagram of initial condition $\begin{bmatrix}b\\ c\end{bmatrix}$ between cell -1 and cell $\max \begin{bmatrix}b\\ c\end{bmatrix}$ is ultimately periodic. However, since RESOCA 115 is reversible, we can conclude that the sequence is periodic. Here are the proofs for the three cases announced above: 1) $b_0 = 0$ and $c_0 = 1$: this is Lemma 2; 2) $b_0 = 1$ and $c_0 = 0$: this is Lemma 5; 3) $b_0 = 1$ and $c_0 = 1$: this is Lemma 4. □

Another interesting consequence of the lemmata in this section is that, for any finite configuration $\vec{c}$, the size of the configuration and its position are essentially preserved along the evolution. More precisely, we can give the following.

Corollary 1. *Let f be the global rule RESOCA 115. For any finite configuration $\vec{c}$, we have $\forall t \in \mathbb{N}$ $\max \vec{c} = \max f^t(\vec{c})$ and $\forall t \in \mathbb{N} \setminus \{0\}$ $\min f(\vec{c}) = \min f^t(\vec{c})$. Moreover, $\min f(\vec{c}) - \min \vec{c} \leq 1$.*

Although the previous result follows rather directly from earlier results, it is nevertheless of significant importance. Indeed, a result of Moothathu implies that the existence of periodic points with the properties described in the corollary guaranties the existence of infinitely many periodic points with the same period [10]. Explicit examples of this fact are provided in Section 6.

5 The merge construction

We dedicate to this construction a separated section since it is useful in different situations when trying to build configurations with peculiar periods.

Consider two finite configurations $\vec{c}_1, \vec{c}_2$. By Theorem 1, these configurations are periodic of period p and q, respectively. Using Lemma 2 through 5, we remark that along the evolution of both configurations no modification (with respect to background) is made before $\min \vec{c}_1 - 1$ (resp., $\min \vec{c}_2 - 1$) or after $\max \vec{c}_1$ (resp., $\max \vec{c}_2$). These facts suggest that if we build an initial configuration in which we position the non-background part of $\vec{c}_1$ on the left of the cell of index 0 and $\vec{c}_2$ on the right, leaving a sufficient number of cells in state 0 separating them, then we should have a new periodic configuration with the period which is some combination of the two original periods. More formally, define u as the largest integer such that $\max \sigma^u(\vec{c}_1) \leq -3$ and v as the least integer such that $\min \sigma^v(\vec{c}_2) \geq 3$. Let us build a new finite configuration $\vec{c}_3$ that has a first component

$$\forall i \in \mathbb{Z} \quad ((\vec{c}_3)_1)_i = \begin{cases} \sigma^u((\vec{c}_1)_1)_i & \text{if } i \leq 0 \\ \sigma^v((\vec{c}_2)_1)_i & \text{if } i > 0 \end{cases}$$

and as a second component

$$\forall i \in \mathbb{Z} \quad ((\vec{c}_3)_2)_i = \begin{cases} \sigma^u((\vec{c}_1)_2)_i & \text{if } i \leq 0 \\ \sigma^v((\vec{c}_2)_2)_i & \text{if } i > 0. \end{cases}$$

Proposition 3. *Let $\vec{c}_1, \vec{c}_2$ and $\vec{c}_3$ as above. Let p (resp., q) be the period of $\vec{c}_1$ (resp., $\vec{c}_2$). Then, $\vec{c}_3$ is periodic of period* $\mathrm{lcm}(p,q)$.

Proof. The proof is trivial and left to the reader. □

Remark that the merge construction can be generalized, in the obvious way, to work with an arbitrary (but finite) number of finite configurations.

6 Families of configurations and their period functions

In this section, we focus our attention on families of initial configurations. Since all initial configurations share the same memory $b = \underline{u}$, each family will be described solely in terms of its set of current configurations. Every family is defined by a regular expression depending on a parameter $k \in \mathbb{N}$. For each value of k, the corresponding regular expression generates a binary word w of odd length, from which the current current configuration is constructed by setting $c_{-k+i} = w_i$ for $i \in [0, |w|]$, where $k = \frac{|w|-1}{2}$. Our objective is to determine the **period function** $P(k)$ associated with the family, that is, the function assigning the initial configuration of parameter k its corresponding period.

6.1 Family $10^{2k-1}1$ ($k \geq 1$)

This is a very simple family of configurations as it is confirmed by the following:

Proposition 4. *The period function $P(k)$ of the family $10^{2k-1}1$ is as follows:*

$$P(k) = \begin{cases} 2^{2k+4} - 1, & \text{if } k = 1,2 \\ 15, & \text{otherwise.} \end{cases}$$

6.2 Family 1^{2k+1}

Also this family of periods is pretty simple but we need a different proof technique than the previous one.

Proposition 5. *The period function $P(k)$ of the family 1^{2k+1} is as follows:*

$$\forall k \in \mathbb{N}, P(k) = \begin{cases} 3 & \text{if } k = 0 \\ 12 & \text{if } k = 1 \\ 6 & \text{otherwise.} \end{cases}$$

6.3 Family $1(01)^k$

For this family, the period function appears to be much more complex than the one of the two previous families. Using a computer program, we produced a long sequence of periods. However, we have no clue for a closed formula for the period function.

Looking more carefully, we remarked that $P(2h+1) = 4^{2h+3} - 1$. We suspect that this is true for any $h \in \mathbb{N}$. Hence, we formulate the following.

Conjecture 2. *The period function $P(k)$ of the family $1(01)^k$ is such that $P(2i+1) = 4^{2i+3} - 1$ for $i \in \mathbb{N}$.*

For even values of k, the period function $P(k)$ seems more complicate. However, from our experiments we could recognize a pattern. For $i > 2$, call q_i the number obtained by adding 1 to $P(2i)$ and then dividing the result by 8. We could remark that the representation in base 4 of $q_i = (P(2i)+1)/8$ is given by $1^i 0^2 2^{i-2} 3$ (the highest weight is at the left). Therefore, q_i can be written as $q_i = 3 \cdot 4^0 + 2 \cdot \sum_{j=1}^{i-2} 4^j + \sum_{j=i+1}^{2i} 4^j = 3 + \frac{4^{2i+1} - 14 \cdot 4^{i-1} - 8}{3}$. Recalling the relation between q_i and p_i, we get $P(2i) = 8p_i - 1 = 23 + \frac{2 \cdot 4^{2i+2} - 7 \cdot 4^{i+1} - 4^3}{3} = \frac{2 \cdot 4^{2i+2} - 7 \cdot 4^{i+1} + 5}{3}$. We can therefore emit the following.

Conjecture 3. *The period function $P(k)$ of the family $1(01)^k$ is such that $P(2i) = \frac{2 \cdot 4^{2i+2} - 7 \cdot 4^{i+1} + 5}{3}$ for $i \in \mathbb{N}$.*

7 Conclusions and perspectives

In this paper, we investigated the behavior of RESOCA 115. In particular, we proved that, for every finite initial configuration, the evolution is periodic with period divisible by 3. We also identified families of finite configurations exhibiting either a simple period function or a considerably more intricate one. Nevertheless, the relationship between the structure of a finite initial configuration and the exact value of its period remains poorly understood. Clarifying this connection constitutes a challenging and promising direction for future research.

Through computational experiments, we identified the following periods: $3, 6, 9^*, 12, 15, 27, 30, 54, 60, 63, 123, 126, 135, 189, 246, 252, 255, 270, 315, 378, 507, 510, 615, 630, 1020, 1023, 1230, 1260, 1890, 2043, 2046, 2535, 2583$ and many others. We emphasize that every period in this list arises from a finite configuration, with the notable exception of period 9. Indeed, rather curiously, we were only able to construct non-finite configurations exhibiting this period.

This observation prevents the use of the merge construction to fill the gap at period 18. More precisely, the technique employed to establish the existence of period 9 does not extend to period 18, as it would require the analysis of graphs with 2^{72} vertices and 2^{74} edges; far beyond the capabilities of our machines[1]. Developing more efficient computational methods for proving or disproving the existence of a prescribed period therefore constitutes another compelling direction for future research.

All the hype surrounding the periods is that if we succeed in getting a certain amount of knowledge about them, then we could provide a concrete playground for a new secret sharing protocol in which the shares are given by some finite configurations and the secret is hidden in the period of a configuration obtained by the shares via the merge construction. We highlight several advantages of this new protocol. Suppose there are n shares, where n itself is secret. In this setting, the presence of all shares is not necessarily required to decrypt the secret; however, one is not completely sure that the decryption is correct until every share has been taken into account. Moreover, the size of each share can be chosen to be logarithmic in the bit-size of its period, and the order in which the shares are processed is irrelevant.

A thorough analysis of this protocol is currently under investigation and will be presented in a separate forthcoming paper.

[1] The technique used for period 9 was suggested by Jarkko Kari in a private communication.

References

[1] G. Braga, Gianpiero Cattaneo, Paola Flocchini & C. Quaranta Vogliotti (1995): *Pattern Growth in Elementary Cellular Automata*. *Theor. Comput. Sci.* 145(1&2), pp. 1–26, doi:10.1016/0304-3975(94)00155-C.

[2] L. Le Bruyn & M. Van den Bergh (1991): *Algebraic properties of linear cellular automata*. *Linear algebra and its applications* 157, pp. 217–234, doi:10.1016/0024-3795(91)90116-E.

[3] Gianpiero Cattaneo, Enrico Formenti, Luciano Margara & G Mauri (1999): *On the Dynamical Behavior of Chaotic Cellular Automata*. *Theoretical Computer Science* 217(1), pp. 31 – 51, doi:10.1016/S0304-3975(98)00149-2.

[4] Karel Culik II & Sheng Yu (1988): *Undecidability of CA Classification Schemes*. *Complex Systems* 2(2), pp. 177–190, doi:10.5555/45269.45271.

[5] Alberto Dennunzio, Enrico Formenti, Luca Manzoni, Luciano Margara & Antonio E. Porreca (2019): *On the dynamical behaviour of linear higher-order cellular automata and its decidability*. *Inf. Sci.* 486, pp. 73–87, doi:10.1016/J.INS.2019.02.023.

[6] Enrico Formenti & Supreeti Kamilya (2024): *On Elementary Second Order Cellular Automata*. In: *Asian Symposium on Cellular Automata Technology*, Springer, pp. 204–218, doi:10.1007/978-3-031-56943-2_16.

[7] Gustav A Hedlund (1969): *Endomorphisms and automorphisms of the shift dynamical system*. *Mathematical systems theory* 3(4), pp. 320–375, doi:10.1007/BF01691062.

[8] Jarkko Kari (2000): *Linear Cellular Automata with Multiple State Variables*. In H. reichel & Sophie Tison, editors: *STACS 2000*, *LNCS* 1770, Springer-Verlag, pp. 110–121, doi:10.1007/3-540-46541-3_9.

[9] Petr Kůrka (1997): *Languages, Equicontinuity and Attractors in Cellular Automata*. *Ergodic theory and dynamical systems* 17(2), pp. 417–433, doi:10.1017/S014338579706985X.

[10] T.K. Subrahmonian Moothathu (2005): *Homogeneity of surjective cellular automata*. *Discrete and Continuous Dynamical Systems* 13(1), pp. 195–202, doi:10.3934/dcds.2005.13.195.

[11] Martin Schüle & Ruedi Stoop (2012): *A Full Computation-relevant Topological Dynamics Classification of Elementary Cellular Automata*. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 22(4), p. 043143, doi:10.1063/1.4771662.

Generation of Maximal Snake Polyominoes Using a Deep Neural Network

Benjamin Gauthier Alain Goupil Fadel Toure
Université du Québec à Trois-Rivières, Trois-Rivières, Canada
Benjamin.Gauthier@uqtr.ca Alain.Goupil@uqtr.ca Fadel.Toure@uqtr.ca

Maximal snake polyominoes are difficult to study numerically in large rectangles, as computing them requires the complete enumeration of all snakes for a specific rectangle size, which corresponds to a brute force algorithm. This hinders the study of maximal snakes in larger rectangles. Moreover, most enumerable snakes lie in small rectangles, obscuring large-scale patterns. In this paper, we investigate the contribution of a deep neural network to the generation of maximal snake polyominoes from a data-driven training, where the maximality and adjacency constraints are not encoded explicitly, but learned. To this extent, we experiment with a denoising diffusion model, which we referred as Structured Pixel Space Diffusion (SPS Diffusion). We find that SPS Diffusion generalizes from small rectangles to larger ones, generating valid snakes up to 28×28 squares and producing maximal snake candidates on squares close to the current computational limit. The model is, however, prone to errors such as branching, cycles, or multiple snake components. Overall, the diffusion model is promising and suggests that complex combinatorial objects can be understood by deep neural networks, which is useful in their investigation.

1 Introduction

In the past 40 years polyominoes and polycubes have been the object of intense investigations from different points of view: combinatorial, algorithmic and statistical physics ([6] and ref. therein, [12]). The problem of finding the length of maximal snake polyominoes in an arbitrary rectangle is still open. The most effective algorithmic approach currently available consists in enumerating all snake polyominoes and then selecting the maximal ones.

Early foundational work on such brute force algorithms was conducted by Jensen [10], who used transfer-matrix methods and enumeration techniques to count polyominoes and tree-like polyominoes. Later work with Guttmann [9] analyzed the asymptotic behavior of polyomino counts using tools from statistical mechanics, showing that the number of polyominoes grows by a factor of about 4.06 when the number of cells increases by one. Since there exists families of snake polyominoes that are in bijection with Fibonacci numbers ([5]), the number of snakes polyominoes grows exponentially with respect to their size and since there is an injection from the set of snake polyominoes of length n into the set of self-avoiding walks ([5]) the growth constant of snake polyominoes is at most that of self-avoiding walks which is known to be 2.6 ([8]). There is thus an inherent difficulty in generating snakes by exhaustive enumeration, especially in large rectangles with size beyond the range where brute force enumeration is feasible.

Barequet and Ben-Shachar [1] revisited polyomino enumeration by introducing bounding boxes rotated by 45° instead of axis-aligned rectangles. This reformulation improved brute force enumeration and allowed polyominoes to be enumerated up to size 70, compared to Jensen's previous limit of 56. Nevertheless, the enumeration complexity remains exponential, highlighting the difficulty of counting and generating maximal snakes in large rectangles.

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 104–113, doi:10.4204/EPTCS.445.13

© B. Gauthier, A. Goupil & F. Toure
This work is licensed under the
Creative Commons Attribution License.

Another issue is that we observed (without proof) that the general structure of maximal snake polyominoes becomes more apparent in larger rectangles compared to small rectangles. In our brute force computations, we were able to reach sizes of up to 15×15 for squares, where the general pattern begins to emerge, but significantly larger rectangles remain out of reach of conventional methods. It becomes relevant to observe maximal snake polyominoes in large rectangles in the hope of formulating conjectures of exact expressions about their size and their enumeration.

A theoretical upper bound on the maximal length of snake polyominoes is provided by the 2/3 theorem [13]. While these upper bounds provide useful theoretical guidance, they do not help overcome the computational difficulties of generating maximal snake polyominoes in large rectangles.

Given these limitations, deep neural networks (DNNs) offer a possible way to study snakes in larger rectangles. Indeed, DNNs have polynomial complexity, which makes them theoretically faster than the current brute force algorithms used for generating snake polyominoes of maximal length. While DNNs have achieved remarkable success in domains such as image generation and natural language processing [19, 7], their application in discrete combinatorial objects and mathematical structure discovery remains relatively unexplored.

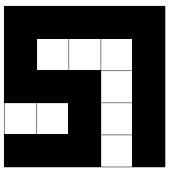
In this work, we generate snake polyominoes using a Structured Pixel Space Diffusion (SPS Diffusion) model. Our goal is not to replace exact methods for enumerating snake polyominoes, but rather to evaluate whether deep neural networks can learn the structural constraints of snake polyominoes and produce candidates for maximal snakes, thus offering an additional tool to classical enumeration techniques and, as a byproduct, an argument for the existence of combinatorial structure of these objects. Section 2 introduce the necessary concepts. In section 3 we introduce two neural network and compare them. In section 4. we present the results obtained.

2 Preliminaries

2.1 Snake polyominoes definitions

We recall the definitions needed below. For a rigorous definition of snake polyominoes, see [11].

A snake polyomino, or more briefly a snake, is a polyomino where all cells are of degree two except for two cells of degree one, called the head and tail. Here, a maximal snake means a snake of maximum area in a given rectangle. In a grid representation, living cells (black) correspond to the snake body and dead cells (white) to empty cells (Fig. 1a). We use a matrix representation where living cells are labeled 1 and dead cells 0 (Fig. 1b). Snakes, and their matrix representations, are contained in discrete rectangles of size $H \times W$. From graph theory, the degree of a cell x is the number of edge-adjacent living cells. We assume the standard 4-connexity of square cells, where each cell has at most four neighbors [2].



(a) Grid representation of a snake.

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

(b) Matrix representation of a snake.

Figure 1: Different representations of snake polyominoes.

To qualitatively assess generated snakes, we identify recurring snake-like structures, i.e., patterns

commonly observed in maximal snakes (Fig. 2). Examples include stairs, which provide a compact folding of cells of degree at most two [13], triangles of six dead cells, and configurations where the head or tail folds compactly. Such structures frequently appear in maximal snakes, although they are not exclusive. For example, head and tail are frequently observed along the borders and corners of the rectangle.

Conversely, we identify snake malformations, which are structures incompatible with valid snakes (Fig. 2). Examples include branching, cycles (absence of head and tail), open heads/tails and forests of snakes where several disconnected components appear. The presence of such structures generally indicates that the snake is not maximal. Note that an open head/tail doesn't imply that the snake is not maximal, rather that it may not be. It is also useful to know that density is in general greater along the edges of the rectangle than in the center. This fact has been proved for $2D$-words ([13]). Border density is thus a good indicator that a snake is maximal.

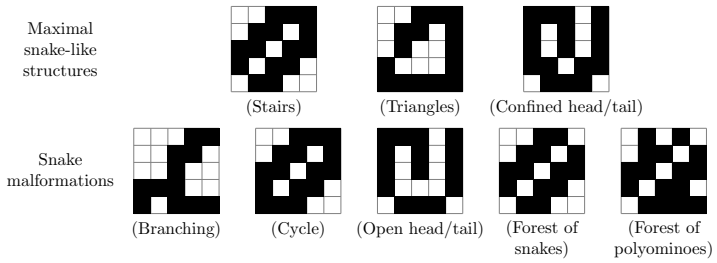


Figure 2: Maximal snake-like structures (above) - stairs, a triangle of dead cells and the confined tail/head. Snake malformations (below) - branching, cycle, forest of snakes and an open head/tail. Note that for the open head/tail, the disjointed segments should be considered part of the same snake, as if the rectangle was bigger than shown here.

2.2 Deep neural network definitions

In the SPS Diffusion model to be introduced later, data are represented as matrices. The source (**src**) is the input of the model, the target (**tgt**) the expected output, and the output (**out**) the model prediction. The model itself is a parameterized function represented by a deep neural network and learned from data.

Training consists in minimizing a loss function that measures the discrepancy between **out** and **tgt**. The parameters of the neural network are updated using gradient descent combined with backpropagation, which computes the gradient of the loss with respect to the model parameters [17].

A fundamental layer of diffusion model is the convolutional layer, which is itself a neural network component that applies small learnable filters (kernels) across an input (such as an image) to extract local patterns. Each filter slides over the input and computes dot products, producing feature maps that detect structures such as edges, textures, or shapes. Convolutional layers are central to diffusion models and enable the efficient capture of spatially localized features, which is particularly important for modeling snakes [16].

Finally, we define self-attention, which is a neural network mechanism that allows a model to contextualize different parts of an input to one another. Given a set of input representations, the layer computes queries, keys, and values, and determines how strongly each element should attend to the others through similarity scores (attention scores). This produces weighted combinations of the values, enabling the model to capture long-range dependencies and global structure [19]. In spatial settings such as snake

polyomino generation, positional information is incorporated using 2D Rotary Positional Embeddings (RoPE-2D) [18], which encode relative spatial relationships while preserving the ability of attention to model interactions across the rectangle.

3 Methodology for the SPS Diffusion model

3.1 Snake generation

We investigate the use of a denoising diffusion probabilistic model (DDPM) for generating maximal snakes. We call our variant structured pixel space diffusion (SPS Diffusion), for reasons detailed in Sec. 3.2.

The process of generating polyominoes with SPS Diffusion is directly analogous to image generation with a standard DDPM and it is described as follows:

1. An image of pure noise, sampled from a normal distribution, is initialized and given to the model.
2. At each step, the model predicts the amount of noise to be removed from the sample in order to move closer to a structured configuration.
3. The predicted noise is subtracted from the current sample.
4. The updated sample is fed back into the model, and the process is repeated for a given number of steps. A larger number of steps typically yields higher-quality outputs.

Fig. 3 illustrates the evolution of an image from pure noise to a maximal snake. This process is called *backward diffusion*. As in standard DDPMs, the backward diffusion process is described by Eq. (1).

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(\mathbf{x}_t, t)\right) + \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}\,\varepsilon \tag{1}$$

where $\mathbf{x}_t$ is the image at step t, $\varepsilon_\theta(\mathbf{x}_t, t)$ is the prediction of the model and $\varepsilon \sim \mathcal{N}(0,1)$ is a statistical variable following a normal distribution of average 0 and variance 1 [3]. The image $\mathbf{x}_0$ is the fully denoised image (a maximal snake ideally), and the image $\mathbf{x}_T$ is the pure noise image (noisy polyomino). The other symbols are constants from the forward diffusion process; see Sec. 3.3 and [3].

3.2 Differences between SPS Diffusion and Stable Diffusion

SPS Diffusion operates similarly to Stable Diffusion: a noisy image is iteratively denoised until the final output emerges. However, maximal snakes are structurally much simpler than natural images. Polyominoes are represented in binary form, with a single feature channel taking values 0 (white) or 1

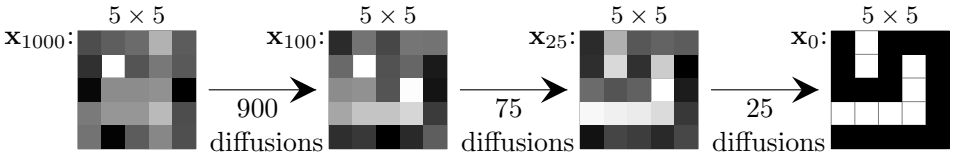


Figure 3: The backward diffusion process to obtain a maximal snake using an SPS Diffusion model. The arrows indicate the number of diffusion steps used for the next image ($\mathbf{x}_t$). The initial pure noise image $\mathbf{x}_{1000}$ is diffused into a maximal snake image $\mathbf{x}_0$.

(black). In practice, this channel initially takes values in $\mathbb{R}$ and is rounded to the nearest integer only at the end, which explains the gray cells visible during the backward diffusion process in Fig. 3.

In contrast, natural images typically use three feature channels (red, green, blue), each ranging from 0 to 255, yielding over 16 million possible colors. Moreover, unlike Stable Diffusion, the model is not conditioned on text or image prompts: the generation always converges to a specific family of images, namely maximal snakes. Finally, snake grids contain far fewer pixels than photographic images; in practice they are smaller than 64×64, whereas natural images often exceed 512×512 pixels. These differences necessitate modifications to the standard Stable Diffusion architecture.

First, our model does not include a Contrastive Language–Image Pretraining (CLIP) encoder. In Stable Diffusion, CLIP is used to project text and image data into a shared embedding space, enabling training on text–image pairs. CLIP can also serve as a loss function to measure how well a generated image matches a textual prompt [14]. Since maximal snake polyomino generation involves no prompts, this module is unnecessary.

Second, the model is composed solely of a U-Net. As in Stable Diffusion, the U-Net predicts the noise to be removed from the current sample. To achieve this, the U-Net reduces the spatial resolution of the input as it applies successive convolutional layers, a process known as downsampling [16]. However, because snake polyomino images are already small, excessive downsampling would collapse the structural information in the image entirely. While a typical Stable Diffusion U-Net may employ 5 or 6 levels of downsampling, SPS Diffusion is restricted to only 3 levels to preserve structural fidelity. For these reasons, this U-Net is better referred to as a mini U-Net. Each level of the mini U-Net applies several convolutions to the image, followed by a self-attention computation which uses RoPE 2D.

Third, the model does not require a Variational Autoencoder (VAE). In Stable Diffusion (or more generally, Latent Diffusion models), the VAE compresses images from pixel space into a latent space to reduce computational cost, since the U-Net cannot practically process very large images directly [15]. Snake images, however, are already very small, making such a compression step useless.

Since there is no VAE, there is no latent space, thus only leaving the pixel space in which the polyomino images live. Also, because this model is specialized in denoising 2D structures, we call it Structured Pixel Space Diffusion (SPS Diffusion). This model is architecturally simpler than Stable Diffusion, having only a single mini U-Net.

Fig. 4 presents the SPS Diffusion model architecture. The model is built from residual blocks, which let information pass through while refining features, and attention blocks, which help to focus on the most important parts of the input. The network has multiple levels which process the image at different resolutions: lower levels handle fine details, higher levels capture larger patterns. The bottleneck is the middle stage where the image is the smallest and most abstract, compressing information before reconstruction. Downsampling is done using a specific way of doing convolution, where the kernel is slid across the image 2 pixels at a time, instead of 1 pixel at a time. Upsampling is done using nearest-neighbor interpolation, which enlarges the image back toward its original size. Skip connections link matching levels in the encoder and decoder, adding the feature channels from the encoder to the decoder to help recover fine details more accurately.

3.3 Training

Sec 3.1 explains how polyomino images are generated using the SPS Diffusion model. This process is called backward diffusion, because noise is removed from the image instead of being added. When training DDPM models, we use *forward diffusion*, where noise is added to a known solution $\mathbf{x}_0$, a maximal snake, in order to generate the **src** data for training. In other words, the noisy polyominoes $\mathbf{x}_t$ (**src**), are

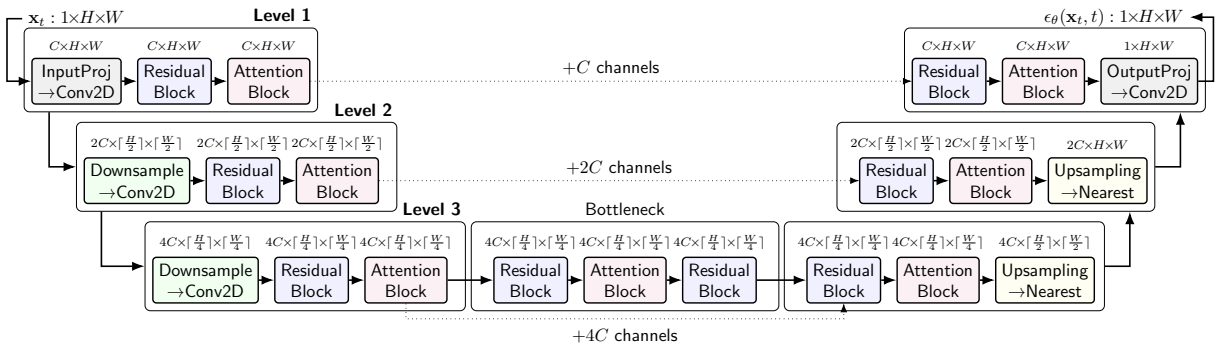


Figure 4: Architecture of the SPS Diffusion model. The model is given $\mathbf{x}_t$, which returns the noise to remove $\varepsilon_\theta(\mathbf{x}_t, t)$ (Sec. 3.3 discusses this further). The labels for the image size indicate the dimensions after the operation of the block below has been applied.

computed in training. Eq. (2) shows how $\mathbf{x}_t$ is computed. The constant $\bar{\alpha}_t$ is related to the mean and variance parameters of the normal distribution governing the forward diffusion process. See [3] for the full definition.

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon \tag{2}$$

Since the model estimates the noise term $\varepsilon_\theta(\mathbf{x}_t, t)$ (see Eq. (1)) that should be removed from $\mathbf{x}_t$ to recover a sample closer to $\mathbf{x}_0$, the training target **tgt** corresponds to the actual noise ε that was added during the forward diffusion step linking $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ (Eq. (2)). Thus, the training pair $(\mathbf{src}, \mathbf{tgt})$ consists of $\mathbf{src} = \mathbf{x}_t$, the noisy polyomino at timestep t, and $\mathbf{tgt} = \varepsilon$, the true noise added at that step.

The SPS Diffusion model presented here was trained by computing the **src** (a noisy polyomino) using known maximal snakes for rectangles of dimension $W \leq 50$ and $H \leq 14$. The noise ε that was added to the maximal snakes during the computation is the **tgt**, as per Eq. (2).

For the loss function, we use the Mean Squared Error (MSE), which was the only tested loss to consistently produce organized snake-like structures (Sec. 2) across all rectangle sizes tested. Losses based on structural penalties, such as degree-three cells or disconnected components, were less effective because early generated images are too noisy for these criteria to be reliable. MSE is defined as follows:

$$\text{MSE}(\mathbf{out}, \mathbf{tgt}) = \frac{1}{N}\sum_{ij}^{N}(\mathbf{tgt}_{ij} - \mathbf{out}_{ij})^2 \tag{3}$$

4 Results for the SPS Diffusion model

We evaluate SPS Diffusion on different rectangle sizes. We consider a SPS Diffusion model trained on over 200 different rectangle sizes, ranging from small 4×4 squares to 50×10 rectangles. We start with rectangles for which the sizes have been seen by the model during training, as shown in Fig. 5. SPS Diffusion has little difficulty generating maximal snake polyominoes in all rectangle sizes seen during training.

Looking at rectangles not seen by the model during training in Fig. 6, SPS Diffusion is able to generalize its understanding of snake polyominoes in small rectangles to much larger sizes at inference time, that is, generate snake-like structures even in large rectangles where the positional encoding is

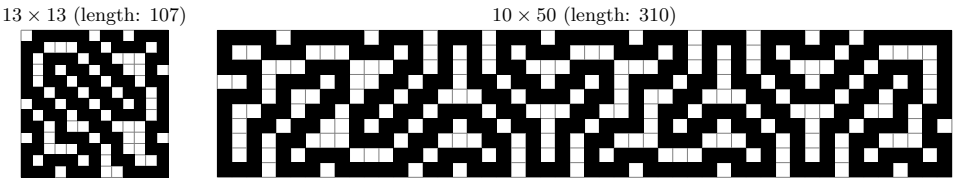


Figure 5: Two maximal snakes generated by the SPS Diffusion model in rectangles seen during training.

technically different. Furthermore, we were able to generate snake polyominoes in squares for sizes up to 28×28, which is bigger than the maximum size used in training for squares (14×14). Similarly, for many rectangles, ranging from near squares to very elongated rectangles of size 10×60, we were also able to generate several snakes. When trying to generate snakes in very large rectangle sizes, we hit the limitations of the model. Fig. 6 shows a forest of polyominoes generated in a large square of size 50×50. Although we observe snake-like structures in the square, the structure itself is not a snake. In some cases, we obtain a forest of snakes, in others, we obtain forests of polyominoes, which are not snakes. Considering the complexity of maximal snakes, the fact that SPS Diffusion is able to get close to a valid solution is encouraging.

Moreover, we were able to generate maximal snakes for rectangle sizes up to 17×17, which is the current computational limit for squares ([4]). After this bound, we cannot know for certain if the generated snakes are maximal. However, we can estimate a lower bound on the minimal length of maximal snakes and, for several rectangle sizes, we can sometimes exceed the lower bound by up to 3 cells. We observe that as the rectangle size increases, it becomes harder for the model to generate a snake, and even harder for a maximal snake. In other words, the error rate increases with the rectangle size. We also observe that the SPS Diffusion model makes fewer errors on elongated rectangles than on squares, since the corresponding snakes are more predictable and therefore easier to generate.

The aformentioned limit of 28×28 for squares containing a snake is a consequence of this fact. For example, for more than 25 000 images generated, no snakes were produced for a 29×29 square, only forests of snakes and forests of polyominoes. We expect that with more calculation time, snakes could be generated for larger square sizes, but then we would encounter the initial limitations of the brute force algorithm, that is, calculation time.

While the SPS Diffusion model generates valid snakes in large rectangles, it has also produced snakes whose lengths exceed the best previously reported lower bounds. These bounds are rigorous rather than conjectural, as snakes of these lengths have been explicitly constructed, often by hand.

Before these results were obtained, the SPS Diffusion model was trained on a smaller dataset of rectangles smaller than 20×10. In this setting, the model generated snakes in squares up to 23×23. After adding squares up to 14×14 (and other similarly sized rectangles) to the training data, the model generated snakes up to 28×28. This improvement suggests that the model benefits substantially from additional data. Consequently, one possible approach is to generate new snakes with the model itself and include them in the training set for retraining. Such an iterative procedure could improve predictions in larger rectangles, although it cannot guarantee that the produced snakes are maximal and mainly serves to guide theoretical investigation of maximal snake lengths.

This procedure can in principle be repeated to generate increasingly large snakes, although it currently requires significant computational time. The SPS Diffusion model also makes several errors and produces forbidden patterns such as cells of degree 3 (branching), cycles, forests of snakes and polyominoes, requiring more computation time in order to generate valid snakes, which, even then, yields only a

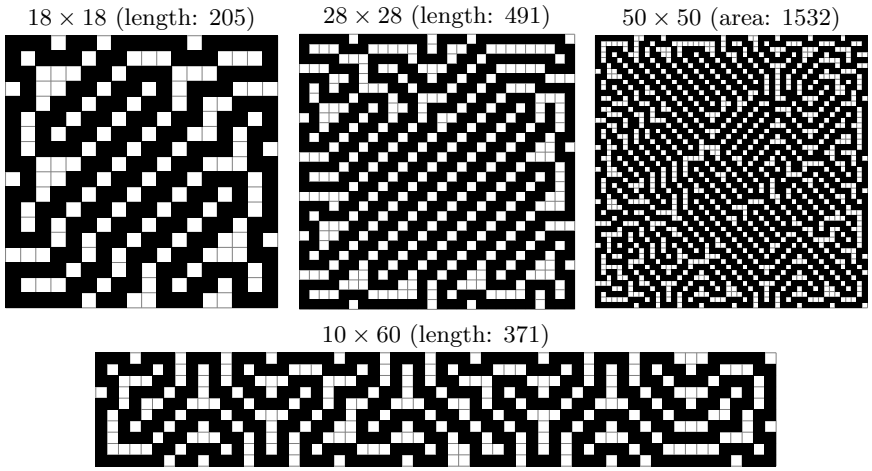


Figure 6: Results generated by the SPS Diffusion model in squares not seen during training.

few snakes, and even less maximal snakes.

Moreover, in larger rectangles there is no guarantee that the positional encoding used for smaller rectangles (RoPE 2D in the attention layers) will remain effective. The current mini-U-Net architecture is also fixed, with only three downsampling layers, which may be insufficient to capture global positional context in larger rectangles. This limitation could be mitigated by training a larger U-Net while excluding smaller rectangles from the training data. Overall, further optimization of the model implementation and evaluation on larger rectangles will be necessary for long-term applicability.

5 Conclusion

In this work, we explored whether a deep neural network can generate maximal snake polyominoes beyond the range of brute force enumeration. We used a diffusion model called Structured Pixel Space Diffusion (SPS Diffusion).

Our experiments showed that SPS Diffusion generates good results, as it can easily reproduce the data it was trained with, as well as generalizing to larger rectangles. More specifically, for squares, snakes were generated up to 28×28, and their length was close to the previously known lower bound. For some configurations, the previously known bound was exceeded by one cell, which demonstrates the potential of the approach when given more training and refinements. Its main weakness is the generation of malformations, including branching, cycles, and forests. Also the likelihood of generating a valid snake decreases as the rectangle size increases. Overall, the results suggest that diffusion models can learn nontrivial combinatorial structures from examples, even when constraints are not explicitly encoded.

Several directions remain open. On the modeling side, reducing invalid generations and improving scalability to larger rectangles are key objectives, for instance by modifying the architecture and the positional encoding of the cells. Alternatively, the loss function could also be tweaked to speed up and improve the training.

On the combinatorial side, a more systematic analysis of recurring substructures in the generated candidates could help guide future conjectures about maximal snake size in large rectangles. Overall, deep neural network models, and SPS Diffusion in particular, provide a new numerical tool to explore maximal snakes beyond the range of exhaustive enumeration.

References

[1] Gill Barequet & Gil Ben-Shachar (2024): Counting Polyominoes, Revisited. In Rezaul Chowdhury & Solon P. Pissis, editors: *2024 Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX 2024*, Proceedings of the Workshop on Algorithm Engineering and Experiments, Society for Industrial and Applied Mathematics, Alexandria, VA, USA, pp. 133–143, doi:10.1137/1.9781611977929.10.

[2] Reinhard Diestel (2025): Graph Theory, 6 edition. *Graduate Texts in Mathematics* 173, Springer, Berlin, Heidelberg, doi:10.1007/978-3-662-70107-2.

[3] Benyamin Ghojogh & Ali Ghodsi (2024): Diffusion Models: Tutorial and Survey, doi:10.31219/osf.io/w7jcm.

[4] Alain Goupil (2020): A331968: Maximum number of unit squares of a snake-like polyomino in an $n \times n$ square box. The On-Line Encyclopedia of Integer Sequences. Available at https://oeis.org/A331968.

[5] Alain Goupil, Marie-Eve Pellerin & Jérôme de Wouters d'Oplinter (2018): Partially Directed Snake Polyominoes. *Discrete Applied Mathematics* 236, pp. 223–234, doi:10.1016/j.dam.2017.09.003.

[6] Anthony J. Guttmann (2009): History and Introduction to Polygon Models and Polyominoes. In Anthony J. Guttmann, editor: *Polygons, Polyominoes and Polycubes*, Springer Netherlands, Dordrecht, pp. 1–21, doi:10.1007/978-1-4020-9927-4_1.

[7] Jonathan Ho, Ajay N. Jain & Pieter Abbeel (2020): Denoising Diffusion Probabilistic Models. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan & H. Lin, editors: *Advances in Neural Information Processing Systems*, 33, Curran Associates, Inc., pp. 6840–6851. Available at https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[8] Jesper Lykke Jacobsen, Christian R. Scullard & Anthony J. Guttmann (2016): On the Growth Constant for Square-Lattice Self-Avoiding Walks. *Journal of Physics A: Mathematical and Theoretical* 49(49):494004, doi:10.1088/1751-8113/49/49/494004.

[9] Iwan Jensen (2001): Enumerations of Lattice Animals and Trees. *Journal of Statistical Physics* 102(3–4), pp. 865–881, doi:10.1023/A:1004855020556.

[10] Iwan Jensen & Anthony J. Guttmann (2000): Statistics of Lattice Animals (Polyominoes) and Polygons. *Journal of Physics A: Mathematical and General* 33(29), pp. L257–L263, doi:10.1088/0305-4470/33/29/102.

[11] Raphaël L'Heureux (2022): Recherche des serpents et des forêts de serpents maximaux dans le plan: approches exhaustives et apprentissage machine. Master's thesis, Université du Québec à Trois-Rivières, Trois-Rivières, QC, Canada. Available at https://depot-e.uqtr.ca/id/eprint/10534.

[12] Alexandre Blondin Massé, Julien de Carufel & Alain Goupil (2018): Saturated Fully Leafed Tree-Like Polyforms and Polycubes. *Journal of Discrete Algorithms* 52–53, pp. 38–54, doi:10.1016/j.jda.2018.11.004. Combinatorial Algorithms – Special Issue Devoted to Life and Work of Mirka Miller.

[13] Alexandre Blondin Massé, Alain Goupil, Raphael L'Heureux & Louis Marin (2025): Maximal 2-Dimensional Binary Words of Bounded Degree. In Guilhem Gamard & Julien Leroy, editors: *Combinatorics on Words*, *Lecture Notes in Computer Science* 15729, Springer Nature Switzerland, Cham, pp. 37–48, doi:10.1007/978-3-031-97548-6_4.

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger & Ilya Sutskever (2021): Learning Transferable Visual Models from Natural Language Supervision. In Marina Meila & Tong Zhang, editors: *Proceedings of the 38th International Conference on Machine Learning*, *Proceedings of Machine Learning Research* 139, PMLR, pp. 8748–8763. Available at https://proceedings.mlr.press/v139/radford21a.html.

[15] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser & Björn Ommer (2022): High-Resolution Image Synthesis with Latent Diffusion Models. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, doi:10.1109/CVPR52688.2022.01042.

[16] Olaf Ronneberger, Philipp Fischer & Thomas Brox (2015): U-Net: Convolutional Networks for Biomedical Image Segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells & Alejandro F. Frangi, editors: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, *Lecture Notes in Computer Science* 9351, Springer International Publishing, Cham, pp. 234–241, doi:10.1007/978-3-319-24574-4_28.

[17] Sebastian Ruder (2017): An Overview of Gradient Descent Optimization Algorithms. Available at https://arxiv.org/abs/1609.04747.

[18] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo & Yunfeng Liu (2024): RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing* 568:127063, doi:10.1016/j.neucom.2023.127063.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser & Illia Polosukhin (2017): Attention Is All You Need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan & Roman Garnett, editors: *Advances in Neural Information Processing Systems*, 30, Curran Associates, Inc., pp. 5998–6008. Available at https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

(Un)ranking Permutation Classes

Nathanaël Hassler Vincent Vajnovszki
Laboratoire d'Informatique de Bourgogne
Université Bourgogne Europe
Dijon, France
nathanael.hassler@ens-rennes.fr vincent.vajnovszki@u-bourgogne.fr

Permutations avoiding a pattern of length three are enumerated by the Catalan numbers. In this work, we present methods for ranking and unranking such permutations in lexicographic or colexicographic order.

1 Introduction

For an ordered set of combinatorial objects we can ask which position in the ordered list of objects a given object has, or conversely determine the objet in a given position. Formally for a total ordered set C we have:

Definition 1.1. The *ranking* over C is the function

$$\mathrm{rank}_C : C \to \{0, 1, \ldots, |C| - 1\}$$

that maps x to the number of elements of C that are less than x. The *unranking* is the inverse function

$$\mathrm{unrank}_C : \{0, 1, \ldots, |C| - 1\} \to C$$

that maps p to the object x such that $\mathrm{rank}_C(x) = p$.

Here, we restrict ourselves to sets C of restricted permutations of the same length in one-line representation, ordered lexicographically or colexicographically. Whenever the set C is clear from the context, the subscripts in the functions rank and unrank will be omitted.

Lehmer [8] may have been the first to suggest studying combinatorial objects by imposing an order on them and then developing corresponding ranking and unranking algorithms. Ruskey in his book [10] considers such algorithms for unrestricted permutations in both (co)lexicographic order and Steinhaus–Johnson–Trotter Gray code order. Among recent contributions in the field, we can cite [2, 3], and more particularly [5] for ranking and unranking algorithms for restricted permutations (derangements and ménage permutations).

Here, we are interested in ranking and unranking algorithms for simple yet non-trivial combinatorial classes, namely permutations avoiding a pattern of length three.

Aside from their intrinsic theoretical interest, such methods also have practical value. Indeed, given an unranking method, the ordered list can be computed as

$$\mathrm{unrank}(0), \mathrm{unrank}(1), \mathrm{unrank}(2), \ldots$$

and in the context of random sampling, $\mathrm{unrank}(p)$ yields a random object as long as p is a random integer less than the cardinality of the combinatorial class. In addition, having a ranking method, the successor of a given object x can be computed as $\mathrm{unrank}(\mathrm{rank}(x) + 1)$. Even though these techniques are not efficient

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 114–120, doi:10.4204/EPTCS.445.14

© N. Hassler & V. Vajnovszki
This work is licensed under the
Creative Commons Attribution License.

in general, they can provide hints for solutions to the underlying problems and enhance understanding of the concerned objects.

A permutation of the set $[n] = \{1, 2, \ldots, n\}$ is a bijection from $[n]$ onto itself, and $\mathfrak{S}_n$ denotes the set of such permutations. We will represent permutations by their one-line notation, that is, the permutation $\pi \in \mathfrak{S}_n$ by the word $\pi(1)\pi(2)\cdots\pi(n) \in [n]^n$.

Let k and n be integers, $0 < k \leq n$, and $\sigma = \sigma(1)\sigma(2)\cdots\sigma(k) \in \mathfrak{S}_k$ and $\pi = \pi(1)\pi(2)\cdots\pi(n) \in \mathfrak{S}_n$ be two permutations. One says that π contains σ if π contains a (not necessarily contiguous) subsequence $\pi(i_1)\pi(i_2)\cdots\pi(i_k)$, $i_1 < i_2 < \cdots < i_k$, order isomorphic to σ; otherwise one says that π avoids σ, or π is σ-avoiding. In this context σ is called a (classical) *pattern*, and the set of length-n permutations avoiding the pattern σ is denoted by $\mathrm{Av}_n(\sigma)$, and $\mathrm{Av}(\sigma) = \cup_{n\geq 0}\mathrm{Av}_n(\sigma)$.

The lexicographic order on $[n]^n$ induces the lexicographic order on permutations in $\mathfrak{S}_n$ in their one-line representation, and it generalizes to words over any ordered alphabet. For two same length words $v = v_1 v_2 \ldots v_n$ and $w = w_1 w_2 \ldots w_n$ one says that v is less than w in *colexicographic order* if $v_n v_{n-1} \ldots v_1$ is less than $w_n w_{n-1} \ldots w_1$ in lexicographic order.

In the following we need the next easy to check result.

Remark 1.2. If C is a set of length-n words over an ordered alphabet, then the rank of $w = w_1 w_2 \ldots w_n \in C$, in lexicographic order, is

$$\mathrm{rank}(w) = \sum_{i=1}^{n} |E_i|, \tag{1}$$

where for each i, E_i is the set of words $u_1 u_2 \ldots u_n \in C$ with $u_j = w_j$ for $j < i$ and $u_i < w_i$.

The number of permutations in $\mathfrak{S}_n$ avoiding a pattern of length three is given by the Catalan number $c_n = \frac{1}{n+1}\binom{2n}{n}$, see [7], which is sequence A000108 in [4]. Often it is convenient to represent pattern avoiding permutations $\pi \in \mathfrak{S}_n$ by an $n \times n$ array with a dot in each one of the squares $(i, \pi(i))$. See Figures 1 (b) and (c) for two examples.

2 Avoiding the 231 pattern

The *reduction* of a word α over a finite alphabet $A \subset \mathbb{N}$ is obtained by replacing each occurrence of the smallest symbol by 1, each occurrence of the second smallest symbol by 2, and so on. The *direct sum* of two permutations $\sigma \in \mathfrak{S}_k$ and $\tau \in \mathfrak{S}_\ell$, denoted by $\sigma \oplus \tau$, is the permutation $\pi \in \mathfrak{S}_{k+\ell}$ where $\pi(1)\pi(2)\cdots\pi(k) = \sigma$ and $\pi(k+1)\pi(k+2)\cdots\pi(k+\ell)$ is a permutation of $\{k+1, k+2, \ldots, k+\ell\}$ that reduces to τ. Similarly, the *skew sum* of σ and τ, denoted by $\sigma \ominus \tau$, is the permutation $\pi \in \mathfrak{S}_{k+\ell}$ where $\pi(1)\pi(2)\cdots\pi(k)$ is a permutation of $\{\ell+1, \ell+2, \ldots, \ell+k\}$ that reduces to σ and $\pi(k+1)\pi(k+2)\cdots\pi(k+\ell) = \tau$. With this notation, if $\pi = 1 \ominus \sigma$, we have $\pi(1) = k+1$.

The next easy to understand fact is folklore.

Fact 2.1. A permutation avoids 231 if and only if it has the form $(1 \ominus \sigma) \oplus \tau$, where σ and τ are (possibly empty) 231-avoiding permutations.

See Figure 1 (a) for the general shape of a 231-avoiding permutation, and (b) for the 231-avoiding permutation $2173546 = (1 \ominus 1) \oplus (1 \ominus (1 \oplus ((1 \ominus 1) \oplus 1)))$.

Proposition 2.2. *Let $\pi = \pi(1)\pi(2)\cdots\pi(n) = (1 \ominus \sigma) \oplus \tau$ be a length-n 231-avoiding permutation, with $\sigma \in \mathrm{Av}_k(231)$ and $\tau \in \mathrm{Av}_\ell(231)$ (and thus, $n = k+\ell+1$). Then the rank of π, in lexicographical order, is given recursively by*

$$\mathrm{rank}_{\mathrm{Av}_n(231)}(\pi) = \sum_{i=1}^{\pi(1)-1} c_i \cdot c_{n-i-1} + \mathrm{rank}_{\mathrm{Av}_k(231)}(\sigma) \cdot c_\ell + \mathrm{rank}_{\mathrm{Av}_\ell(231)}(\tau).$$

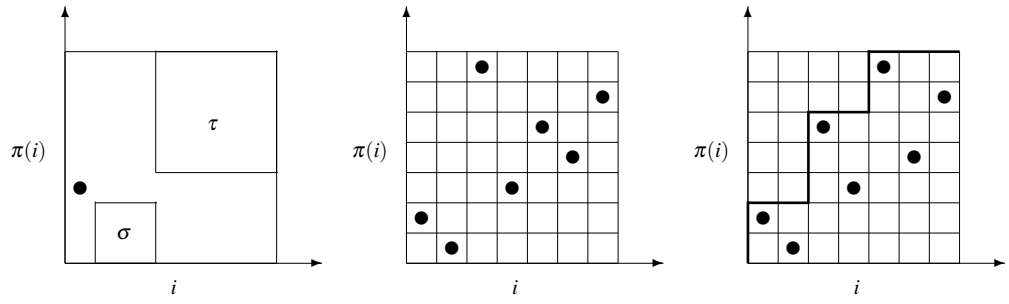


Figure 1: The arrays representation of: (a) the permutation $\pi = (1 \ominus \sigma) \oplus \tau$; (b) the 231-avoiding permutation 2173546; (c) the 321-avoiding permutation 2153746 corresponding to the Dyck word *bbaabbbaabbaaa* which codes the over-diagonal path in bold.

Proposition 2.3. *Let n, p be integers, $0 \leq p < c_n$. The permutation* $\mathrm{unrank}_{\mathrm{Av}_n(231)}(p)$ *is obtained recursively as follows. Determine the smallest k such that*

$$p < \sum_{i=0}^{k} c_i c_{n-i-1}$$

and denote

$s := \sum_{i=0}^{k-1} c_i c_{n-i-1}$,

$u := \lfloor \frac{p-s}{c_{n-k}} \rfloor$, *and*

$v := p - s - u \cdot c_{n-k}$.

Then $\mathrm{unrank}_{\mathrm{Av}_n(231)}(p) = (1 \ominus \sigma) \oplus \tau$ *where*

$\sigma = \mathrm{unrank}_{\mathrm{Av}_{k-1}(231)}(u)$, *and*

$\tau = \mathrm{unrank}_{\mathrm{Av}_{n-k}(231)}(v)$.

In fact, the value k defined in the previous proposition is the first entry of $\mathrm{unrank}_{\mathrm{Av}_n(231)}(p)$.

3 Avoiding the 321 pattern

A *left-to-right maximum position* (or *lrmp* for short) in a permutation π is an index i such that, for all $j < i$, we have $\pi(j) < \pi(i)$. The value $\pi(i)$ is then called a *left-to-right maximum value* (or *lrmv* for short). An *excedance* is an index i such that $\pi(i) > i$. It is known (see, e.g., [9]) that a permutation is 321-avoiding if and only if both the subsequence of its left-to-right maximum values and the subsequence of its remaining elements are increasing. Moreover, such a permutation is uniquely determined by the set $\{(i, \pi(i))\}$, where i ranges over its left-to-right maximum positions. Even though we will not use it later on, it is worth mentioning that, for a 321-avoiding permutation, the left-to-right maxima positions coincide with the excedances of the permutation.

Another combinatorial class closely related to length-3 pattern-avoiding permutations is that of *Dyck words*. Let D_n denote the set of Dyck words of semilength n, that is, words $w \in \{a, b\}^{2n}$ containing the

same number of occurrences of a and b, and such that every prefix of w contains at least as many b's as a's. (We adopt this unusual notation for convenience; traditionally, the roles of a and b are interchanged.) Dyck words in D_n are counted by the Catalan number c_n.

The bijection $\psi : \mathrm{Av}_n(321) \to D_n$ appeared originally in [7, p. 89] in a slightly different form. Consider the array representation corresponding to $\pi \in \mathrm{Av}_n(321)$ and the path with *east* and *north* steps along the edges of the array that goes from the lower-left corner to the upper-right corner of the array, leaving all the dots to the right and remaining always as close to the main diagonal as possible. Let P be the resulting path. Then $\psi(\pi)$ can be obtained from P just by reading its steps, a north step corresponding to the letter b and an east step to the letter a. See Figure 1 (c) for an example.

The next proposition says that ψ preserves the lexicographic order.

Proposition 3.1. *If $\pi, \tau \in \mathrm{Av}_n(321)$ with $\pi < \tau$ in lexicographic order, then $\psi(\pi) < \psi(\tau)$ in lexicographic order as well.*

Ballot sequences are prefixes of Dyck words. We denote by $B(i,j)$ the set of ballot sequences with i occurrences of b and j occurrences of a. Similarly, we denote by $\bar{B}(i,j)$ the set of *suffixes of Dyck words*.

Clearly, $B(i,j)$ is empty if $i < j$, and there is a unique word in $B(i,0)$, namely b^i. Moreover, if we ask to obtain a ballot sequence $w \in B(i,j)$ by appending a single letter to a shorter one, then

- if $i = j$, this can be obtained in a unique way: $w = ua$ for some $u \in B(i, j-1)$, and
- if $0 < j < i$, this can be obtained in two different ways, namely $w = ua$, for some $u \in B(i, j-1)$, and $w = ub$, for some $u \in B(i-1, j)$.

The considerations above (see for instance [6] and the references therein) lead us to the following recursive definition of $t(i,j)$, the cardinality of $B(i,j)$:

$$t(i,j) = \begin{cases} 0 & \text{if} \quad j > i, \\ 1 & \text{if} \quad j = 0, \\ t(i,j-1) & \text{if} \quad j = i > 0, \\ t(i-1,j) + t(i,j-1) & \text{if} \quad 0 < j < i. \end{cases} \tag{2}$$

The sequence $\big(t(i,j)\big)_{0 \le j \le i}$ is the sequence A009766 in [4], which has the closed form:

$$t(i,j) = \binom{i+j}{i} \frac{i-j+1}{i+1}, \text{ for } 0 \le j \le i, \tag{3}$$

and its first values are shown in Table 1.

Proposition 3.2 ([1])**.** *For the double indexed sequence $s(i,j) = \sum_{k=0}^{j} t(i,k)$ we have*

$$s(i,j) = t(i+1,j).$$

Representing a 321-avoiding permutation by both the sequence of its left-to-right maxima and the corresponding Dyck word (see Figure 2), we have:

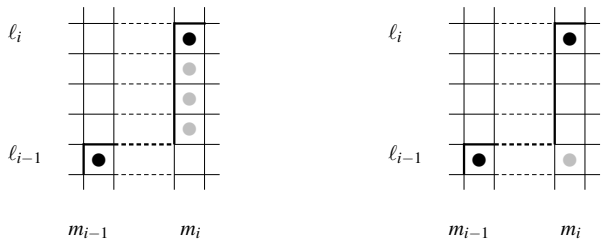
Proposition 3.3. *Let π be a length-n 321-avoiding permutation, and let $(m_i, \ell_i)_{i=1}^{k}$ be the sequence where the m_i's are the* lrmp*'s of π, and $\ell_i = \pi(m_i)$ are the* lrmv*'s. Then,*

$$\mathrm{rank}_{\mathrm{Av}_n(321)}(\pi) = \sum_{i=1}^{k} \left(s(n - m_i + 1, n - \max\{\ell_{i-1}, m_i\}) - s(n - m_i + 1, n - \ell_i + 1) \right),$$

where for convenience $\ell_0 = 0$.

$i \setminus j$	0	1	2	3	4	5	6	7
0	1							
1	1	1						
2	1	2	2					
3	1	3	5	5				
4	1	4	9	14	14			
5	1	5	14	28	42	42		
6	1	6	20	48	90	132	132	
7	1	7	27	75	165	297	429	429

Table 1: The first values of the double indexed sequence $\big(t(i,j)\big)_{0\le j\le i}$.



(a) (b)

Figure 2: The black dots correspond to two consecutive *lrmp*'s, m_{i-1} and m_i, of a 321-avoiding permutation π, together with their *lrmv*'s $\ell_{i-1} = \pi(m_{i-1})$ and $\ell_i = \pi(m_i)$. Let σ be another 321-avoiding permutation of the same length that is smaller than π in lexicographic order and has the same first $(i-1)$ *lrmp*'s $m_1, m_2, \ldots, m_{i-1}$, and the same first $(i-1)$ *lrmv*'s, that is $\sigma(m_j) = \pi(m_j)$ for $j < i$. (a) The gray dots indicate the possible values for the ith *lrmv* of σ if m_i is still an *lrmp* of σ, with $\sigma(m_i) < \ell_i$. (b) The gray dot indicates the value of $\sigma(m_i)$ if m_i is no longer an *lrmp* of σ (and thus $\ell_{i-1} \ge m_i$).

Algorithm 1 Procedure for computing the Dyck word of semilength n and rank p. Here $\bar{t}(i,j) = \mathrm{card}(\overline{B}(i,j))$, so $\bar{t}(i,j) = t(j,i)$.

```
procedure UNRANKDYCK(n, p: integer)
    global d: array
    i ← n, j ← n
    for k from 1 to 2n do
        if i = j or p ≥ t̄(i − 1, j) then
            d_k ← b; p ← p − t̄(i − 1, j); i ← i − 1
        else d_k ← a; j ← j − 1
        end if
end procedure
```

Let π be a 321-avoiding permutation of length n, and let $w = w_1 w_2 \dots w_{2n}$ be its corresponding Dyck word, that is $w = \Psi(\pi) \in D_n$. Obviously, the number of left-to-right maxima in π is equal to the number of occurrences of the factor ba (i.e., number of peaks) in w. Moreover, if $w_{j-1}w_j$ is the ith occurrence of the factor ba, then the ith left-to-right maximum (m_i, ℓ_i) of π is given by $m_i = |w_1 w_2 \dots w_j|_a$ and $\ell_i = |w_1 w_2 \dots w_j|_b$.

Proposition 3.4. *Let n, p be integers with $0 \le p < c_n$. The permutation $\mathrm{unrank}_{\mathrm{Av}_n(321)}(p)$ is obtained as follows:*

- *Run the* UNRANKDYCK *procedure in Algorithm 1 to construct the Dyck word w of semilength n and rank p.*
- *Using the considerations above, construct the set $\{(m_i, \pi(m_i))\}_{i=1}^{k}$ of left-to-right maxima (values, positions) of the permutation $\Psi^{-1}(w)$.*
- *Finally, construct the permutation $\mathrm{unrank}_{\mathrm{Av}_n(321)}(p)$.*

4 Trivial symmetries

Here, based on the previous results and the trivial symmetries between permutation classes, we derive ranking and unranking algorithms for the remaining length-3 avoiding patterns: 123, 132, 213, and 312.

Let $\mathfrak{c}$ be the complement operation on $\mathfrak{S}_n$, that is the involution defined as: $\sigma = \mathfrak{c}(\pi)$ if $\sigma(i) = n+1-\pi(i)$ for $1 \le i \le n$. Clearly, the complement operation $\mathfrak{c}$ induces a bijection between $\mathrm{Av}_n(231)$ and $\mathrm{Av}_n(213)$, and between $\mathrm{Av}_n(321)$ and $\mathrm{Av}_n(123)$. Moreover, if $\pi < \rho$ in lexicographic order, then $\mathfrak{c}(\rho) < \mathfrak{c}(\pi)$ in lexicographic order. With the considerations above we have:

Proposition 4.1. *Let τ be one of the length-3 patterns* 213 *or* 123 *(and so, $\mathfrak{c}(\tau)$ is* 231 *or* 321*). Then, for $n \ge 0$ and*

- *for $\pi \in \mathrm{Av}_n(\tau)$, $\mathrm{rank}_{\mathrm{Av}_n(\tau)}(\pi) = c_n - \mathrm{rank}_{\mathrm{Av}_n(\mathfrak{c}(\tau))}(\mathfrak{c}(\pi)) - 1$, and*
- *for an integer p, $0 \le p < c_n$, $\mathrm{unrank}_{\mathrm{Av}_n(\tau)}(p) = \mathrm{unrank}_{\mathrm{Av}_n(\mathfrak{c}(\tau))}(c_n - p - 1)$.*

Let now $\mathfrak{r}$ be the reverse operation on $\mathfrak{S}_n$, that is the involution on $\mathfrak{S}_n$ defined as: $\sigma = \mathfrak{r}(\pi)$ if $\sigma(i) = \pi(n+1-i)$ for $1 \le i \le n$. The reverse operation $\mathfrak{r}$ induces a bijection between $\mathrm{Av}_n(231)$ and $\mathrm{Av}_n(132)$, and between $\mathrm{Av}_n(213)$ and $\mathrm{Av}_n(312)$. Moreover, if $\pi < \rho$ in lexicographic order, then $\mathfrak{r}(\pi) < \mathfrak{r}(\rho)$ in colexicographic order. Denoting rank^* (resp. unrank^*) the ranking (resp. unranking) function in colexicographic order we have:

Proposition 4.2. *Let τ be one of the length-3 patterns* 132 *or* 312 *(and so, $\mathfrak{r}(\tau)$ is* 231 *or* 213*). Then, for $n \ge 0$ and*

- *for $\pi \in \mathrm{Av}_n(\tau)$, $\mathrm{rank}^*_{\mathrm{Av}(\tau)}(\pi) = \mathrm{rank}_{\mathrm{Av}(\mathfrak{r}(\tau))}(\mathfrak{r}(\pi))$,*
- *for an integer p, $0 \le p < c_n$, $\mathrm{unrank}^*_{\mathrm{Av}(\tau)}(p) = \mathfrak{r}(\mathrm{unrank}_{\mathrm{Av}(\mathfrak{r}(\tau))}(p))$.*

Open problems and future directions:

Even when the counting sequences for permutations avoiding multiple patterns of length three are trivial (e.g., 2^{n-1}, Fibonacci numbers), ranking and unranking methods in lexicographic order still seem less straightforward. It would also be interesting to consider the avoidance of patterns of size greater than three. Finally, the efficient implementation of these methods remains an open problem.

References

[1] D.F. Bailey (1996): *Counting arrangements of 1's and -1's*. Mathematics Magazine 69(2), pp. 128–131.

[2] A. Curiel & A. Genitrini (2024): *Lexicographic unranking algorithms for the twelvefold way*. Processing of AofA 2024, Leibniz International Proceedings in Informatics 302, pp. 17:1–17:14, doi:10.4230/LIPIcs.AofA.2024.17.

[3] A. Genitrini & M. Pépin (2021): *Lexicographic unranking of combinations revisited*. Algorithms 4(97), doi:10.3390/a14030097.

[4] The On-Line Encyclopedia of Integer Sequences: Available at `http://oeis.org`.

[5] P. Kagey (2024): *Ranking and unranking restricted permutations*. Discrete Applied Mathematics 355, pp. 247–261, doi:10.1016/j.dam.2024.05.010.

[6] Z. Kása (2009): *Generating and ranking of Dyck words*. Acta Univ. Sapientiae, Informatica 1(1), pp. 109–118. Available at `https://arxiv.org/pdf/1002.2625`.

[7] D. Knuth (1973): *The Art of Computer Programming, III*. Addison-Wesley, Reading, MA.

[8] D.H. Lehmer (1964): *The machine tools of combinatorics, Applied Combinatorial Mathematics*. Wiley.

[9] A. Reifegerste (2003): *On the diagram of 132-avoiding permutations*. European J. Combin. 24, pp. 759–776, doi:10.1016/S0195-6698(03)00065-9.

[10] F. Ruskey (2003): *Combinatorial Generations*. Available at `https://page.math.tu-berlin.de/~felsner/SemWS17-18/Ruskey-Comb-Gen.pdf`.

Frequency of Patterns in Smooth Sequences over the Alphabet $\{1,3\}$

Damien Jamet
Université de Lorraine, CNRS, LORIA, UMR 7503, F-54000 Nancy, France
damien.jamet@loria.fr

Irène Marcovici
Univ Rouen Normandie, CNRS, Normandie Univ, LMRS UMR 6085, F-76000 Rouen, France
irene.marcovici@univ-rouen.fr

Léo Poirier
Aix-Marseille Université, CNRS, I2M UMR 7373, F-13000 Marseille, France
leo.poirier@univ-amu.fr

Thierry de la Rue
Univ Rouen Normandie, CNRS, Normandie Univ, LMRS UMR 6085, F-76000 Rouen, France
thierry.de-la-rue@univ-rouen.fr

We provide an ergodic theory framework to study statistical properties of smooth sequences over the odd alphabet $\{1,3\}$. The arithmetic nature of this alphabet yields a partition of the subshift of smooth sequences based on their local structure, defining a notion of type for those sequences. We describe the substitutive structure of the smaller subshifts obtained by fixing the sequence of types of the successive derivatives of smooth sequences, from which we obtain the unique ergodicity of all these subshifts. A direct consequence is that the asymptotic frequency of any finite pattern in a smooth sequence over $\{1,3\}$ is always well-defined and depends on its type sequence. Finally, we characterize the minimality of these subshifts.

1 Introduction

The study of combinatorial structures arising from run-length encoding has been a subject of enduring interest in discrete mathematics. For many years, the canonical reference for this object has been the problem posed by Kolakoski in 1965 [9], asking for a rule describing the self-reading sequence $\mathbb{K} = 1221121221\ldots$. However, this sequence was introduced and extensively investigated much earlier by Oldenburger in 1939 [10]. The **Oldenburger–Kolakoski sequence** $\mathbb{K}$, defined over $\{1,2\}$, is the unique fixed point of the run-length encoding operator Δ (which we call the *derivative* operator) that starts with 1.

While its definition is elementary, the dynamical properties of $\mathbb{K}$ remain largely unknown. Dekking [6] highlighted two fundamental open problems: *mirror invariance* (a word is a factor iff its reversal is) and *recurrence*. He established that mirror invariance implies recurrence and is equivalent to stating that every smooth word appears as a factor of $\mathbb{K}$. Regarding complexity, he conjectured a polynomial growth $p_{\mathbb{K}}(n) \asymp n^{\rho}$ with $\rho \approx 2.71$ [7]. Recent progress by Cassaigne and Henry [5] provides new bounds for this complexity and formalizes the class of *f-smooth* words. Other conjectures, such as the frequency of the symbol 1 being $1/2$ [8], also remain open, with current rigorous bounds at $[0.49992, 0.50008]$ [11].

To gain insight into such self-descriptive structures, the notion of *smooth sequences* (or C^{∞}-sequences) was introduced as a natural generalization [4, 2]. A sequence x over a finite alphabet $\Sigma \subset \mathbb{N}$ is said to be **differentiable** if its derivative Δx (the sequence of lengths of its consecutive mono-symbol blocks) remains a sequence over Σ. It is **smooth** if it is infinitely differentiable, i.e., $\Delta^k x \in \Sigma^{\mathbb{N}}$ for all $k \geq 0$.

Research on smooth sequences is traditionally categorized by the parity of the alphabet. On a two-letter alphabet, when symbols share an *even* parity, the structure is perfectly balanced, and symbol fre-

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 121–129, doi:10.4204/EPTCS.445.15

quencies are trivially $1/2$. Odd alphabets such as $\{1,3\}$ exhibit intermediate behavior: for instance, Baake and Sing [1] showed that the Kolakoski-(3,1) sequence relates to model sets with pure point diffraction spectra, providing a structural bridge between these sequences and quasicrystals.

In the present paper, we also focus on the dynamical and statistical properties of smooth sequences over $\Sigma = \{1,3\}$. While specific extremal cases have been studied [3], the general distribution of frequencies in this setting remains poorly understood.

Main Results – To address these questions, we extend the study of smooth sequences to the bi-infinite setting. This transition is essential for ergodic approach, as it allows us to work within a compact, shift-invariant space: the subshift $X \subset \{1,3\}^{\mathbb{Z}}$ of bi-infinite smooth sequences. Our analysis relies on two main tools:

1. A **recoding** of smooth sequences over a four-letter alphabet $\mathscr{A} = \{\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}\}$ based on the four possible mono-symbol blocks.
2. A **classification** of these sequences into two distinct *types* (0 and 1) based on their local structure, which is preserved under derivation.

This framework reveals that X decomposes into a disjoint union $X = \bigsqcup_{\tau\in\{0,1\}^{\mathbb{N}}} X_\tau$, where each X_τ is defined by a fixed sequence of types τ. Our first main result is that each subshift X_τ (and its recoded counterpart Y_τ) is uniquely ergodic (Corollary 3.2). As a direct consequence of this unique ergodicity, the asymptotic frequency of any finite pattern w exists and is well-defined for both bi-infinite and one-sided smooth sequences (Theorem 3.1). Furthermore, we provide a complete characterization of the minimality of these subshifts and we establish that X_τ (resp. Y_τ) is minimal if and only if τ contains infinitely many zeros (Theorem 4.1).

2 Smooth sequences over $\{1,3\}$ and their recodings

Bi-infinite smooth sequences over $\{1,3\}$ – The operator Δ maps any non-stationary bi-infinite sequence x to the sequence of lengths of its consecutive mono-symbol blocks. To properly index this derived sequence, we adopt the following convention: the value $(\Delta x)_0$ at the origin of the derivative is the length of the mono-symbol block in x that contains the term x_0.
Having defined the action of the derivative operator Δ on bi-infinite sequences, we now formalize the concepts of differentiability and smoothness in this setting:

Definition 2.1. *A bi-infinite sequence $x \in \{1,3\}^{\mathbb{Z}}$ is said to be* differentiable *if its derivative Δx is well-defined (i.e., x is non-stationary) and belongs to $\{1,3\}^{\mathbb{Z}}$. Furthermore, x is said to be* smooth *if it is infinitely differentiable.*

Let us denote by $X \subset \{1,3\}^{\mathbb{Z}}$ the set of all bi-infinite smooth sequences. It is obviously invariant under the shift map S. Furthermore, X is a closed subset of the product space $\{1,3\}^{\mathbb{Z}}$. As a closed and shift-invariant subset of a compact space, X constitutes a compact subshift, providing the exact topological framework required for our ergodic study.

Recodings of smooth sequences – Since the derivative of a smooth sequence is also a sequence over the same alphabet, any $x \in X$ is necessarily a concatenation of blocks of lengths 1 and 3. Consequently, a smooth sequence is composed of only four types of mono-symbol blocks, which we recode using the alphabet $\mathscr{A} = \{\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}\}$ defined as follows: $\mathsf{A} = 1$, $\mathsf{B} = 3$, $\mathsf{C} = 111$, $\mathsf{D} = 333$. The *recoding* of

a smooth sequence $x \in X$ is the sequence $y = \mathtt{rec}(x) \in \mathscr{A}^{\mathbb{Z}}$ where each term y_i encodes a consecutive mono-symbol block (or run) in x. By convention, y_0 is the symbol encoding the block that contains x_0. We denote by $Y = \mathrm{rec}(X) \subset \mathscr{A}^{\mathbb{Z}}$ the subshift of all such recodings. An example of this correspondence is shown below, where the underlined symbol indicates the zero position of the sequence:

$$\begin{array}{lll} x & = \cdots 333\ 1\underline{1}1\ 333\ 1\ 3\ 1\ 333\ 111\ 333\ 1\cdots \\ y = \mathrm{rec}(x) & = \quad \cdots \mathtt{D}\ \ \underline{\mathtt{C}}\ \ \mathtt{D}\ \ \mathtt{A}\ \mathtt{B}\ \mathtt{A}\ \ \mathtt{D}\ \ \ \mathtt{C}\ \ \ \mathtt{D}\ \ \mathtt{A}\cdots \end{array}$$

Observe that each symbol in the recoding sequence $y = \mathrm{rec}(x)$ corresponds to a single symbol in the derivative $\Delta(x)$: a 1 in $\Delta(x)$ gives rise either to an $\mathtt{A}$ or to a $\mathtt{B}$ in y, whereas a 3 in $\Delta(x)$ gives rise either to a $\mathtt{C}$ or to a $\mathtt{D}$ in y. Now, the structuring of $\Delta(x)$ into mono-symbol blocks yields a central property of $y = \mathrm{rec}(x)$: its decomposition into so-called *elementary blocks*, described in the following array.

mono-symbol block in $\Delta(x)$	corresponding factor in x	elementary block in y
1	1 or 3	$\mathtt{A}$ or $\mathtt{B}$
3	111 or 333	$\mathtt{C}$ or $\mathtt{D}$
111	131 or 313	$\mathtt{ABA}$ or $\mathtt{BAB}$
333	111333111 or 333111333	$\mathtt{CDC}$ or $\mathtt{DCD}$

Note that these elementary blocks are either factors of y of length 3, of the form $\mathtt{ABA}$, $\mathtt{BAB}$, $\mathtt{CDC}$, $\mathtt{DCD}$, or single letters $\mathtt{A}$, $\mathtt{B}$, $\mathtt{C}$, $\mathtt{D}$ that are not part of larger elementary blocks. Observe also that, since mono-symbol blocks in $\Delta(x)$ have length at most 3, elementary blocks in y cannot overlap, and therefore there is a unique way of decomposing y as a bi-infinite concatenation of elementary blocks.
We can then naturally define a notion of derivation for recodings as $\Delta(rec(x)) = rec(\Delta x)$. In practice, using the elementary block decomposition of any $y \in Y$, this results in the following local rules:

$$\Delta : \left| \begin{array}{lcl} \mathtt{ABA}, \mathtt{BAB} & \longmapsto & \mathtt{C} \\ \mathtt{CDC}, \mathtt{DCD} & \longmapsto & \mathtt{D} \\ \mathtt{A}, \mathtt{B} & \longmapsto & \mathtt{A} \\ \mathtt{C}, \mathtt{D} & \longmapsto & \mathtt{B} \end{array} \right.$$

Two types of smooth sequences and recodings – The above local rules give two possible preimages for mono-symbol blocks in $\Delta(x)$. As it turns out, the arithmetic nature of the alphabet $\Sigma = \{1,3\}$ imposes a strict parity constraint on the positions of the blocks, which implies that the choice of the preimage of one mono-symbol block determines all the others in the process of taking the primitive. This yields the following proposition.

Proposition 2.2. *A recoding sequence $y \in Y$ cannot simultaneously contain an elementary block in $\{\mathtt{ABA}, \mathtt{DCD}\}$ and an elementary block in $\{\mathtt{BAB}, \mathtt{CDC}\}$.*

This internal consistency allows us to classify smooth sequences into two distinct categories based on their local structure.

Definition 2.3. *Let $x \in X$ be a smooth sequence and $y = \mathrm{rec}(x)$ its recoding. We say that x and y are of* type 0 *if y contains an elementary block in $\{\mathtt{ABA}, \mathtt{DCD}\}$. Conversely, we say that x and y are of* type 1 *if y contains an elementary block in $\{\mathtt{BAB}, \mathtt{CDC}\}$.*

We denote the type of x and y by $\mathrm{type}(x) = \mathrm{type}(y) \in \{0,1\}$. The sequence of types of all iterated derivatives is denoted by:

$$\mathrm{Types}(x) = \mathrm{Types}(y) := \left(\mathrm{type}\left(\Delta^n x\right)\right)_{n \in \mathbb{N}} \in \{0,1\}^{\mathbb{N}}.$$

This classification gives us a way to partition X and Y by specifying the sequence of types of their elements.

Definition 2.4. *For a sequence of types $\tau \in \{0,1\}^{\mathbb{N}}$, we define X_τ as the set of smooth bi-infinite sequences whose type sequence is exactly τ:*

$$X_\tau := \{x \in X \mid \mathrm{Types}(x) = \tau\}.$$

The set of corresponding recodings is denoted by $Y_\tau := \mathrm{rec}(X_\tau) = \{y \in Y \mid \mathrm{Types}(y) = \tau\}$.

X_τ and Y_τ are clearly closed and shift-invariant. We also prove that they are always nonempty: therefore, they are subshifts over respectively the alphabets $\{1,3\}$ and $\{\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}\}$.

Substitutive structure – We now define two substitutions that will, given the type of a recoding $y \in Y_\tau$ and its derivative Δy, help us reconstruct back y from local rules:

$$\varphi_0 : \begin{cases} \mathsf{A} \mapsto \mathsf{A} \\ \mathsf{B} \mapsto \mathsf{D} \\ \mathsf{C} \mapsto \mathsf{ABA} \\ \mathsf{D} \mapsto \mathsf{DCD} \end{cases} \qquad \varphi_1 : \begin{cases} \mathsf{A} \mapsto \mathsf{B} \\ \mathsf{B} \mapsto \mathsf{C} \\ \mathsf{C} \mapsto \mathsf{BAB} \\ \mathsf{D} \mapsto \mathsf{CDC} \end{cases}.$$

The action of φ_0 and φ_1 are extended by morphism to bi-infinite sequences: for $t = 0$ or 1, applying φ_t on every letter of $y \in Y$ yields the concatenated sequence of blocks

$$\ldots \varphi_t(y_{-1})\varphi_t(y_0)\varphi_t(y_1)\varphi_t(y_2)\ldots$$

To properly index the resulting sequence, we adopt the convention that the block $\varphi_t(y_0)$ begins exactly at index 0. The positions of all other blocks are then uniquely determined by expanding outwards to the left and right. These two substitutions act as right inverses of the derivative operator Δ: for any $y \in Y$ and any $t \in \{0,1\}$, we have $\Delta(\varphi_t(y)) = y$. Conversely, applying the substitution $\varphi_{\mathrm{type}(y)}$ to the derivative Δy recovers the original sequence y only up to a bounded shift. In fact, for any $L \geq 1$, given the L first terms of the sequence of types of a recoding y, we can recover it from its L-derivative $\Delta^L y$.

Proposition 2.5. *For all $\tau \in \{0,1\}^{\mathbb{N}}$, for all $y \in Y_\tau$, for all $L \geq 1$, there exists $i \in \{0,\ldots,3^L-1\}$ such that*

$$y = S^i \varphi_{\tau_0} \circ \varphi_{\tau_1} \circ \cdots \circ \varphi_{\tau_{L-1}}\left(\Delta^L y\right). \tag{1}$$

3 Main results

3.1 Frequency of patterns and unique ergodicity of X_τ and Y_τ

Let Σ be a finite alphabet, and let $w \in \Sigma^*$ be a finite word. An *occurrence* of w in a sequence $x \in \Sigma^* \cup \Sigma^{\mathbb{N}} \cup \Sigma^{\mathbb{Z}}$ is an index i such that $x_{i+k} = w_k$ for all $0 \leq k < |w|$. We denote by $|x_0 \cdots x_{n-1}|_w$ the number of occurrences of w in the prefix of length n of x. The *frequency (of occurrence)* of w in x is defined as the limit:

$$\mathrm{freq}_x(w) := \lim_{n\to\infty} \frac{|x_0 \cdots x_{n-1}|_w}{n},$$

provided that this limit exists.

Theorem 3.1. *For any $\tau \in \{0,1\}^{\mathbb{N}}$, we have the following results:*

- *$\forall y \in Y_\tau$, $\forall w \in \{\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}\}^*$, the frequency of w in y exists, and depend only on τ and w ;*
- *$\forall x \in X_\tau$, $\forall w \in \{1,3\}^*$, the frequency of w in x exists, and depend only on τ and w.*

We recall that a subshift is said to be *uniquely ergodic* if there is a unique shift-invariant probability measure supported on this subshift. A reformulation of the above result is the following:

Corollary 3.2. *The subshifts X_τ and Y_τ are uniquely ergodic*

We denote by μ_τ and ν_τ the respective unique shift-invariant probability measure on X_τ and Y_τ.

3.2 Sketch of the proof

We first establish the result for Y_τ, by exploiting its substitutive structure. Then we show how X_τ inherits the unique ergodicity property from Y_τ.

We recall that, as for any subshift over a finite alphabet, the set of Borel probability measures over Y_τ is nonempty, and the topology of weak convergence of probability measures turns this set into a metrizable compact space. For $y \in Y_\tau$ and $n \geq 1$, we define the *empirical measure*

$$m_y^n := \frac{1}{n}\sum_{j=1}^{n} \delta_{S^j y}.$$

By compacity, there exists an increasing sequence (n_k) along which the sequence of empirical measure converges to an accumulation point ν. This accumulation point is always shift-invariant. Moreover, this weak convergence amounts to saying that the frequency of any finite word $w \in \{\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}\}^*$ exists along (n_k), and is given by $\nu([w])$, the measure of the cylinder set associated to the word w. The core of the argument consists in proving that this frequency depends neither on the choice of the increasing sequence (n_k) along which the convergence holds, nor on the choice of $y \in Y_\tau$.

A preliminary step shows that, for all $L \geq 0$, there exists an increasing sequence of integers $(n_k^L)_k$ (depending on (n_k) and y) along which we also have the weak convergence of the empirical measures $m^n_{\Delta^L y}$. We can then consider a^L, b^L, c^L and d^L to be the respective frequencies of the symbols A, B, C and D in $\Delta^L y$ measured along (n_k^L). By alternance of the blocks of 1s and the blocks of 3s in any smooth sequence, we easily see that

$$a_L + c_L = b_L + d_L = 1/2.$$

Therefore we can focus on the study of a^L and b^L (from which the other two can be deduced). The substitutive structure of Y_τ yields the following result:

Lemma 3.3. *For any $L \geq 1$, if $\tau_L = i \in \{0,1\}$, then $\left(a^L, b^L\right) = h_i\left(a^{L+1}, b^{L+1}\right)$, where h_0 and h_1 are the following homographies:*

$$h_0(a,b) := \frac{1}{3-2(a+b)}\left(1-a, \frac{1}{2}-a\right), \qquad h_1(a,b) := \frac{1}{3-2(a+b)}\left(\frac{1}{2}-a, 1-a\right).$$

As can be observed on Figure 1, taking Lemma 3.3 into account, their respective actions on the square K imply that any (a^L, b^L) is actually in $E := \{(a,b) \in K \mid a+b \leq 3/4\}$, so we consider h_0 and h_1 as maps: $E \to E$. Seen as such, both homographies can be shown to be contracting for $\|\cdot\|_\infty$, which implies the following lemma

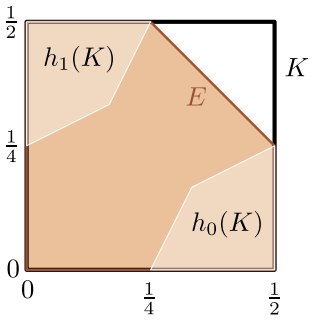


Figure 1: Homographies h_0 and h_1 acting on the square $K = [0, \frac{1}{2}] \times [0, \frac{1}{2}]$

Lemma 3.4. *For every sequence $t = (t_L)_{L\geq 0} \in \{0,1\}^{\mathbb{N}}$, there exists $\big(a(t), b(t)\big) \in E$ such that*

$$\bigcap_{L\geq 0} h_{t_0} \circ \cdots \circ h_{t_L}(E) = \big\{\big(a(t), b(t)\big)\big\}.$$

Coming back to the frequencies of letters observed in $\Delta^L y$ along the increasing sequence $(n_k^L)_{k\geq 1}$, we can conclude using Lemma 3.3 that these frequencies are dependent neither on the choice of (n_k) nor on that of y, they depend only on τ:

Corollary 3.5. *For each $L \geq 0$, we have*

$$a^L = a(S^L\tau), \quad b^L = b(S^L\tau), \quad c^L = \frac{1}{2} - a(S^L\tau), \quad \text{and} \quad d^L = \frac{1}{2} - b(S^L\tau).$$

Once we know that the frequencies of symbols in all the derivatives of y depend only on τ, we exploit again the substitutive structure of Y_τ to prove that similar results hold for the frequencies of larger patterns measured along (n_k), and the unique ergodicity of Y_τ follows.

Coming back to X_τ – To come back to the subshift X_τ of smooth sequences of Types τ, we establish a conjugacy between Y_τ and the system induced by X_τ on some subset $B \subset X_\tau$. By isomorphism, the system (B, S_B) (where S_B is the transformation induced on B by the shift map S) is also uniquely ergodic. Then the unique ergodicity of X_τ follows.

4 Minimality

Once we know that (Y_τ, S) and (X_τ, S) are uniquely ergodic for all τ, the question of the minimality of these systems arises naturally. We recall the following classical result (see *e.g.* [12, Theorem 6.17]): if (X,T) is a uniquely ergodic topological dynamical system, with unique invariant probability measure μ, then (X,T) is minimal if and only if $\mu(U) > 0$ for all non-empty open set $U \subset X$. In the context of a symbolic dynamical system, this condition is equivalent to saying that the unique invariant measure charges every cylinder $[w]$ for all finite word w in the language of the subshift. We have therefore to study whether, for a given $\tau \in \{0,1\}^{\mathbb{N}}$, the following is true:

$$\forall w \in \mathscr{L}(Y_\tau),\ \nu_\tau([w]) > 0. \tag{2}$$

Theorem 4.1. *For all $\tau \in \{0,1\}^{\mathbb{N}}$, the subshift (Y_τ, S) is minimal if and only if τ has infinitely many* 0*s*

Non-minimal cases – By looking back at the definition of φ_1, the only way to produce a symbol D through this substitution is by applying it to some already-present D: $\varphi_1(\mathsf{D}) = \mathsf{CDC}$. This results in the fact that whenever there exists an $L \geq 0$ such that $S^L\tau = 1^\infty$, the corresponding subshift Y_τ will contain words that have a finite amount of occurrences of the symbol D in them, contradicting (2) because in that case, $\mathsf{D} \in \mathscr{L}(Y_\tau)$ and $\nu_\tau([\mathsf{D}]) = 0$.

Other cases are minimal – On the other hand, a closer study of the actions of h_0 and h_1 on the square K yields a characterisation of the respective cases where each symbol frequency equals zero. Then, for a larger word w, we can always find a strictly shorter word w' which occurrence can be associated with that of w; hence the ability to inherit minimality from the strict positivity of the frequencies of symbols.

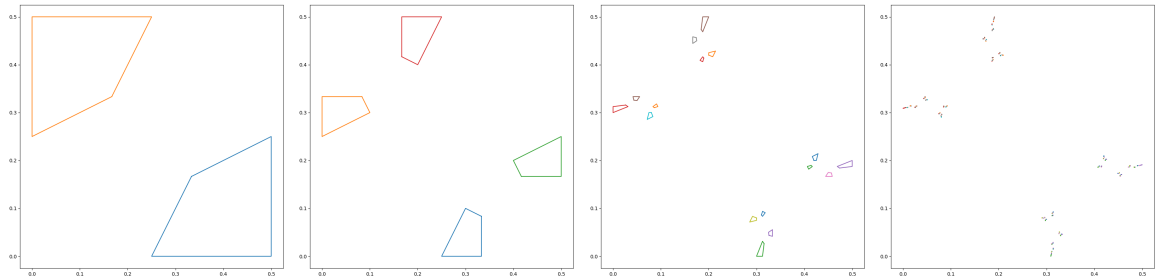
5 Perspectives

Study of the set of possible frequencies – Given a sequence type τ, we recall the notation $a = a(\tau) := \nu_\tau([\mathsf{A}])$, and likewise for B, C and D. By the unique ergodicity of (X_τ, S), all the sequences of X_τ have the same frequency of 1s. We denote by $f_\tau := \mu_\tau([1])$ the value of this frequency.

Let $F := \{f_\tau \mid \tau \in \{0,1\}^{\mathbb{N}}\}$ be the set of possible values for the frequency of 1s in smooth sequences. Figure 3 illustrates the successive approximations of F derived from the approximations of Figure 2. Note that F is symmetrical with respect to $1/2$, since swapping the 1s and the 3s in a sequence results in moving from a frequency f to a frequency $1-f$.

When the sequence τ is ultimately periodic, the value of f_τ has an explicit description. In particular, for $\tau_1 = 0001^\infty$, we obtain that $f_{\tau_1} = \frac{1}{2} - \frac{1}{13-\sqrt{5}}$, and for $\tau_2 = 1^\infty$, $f_{\tau_2} = \frac{1}{2} + \frac{2}{3+\sqrt{5}}$. We conjecture that these sequences achieve the maxima of F on the intervals $[0, 1/2]$ and $[0,1]$.

Problem 1. *Examine in further detail the fractal structure of the set F of possible values for the frequency of 1s in smooth sequences. Is it true that* $\sup F \cap [0,1/2] = \frac{1}{2} - \frac{1}{13-\sqrt{5}}$, *and* $\sup F = \frac{1}{2} + \frac{2}{3+\sqrt{5}}$ *?*



(a) 1 iteration (b) 2 iterations (c) 4 iterations (d) 8 iterations

Figure 2: Successive approximations of the set $\{(a(\tau), b(\tau)) \mid \tau \in \{0,1\}^{\mathbb{N}}\}$ obtained by computing the images of the square K by sequences of iterations of the two homographies h_0 and h_1.

Generalisation to other alphabets – The bi-infinite smooth sequence and recoding formalisms can be extended to any two-integers alphabet $\{\alpha, \beta\}$ with $\mathsf{A} = \alpha^\alpha, \mathsf{B} = \beta^\alpha, \mathsf{C} = \alpha^\beta$ and $\mathsf{D} = \beta^\beta$. When both α and β are odd integers, the properties of Section 2 still hold, with new substitutions φ_0 and φ_1.

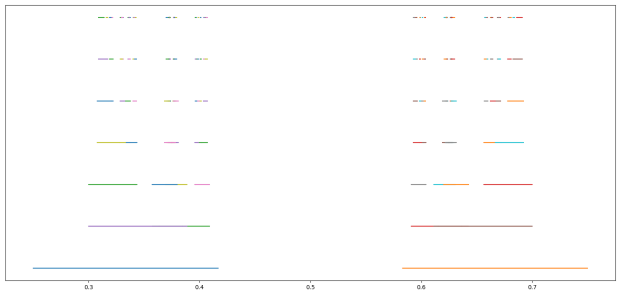
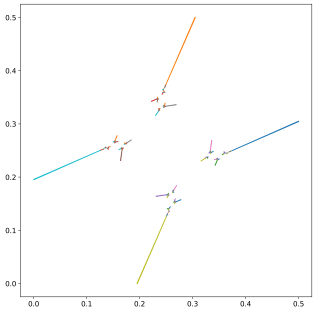


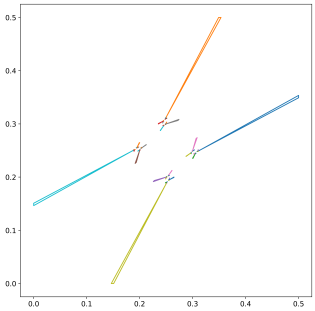
Figure 3: Successive approximations of F (from bottom up), up to rank 7.

However, if we want to use the same arguments as in Section 3 to prove the unique ergodicity, we need to ensure that the corresponding homographies are contracting on a suitable domain. But this does not seem to be straightforward, depending on the values of α and β. Indeed, finite approximations suggest that, for alphabets of the form $\{1,\beta\}$ with $\beta \geq 5$, the contraction property of the homographies might no longer be valid (see Figures 4a and 4b).

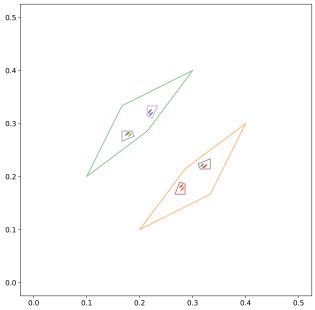
On the other hand, when $\min\{\alpha,\beta\} \geq 3$, the homographies seem to contract event faster than for $\{1,3\}$. See for example Figures 4c and 4d where we had to left the traces of the first iterations since the contraction was happening really fast.



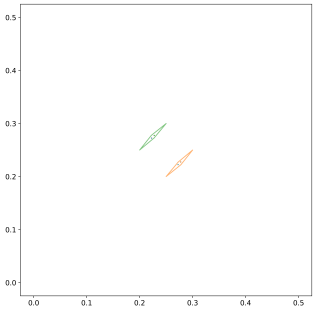
(a) $\{\alpha,\beta\} = \{1,9\}$



(b) $\{\alpha,\beta\} = \{1,17\}$



(c) $\{\alpha,\beta\} = \{3,7\}$



(d) $\{\alpha,\beta\} = \{9,11\}$

Figure 4: Images of the square K by sequences (of length 13 for (a) and (b), and up to length 6 for (c) and (d)) of iterations of the two homographies h_0 and h_1 associated with the given alphabet.

Problem 2. *For which values of α and β can we use a contracting property of the homographies h_0 and h_1 to prove the unicity of $a(\tau)$ and $b(\tau)$? Are the subshifts $(X_\tau)_{\tau\in\{0,1\}^{\mathbb{N}}}$ of smooth bi-infinite sequences on the alphabet $\{\alpha,\beta\}$ uniquely ergodic for* any *choice of odd integers α and β?*

References

[1] M. Baake & B. Sing (2004): *Kolakoski-(3, 1) Is a (Deformed) Model Set*. Canadian Mathematical Bulletin 47(2), p. 168–190, doi:10.4153/CMB-2004-018-6.

[2] S. Brlek, S. Dulucq, A. Ladouceur & L. Vuillon (2006): *Combinatorial properties of smooth infinite words*. Theor. Comput. Sci. 352(1-3), pp. 306–317, doi:10.1016/J.TCS.2005.12.003.

[3] S. Brlek, D. Jamet & G. Paquin (2008): *Smooth words on 2-letter alphabets having same parity*. Theoretical Computer Science 393(1-3), pp. 166–181, doi:10.1016/j.tcs.2007.11.019.

[4] S. Brlek & A. Ladouceur (2003): *A note on differentiable palindromes*. Theoretical Computer Science 302(1-3), pp. 167–178, doi:10.1016/S0304-3975(02)00746-6.

[5] J. Cassaigne & R. Henry (2026): *The complexity of finite smooth words over binary alphabets*. arXiv:2603.10733.

[6] F. M. Dekking (1980): *On the structure of self-generating sequences*. Séminaire de Théorie des Nombres de Bordeaux 2, pp. 1–6.

[7] F. M. Dekking (1981): *Regularity and irregularity of sequences generated by automata*. Séminaire de théorie des nombres de Bordeaux, pp. 901–910. Exposé no. 9.

[8] M. S. Keane (1991): *Ergodic theory and subshifts of finite type*. In T. Bedford, M. Keane & C. Series, editors: Ergodic theory, symbolic dynamics, and hyperbolic spaces, Oxford University Press, pp. 35–70.

[9] W. Kolakoski (1965): *Advanced Problems 5304*. The American Mathematical Monthly 72, p. 674. Solution in *Amer. Math. Monthly* 73 (1966), 681–682.

[10] R. Oldenburger (1939): *Exponent trajectories in symbolic dynamics*. Transactions of the American Mathematical Society 46(3), pp. 453–466, doi:10.1090/S0002-9947-1939-0000867-8.

[11] M. Rao (2012): *Trucs et bidules sur la séquence de Kolakoski*. Available at `https://www.arthy.org/kola/kola.php`. Unpublished manuscript.

[12] P. Walters (1982): *An introduction to ergodic theory*. Grad. Texts Math. 79, Springer, Cham.

Exhaustive Generation of Genus-One Knot and Link Diagrams via Maps on the Torus

Alexander Omelchenko
Constructor University Bremen
Campus Ring 1, 28759 Bremen, Germany
aomelchenko@constructor.university

We present an algorithmic framework for the exhaustive generation and tabulation of knot and link diagrams on the thickened torus $T^2 \times I$, based on the theory of maps on surfaces. Cellular 4-regular torus projections are encoded by permutation pairs (α, σ), and unsensed equivalence classes are enumerated completely and without duplication via canonical representatives. Crossing assignments, local diagram-level reductions, and the generalized Kauffman-type bracket are formulated entirely within the same permutation model. The pipeline is validated against published genus-one classifications for crossing numbers $N \leq 5$ and then extended to $N = 6, 7, 8$, producing, to our knowledge, the first complete genus-one tabulation at these crossing numbers under the stated comparison conventions. The resulting dataset contains more than 33 000 knot and link types. Besides the tables, the computation yields proved structural facts, including a parity statement for the a-span of the bracket and a sharp upper bound $N-1$ for the number of bigon faces in a 4-regular torus map. It also suggests several conjectures, among them a formula for the maximum number of straight-ahead components, the absence of equi-quadrilateral knot projections, and a $4N$ upper bound for the genus-one bracket span.

1 Introduction

Tabulation of knots and links has a long tradition going back to Tait [19], culminating in very large classical tables [7]. Virtual knots [8] and links in thickened surfaces $\Sigma_g \times I$ provide a natural extension of this program beyond S^3. A fundamental result of Kuperberg [13] asserts that every virtual link admits a unique minimal-genus surface representative. We work throughout with this surface-based viewpoint.

For the thickened torus $T^2 \times I$, explicit genus-one tables at small crossing numbers appeared in [15, 16, 17, 2, 3, 1]. These works typically start from abstract 4-regular graphs, then classify inequivalent embeddings into the surface, and only afterwards assign crossing data. Controlling equivalence under surface homeomorphisms is the main bottleneck.

The central idea of this paper is that the embedding should be treated as the *primary* combinatorial object. This is precisely the subject of the classical theory of *maps on surfaces*. A cellular embedding of a connected graph into a closed orientable surface is a map, and a cellular 4-regular knot projection is simply a 4-regular map. Maps admit a compact permutation encoding [18]: an involution α pairs the two darts of each edge, and a permutation σ records the cyclic order of darts at each vertex.

This approach is related to general-purpose map and graph generators, but it is not a direct invocation of them. The program `plantri` is optimized for many classes of plane maps and planar graphs [5]. More recent work of Brinkmann gives a general generator for maps on oriented surfaces by combining abstract graph generators with embedding and isomorphism rejection routines [4]. Our setting is more specialized: the generated objects are loopless cellular 4-regular torus maps satisfying knot-theoretic filters, and

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 130–138, doi:10.4204/EPTCS.445.16

© A. Omelchenko
This work is licensed under the
Creative Commons Attribution License.

the subsequent crossing assignments and state-sum invariants are computed in the same dart-permutation model. Thus Brinkmann's framework provides a natural independent benchmark or alternative generator, whereas the present implementation is designed as an integrated pipeline for genus-one knot and link diagrams.

In this encoding, unsensed equivalence of projections reduces to a finite conjugacy relation on permutation pairs, so that canonical representatives yield a duplication-free enumeration without a separate classification of embeddings. Moreover, the number of straight-ahead components (hence the knot/link distinction) is read off directly from the permutations.

Building on the map-enumeration line developed for degree-4 structures [9, 10, 11, 12], we turn these map-theoretic tools into a structural engine for knot and link tabulation on T^2. The implementation is validated against published genus-one tables for $N \leq 5$ and then extended to $N = 6,7,8$. A public reference implementation and datasets are available [14].

2 The Permutation Model for Maps

Let Σ_g be the closed orientable surface of genus g. A *map* on Σ_g is a cellular embedding of a connected graph into Σ_g. Permutations act on the right: $(pq)(h) = q(p(h))$.

Definition 1 (Permutation encoding)**.** *A labelled orientable map is encoded by a pair (α,σ) of permutations on a dart set H such that: (i) α is a fixed-point-free involution (edge pairing); (ii) the cycles of σ record the cyclic order of darts at vertices; (iii) $\langle\alpha,\sigma\rangle$ acts transitively on H (connectedness). The face permutation is $\varphi := \sigma\alpha$, and the genus is determined by*

$$\#\mathrm{cyc}(\sigma) - \#\mathrm{cyc}(\alpha) + \#\mathrm{cyc}(\varphi) = 2 - 2g.$$

Two encodings define the same *sensed* map if one is obtained from the other by simultaneous conjugation; they define the same *unsensed* map if, additionally, $\sigma \mapsto \sigma^{-1}$ (orientation reversal) is allowed.

For 4-regular maps, each σ-cycle has length 4, and the involution σ^2 pairs opposite darts at each vertex. Define $\rho := \sigma^2\alpha$. Then the number of *straight-ahead components* of the corresponding projection (i.e. the components of the projected curve) is

$$c(P) = \tfrac{1}{2}\#\mathrm{cyc}(\rho).$$

This detects the knot/link distinction directly from the permutation data.

Monogon faces (fixed points of φ) are the projection-level shadow of immediate Reidemeister I reductions and are excluded from our enumeration.

Example. Consider the dart set $H = \{1,\dots,8\}$ with $\sigma_0 = (1234)(5678)$ ($N = 2$ vertices) and $\alpha = (16)(28)(35)(47)$. Then $\varphi = \sigma_0\alpha = (17)(25)(38)(46)$, giving $F = 4$ cycles (each of length 2) and $V - E + F = 2 - 4 + 4 = 2 \neq 0$; this is a sphere map, not a torus one. The torus condition $F = N$ is therefore a genuine constraint on the matching α.

3 Canonical Enumeration of Projections on the Torus

Fix $N \geq 1$ and let $H = \{1,\dots,4N\}$. We fix a standard vertex rotation

$$\sigma_0 = (1234)(5678)\cdots(4N{-}3\ 4N{-}2\ 4N{-}1\ 4N).$$

Every unlabelled 4-regular map admits a labelled encoding of the form (α, σ_0) for some fixed-point-free involution α.

Definition 2 (Candidate torus projection)**.** *A pair (α, σ_0) is a candidate projection on T^2 with N crossings if: (1) α is a fixed-point-free involution; (2) $\langle \alpha, \sigma_0 \rangle$ acts transitively on H; (3) $\#\mathrm{cyc}(\sigma_0\alpha) = N$ (torus condition); (4) $\varphi = \sigma_0\alpha$ has no fixed points (no monogons); (5) no loop edges (for every h, h and $\alpha(h)$ lie in different σ_0-cycles).*

Canonical representatives and rooted normalization. Fix a total order $\prec$ on labelled pairs. The unsensed canonical representative of (α, σ) is the $\prec$-minimal element of the union of the conjugacy orbits of (α, σ) and (α, σ^{-1}) under S_H. Two encodings represent the same unsensed map if and only if their canonical representatives coincide.

In practice this minimum is computed by rooted normalization rather than by enumerating S_H. For each root dart r and for each of the two orientations σ and σ^{-1}, we perform a deterministic traversal of the dart graph, inspecting from a dart h the neighbours in the fixed order $(\sigma(h), \alpha(h))$. Darts are relabelled $1, 2, \ldots, 4N$ in order of first discovery, and the resulting labelled pair is recorded. The canonical form is the lexicographically smallest normal form over all $4N$ roots and both orientations. A rooted connected map has no non-trivial label ambiguity once the root and orientation are fixed, so this procedure gives the same complete invariant as the abstract orbit minimum. Its cost is $O(N)$ per root and hence $O(N^2)$ per accepted labelled encoding.

Enumeration algorithm and crude complexity bounds. The conceptual algorithm enumerates fixed-point-free involutions α on H; for each one it tests the defining conditions, computes the canonical representative, and stores it. After deduplication, this yields exactly one representative per unsensed class. The naive worst-case number of matchings is

$$(4N-1)!! = \frac{(4N)!}{2^{2N}(2N)!},$$

so the corresponding crude time bound is $O(N^2(4N-1)!!)$ and the storage needed after deduplication is $O(N|\mathscr{C}_N|)$. This bound is intentionally pessimistic. The implementation uses a canonical construction path with activated vertices and early rejection of partial matchings; it does not materialize all matchings. The lower bound is the output size: any complete enumeration must take $\Omega(|\mathscr{C}_N|)$ time.

Theorem 3 (Correctness)**.** *The procedure outputs a set $\mathscr{C}_N$ that is sound (every element is a valid candidate), complete (every unsensed class is represented), and non-duplicating (no two elements represent the same class).*

Primeness filters. A projection is called *prime* in the computational sense used here if its underlying multigraph has no 2-edge-cut and no splitness witness based on mixed vertices. These are explicit combinatorial witnesses, stable under unsensed equivalence. They are sufficient for matching the projection-level conventions used in the comparison tables below, but they are not presented as a complete topological characterization of primeness in every possible convention.

Benchmark data. Table 1 records the sizes of the enumerated sets. Table 2 shows the distribution of prime link projections by component count.

N	candidates	removed	prime total	knot proj.	link proj.
3	6	0	6	2	4
4	28	5	23	10	13
5	109	28	81	34	47
6	595	216	379	170	209
7	3216	1421	1795	777	1018
8	19956	10141	9815	4308	5507

Table 1: Unsensed 4-regular projection counts on the torus for $N = 3,\dots,8$. The column "removed" records removals by the 2-edge-cut compositeness witness; in this range the splitness witness removes no further candidates.

N	$c=2$	$c=3$	$c=4$	$c=5$	$c=6$
3	3	1	—	—	—
4	9	3	1	—	—
5	37	9	1	—	—
6	150	51	7	1	—
7	775	212	30	1	—
8	4030	1293	169	14	1

Table 2: Prime link projections by number of straight-ahead components $c(P) \geq 2$.

4 From Projections to Diagrams

For a fixed projection P with N crossings, a *diagram* $D = (P,b)$ is obtained by choosing a crossing assignment $b : \mathrm{Vert}(P) \to \{0,1\}$, selecting which strand is overpassing at each vertex. Thus b is the over/under datum of the diagram. This gives 2^N diagrams per projection, or half as many when the comparison convention identifies the global crossing switch $b \mapsto 1-b$.

Bigon reduction. A bigon face is a 2-cycle $(i\,j)$ of $\varphi = \sigma\alpha$. Suppose that it is incident to two distinct vertices u and v. Choose local cyclic orders $(h_0\,h_1\,h_2\,h_3)$ at u and $(k_0\,k_1\,k_2\,k_3)$ at v, where $b(u) = 0$ means that $\{h_0,h_2\}$ is overpassing and $b(v) = 0$ means that $\{k_0,k_2\}$ is overpassing. If $i = h_r$ and $j = k_s$, put $p_u(i) = r \pmod 2$ and $p_v(j) = s \pmod 2$. Then the bigon supports an immediate Reidemeister II reduction if and only if

$$b(u) + b(v) + p_u(i) + p_v(j) \equiv 1 \pmod 2.$$

The reduced-diagram convention used in the comparison tables keeps precisely those assignments for which this congruence fails for every two-vertex bigon. The simpler rule $b(u) \neq b(v)$ is valid only in local labellings where $p_u(i) = p_v(j) = 0$.

State-sum invariants. The generalized Kauffman bracket on the torus [2] is formulated entirely in the permutation model. At a vertex, the crossing assignment b selects the overpassing local strand. An A- or B-smoothing replaces the crossing by one of the two non-crossing pairings of the four incident darts; the product over all vertices is a smoothing involution $\tau_{b,s}$, where the state s records the A/B choice at each vertex. The *state circles* are the connected components of the 2-regular dart graph whose edges

are the α-pairs and the $\tau_{b,s}$-pairs. Equivalently, each state circle gives two cycles of the permutation $\pi_{b,s} := \alpha\tau_{b,s}$.

The torus bracket distinguishes contractible and essential state circles. Let C_1 be the $\mathbb{F}_2$-vector space generated by the projection edges and let $\partial_2 : C_2 \to C_1$ be the cellular boundary map sending each face to its mod-2 boundary vector. A state circle is contractible if and only if its edge incidence vector lies in $\operatorname{im}\partial_2$; otherwise it is essential. Thus no drawing in a fundamental polygon is required. The bracket is

$$\langle D\rangle(a,x) = \sum_{s\in\{A,B\}^N} a^{A(s)-B(s)}(-a^2-a^{-2})^{\gamma(s)}x^{\delta(s)}, \tag{1}$$

where $A(s)$ and $B(s)$ are the numbers of A- and B-smoothings in the state, and $\gamma(s)$ and $\delta(s)$ count contractible and essential state circles. For knots, $w(D)$ denotes the usual writhe, and the normalized polynomial

$$X_D(a,x) = (-a)^{-3w(D)}\langle D\rangle(a,x)$$

is invariant under all Reidemeister moves.

An efficient implementation precomputes the pair $(\gamma(t),\delta(t))$ for every smoothing vector $t \in \{0,1\}^N$ once per projection, then reuses the data for all crossing assignments via the bijection $s = b \oplus t$.

Validation. For $N \le 5$ we compare raw locally reduced diagram classes with the published lists. The comparison is summarized in Table 3.

N	knots	links
2	1/1	1/1
3	3/3	4/4
4	18/17	22/21
5	71/69	99/99

Table 3: Raw locally reduced counts: this work / published count.

The discrepancies are convention-dependent and are not treated as contradictions with the published classifications. For $N = 4$ knots, the additional class in our raw list is supported by the unique $N = 4$ projection without bigon faces; it is excluded in the published table by a global primeness convention not captured by our local bigon rule. For $N = 4$ links, the published count 21 is obtained from an original list of 22 by a global equivalence already noted in the link classification. For $N = 5$ knots, one of the two extra raw classes is equivalent to a smaller-crossing class under the comparison conventions. The remaining difference reflects an additional global convention in the published knot table, such as the central-fragment convention related to Ω_3 moves, rather than a failure of the permutation enumeration itself. For $N = 5$ links the count agrees exactly with the published table when the optional over/under participation prefilter is not imposed.

5 Tabulation Data and Structural Observations

The computation is extended to $N = 6, 7, 8$. Table 4 records the new diagram types at each crossing number.

N	knots	links	total	ratio to $N-1$
2	1	1	2	—
3	3	4	7	3.5
4	18	22	40	5.7
5	71	99	170	4.2
6	378	525	903	5.3
7	1743	2909	4652	5.2
8	10704	16752	27456	5.9

Table 4: New genus-one diagram types at each crossing number. Through $N = 8$ the tabulation contains 12918 knot and 20312 link types (33230 total).

A log-linear fit over $N = 3,\dots,8$ is numerically close to $0.052 \cdot 5.14^N$, but this is only a descriptive summary of the available data. With six data points we make no asymptotic claim about the true exponential growth constant.

5.1 Maximum number of straight-ahead components

For each N, let $c_{\max}(N)$ denote the maximum number of straight-ahead components among prime projections. In the computed range $3 \le N \le 8$:

Proposition 4 (Lattice lower bound)**.** *For every even $N \ge 4$, let P_N be the 4-regular map on the torus obtained from the square lattice $\mathbb{Z}^2$ by identifying under $\Lambda_N = \langle (2,0),(0,N/2)\rangle$. Then P_N has N vertices, N quadrilateral faces, $N/2+2$ straight-ahead components, and no 2-edge-cut.*

Conjecture 5. *For all $N \ge 3$, $c_{\max}(N) = \lfloor N/2 \rfloor + 2$. In the computed range the maximizer is unique: the lattice projection for even N; a projection with face-degree vector $(3,3,4^{N-3},6)$ for odd N.*

The proved even-N lattice construction gives the lower bound. The remaining problem is the upper bound. A possible approach is to study the intersection pattern of straight-ahead components as a collection of essential curves on the torus and to show that primeness forces enough mixed vertices between them. For odd N, the computed maximizers suggest a triangle-hexagon defect inserted into the even lattice family; constructing and analyzing this family explicitly would be a natural first step.

N	3	4	5	6	7	8
$c_{\max}(N)$	3	4	4	5	5	6
$\lfloor N/2 \rfloor + 2$	3	4	4	5	5	6

Table 5: Maximum component count among prime torus projections.

5.2 Equi-quadrilateral projections are always links

A 4-regular torus map is *equi-quadrilateral* if every face has degree 4. In the computed range, every prime equi-quadrilateral projection has $c(P) \ge 2$: no equi-quadrilateral projection supports a knot diagram.

Conjecture 6. *Every prime equi-quadrilateral 4-regular map on the torus has at least 2 straight-ahead components.*

This admits a purely permutation-theoretic reformulation: if σ and $\varphi = \sigma\alpha$ both have cycle type (4^N) and $\langle \sigma, \alpha \rangle$ acts transitively, must $\rho = \sigma^2\alpha = \sigma\varphi$ have at least four cycles? Equivalently, can the straight-ahead component count be one? The computation suggests that homological restrictions on the straight-ahead cycles may rule out this case.

5.3 The a-span of the genus-one bracket

For a Laurent polynomial $Q(a,x)$, define

$$\operatorname{span}_a(Q) = \max\{k : [a^k]Q \neq 0\} - \min\{k : [a^k]Q \neq 0\}.$$

Proposition 7 (Span parity)**.** *For any torus diagram with N crossings, every monomial in $\langle D \rangle(a,x)$ has a-exponent of parity N. In particular,* $\operatorname{span}_a \langle D \rangle$ *is even.*

Proof. In the state sum (1), $A(s) - B(s) = 2A(s) - N \equiv N \pmod 2$, and $(-a^2 - a^{-2})^{\gamma(s)}$ contributes only even exponents. □

Conjecture 8 ($4N$ span bound)**.** *For every genus-one knot diagram with N crossings,* $\operatorname{span}_a(X_D) \leq 4N$. *The same bound holds for the bracket of link diagrams in the computed range $3 \leq N \leq 8$.*

This is motivated by the classical Kauffman–Murasugi–Thistlethwaite span theorem, but the present statement is only conjectural in genus one. A possible route to a proof would be to adapt the state-graph argument behind the classical span inequality, while controlling cancellations between states with contractible and essential circles. Table 6 confirms the bound in the computed range.

N	min span	max span	$4N$	avg span
3	6	12	12	9.0
4	8	16	16	11.9
5	10	20	20	15.4
6	4	24	24	17.7
7	4	28	28	21.0
8	4	32	32	23.6

Table 6: Observed a-span of $X_D(a,x)$ for genus-one knot types.

5.4 Further observations

Purely essential knots. A genus-one knot is *purely essential* if $X_D(a,x)$ has no x^0 term. Equivalently at the level of the invariant, all purely contractible contributions cancel or are absent. This phenomenon, vacuous on S^2, accounts for a large fraction of genus-one knot types (Table 7).

Bigon prevalence. A 4-regular torus map with N vertices has at most $N-1$ bigon faces: since Euler's formula gives $F = N$, not all faces can be bigons, and if $N-1$ faces are bigons then the remaining face has degree $2N+2$. This bound is sharp at every $N \leq 8$ in our data. The fraction of prime projections containing at least one bigon reaches 97% at $N = 8$. This trend is qualitatively consistent with general 0–1 laws for submaps: in broad map classes, an admissible fixed planar submap appears linearly often

N	purely ess.	total knots	fraction
3	2	3	0.67
4	14	18	0.78
5	34	71	0.48
6	276	378	0.73
7	1053	1743	0.60
8	7805	10704	0.73

Table 7: Fraction of purely essential knot types at each crossing number.

with probability tending to one [6]. Our loopless, 4-regular, prime torus projections form a more constrained family, so this theorem does not by itself prove the observed statement here; it explains why rapid appearance of bigons should not be surprising.

6 Outlook

The permutation framework applies without modification to any Σ_g. On the torus, the contractible/essential dichotomy for state circles reduces to an $\mathbb{F}_2$ boundary test, and a single variable x suffices for the bracket. For $g \geq 2$, separating essential circles can be null-homologous, requiring integral homology classes and mapping-class-group canonicalization; this is the main obstacle to scaling the pipeline beyond genus one.

References

[1] S. V. Matveev A. A. Akimova & V. V. Tarkaev (2020): *Classification of prime links in the thickened torus having crossing number 5*. *J. of Knot Theory and Its Ramifications* 29(03), p. 2050012, doi:10.1142/S0218216520500121.

[2] A. A. Akimova & S. Matveev (2012): *Classification of knots in T x I with at most 4 crossings*. *arXiv: Geometric Topology*. Available at https://api.semanticscholar.org/CorpusID:119621553.

[3] A. A. Akimova & S. V. Matveev (2014): *Classification of genus 1 virtual knots having at most five classical crossings*. *J. of Knot Theory and Its Ramifications* 23(06), p. 1450031, doi:10.1142/S021821651450031X.

[4] G. Brinkmann (2024): *Generating Maps on Oriented Surfaces Using the Homomorphism Principle*. *Discrete & Computational Geometry*, doi:10.1007/s00454-025-00734-5.

[5] G. Brinkmann & B. D. McKay (2007): *Fast generation of planar graphs*. *Match-communications in Mathematical and in Computer Chemistry* 58, pp. 323–357. Available at https://api.semanticscholar.org/CorpusID:116425311.

[6] Z.-C. Gao E. A. Bender & L. B. Richmond (1992): *Submaps of maps. I. General 0-1 laws*. *J. Comb. Theory B* 55(1), pp. 104–117, doi:10.1016/0095-8956(92)90034-U.

[7] M. Thistlethwaite J. Hoste & J. Weeks (1998): *The first 1,701,936 knots*. *The Mathematical Intelligencer* 20, pp. 33–48, doi:10.1007/BF03025227.

[8] L. H. Kauffman (1999): *Virtual Knot Theory*. *Eur. J. Comb.* 20(7), pp. 663–691, doi:10.1006/EUJC.1999.0314.

[9] E. Krasko & A. Omelchenko (2017): *Enumeration of 4-regular one-face maps*. *European J. of Combinatorics* 62, pp. 167–177, doi:10.1016/j.ejc.2016.12.004.

[10] E. Krasko & A. Omelchenko (2019): *Enumeration of r-regular maps on the torus. Part I:*. *Discrete Math.* 342(2), pp. 584–599, doi:10.1016/j.disc.2018.07.013.

[11] E. Krasko & A. Omelchenko (2019): *Enumeration of r-regular maps on the torus. Part II: Unsensed maps*. *Discrete Math.* 342(2), pp. 600–614, doi:10.1016/j.disc.2018.09.004.

[12] E. Krasko & A. Omelchenko (2021): *Enumeration of rooted 4-regular maps without planar loops*. *Discrete Math.* 344(8), p. 112442, doi:10.1016/j.disc.2021.112442.

[13] G. Kuperberg (1999): *What is a virtual link?* *Algebr. Geom. Topol.* 3, pp. 587–591, doi:10.2140/agt.2003.3.587.

[14] A. Omelchenko (2026): *Torus_Maps: Reference implementation and datasets for genus-one projection and diagram enumeration on the torus*. Available at https://github.com/avo-travel/Torus_Maps.

[15] V. R. Meshkov S. A. Grishanov & A. V. Omelchenko (2007): *Kauffman-type polynomial invariants for doubly periodic structures*. *Journal of Knot Theory and Its Ramifications* 16(06), pp. 779–788, doi:10.1142/S021821650700549X.

[16] V. Meshkov S. Grishanov & A. Omelchenko (2009): *A Topological Study of Textile Structures. Part I: An Introduction to Topological Methods*. *Textile Res. J.* 79, pp. 702 – 713, doi:10.1177/0040517508095600.

[17] V. Meshkov S. Grishanov & A. Omelchenko (2009): *A Topological Study of Textile Structures. Part II: Topological Invariants in Application to Textile Structures*. *Textile Res. J.* 79, pp. 822–836, doi:10.1177/0040517508096221.

[18] Sergei S. K. Lando & A. K. Zvonkin (2004): *Graphs on Surfaces and Their Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, doi:10.1007/978-3-540-38361-1_1.

[19] P. G. Tait (1877): *On knots I, II, III*. *Trans. Roy. Soc. Edinburgh* 28, pp. 35–79.

Permanental Inequalities and Unit Interval Orders

Sihong Pan Mark Skandera Jiayuan Wang
Lehigh University, Bethlehem PA, USA
sip318@lehigh.edu mas906@lehigh.edu jiw922@lehigh.edu

Given an $n \times n$ matrix $A = (a_{i,j})$ and $I,J \subseteq [1,n] := \{1,\dots,n\}$, let $A_{I,J} = (a_{i,j})_{i\in I, j\in J}$ denote the (I,J)-submatrix of A. We consider a class of 0-1 totally nonnegative (TNN) matrices arising as antiadjacency matrices of unit interval orders, and show that for each matrix A in this class, the inequalities

$$\mathrm{per}(A_{[1,h],[1,h]})\,\mathrm{per}(A_{[h+1,n],[h+1,n]}) \;\geq\; \mathrm{per}(A_{[1,n]\cap 2\mathbb{Z},[1,n]\cap 2\mathbb{Z}})\,\mathrm{per}(A_{[1,n]\smallsetminus 2\mathbb{Z},[1,n]\smallsetminus 2\mathbb{Z}}) \qquad (*)$$

hold for $h = 1,\dots,n-1$.

Let $\mathfrak{A}_n$ be the Young Subgroup $\mathfrak{S}_{1,\dots,\lfloor \frac{n}{2}\rfloor} \times \mathfrak{S}_{\lceil \frac{n}{2}\rceil,\dots,n}$, and $\mathfrak{B}_n$ be the parity alternating permutations $\{w = w_1\cdots w_n \in \mathfrak{S}_n \mid i \text{ and } w_i \text{ have the same parity}\}$. We find a bijective map $f_n : \mathfrak{A}_n \to \mathfrak{B}_n$ such that $w \leq f_n(w)$ under the Bruhat order for $n \leq 13$ and conjecture this holds for all n. The existence of the bijective map f_n ensures that inequalities $(*)$ are satisfied for all TNN matrices A when $h = \lfloor \frac{n}{2}\rfloor$. We also conjecture the inequalities $(*)$ to hold for all TNN matrices and all $h = 1,\dots,n-1$.

1 Introduction

Given an $n\times n$ matrix $A = (a_{i,j})$ and subsets $I,J \subseteq [n] := \{1,\dots,n\}$, define the submatrix $A_{I,J} = (a_{i,j})_{i\in I, j\in J}$ of A, and call $\det(A_{I,J})$ a *minor* of A. We call $A \in \mathrm{Mat}_{n\times n}(\mathbb{R})$ *totally nonnegative* if each of its minors is negative. It is known that the defining inequalities $\det(A_{I,J}) \geq 0$ of totally nonnegative matrices imply many other polynomial inequalities in the matrix entries. Some of these involve *immanants* of A and its submatrices, i.e., linear combinations of the $n!$ monomials $\{a_{1,w_1}\cdots a_{n,w_n} \mid w \in \mathfrak{S}_n\}$. In particular for any function $f : \mathfrak{S}_n \to \mathbb{R}$, we define the *$f$-immanant of A* [12] to be the polynomial

$$\mathrm{Imm}_f(A) := \sum_{w\in\mathfrak{S}_n} f(w) a_{1,w_1}\cdots a_{n,w_n} \qquad (1.1)$$

generalizing the permanent and determinant

$$\mathrm{per}(A) = \sum_{w\in\mathfrak{S}_n} a_{1,w_1}\cdots a_{n,w_n}, \qquad \det(A) = \sum_{w\in\mathfrak{S}_n} \mathrm{sgn}(w) a_{1,w_1}\cdots a_{n,w_n}.$$

Skandera [9] gave a necessary and sufficient condition on subsets $I,I',J,J' \subseteq [n] := \{1,\dots,n\}$ such that the inequality

$$\det(A_{I,J})\det(A_{[n]\smallsetminus I,[n]\smallsetminus J}) \;\leq\; \det(A_{I',J'})\det(A_{[n]\smallsetminus I',[n]\smallsetminus J'})$$

holds whenever A is TNN. Some other inequalities [1, 8, 13, 14] suggest that one can use inequalities involving determinants to produce inequalities involving permanents by simply changing the direction of the inequality.

Given a subset $I \subseteq [n]$, let $\bar{I} = [n] \smallsetminus I$. A natural starting point for such an investigation is Skandera's result [9, Thm. 3.2] which provides an *upper bound* for all products $\det(A_{I,I})\det(A_{\bar{I},\bar{I}})$:

$$\det(A_{I,I})\det(A_{\bar{I},\bar{I}}) \;\leq\; \det(A_{[n]\cap 2\mathbb{Z},[n]\cap 2\mathbb{Z}})\det(A_{[n]\smallsetminus 2\mathbb{Z},[n]\smallsetminus 2\mathbb{Z}})$$

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 139–147, doi:10.4204/EPTCS.445.17

© S. Pan, M. Skandera, & J. Wang
This work is licensed under the
Creative Commons Attribution License.

whenever A is TNN. It is natural to ask whether we can find a *lower bound* for all products $\operatorname{per}(A_{I,I})\operatorname{per}(A_{\bar{I},\bar{I}})$:

$$\operatorname{per}(A_{I,I})\operatorname{per}(A_{\bar{I},\bar{I}}) \;\geq\; \operatorname{per}(A_{[n]\cap 2\mathbb{Z},[n]\cap 2\mathbb{Z}})\operatorname{per}(A_{[n]\smallsetminus 2\mathbb{Z},[n]\smallsetminus 2\mathbb{Z}}). \tag{1.2}$$

Our main result in Section 5 shows that inequalities of the form (1.2) hold when

(1) The totally nonnegative matrix A is related to a unit interval order, and

(2) I is the interval $[h]$ for any $h \in [n]$.

We conjecture the inequalities to hold for all totally nonnegative matrices and $I = [h]$. On the other hand, we show that the inequalities do not hold for general I, even when restricted to matrices of the form (1).

2 Four families of sets counted by Catalan numbers

A simple subset of $n \times n$ totally nonnegative matrices is the set of 0-1 matrices which are weakly increasing in columns, weakly decreasing in rows, and all entries on or below the diagonal are 1. Call this set $\mathscr{A}_n$. For $A \in \mathscr{A}_n$, the boundary between entries of A equal to 1 and 0 can be viewed as a $2n$-step *Dyck path*, consisting of n horizontal and n vertical steps, none below the diagonal, from the upper left corner of the matrix to the lower right corner. For example, we can draw an 8-step Dyck path on a matrix in $\mathscr{A}_4$.

$$\begin{array}{|cccc} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{array}$$

The bijection between $\mathscr{A}_n$ and $2n$-step Dyck paths implies that the cardinality of $\mathscr{A}_n$ is the nth Catalan number, $\frac{1}{n+1}\binom{2n}{n}$. (See [11, Exercise 6.19 (i)].)

Observation 2.1. *Each matrix in $\mathscr{A}_n$ is totally nonnegative.*

Elements of $\mathscr{A}_n$ correspond bijectively to n-element posets called *unit interval orders*. Call this set $\mathscr{U}_n$. (See [5, p. 33], [11, Exercise 6.19 (ddd)], [15, §8.2].) These are the posets in which no induced four-element subposet is isomorphic to either of the posets

$$\mathbf{3+1} = \;\vdots\!\bullet, \qquad \mathbf{2+2} = \;\vdots\!\vdots.$$

The bijection $A \mapsto P(A)$ is given by declaring $i \leq_{P(A)} j$ if and only if $a_{i,j} = 0$. Each matrix A is said to be the *antiadjacency matrix* of $P(A)$.

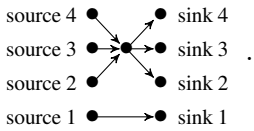
Define a *planar network of order n* to be a directed, planar, acyclic multigraph which can be embedded in a disc so that $2n$ boundary vertices can be labeled clockwise as *source* $1,\ldots,$ *source* n, *sink* $n,\ldots,$ *sink* 1. We will assume all sources to have indegree 0 and outdegree 1, and all sinks to have indegree 1 and outdegree 0. Let $\mathscr{F}_n$ denote the set of such networks.

For each subinterval $[a,b]$ of $[n]$ we define a *simple star network* $F_{[a,b]} \in \mathscr{F}_n$ by

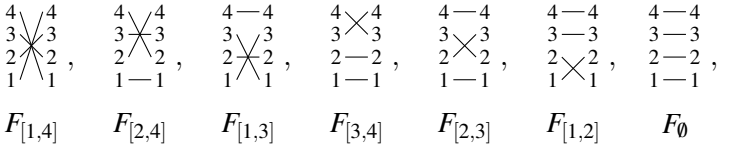
1. Sources $1,\ldots,n$ lie on a vertical line to the left; sinks $1,\ldots,n$ lie on a vertical line to the right. Both are labeled from bottom to top.
2. An interior vertex lies between the sources and sinks.
3. For $i = 1,\ldots,a-1$ and $i = b+1,\ldots,n$, a directed edge begins at source i and terminates at sink i.

4. For $i = a, \ldots, b$, a directed edge begins at source i and terminates at the interior vertex, and another directed edge begins at the interior vertex and terminates at sink i.

For zero- and one-element subintervals we define the trivial network $F_\emptyset = F_{[1,1]} = \cdots = F_{[n,n]}$ to have no interior vertex, and n horizontal edges, each from source i to sink i, for $i = 1, \ldots, n$. For example, the element $F_{[2,4]} \in \mathscr{F}_4$ is



For economy, we will omit edge orientations and vertices from drawings of planar networks. Thus we will draw the seven simple star networks in $\mathscr{F}_4$ as



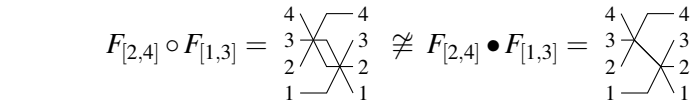
(2.1)

where $F_\emptyset = F_{[1,1]} = F_{[2,2]} = F_{[3,3]} = F_{[4,4]}$. When there is no danger of confusion, we will omit source and sink labels as well.

The set $\mathscr{F}_n$ is generated by simple star networks and two types of concatenation operations. Given networks $E, F \in \mathscr{F}_n$, define the (ordinary) concatenation $E \circ F$ as follows. For $i = 1, \ldots, n$, do

1. Remove sink i of E and source i of F,
2. Merge each edge $(x, \text{sink } i)$ in E with each edge $(\text{source } i, y)$ in F to form a single edge (x, y) in $E \circ F$.

Sometimes in a concatenation $E \circ F$, there may exist vertices x in E, y in F with more than one edge incident upon both. Let $m(x,y)$ be this number. Define the *condensed concatenation* $E \bullet F$ to be the subdigraph of $E \circ F$ obtained by removing, for all such pairs (x,y), all but one of the $m(x,y)$ edges incident upon both. For example, in $\mathscr{F}_4$ we have



(2.2)

Definition 2.2. Define a *descending star network* to be a condensed concatenation of the form

$$F = F_{[c_1,d_1]} \bullet \cdots \bullet F_{[c_t,d_t]}, \tag{2.3}$$

with intervals $[c_1,d_1], \ldots, [c_t,d_t]$ distinct, pairwise nonnesting, and satisfying $c_1 > \cdots > c_t$.

Let $\mathscr{D}_n$ be the set of all descending star networks of order n. In the notation of Definition 2.2, these are $F_{[1,4]}$, $F_{[2,4]} \bullet F_{[1,3]}$, $F_{[2,4]} \bullet F_{[1,2]}$, etc. By [3, Thm. 3.6], [10, Thm. 3.5, Lem 5.3], the number of descending star networks of order n is $\frac{1}{n+1}\binom{2n}{n}$. Explicit bijections between $\mathscr{U}_n$ and $\mathscr{D}_n$ can be found at [2].

Given a planar network F of order n, define the *path matrix* $B = (b_{i,j})$ of F by

$$b_{i,j} = \text{number of paths in } F \text{ from source } i \text{ to sink } j. \tag{2.4}$$

Let $\mathscr{P}_n$ be the set of $\frac{1}{n+1}\binom{2n}{n}$ path matrices of planar networks in $\mathscr{D}_n$.

Some, but not all, of the matrices in $\mathscr{A}_n$ appear in $\mathscr{P}_n$. The precise connection between $\mathscr{A}_n$ and $\mathscr{P}_n$ can be described using *block structures* of these matrices. Let $\alpha = (\alpha_1, \ldots, \alpha_r)$ be an integer composition of n, and define the (ordered) set partition $(I_1, \ldots, I_r)$ of $[n]$ into intervals of lengths $\alpha_1, \ldots, \alpha_r$ by

$$I_1 = [1, \alpha_1], \quad I_2 = [\alpha_1 + 1, \alpha_1 + \alpha_2], \quad \ldots, \quad I_r = [n - \alpha_r, n]. \tag{2.5}$$

For any $n \times n$ matrix A, we can write it in terms of *block submatrices* A_{I_i,I_j} by

$$A = (A_{I_i,I_j}) = \begin{bmatrix} A_{I_1,I_1} & A_{I_1,I_2} & \cdots & A_{I_1,I_r} \\ A_{I_2,I_1} & A_{I_2,I_2} & \cdots & A_{I_1,I_r} \\ \vdots & \vdots & & \vdots \\ A_{I_r,I_1} & A_{I_r,I_2} & \cdots & A_{I_r,I_r} \end{bmatrix}. \tag{2.6}$$

If $A_{I_i,I_j} = 0$ whenever $i < j$, we say A is *α-block lower-triangular*; if A and $A^\top$ are both α-block lower-triangular, we say that A is *α-block diagonal*. In particular, for any square matrix A there is a finest composition $\alpha = \alpha(A)$ of n for which A is α-block lower-triangular. Next, we precisely state the relationship between $\mathscr{A}_n$ and $\mathscr{P}_n$. Define the map

$$\begin{aligned} \psi : \mathscr{A}_n &\to \mathscr{P}_n \\ A &\mapsto A_{I_1,I_1} \oplus \cdots \oplus A_{I_r,I_r}, \end{aligned} \tag{2.7}$$

where $(I_1, \ldots, I_r)$ is the set partition associated to the finest composition $\alpha(A)$ as in (2.5).

Proposition 2.3. *Given unit interval order $P \in \mathscr{U}_n$ with antiadjacency matrix A, the descending star network $F = F(P) \in \mathscr{D}_n$ has path matrix $\psi(A)$.*

3 Permanents and path families

The planar networks defined in Section 2 have close connections to permanents. Let $\pi = (\pi_1, \ldots, \pi_n)$ be a sequence of source-to-sink paths in a planar network $F \in \mathscr{F}_n$. We call π a *path family* if there exists a permutation $w = w_1 \cdots w_n \in \mathfrak{S}_n$ such that π_i is a path from source i to sink w_i. In this case, we say that π has *type* w. Further, we say that the path family *covers* F if it contains every edge exactly once.

Let F be a planar network of order n and let I, J be subsets of $[n]$ with $|I| = |J|$. Let $\Pi_{I,J}(F)$ denote the set of path families from sources indexed by I to sinks indexed by J. The following interpretation of the determinent is due to Karlin and McGregor [6] and usually attributed to Lindstrom [7].

Lemma 3.1. *For $I, J \subseteq [n]$ with $|I| = |J| = k$, we have*

$$\det(A_{I,J}) = \#\{\pi = (\pi_1, \ldots, \pi_k) \in \Pi_{I,J}(F) \,|\, \pi_1, \ldots, \pi_k \text{ pairwise nonintersecting } \}.$$

We have the following combinatorial interpretation of the permanent using path families.

Lemma 3.2. *For $I, J \subseteq [n]$ with $|I| = |J| = k$, we have* $\operatorname{per}(A_{I,J}) = |\Pi_{I,J}(F)|$.

There is a simple formula counting path families covering a descending star network.

Lemma 3.3. *Given descending star network $F = F_{[c_1,d_1]} \bullet \cdots \bullet F_{[c_r,d_r]}$, define $m_1, \ldots, m_r, k_1, \ldots, k_{r-1}$ by*

$$m_i = d_i - c_i + 1, \quad k_i = \max\{d_{i+1} - c_i + 1, 0\}. \tag{3.1}$$

Then we have

$$|\Pi_{[n],[n]}(F)| = \frac{m_1! \cdots m_{r-1}!}{k_1! \cdots k_{r-1}!} \cdot m_r!. \tag{3.2}$$

Let $h \in [n]$, then define sets $J = [h]$, $\bar{J} = [h+1, n]$, $I = [n] \cap 2\mathbb{Z}$, and $\bar{I} = [n] \smallsetminus 2\mathbb{Z}$. Lemma 3.3 and several factorial inequalities yield the following relation for path families.

Lemma 3.4. *Fix $h \in [n]$ and let $F = F_{[c_1,d_1]} \bullet \cdots \bullet F_{[c_r,d_r]}$ be a descending star network such that $F_{[c_p,d_p]}$ is the only star with $[c_p, d_p] \cap [h, h+1] \neq \emptyset$. We have*

$$|\Pi_{I,I}(F)| \cdot |\Pi_{\bar{I},\bar{I}}(F)| \leq |\Pi_{J,J}(F)| \cdot |\Pi_{\bar{J},\bar{J}}(F)|.$$

4 Merging stars in a star network

For a general descending star network F, there may be multiple stars $F_{[c_p,d_p]}$ with $[c_p, d_p] \cap [h, h+1] \neq \emptyset$. We now define a function that can modify the star network F. For each number $h \in [n]$, define the *h-merge function* $\phi_h : \mathscr{D}_n \to \mathscr{D}_n$ as follows.

1. Given $F = F_{[c_1,d_1]} \bullet \cdots \bullet F_{[c_r,d_r]}$, define

$$p = \max\{k \,|\, [c_k, d_k] \cap [h, h+1] \neq \emptyset\}, \quad q = \min\{k \,|\, [c_k, d_k] \cap [h, h+1] \neq \emptyset\}.$$

2. Define $\phi_h(F) = F_{[c_1,d_1]} \bullet \cdots \bullet F_{[c_{q-1},d_{q-1}]} \bullet F_{[c_p,d_q]} \bullet F_{[c_{p+1},d_{p+1}]} \bullet \cdots \bullet F_{[c_r,d_r]}$.

The h-merge function gives an equality on the cardinality of path families

Proposition 4.1. *Fix $h \in [n]$, and let F be a descending star network of order n with path matrix A. Let B be the path matrix of the descending star network $\phi_h(F)$. Then we have*

(i) $A_{[h],[h]} = B_{[h],[h]}$ *and* $A_{[h+1,n],[h+1,n]} = B_{[h+1,n],[h+1,n]}$.

(ii) $|\Pi_{[h],[h]}(F)| \cdot |\Pi_{[h+1,n],[h+1,n]}(F)| = |\Pi_{[h],[h]}(\phi_h(F))| \cdot |\Pi_{[h+1,n],[h+1,n]}(\phi_h(F))|$.

5 Main results

Now we state our main result about products of permanents.

Theorem 5.1. *Let A be the antiadjacency matrix of a unit interval order. Fix $h \in [n]$ and define sets*

$$J = [h], \qquad I = [n] \cap 2\mathbb{Z}.$$

Then we have

$$\operatorname{per}(A_{J,J}) \operatorname{per}(A_{\bar{J},\bar{J}}) \geq \operatorname{per}(A_{I,I}) \operatorname{per}(A_{\bar{I},\bar{I}}). \tag{5.1}$$

Proof omitted.

Theorem 5.2. *Let A be the antiadjacency matrix of a unit interval order. Let $I = [n] \cap 2\mathbb{Z}$. Then for any $i \in [n]$, we have*

$$\operatorname{per}(A_{[n]\setminus\{i\},[n]\setminus\{i\}}) \operatorname{per}(A_{\{i\},\{i\}}) \geq \operatorname{per}(A_{I,I}) \operatorname{per}(A_{\bar{I},\bar{I}}). \tag{5.2}$$

Proof omitted.

6 A Generalization on I

A natural generalization of a product of two permanents indexed by even or odd integers is a product of s permanents indexed by integers which are equivalent modulo s. Define

$$[c,d]_r^s = \{x \in [c,d] : \ x \equiv r \pmod{s}\}.$$

Next, we have a generalization of Lemma 3.3.

Lemma 6.1. *Let $F = F_{[c_0,d_0]} \bullet \cdots \bullet F_{[c_t,d_t]}$ be a descending star network. Fix $s \in [n]$. For every $r \in [0, s-1]$, We have*

$$|\Pi_{[1,n]_r^s,[1,n]_r^s}(F)| = \frac{|[c_0,d_0]_r^s|! \cdot |[c_1,d_1]_r^s|! \cdots |[c_t,d_t]_r^s|!}{|[c_0,d_1]_r^s|! \cdot |[c_1,d_2]_r^s|! \cdots |[c_{t-1},d_t]_r^s|!}. \tag{6.1}$$

We can now extend Theorem 5.1 to a more general result.

Theorem 6.2. *Let A be the antiadjacency matrix of an n-element unit interval order.*
Then for $s = 2, \ldots, n-1$, we have

$$\operatorname{per}(A_{[1,n]_0^s,[1,n]_0^s}) \cdots \operatorname{per}(A_{[1,n]_{s-1}^s,[1,n]_{s-1}^s}) \geq \operatorname{per}(A_{[1,n]_0^{s+1},[1,n]_0^{s+1}}) \cdots \operatorname{per}(A_{[1,n]_s^{s+1},[1,n]_s^{s+1}}). \tag{6.2}$$

Proof omitted.

7 A Generalization to all TNN matrices

In this section, we will show that (1.2) holds for $I = [1, \lfloor \frac{n}{2} \rfloor]$.

A polynomial $p(x_{11}, x_{12}, \cdots, x_{nn})$ in n^2 variables is called totally non-negative (TNN) if it satisfies

$$p(A) := p(a_{11}, a_{12}, \cdots, a_{nn}) \geq 0 \tag{7.1}$$

for each $n \times n$ totally non-negative matrix $A = (a_{ij})$. Therefore, (1.2) is true if and only if the polynomial

$$\operatorname{per}(X_{[1,\lfloor \frac{n}{2} \rfloor],[1,\lfloor \frac{n}{2} \rfloor]}) \operatorname{per}(X_{[\lceil \frac{n}{2} \rceil,n],[\lceil \frac{n}{2} \rceil,n]}) - \operatorname{per}(X_{[n]\cap 2\mathbb{Z},[n]\cap 2\mathbb{Z}}) \operatorname{per}(X_{[n]\setminus 2\mathbb{Z},[n]\setminus 2\mathbb{Z}}) \tag{7.2}$$

is TNN.

Drake, Gerrish and Skandera [4] gave a necessary and sufficient condition for a polynomial which is a difference of "permutation monomials" to be TNN.

Theorem 7.1 (Drake, Gerrish, Skandera)**.** *Let w and u be two permutations in $\mathfrak{S}_n$. Then w is less than or equal to u in the Bruhat order if and only if the polynomial*

$$x_{1,w_1} \cdots x_{n,w_n} - x_{1,u_1} \cdots x_{n,u_n}$$

is totally non-negative.

Let $t = \lfloor \frac{n}{2} \rfloor$. The term $\operatorname{per}(X_{\lfloor \frac{n}{2} \rfloor,\lfloor \frac{n}{2} \rfloor}) \operatorname{per}(X_{\lceil \frac{n}{2} \rceil,\lceil \frac{n}{2} \rceil})$ contains monomials $x_{1,w_1} \cdots x_{t,w_t} x_{t+1,w_{t+1}} \cdots x_{n,w_n}$ such that

$$w_1 \cdots w_t \ \text{ permutes } \{1, \ldots, t\}, \qquad w_{t+1} \cdots w_n \ \text{ permutes } \{t+1, \ldots, n\}.$$

For convenience, we set

$$\begin{aligned} \mathfrak{A}_n &= \mathfrak{S}_{1,\ldots,t} \times \mathfrak{S}_{t+1,\ldots,n} \\ &= \{w \in \mathfrak{S}_n \mid w_1 \cdots w_t \text{ permutes } \{1, \ldots, t\} \text{ and } w_{t+1} \cdots w_n \text{ permutes } \{t+1, \ldots, n\}\}. \end{aligned} \tag{7.3}$$

Also, for future reference, we define

$$\tilde{\mathfrak{A}}_n = \{w \in \mathfrak{S}_n \mid w_1 \cdots w_{t+1} \text{ permutes } \{1,\ldots,t+1\} \text{ and } w_{t+2}\cdots w_n \text{ permutes } \{t+2,\ldots,n\}\}. \quad (7.4)$$

It is easy to see that $|\mathfrak{A}_n| = \lfloor \frac{n}{2} \rfloor ! \lceil \frac{n}{2} \rceil !$. For example, $\mathfrak{A}_4 = \{1234, 1243, 2134, 2143\}$. The product of permanents $\mathrm{per}(X_{[n]\cap 2\mathbb{Z},[n]\cap 2\mathbb{Z}})\,\mathrm{per}(X_{[n]\setminus 2\mathbb{Z},[n]\setminus 2\mathbb{Z}})$ contains monomials $x_{1,u_1}\cdots x_{n,u_n}$ such that

$$u_1u_3u_5\cdots \text{ permutes } \{1,3,5,\cdots\} = [n]\setminus 2\mathbb{Z}, \quad u_2u_4u_6\cdots \text{ permutes } \{2,4,6,\cdots\} = [n]\cap 2\mathbb{Z}.$$

We call the set that consists of $u = u_1\cdots u_n$ satisfying the above conditions $\mathfrak{B}_n$, i,e,

$$\mathfrak{B}_n = \{u \in \mathfrak{S}_n \mid u_1u_2\cdots u_n \text{ takes odd and even integers alternately and } u_1 \text{ is odd}\} \quad (7.5)$$

It is easy to see that $|\mathfrak{B}_n| = \lfloor \frac{n}{2} \rfloor ! \lceil \frac{n}{2} \rceil !$. Then, the polynomial (7.2) has this number of positive terms (on the left) and negative terms (on the right). Therefore, if we can find a bijective map $f_n : \mathfrak{A}_n \to \mathfrak{B}_n$, such that $w \le f_n(w)$ for all $w \in \mathfrak{A}_n$, by Theorem 7.1, the polynomial (7.2) is TNN.

Next, we introduce a few necessary maps. For $p,q \in [n]$, define an insertion map ins_p and an insertion-swapping map inss_q (when $n+1-q$ is even) given by

$$\begin{aligned} \mathrm{ins}_p &: \mathfrak{S}_n \to \mathfrak{S}_{n+1}, \\ w = w_1\cdots w_p\cdots w_n &\mapsto w_1\cdots w_{p-1}(n+1)w_p\cdots w_n, \\ \mathrm{inss}_q &: \mathfrak{S}_n \to \mathfrak{S}_{n+1}, \\ w = w_1\cdots w_q\cdots w_n &\mapsto w_1\cdots w_{q-1}(n+1)w_{q+1}w_qw_{q+3}w_{q+2}\cdots w_nw_{n-1}. \end{aligned}$$

For example, if $w = 216435$, then $\mathrm{ins}_3(w) = 2176435$, $\mathrm{inss}_3(w) = 2174653$, but $\mathrm{inss}_4(w)$ is not well-defined since $n+1-q = 7-4 = 3$ is not even.

For $w \in \mathfrak{S}_n$, define the inversion statistic by $\mathrm{INV}(w) := |\{(w_i,w_j) \mid i<j \text{ and } w_i > w_j\}|$. Observe that $|\mathfrak{A}_n| = |\tilde{\mathfrak{A}}_n| = \lfloor \frac{n}{2} \rfloor ! \lceil \frac{n}{2} \rceil !$. Then there exists a bijective map between $\mathfrak{A}_n$ and $\tilde{\mathfrak{A}}_n$. Let $w = w_1\cdots w_n \in \mathfrak{S}_n$. Define two maps U and R given by

$$\begin{aligned} U &: \mathfrak{S}_n \to \mathfrak{S}_n, \\ &\quad w \mapsto w^U := (n+1-w_1)(n+1-w_2)\cdots(n+1-w_n), \\ R &: \mathfrak{S}_n \to \mathfrak{S}_n, \\ &\quad w \mapsto w^R := w_n\cdots w_1. \end{aligned}$$

We abbreviate $U\circ R = R\circ U = RU$. This induces bijective maps between $\mathfrak{A}_n$ and $\tilde{\mathfrak{A}}_n$, and $\mathfrak{B}_n$ to itself.

Proposition 7.2. *The map $U\circ R : \mathfrak{S}_n \to \mathfrak{S}_n$ sending $w \mapsto w^{UR}$ restricts to a bijection between $\mathfrak{A}_n$, $\tilde{\mathfrak{A}}_n$, and to an involution on $\mathfrak{B}_n$.*

The map $U\circ R$ preserves the Bruhat order.

Proposition 7.3. *Let $v,\ w \in \mathfrak{S}_n$ with $v \le w$. Then, $v^{RU} \le w^{RU}$.*

Next, we define a map $\tilde{f}_n : \tilde{\mathfrak{A}}_n \to \mathfrak{B}_n$. Let $\tilde{w} \in \tilde{\mathfrak{A}}_n$. Define $\tilde{f}_n$ by

$$\begin{aligned} \tilde{f}_n &: \tilde{\mathfrak{A}}_n \to \mathfrak{B}_n, \\ &\quad \tilde{w} \mapsto [f_n(\tilde{w}^{RU})]^{RU}. \end{aligned} \quad (7.6)$$

We can show that $\tilde{f}_n$ is bijective if f_n is bijective, and many properties of f_n are inherited in $\tilde{f}_n$.

Lemma 7.4. *Let $f_n : \mathfrak{A}_n \to \mathfrak{B}_n$ be a bijection such that $w \leq f_n(w)$ for all $w \in \mathfrak{A}_n$. Then, the bijection $\tilde{f}_n$ (7.6) satisfies $\tilde{w} \leq \tilde{f}_n(\tilde{w})$ for every $\tilde{w} \in \tilde{\mathfrak{A}}_n$.*

Now, we define $f_n : \mathfrak{A}_n \to \mathfrak{B}_n$ inductively.

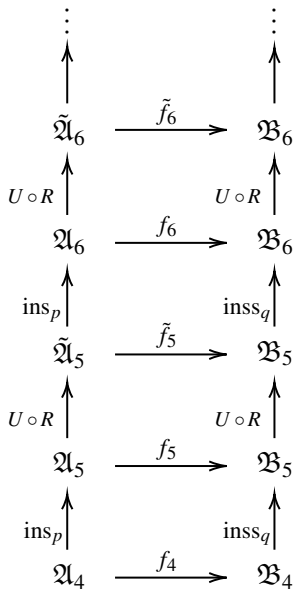
Algorithm 7.5. Let $\mathfrak{A}_n$, $\tilde{\mathfrak{A}}_n$, $\mathfrak{B}_n$ be defined as above. and let f_4 be the map

$$1234 \longmapsto 1234 \quad 1243 \longmapsto 1432 \quad 2134 \longmapsto 3214 \quad 2143 \longmapsto 3412. \tag{7.7}$$

For $n = 5, 6, 7, \ldots$, do

1. If n is odd, do
   a. Write $n - 1 = 2k$.
   b. For each $w \in \mathfrak{A}_n$, find the unique pair $(a, p) \in \mathfrak{A}_{n-1} \times \mathbb{N}$ such that $w = \mathrm{ins}_p(a)$.
   c. Define $f_n(w) = \mathrm{inss}_{2(p-k)-1}(f_{n-1}(a))$.
2. If n is even, do
   a. Write $n - 1 = 2k + 1$.
   b. Define $\tilde{f}_{n-1} : \tilde{\mathfrak{A}}_n \to \mathfrak{B}_n$ by f_{n-1} as in (7.6).
   c. For each $u \in \mathfrak{A}_n$, $(\tilde{w}, q) \in \tilde{\mathfrak{A}}_n \times \mathbb{N}$ such that $u = \mathrm{ins}_q(\tilde{w})$.
   d. Define $f_n(u) = \mathrm{inss}_{2(q-k-1)}(\tilde{f}_{n-1}(\tilde{w}))$.

We illustrate the inductive definition of f_n in Algorithm 7.5 with the following diagram.



Our algorithm behaves nicely with respect to inversion and the Bruhat order. We have the following partial results and conjecture for the function f_n defined by Algorithm 7.5.

Theorem 7.6. *For all $w \in \mathfrak{A}_n \setminus \{e\}$, we have* $\mathrm{INV}(w) < \mathrm{INV}(f_n(w))$.

Proof omitted.

Theorem 7.7. *For $n \leq 13$ and all $w \in \mathfrak{A}_n$, we have $w \leq f_n(w)$ in the Bruhat order.*

Proof omitted.

Conjecture 7.8. *For all n and all $w \in \mathfrak{A}_n$ we have $w \leq f_n(w)$ in the Bruhat order.*

A proof of Conjecture 7.8 would extend Theorem 5.1 to all totally nonnegative matrices.

References

[1] D. Carlson (1967): *Weakly sign-symmetric matrices and some determinantal inequalities*. Colloq. Math. 17, pp. 123–129, doi:10.4064/cm-17-1-123-129.

[2] S. Clearman, M. Hyatt, B. Shelton & M. Skandera (2013): *Evaluations of Hecke algebra traces at Kazhdan-Lusztig basis elements*. In: *25th International Conference on Formal Power Series and Algebraic Combinatorics (FPSAC 2013)*, Discrete Math. Theor. Comput. Sci. Proc., AU, Assoc. Discrete Math. Theor. Comput. Sci., Nancy, pp. 1119–1130.

[3] S. Clearman, M. Hyatt, B. Shelton & M. Skandera (2016): *Evaluations of Hecke algebra traces at Kazhdan-Lusztig basis elements*. Electron. J. Combin. 23(2), doi:10.37236/5021. Paper 2.7, 56 pages.

[4] B. Drake, S. Gerrish & M. Skandera (2004): *Two new criteria for comparison in the Bruhat order*. Electron. J. Combin. 11(1), doi:10.37236/1847. Note 6, 4 pp. (electronic).

[5] P. C. Fishburn (1985): *Interval Graphs and Interval Orders*. Wiley, New York.

[6] S. Karlin & G. McGregor (1959): *Coincidence probabilities*. Pacific J. Math. 9, pp. 1141–1164. Available at http://projecteuclid.org/euclid.pjm/1103038889.

[7] B. Lindström (1973): *On the vector representations of induced matroids*. Bull. London Math. Soc. 5, pp. 85–90, doi:10.1112/blms/5.1.85.

[8] M. Marcus (1963): *The permanent analogue of the Hadamard determinant theorem*. Bulletin of the American Mathematical Society 69(4), pp. 494–496, doi:10.1090/S0002-9904-1963-10975-1.

[9] M. Skandera (2004): *Inequalities in products of minors of totally nonnegative matrices*. J. Algebraic Combin. 20(2), pp. 195–211, doi:10.1023/B:JACO.0000047282.21753.ae.

[10] M. Skandera (2008): *On the dual canonical and Kazhdan-Lusztig bases and 3412, 4231-avoiding permutations*. J. Pure Appl. Algebra 212, doi:10.1016/j.jpaa.2007.09.007.

[11] R. Stanley (1999): *Enumerative Combinatorics*. 2, Cambridge University Press, Cambridge, doi:10.1017/CBO9780511609589.

[12] R. Stanley (2000): *Positivity problems and conjectures in algebraic combinatorics*. In V. Arnold, M. Atiyah, P. Lax & B. Mazur, editors: *Mathematics: Frontiers and Perspectives*, American Mathematical Society, Providence, RI, pp. 295–319.

[13] J. Stembridge (1991): *Immanants of totally positive matrices are nonnegative*. Bull. London Math. Soc. 23, pp. 422–428, doi:10.1112/blms/23.5.422.

[14] J. Stembridge (1992): *Some conjectures for immanants*. Canad. J. Math. 44(5), pp. 1079–1099, doi:10.4153/CJM-1992-066-1.

[15] W. T. Trotter (1992): *Combinatorics and Partially Ordered Sets: Dimension Theory*. Johns Hopkins University Press, Baltimore.

On the Impossibility of Parabolic Factorization of certain Kazhdan-Lusztig Basis Elements

Tommy Parisi Mark Skandera Ben Spahiu Jiayuan Wang
Lehigh University, Bethlehem PA, USA
tjp225@lehigh.edu mas906@lehigh.edu bes226@lehigh.edu jiw922@lehigh.edu

For w in the symmetric group $\mathfrak{S}_n$, let $\widetilde{C}_w$ be the corresponding modified, signless Kazhdan-Lusztig basis element of the type-A Hecke algebra $H_n(q)$. An extension [*Ann. Comb.* **25**, no. 3 (2021) pp. 757–787] of a result of Deodhar [*Geom. Dedicata* **36**, (1990) pp. 95–119] implies that any factorization of the form $\widetilde{C}_w = \frac{1}{f(q)}\widetilde{C}_{v^{(1)}} \cdots \widetilde{C}_{v^{(r)}}$, with $v^{(1)}, \ldots, v^{(r)}$ maximal elements of parabolic subgroups of $\mathfrak{S}_n$ and $f(q) \in \mathbf{N}[q]$ depending on these, provides cancellation-free combinatorial interpretations of the polynomials $\{P_{v,w}(q) \,|\, v \in \mathfrak{S}_n\}$ appearing in the expansion $\sum_v P_{v,w}(q)T_v$ of $\widetilde{C}_w$ in terms of the natural basis $\{T_v \,|\, v \in \mathfrak{S}_n\}$ of $H_n(q)$. While the set of permutations $w \in \mathfrak{S}_n$ admitting such a factorization of $\widetilde{C}_w$ has not yet been characterized, we apply a result of Gaetz–Gao [*Adv. Math.* **457** (2024) Paper No. 109941] to describe a set for which such a factorization cannot exist.

1 Introduction

The *Kazhdan–Lusztig polynomials* $\{P_{v,w}(q) \,|\, v,w \in \mathfrak{S}_n\} \subset \mathbf{N}[q]$ are entries of the change-of-basis matrix relating a certain *Kazhdan–Lusztig basis* of the Hecke algebra with another *natural basis.* First appearing in the study of representations of the Hecke algebra, they were given existential and recursive definitions in [19]. Appearances of the polynomials in other areas such as Lie Theory [1], [2], [9], quantum groups [14], and Schubert varieties [19], [20] have inspired a search for simpler descriptions. Ideally, such a description should interpret each coefficient of $P_{v,w}(q)$ as a set cardinality.

Some famous alternative formulas for the Kazhdan–Lusztig polynomials are due to Brenti and Deodhar. Brenti expressed $P_{v,w}(q)$ in two different ways as simple linear combinations of recursively defined polynomials in $\mathbf{Z}[q]$ having both positive and negative coefficients [7, §3], [8, §3]. Because of negative coefficients and recursive definitions, these formulas do not interpret coefficients in $P_{v,w}(q)$ as set cardinalities. Deodhar [13] developed an algorithm which takes any reduced expression for w as an input, and outputs a set $\mathscr{E}_{\min}$ of (not necessarily reduced) expressions for other permutations in $\mathfrak{S}_n$. For each $v \in \mathfrak{S}_n$ and $k > 0$, the coefficient of q^k in $P_{v,w}(q)$ is equal to the cardinality of a certain subset of $\mathscr{E}_{\min}$. On the other hand, the algorithmic component of Deodhar's method makes it difficult to apply his combinatorial interpretation in practice.

Billey and Warrington showed [5, Thm. 1, Rmk. 6] that when w has certain properties, Deodhar's algorithm is trivial, and the output set $\mathscr{E}_{\min}$ of expressions can be replaced by a more visually appealing set of path families in a certain wiring diagram. Again for each v and k, the coefficient of q^k in $P_{v,w}(q)$ is equal to the cardinality of a subset of these path families. Clearwater and the second author [11, Cor. 5.3] then extended this result to permutations w for which the Kazhdan–Lusztig basis element $\widetilde{C}_w$ factors nicely, but did not solve the problem [23, Quest. 4.5] of characterizing such permutations w.

In Sections 2 – 3 we review basic facts about the symmetric group, planar networks, the Hecke algebra, and the Kazhdan–Lusztig basis and polynomials. In Section 4 we use the result [11, Cor. 5.3] to state properties of polynomials which arise in the natural expansion of products of certain Kazhdan–Lusztig

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 148–157, doi:10.4204/EPTCS.445.18

© T. Parisi, M. Skandera, B. Spahiu, & J. Wang
This work is licensed under the
Creative Commons Attribution License.

basis elements of the Hecke algebra. This leads to a partial answer in Section 5 to the characterization question [23, Quest. 4.5]: a description of certain Kazhdan–Lusztig basis elements which do not factor as desired.

2 The symmetric group and planar networks

Let $\mathfrak{S}_n$ be the symmetric group, with standard generators $s_1,\dots,s_{n-1}$, length function ℓ, and Bruhat order $\leq$. (See, e.g., [6] for definitions.) Given a word $u = u_1\cdots u_k$ in $\mathfrak{S}_k$, and a word $y = y_1\cdots y_k$ having k distinct letters, we say that y *matches the pattern* u if the letters of y appear in the same relative order as those of u; that is, if we have $u_i < u_j$ if and only if $y_i < y_j$ for all $i,j \in [k] := \{1,\dots,k\}$. On the other hand, say that $w \in \mathfrak{S}_n$ *avoids the pattern* u if no subword y of w matches the pattern u. For example, the subword 423 of the permutation 4231 matches the pattern 312; thus 4231 does not avoid the pattern 312.

It is easy to see that for each subinterval $[a,b] := \{a,\dots,b\}$ of $[n]$, the *reversal*

$$s_{[a,b]} := 1\cdots(a-1)b\cdots a(b+1)\cdots n \in \mathfrak{S}_n \tag{1}$$

avoids the patterns 3412 and 4231. This element is the unique longest (maximum length) element of the subgroup of $\mathfrak{S}_n$ generated by $s_a,\dots,s_{b-1}$. More generally, each *parabolic* subgroup of $\mathfrak{S}_n$ generated by a subset of generators has longest element equal to a product of reversals on disjoint intervals. Multiplication of reversals in $\mathfrak{S}_n$ or of related elements

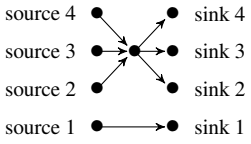
$$D_{[a,b]} := \sum_{v \leq s_{[a,b]}} v \tag{2}$$

in $\mathbf{Z}[\mathfrak{S}_n]$ can be performed graphically with certain planar networks.

Define a *planar network of order n* to be a directed, planar, acyclic multigraph with $2n$ boundary vertices having n source vertices on the left and n sink vertices on the right, both labeled $1,\dots,n$ from bottom to top. We will allow edges (x,y) to be marked by a positive integer multiplicity $m(x,y)$. Let $\mathscr{F}_n$ denote the set of such networks. For each subinterval $[a,b]$ of $[n]$ we define a *simple star network* $F_{[a,b]} \in \mathscr{F}_n$ by

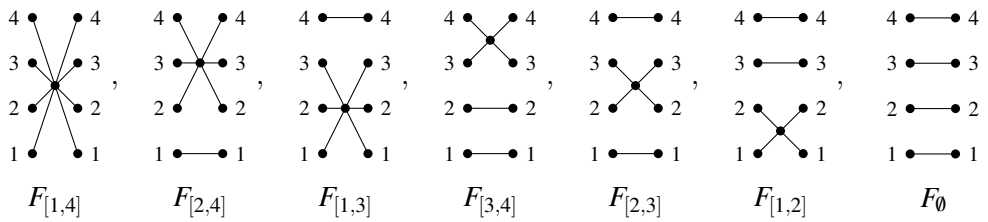
1. an interior vertex x lies between the sources and sinks,
2. for $i \in [a,b]$ we have edges from source i to x and from x to sink i,
3. for $i \notin [a,b]$ we have edges from source i to sink i.

For example, the simple star network $F_{[2,4]} \in \mathscr{F}_4$ is



. (3)

For economy, we will omit edge orientations and the words "source" and "sink" from figures. Thus the seven simple star networks in $\mathscr{F}_4$ are



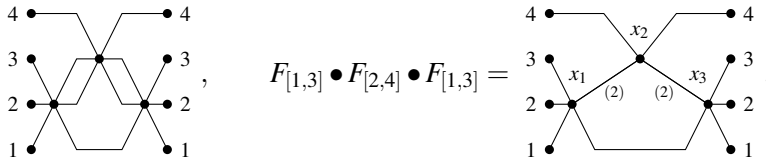
. (4)

Given networks $E, F \in \mathscr{F}_n$, in which all sources have outdegree 1 and all sinks have indegree 1, define the concatenation $E \circ F$ of E and F as follows. For $i = 1, \ldots, n$, do

1. remove sink i of E and source i of F,
2. merge each edge $(x, \text{sink } i)$ in E with each edge $(\text{source } i, y)$ in F to form a single edge (x,y) in $E \circ F$.

Thus a concatenation of the form $F_{[a_1,b_1]} \circ \cdots \circ F_{[a_m,b_m]} \in \mathscr{F}_n$ has $2n+m$ edges: n sources inherited from $F_{[a_1,b_1]}$, n sinks inherited from $F_{[a_m,b_m]}$, and m internal vertices $x_1, \ldots, x_m$, where x_j is inherited from $F_{[a_j,b_j]}$. Sometimes in a concatenation $E \circ F$, there may exist internal vertices x in E, y in F with $m(x,y) > 1$ multiplicity-1 edges incident upon both. Define the *condensed concatenation* $E \bullet F$ to be the subdigraph of $E \circ F$ obtained by removing, for all such pairs (x,y), all but one of the $m(x,y)$ edges incident upon both, and by marking this edge with the multiplicity $m(x,y)$. For example, in $\mathscr{F}_4$ we have the graphs

$$F_{[1,3]} \circ F_{[2,4]} \circ F_{[1,3]} = \quad , \qquad F_{[1,3]} \bullet F_{[2,4]} \bullet F_{[1,3]} = \quad , \tag{5}$$



in which the two multiplicity-2 edges (x_1,x_2), (x_2,x_3) of $F_{[1,3]} \bullet F_{[2,4]} \bullet F_{[1,3]}$ are the remnants of pairs of edges incident upon the same internal vertices in $F_{[1,3]} \circ F_{[2,4]} \circ F_{[1,3]}$.

Define a *star network* to be an element of $\mathscr{F}_n$ constructed by concatenation or condensed concatenation of simple star networks. Let $\mathscr{F}_n^\bullet$ denote the subset of $\mathscr{F}_n$ consisting of condensed concatenations of finitely many simple star networks. Call a sequence $\pi = (\pi_1, \ldots, \pi_n)$ of source-to-sink paths in a star network $F \in \mathscr{F}_n^\bullet$ a *path family of type* $v = v_1 \cdots v_n \in \mathfrak{S}_n$ if for all i, path π_i begins at source i and terminates at sink v_i. Say that π *covers* F if each edge (x_i,x_j) of F belongs to $m(x_i,x_j)$ of the paths in π, and define the sets

$$\begin{aligned} \Pi(F) &= \{\pi \,|\, \pi \text{ a path family covering } F\}, \\ \Pi_v(F) &= \{\pi \in \Pi(F) \,|\, \text{type}(\pi) = v\}. \end{aligned} \tag{6}$$

In terms of the definitions (6), we may combinatorially interpret products of elements (2) quite simply. We say that F *graphically represents*

$$\sum_{v \in \mathfrak{S}_n} |\Pi_v(F)|\, v \tag{7}$$

as an element of $\mathbf{Z}[\mathfrak{S}_n]$. For $F = F_{[a_1,b_1]} \circ \cdots \circ F_{[a_m,b_m]}$, this element is $D_{[a_1,b_1]} \cdots D_{[a_m,b_m]}$; for $F = F_{[a_1,b_1]} \bullet \cdots \bullet F_{[a_m,b_m]}$, it is $D_{[a_1,b_1]} \cdots D_{[a_m,b_m]}$ divided by the product, over all edges (x_i,x_j), of the numbers $m(x_i,x_j)!$.

3 The Hecke algebra and planar networks

Define the *(type-A Iwahori–) Hecke algebra* $H_n(q)$ to be the $\mathbf{Z}[q^{\frac{1}{2}}, q^{-\frac{1}{2}}]$-span of its *natural basis* $\{T_w \,|\, w \in \mathfrak{S}_n\}$, with multiplication given by $T_e T_w = T_w$ where $e = 1 \cdots n$ is the identity element of $\mathfrak{S}_n$, and

$$T_{s_i} T_w = \begin{cases} T_{s_i w} & \text{if } s_i w > w, \\ (q-1) T_{s_i w} + q T_w & \text{if } s_i w < w. \end{cases}$$

Specializing at $q^{\frac{1}{2}} = 1$ we have $H_n(1) \cong \mathbf{Z}[\mathfrak{S}_n]$ with $T_w \mapsto w$.

A semilinear involution on $H_n(q)$, known as the *bar involution*, is defined by

$$\overline{q^{\frac{1}{2}}} = q^{\frac{-1}{2}}, \qquad \overline{T_w} = (T_{w^{-1}})^{-1}, \qquad \overline{\sum_{w\in\mathfrak{S}_n} B_w(q)T_w} = \sum_{w\in\mathfrak{S}_n} \overline{B_w(q)}\,\overline{T_w}.$$

Kazhdan and Lusztig showed [19] that $H_n(q)$ has a unique basis $\{C'_w \,|\, w \in \mathfrak{S}_n\}$ satisfying $\overline{C'_w} = C'_w$ and

$$q^{\frac{\ell(w)}{2}} C'_w = \sum_{v\leq w} P_{v,w}(q)T_w, \tag{8}$$

where coefficients $P_{v,w}(q) \in \mathbf{Z}[q]$ satisfy $\deg(P_{v,w}(q)) < \frac{\ell(w)-\ell(v)-1}{2}$ for $v < w$, and $P_{w,w}(q) = 1$ for all w. It is known that these *Kazhdan–Lusztig polynomials* satisfy $P_{v,w}(q) \in \mathbf{N}[q]$, and $P_{v,w}(0) = 1$ for $v \leq w$. We also have [21] that if w avoids the patterns 3412 and 4231, then $P_{v,w}(q) = 1$ for all $v \leq w$. For convenience, we define

$$\widetilde{C}_w := q^{\frac{\ell(w)}{2}} C'_w. \tag{9}$$

Kazhdan–Lusztig basis elements and their products appear in various settings, including intersection homology [3], [25], algorithmic and combinatorial description of Kazhdan–Lusztig basis elements themselves [5], [13], Schubert varieties [5], total nonnegativity [16], [23], [27], [28], trace evaluations [10], [11], [17], [18], [24], and chromatic symmetric functions [10], [24].

Deodhar [13, Prop. 3.5] studied sequences $(s_{i_1}, \ldots, s_{i_k})$ of generators of $\mathfrak{S}_n$, products of the corresponding Kazhdan–Lusztig basis elements $\widetilde{C}_{s_{i_j}} = T_e + T_{s_{i_j}}$ of $H_n(q)$, and their natural expansions

$$\widetilde{C}_{s_{i_1}} \cdots \widetilde{C}_{s_{i_m}} = \sum_{v\in\mathfrak{S}_n} A_v(q)T_v. \tag{10}$$

He described the coefficients $\{A_v(q) \,|\, v \in \mathfrak{S}_n\} \subset \mathbf{Z}[q]$ in terms of *subexpressions* of $(s_{i_1}, \ldots, s_{i_m})$, sequences $\sigma = (\sigma_1, \ldots, \sigma_m)$ with $\sigma_j \in \{e, s_{i_j}\}$ for $j = 1, \ldots, m$. (Our treatment here differs slightly from that of [13] but is equivalent.) Call index j a *defect* of σ if

$$\sigma_1 \cdots \sigma_{j-1} s_{i_j} < \sigma_1 \cdots \sigma_{j-1} \tag{11}$$

and let $\mathrm{dfct}(\sigma)$ denote the number of defects of σ. (Observe that $j = 1$ cannot be a defect: we have $s_{i_1} > e$ always.) Each coefficient on the right-hand side of (10) is given by

$$A_v(q) = \sum_{\sigma} q^{\mathrm{dfct}(\sigma)}, \tag{12}$$

where the sum is over all subexpressions σ of $(s_{i_1}, \ldots, s_{i_m})$ satisfying $\sigma_1 \cdots \sigma_m = v$.

Billey and Warrington observed [5, Rmk. 6] that the defect statistic has a simple graphical interpretation. Specifically, subexpressions of $(s_{i_1}, \ldots, s_{i_m})$ correspond bijectively to path families covering

$$F = F_{[i_1, i_1+1]} \bullet \cdots \bullet F_{[i_m, i_m+1]} \tag{13}$$

in $\mathscr{F}_n^\bullet$ with $(\sigma_1, \ldots, \sigma_m)$ corresponding to the family $\pi \in \Pi(F)$ constructed by prescribing

$$\text{the paths meeting at } x_j \begin{cases} \text{cross there} & \text{if } \sigma_j = s_{i_j}, \\ \text{do not cross there} & \text{if } \sigma_j = e. \end{cases}$$

By this bijection, index j is a defect of σ in the sense of (11) if and only if the paths meeting at x_j have previously crossed an odd number of times.

Clearwater–Skandera extended this result [11, Cor. 5.3] to products of the form

$$\widetilde{C}_{s_{[a_1,b_1]}}\cdots\widetilde{C}_{s_{[a_m,b_m]}} = \sum_{v\in\mathfrak{S}_n} A_v(q)T_v, \quad \text{where} \quad \widetilde{C}_{s_{[a_j,b_j]}} = \sum_{u\leq s_{[a_j,b_j]}} T_u, \tag{14}$$

since reversals avoid the patterns 3412 and 4231. This extension requires a more general definition of defects. While the intersection of two paths in (13) is a union of vertices, the intersection of two paths in

$$F = F_{[a_1,b_1]}\bullet\cdots\bullet F_{[a_m,b_m]} \tag{15}$$

is a subgraph of F whose connected components are vertices or paths of the form

$$(x_k,\ldots,x_\ell) \tag{16}$$

for some $k<\ell$. For each initial vertex x_k in a component (16) of the intersection of two paths, we will say that the paths *meet* at x_k. Our embedding of star networks in the plane naturally allows us to declare an edge entering (exiting) a vertex x_k to be above or below another edge entering (exiting) x_k. We will call a component (16) in the intersection of paths π_i, π_j, a *crossing* of π_i and π_j if the two paths enter x_k and exit x_ℓ in different orders. Extending the Billey–Warrington definition of defect to accommodate three or more paths passing through a vertex, we have the following [11, §5].

Definition 3.1. *Given a path family π covering $F = F_{[a_1,b_1]}\bullet\cdots\bullet F_{[a_m,b_m]}$, define a* defect *of π at x_k to be a triple (π_i,π_j,k) with $i<j$ and π_i and π_j meeting at x_k after having crossed an odd number of times. Define* $\mathrm{dfct}(\pi)$ *to be the number of defects of π.*

Extending the interpretation (7) of a planar network, we define the set

$$\Pi_{v,d}(F) = \{\pi\in\Pi_v(F)\,|\,\mathrm{dfct}(\pi)=d\} \tag{17}$$

and we say that F *graphically represents the element*

$$\sum_{v\in\mathfrak{S}_n}\sum_{d\geq 0}|\Pi_{v,d}(F)|q^dT_v = \sum_{\pi\in\Pi(F)} q^{\mathrm{dfct}(\pi)}T_{\mathrm{type}(\pi)} \tag{18}$$

as an element of $H_n(q)$. For $F = F_{[a_1,b_1]}\circ\cdots\circ F_{[a_m,b_m]}$, this element is $\widetilde{C}_{s_{[a_1,b_1]}}\cdots\widetilde{C}_{s_{[a_m,b_m]}}$; for $F = F_{[a_1,b_1]}\bullet\cdots\bullet F_{[a_m,b_m]}$, it is $\widetilde{C}_{s_{[a_1,b_1]}}\cdots\widetilde{C}_{s_{[a_m,b_m]}}$, divided by the product, over all edges (x_i,x_j), of the q-factorial polynomials $m(x_i,x_j)_q!$. (See, e.g., [26].)

The denominator above can also be expressed in terms of intervals appearing in reversals. Given a sequence of m intervals

$$\mathscr{A} = ([a_1,b_1],\ldots,[a_m,b_m]), \tag{19}$$

define $\binom{m}{2}$ more intervals $\{I_{i,j}\,|\,i<j\}$ by

$$I_{i,j} = [a_i,b_i]\cap[a_j,b_j]\setminus([a_{i+1},b_{i+1}]\cup\cdots\cup[a_{j-1},b_{j-1}]). \tag{20}$$

Let $f_{\mathscr{A}}(q)$ be the product of the q-factorials of the cardinalities of the intervals (20),

$$f_{\mathscr{A}}(q) := \prod_{i<j}|I_{i,j}|_q!. \tag{21}$$

Say that a Kazhdan–Lusztig basis element $\widetilde{C}_w$ has a *parabolic factorization* if there is a sequence (19) of intervals satisfying

$$\widetilde{C}_w = \frac{1}{f_{\mathscr{A}}(q)} \widetilde{C}_{s_{[a_1,b_1]}} \cdots \widetilde{C}_{s_{[a_m,b_m]}}. \tag{22}$$

Two results on parabolic factorization are the following [5, Thm. 1], [23, Thm. 4.3].

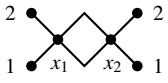
Theorem 3.2. *If $w \in \mathfrak{S}_n$ avoids the patterns* 321, 56781234, 46781235, 56718234, 46718235, *and if $s_{i_1} \cdots s_{i_\ell}$ is a reduced expression for w, then we have $\widetilde{C}_w = \widetilde{C}_{s_{i_1}} \cdots \widetilde{C}_{s_{i_\ell}}$.*

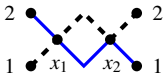
Theorem 3.3. *If $w \in \mathfrak{S}_n$ avoids the patterns* 3412 *and* 4231, *then there exists a sequence* (19) *of intervals such that we have the factorization* (22).

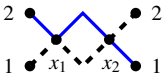
The combination of Theorems 3.2, 3.3 is not the strongest result possible. Indeed, the known parabolic factorization $\widetilde{C}_{4231} = \widetilde{C}_{s_{[1,2]}} \widetilde{C}_{s_{[2,4]}} \widetilde{C}_{s_{[1,2]}}$ is guaranteed by neither theorem.

4 Reduction of defects

Our defect reduction theorem asserts that if a star network can be covered by a path family having d defects, then it can also be covered by a path family of the same type having $d-1$ defects. In certain simple cases, we can easily produce a $(d-1)$-defect family from a d-defect family by swapping a pair of subpaths. For example, consider the star network and path families

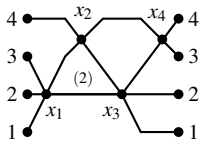
$F_{[1,2]} \circ F_{[1,2]} =$ 

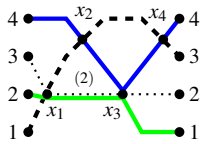
, $\pi =$ 

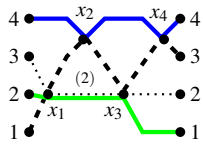
, $\sigma =$ 

,

with $\mathrm{dfct}(\pi) = 1$, σ constructed from π by swapping the two x_1-to-x_2 subpaths of π, and $\mathrm{dfct}(\sigma) = 0$. On the other hand, this simple procedure does not always reduce defects by 1. Consider the star network and path families of type 3124

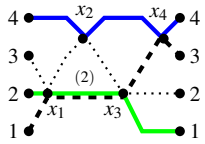
$F_{[1,3]} \bullet F_{[3,4]} \bullet F_{[1,3]} \bullet F_{[3,4]} =$ 

, $\pi =$ 

, $\sigma =$ 

,

with $\mathrm{dfct}(\pi) = 1$, and σ constructed from π by swapping the x_2-to-x_4 subpaths of π_1 and π_4. The swap eliminates the defect $(\pi_1, \pi_4, 4)$, but introduces two more: $(\sigma_1, \sigma_2, 3)$, $(\sigma_1, \sigma_3, 3)$. Thus we have $\mathrm{dfct}(\sigma) = 2$. There is in fact a type-3124 path family

$\tau =$ 

satisfying $\mathrm{dfct}(\tau) = \mathrm{dfct}(\pi) - 1 = 0$, but the naive approach above does not produce it from π.

To describe the process of reducing defects in a path family by exactly 1, we begin by stating a map which modifies a path family by removing a defect at a specified vertex, possibly creating earlier defects. For $F = F_{[a_1,b_1]} \bullet \cdots \bullet F_{[a_m,b_m]}$ and $k \in \{2, \ldots, m\}$, define

$$\begin{aligned} \phi_k : \{\pi \in \Pi(F) \,|\, \pi \text{ has a defect at } x_k\} &\to \Pi(F) \\ \pi &\mapsto \hat{\pi} \end{aligned} \tag{23}$$

by the following algorithm.

Algorithm 4.1. *Given star network $F = F_{[a_1,b_1]} \bullet \cdots \bullet F_{[a_m,b_m]} \in \mathscr{F}_n^\bullet$, path family $\pi \in \Pi(F)$, and index k such that π has a defect at x_k, do*

1. *Let (r,t) be the lexicographically least pair such that (π_r, π_t, k) is a defect.*
2. *Let s be the largest index such that π_s enters vertex x_k through the same edge as π_r and (π_s, π_t, k) is a defect.*
3. *Let x_l be the final vertex in the unique crossing of π_s and π_t prior to x_k.*
4. *Create a new path family $\hat{\pi} = (\hat{\pi}_1, \ldots, \hat{\pi}_n)$ by*
   (a) *$\hat{\pi}_i = \pi_i$ if $i \notin \{s,t\}$.*
   (b) *$\hat{\pi}_s$ is π_s with the x_l-to-x_k subpath replaced by that of π_t.*
   (c) *$\hat{\pi}_t$ is π_t with the x_l-to-x_k subpath replaced by that of π_s.*

Proposition 4.2. *Algorithm 4.1 produces a path family $\phi_k(\pi) = \hat{\pi}$ that satisfies*

1. $\mathrm{type}(\hat{\pi}) = \mathrm{type}(\pi)$,
2. *for each $p > k$, we have $\{(i,j) \,|\, (\hat{\pi}_i, \hat{\pi}_j, p)$ defective$\} = \{(i,j) \,|\, (\pi_i, \pi_j, p)$ defective$\}$,*
3. *$\#\{(i,j) \,|\, (\hat{\pi}_i, \hat{\pi}_j, k)$ defective$\} = \#\{(i,j) \,|\, (\pi_i, \pi_j, k)$ defective$\} - 1$.*

Proof. Omitted. □

We may further modify the path family $\hat{\pi}$ to produce a path family having no defects at $x_1, \ldots, x_{k-1}$.

Proposition 4.3. *Fix $F = F_{[a_1,b_1]} \bullet \cdots \bullet F_{[a_m,b_m]} \in \mathscr{F}_n^\bullet$, and path family $\pi \in \Pi_v(F)$ having earliest defect at x_k. Then there exists a path family $\sigma \in \Pi_v(F)$ which satisfies*

1. *for all $p \neq k$, we have $\{(i,j) \,|\, (\sigma_i, \sigma_j, p)$ defective$\} = \{(i,j) \,|\, (\pi_i, \pi_j, p)$ defective$\}$,*
2. *for $p = k$, we have $\#\{(i,j) \,|\, (\sigma_i, \sigma_j, p)$ defective$\} = \#\{(i,j) \,|\, (\pi_i, \pi_j, p)$ defective$\} - 1$.*

Proof. Omitted. □

By Proposition 4.3, we see that a path family having type v and d defects (i.e., in $\Pi_{v,d}(F)$) implies the existence of other path families of type v having $d-1, \ldots, 0$ defects.

Theorem 4.4. *Fix star network $F \in \mathscr{F}_n^\bullet$. If for some $v \in \mathfrak{S}_n$ and $d \geq 1$ the set $\Pi_{v,d}(F)$ is nonempty, then the sets $\Pi_{v,d-1}(F), \ldots, \Pi_{v,0}(F)$ are also nonempty, and $|\Pi_{v,0}(F)| = 1$.*

Proof. Omitted. □

5 Main Result

Every polynomial in $\mathbf{N}[q]$ with constant term 1 arises as a Kazhdan–Lusztig polynomial [22]. Gaetz–Gao [15] studied the sequences of coefficients in these polynomials, especially coefficients equal to 0 between other nonzero coefficients. Define a function $\mathrm{singdeg} : \mathfrak{S}_n \to \mathbf{N} \cup \{\infty\}$ by

$$\mathrm{singdeg}(w) = \begin{cases} \infty & \text{if } P_{e,w}(q) = 1, \\ \min\{k > 0 \,|\, \text{coefficient of } q^k \text{ in } P_{e,w}(q) \text{ is nonzero}\} & \text{if } P_{e,w}(q) \neq 1. \end{cases} \tag{24}$$

This is a lower bound on degrees for which Poincaré duality fails in the Schubert variety X_w, and can be computed in terms of patterns in w and a related definition. Specifically, given $w \in \mathfrak{S}_n$ not avoiding the pattern 3412, define the *3412-gap* of w by

$$\mathrm{gap}_{3412}(w) = \min\{w_{i_1} - w_{i_4} \,|\, \text{subword } w_{i_1}w_{i_2}w_{i_3}w_{i_4} \text{ matches the pattern } 3412\}. \tag{25}$$

For w avoiding the patterns 3412 and 4231, we have $\mathrm{singdeg}(w) = \infty$. Otherwise, we can compute $\mathrm{singdeg}(w)$ in terms of $\mathrm{gap}_{3412}(w)$ as follows [15, Thm. 1.6].

Theorem 5.1. *For w not avoiding the patterns* 3412 *and* 4231 *we have*

$$\mathrm{singdeg}(w) = \begin{cases} \mathrm{gap}_{3412}(w) & \text{if } w \text{ avoids the pattern } 4231, \\ 1 & \text{otherwise}. \end{cases}$$

For example, consider the permutation $45312 \in \mathfrak{S}_5$, which avoids the pattern 4231 and has only the subword 4512 matching the pattern 3412. Since $\mathrm{gap}_{3412}(45312) = 2$, Theorem 5.1 implies that the coefficient of q in $P_{e,45312}(q)$ is 0 and that the coefficient of q^2 is not. This is consistent with the fact that $P_{e,45312}(q) = 1 + q^2$. (See [4, p. 75].)

For each permutation w having $\mathrm{singdeg}(w) > 1$, the Kazhdan–Lusztig basis element $\widetilde{C}_w$ has no parabolic factorization (22) and therefore is not graphically representable by a star network, in the sense of (18).

Theorem 5.2. *For $w \in \mathfrak{S}_n$ avoiding the pattern* 4231*, not avoiding the pattern* 3412*, and having* $\mathrm{gap}_{3412}(w) > 1$*, the Kazhdan–Lusztig basis element $\widetilde{C}_w$ has no parabolic factorization.*

Proof. Fix w as above with $k = \mathrm{gap}_{3412}(w)$ and suppose that the star network F graphically represents $\widetilde{C}_w$ as an element of $H_n(q)$,

$$\widetilde{C}_w = \sum_{v \in \mathfrak{S}_n} \sum_{d \geq 0} |\Pi_{v,d}(F)| q^d T_v = \sum_{v \leq w} P_{v,w}(q) T_v. \tag{26}$$

Since the constant term of $P_{e,w}(q)$ is 1, we have $|\Pi_{e,0}(F)| = 1$. By our definition of k and Theorem 5.1, the coefficients of $q, \ldots, q^{k-1}$ in $P_{e,w}(q)$ are 0, while the coefficient of q^k is positive. In particular, we have the cardinalities $|\Pi_{e,1}(F)| = \cdots = |\Pi_{e,k-1}(F)| = 0$ and $|\Pi_{e,k}(F)| > 0$, which contradict Theorem 4.4. □

By Theorem 4.4, parabolic factorization of $\widetilde{C}_w$ implies that *none* of the Kazhdan–Lusztig polynomials $P_{v,w}(q)$ has internal coefficients equal to zero.

Theorem 5.3. *If $\widetilde{C}_w$ has a parabolic factorization, then for every $v < w$ there exists $k = k(v)$ in* **N** *such that we have $P_{v,w}(q) = 1 + a_1 q + \cdots + a_k q^k$ with $a_1, \ldots, a_k > 0$.*

It is easy to show that the inequality $\mathrm{gap}_{3412}(w) > 1$ implies that w does not avoid the pattern 45312. It is also easy to show that no star network graphically represents $\widetilde{C}_{453129786}$, even though the subword 9786 matches the pattern 4231. Indeed, some limited experimentation [12] suggests that avoidance of the pattern 45312 is important and avoidance of the pattern 4231 is unimportant in the classification of permutations w for which $\widetilde{C}_w$ is graphically representable by a star network. We conjecture the following partial answer to [23, Quest. 4.5].

Conjecture 5.4. *If $w \in \mathfrak{S}_n$ does not avoid the pattern* 45312*, then the Kazhdan–Lusztig basis element $\widetilde{C}_w$ has no parabolic factorization.*

References

[1] A. Beĭlinson & J. Bernstein (1981): *Localisation of* $\mathfrak{g}$*-modules*. *C. R. Acad.Sci. Paris Ser. I Math* 292, pp. 15–18.

[2] A. Beĭlinson & J. Bernstein (1993): *A proof of Jantzen conjectures*. In: *I. M. Gelfand Seminar*, *Adv. Soviet Math* 16, Part 1, Amer. Math. Soc., Providence, RI, pp. 1–50.

[3] A. Beĭlinson, J. Bernstein & P. Deligne (1982): *Faisceaux pervers*. In: *Analysis and topology on singular spaces, I (Luminy, 1981)*, *Astérisque* 100, Soc. Math. France, Paris, pp. 5–171.

[4] S. Billey & V. Lakshmibai (2000): *Singular loci of Schubert varieties*. *Progress in Mathematics* 182, Birkhäuser Boston Inc., Boston, MA, doi:10.1007/978-1-4612-1324-6.

[5] S. Billey & G. Warrington (2001): *Kazhdan–Lusztig polynomials for* 321*-hexagon-avoiding permutations*. *J. Algebraic Combin.* 13(2), pp. 111–136, doi:10.1023/A:1011279130416.

[6] A. Björner & F. Brenti (2005): *Combinatorics of Coxeter groups*. *Graduate Texts in Mathmatics* 231, Springer, New York.

[7] F. Brenti (1994): *A combinatorial formula for Kazhdan–Lusztig polynomials*. *Invent. Math.* 118(2), pp. 371–394, doi:10.1007/BF01231537.

[8] F. Brenti (1997): *Combinatorial expansions of Kazhdan–Lusztig polynomials*. *J. London Math. Soc.* 55(2), pp. 448–472, doi:10.1112/S0024610797004948.

[9] J.-L. Brylinski & M. Kashiwara (1981): *Kazhdan–Lusztig conjecture and holonomic systems*. *Invent. Math.* 64(3), pp. 387–410, doi:10.1007/BF01389272.

[10] S. Clearman, M. Hyatt, B. Shelton & M. Skandera (2016): *Evaluations of Hecke algebra traces at Kazhdan–Lusztig basis elements*. *Electron. J. Combin.* 23(2), doi:10.37236/5021. Paper 2.7, 56 pages.

[11] A. Clearwater & M. Skandera (2021): *Total nonnegativity and Hecke algebra trace evaluations*. *Ann. Combin.* 25, pp. 757–787, doi:10.1007/s00026-021-00545-4.

[12] A. Datko & M. Skandera (2022): *Combinatorial interpretation of Kazhdan–Lusztig basis elements indexed by* 45312*-avoiding permutations in* S_6. *Pure Math. Appl.* 30, pp. 68–74, doi:10.2478/puma-2022-0011.

[13] V. Deodhar (1990): *A combinatorial setting for questions in Kazhdan–Lusztig theory*. *Geom. Dedicata* 36(1), pp. 95–119, doi:10.1007/BF00181467.

[14] I. B. Frenkel, M. G. Khovanov & A. A. Kirillov, Jr. (1998): *Kazhdan–Lusztig polynomials and canonical basis*. *Transform. Groups* 3(4), pp. 321–336, doi:10.1007/BF01234531.

[15] C. Gaetz & Y. Gao (2024): *On the minimal power of q in a Kazhdan–Lusztig polynomial*. *Adv. Math.* 457, doi:10.1016/j.aim.2024.109941. Paper No. 109941, 25 pages.

[16] I. Goulden & D. Jackson (1992): *Immanants of combinatorial matrices*. *J. Algebra* 148, pp. 305–324, doi:10.1016/0021-8693(92)90196-S.

[17] C. Greene (1992): *Proof of a conjecture on immanants of the Jacobi–Trudi matrix*. *Linear Algebra Appl.* 171, pp. 65–79, doi:10.1016/0024-3795(92)90250-E.

[18] R. Kaliszewski, J. Lambright & M. Skandera (2019): *Bases of the quantum matrix bialgebra and induced sign characters of the Hecke algebra*. *J. Algebraic Combin.* 49(4), pp. 475–505, doi:10.1007/s10801-018-0832-4.

[19] D. Kazhdan & G. Lusztig (1979): *Representations of Coxeter groups and Hecke algebras*. *Invent. Math.* 53, pp. 165–184, doi:10.1007/BF01390031.

[20] D. Kazhdan & G. Lusztig (1980): *Schubert varieties and Poincaré duality*. *Proc. Symp. Pure. Math., A.M.S.* 36, pp. 185–203.

[21] V. Lakshmibai & B. Sandhya (1990): *Criterion for smoothness of Schubert Varieties in* $SL(n)/B$. *Proc. Indian Acad. Sci. (Math Sci.)* 100(1), pp. 45–52, doi:10.1007/BF02881113.

[22] P. Polo (1999): *Construction of arbitrary Kazhdan–Lusztig polynomials in symmetric groups*. Represent. Theory 3, pp. 90–104, doi:10.1090/S1088-4165-99-00074-6.

[23] M. Skandera (2008): *On the dual canonical and Kazhdan–Lusztig bases and 3412, 4231-avoiding permutations*. J. Pure Appl. Algebra 212, doi:10.1016/j.jpaa.2007.09.007.

[24] M. A. Skandera (2025): *Hyperoctahedral group characters and a type-BC analog of graph coloring*. Algebr. Comb. 8(6), pp. 1651–1711, doi:10.5802/alco.459.

[25] T. Springer (1982): *Quelques aplications de la cohomologie d'intersection*. In: Séminaire Bourbaki, Vol. 1981/1982, Astérisque 92, Soc. Math. France, Paris, pp. 249–273.

[26] R. Stanley (1997): *Enumerative Combinatorics*. 1, Cambridge University Press, Cambridge.

[27] J. Stembridge (1991): *Immanants of Totally Positive Matrices are Nonnegative*. Bull. London Math. Soc. 23, pp. 422–428, doi:10.1112/blms/23.5.422.

[28] J. Stembridge (1992): *Some conjectures for immanants*. Canad. J. Math. 44(5), pp. 1079–1099, doi:10.4153/CJM-1992-066-1.

A Class of Multiparameter Signless Stirling Numbers of the First Kind and their q-Analogues

Violetta E. Piperigou
Department of Mathematics
University of Patras, Greece
vpiperig@math.upatras.gr

Malvina G. Vamvakari
Department of Informatics & Telematics
Harokopio University of Athens, Greece
mvamv@hua.gr

In this work, we provide a probabilistic derivation of a class of multiparameter signless Stirling numbers of the first kind and their q-analogues, and study the associated multivariate discrete distributions.

1 Brief Introduction

The present work is motivated by earlier contributions of Cacoullos and Papageorgiou [1] and Koutras [7], as well as more recent developments by Charalambides [2, 3, 4, 5], Kyriakoussis and Vamvakari [8] and Vamvakari [9]. Inspired by these works, we introduce a class of multiparameter Stirling numbers and their q-analogues through probabilistic models.
In the first part, we extend to the multivariate setting the univariate Stirling distributions from which the corresponding signless Stirling numbers of the first kind arise, as presented in Charalambides [2], together with the associated class of multiparameter signless Stirling numbers of the first kind. In the second part, we proceed analogously by generalizing the univariate discrete q-Stirling distributions presented in Charalambides [3] to their multivariate counterparts, and derive the corresponding class of multiparameter signless q-Stirling numbers of the first kind.
Our approach provides a probabilistic framework for the study of these classes of multiparameter signless Stirling numbers of the first kind and their q-analogues, and leads naturally to the establishment of the associated multivariate discrete distributions.

2 Main Results

2.1 A Probability Model for a Class of Multiparameter Signless Stirling Numbers of the First Kind

Consider a chamber of k consecutive lined cells initially containing m_j balls of color c_j, where c_j denotes the j-th shade of grey, for $j = 1,2,\ldots,k$, and a group of n batches of s red balls. Suppose that n balls are successively drawn one after the other from the first lined cell where after each trial the drawn ball is placed back in the first lined cell along with s red balls.
Let $A_{1,i}$ be the event of drawing a ball of color c_1 at the i-th trial, $i = 1,2,\ldots,n$. Then, setting $\theta_1 = m_1/s$, we get

$$p_{1,i} = P(A_{1,i}) = \frac{\theta_1}{\theta_1 + i - 1},\ p'_{1,i} = P\left(A'_{1,i}\right) = \frac{i-1}{\theta_1 + i - 1}, i = 1,2,\ldots,n,$$

S. Brlek and L. Ferrari (Eds.): GASCom 2026
EPTCS 445, 2026, pp. 158–163, doi:10.4204/EPTCS.445.19

© V. E. Piperigou & M.G. Vamvakari
This work is licensed under the
Creative Commons Attribution License.

where $A'_{1,i}$ denotes the complementary event of drawing a ball of color red at the i-th trial, $i = 1,2,\ldots,n$. After n drawings (trials) we remove from the first cell $n-x_1$ batches of s red balls, where x_1 is the number of balls of color c_1 drawn and placed back to the first cell, $x_1 = 1,2,\ldots,n$. Now, suppose that $n-x_1$ balls are successively drawn one after the other from the second lined cell where after each trial the drawn ball is placed back along with s red balls.
Let $A_{2,i}$ be the event of drawing a ball of color c_2 at the i-th trial, $i = 1,2,\ldots,n-x_1$. Then, setting $\theta_2 = m_2/s$, we get

$$p_{2,i} = P(A_{2,i}) = \frac{\theta_2}{\theta_2+i-1}, p'_{2,i} = P\left(A'_{2,i}\right) = \frac{i-1}{\theta_2+i-1},$$

$$i = 1,2,\ldots,n-x_1,\ x_1 = 1,2,\ldots,n.$$

After $n-x_1$ drawings (trials) we remove from the second cell $n-x_1-x_2$ batches of s red balls, where x_2 is the number of balls of color c_2 drawn and placed back to the second cell, $x_2 = 1,2,\ldots,n-x_1$.
As regards the drawings from the k-th cell, there is a group of $n-\sum_{j=0}^{k-1}x_j$ batches of s red balls. Suppose that we successively draw $n-\sum_{j=0}^{k-1}x_j$ balls from the k-th cell, where after each trial the drawn ball is placed back along with s red balls. Let $A_{k,i}$ be the event of drawing a ball of color c_k at the i-th trial, $i = 1,2,\ldots,n-\sum_{j=0}^{k-1}x_j$. Then, setting $\theta_k = m_k/s$, we get

$$p_{k,i} = P(A_{k,i}) = \frac{\theta_k}{\theta_k+i-1}, p'_{k,i} = P\left(A'_{k,i}\right) = \frac{i-1}{\theta_k+i-1},$$

$i = 1,2,\ldots,n-\sum_{j=0}^{k-1}x_j$, $x_k = 1,2,\ldots,n-\sum_{j=0}^{k-1}x_j$, $k \geq 1$, where x_k are the number of balls of color c_k drawn and placed back to the k-th cell in $n-\sum_{j=0}^{k-1}x_j$ trials, with $x_0 = 0$. So,

$$\begin{aligned}
&P\left(\prod_{j=1}^{k} A_{j,i_{j,1}} A_{j,i_{j,2}} \ldots A_{j,i_{j,x_j}} A'_{j,i_{j,x_j+1}} \ldots A'_{j,i_{j,n-\sum_{\nu=0}^{j-1}x_\nu}}\right) \\
&= \prod_{j=1}^{k} P\left(A_{j,i_{j,1}}\right) P\left(A_{j,i_{j,2}}\right) \ldots P\left(A_{j,i_{j,x_j}}\right) \\
&\qquad \cdot P\left(A'_{j,i_{j,x_j+1}}\right) \ldots P\left(A'_{j,i_{j,n-\sum_{\nu=0}^{j-1}x_\nu}}\right)
\end{aligned}$$

Therefore, the probability $p(x_1,x_2,\ldots,x_k;n)$ of drawing x_j balls of color c_j from the j-th cell, $j = 1,2,\ldots,k$, $k \geq 1$, under the described probability model, is given by the equation

$$\begin{aligned}
p(x_1,x_2,\ldots,x_k;n) &= \frac{\prod_{j=1}^{k}\theta_j^{x_j}}{\prod_{j=1}^{k}\left(\theta_j+n-\sum_{\nu=0}^{j-1}x_\nu-1\right)_{n-\sum_{\nu=0}^{j-1}x_\nu}} \\
&\cdot \sum (i_{1,x_1+1}-1)\cdots(i_{1,n}-1)(i_{2,x_2+1}-1)\cdots(i_{2,n-x_1}-1) \\
&\quad \cdots (i_{k,x_k+1}-1)\cdots\left(i_{1,n-\sum_{j=0}^{k-1}x_j}-1\right), \sum_{j=1}^{k}x_j \leq n, k \geq 1, \\
&\quad x_j \geq 1, j = 1,2,\ldots,k, x_0 = 0,
\end{aligned}$$

where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{i_{j,x_j+1},i_{j,x_j+2},\ldots,i_{j,n-\sum_{\nu=0}^{j-1}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu-1$ positive integers $\left\{2,3,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu\right\}$, $j=1,2,\ldots,k$, $k\geq 1$. Thus,

$$p\left(x_1,x_2,\ldots,x_k;n\right)=\frac{\prod_{j=1}^{k}\theta_j^{\,x_j}}{\prod_{j=1}^{k}\left(\theta_j+n-\sum_{\nu=0}^{j-1}x_\nu-1\right)_{n-\sum_{\nu=0}^{j-1}x_\nu}}$$
$$\cdot\sum m_{1,1}\cdots m_{1,n-x_1}m_{2,1}\cdots m_{2,n-x_1-x_2}\cdots m_{k,1}\cdots m_{k,n-\sum_{\nu=1}^{k}x_\nu},$$
$$\sum_{j=1}^{k}x_j\leq n,k\geq 1,x_j\geq 1,j=1,2,\ldots,k,x_0=0,$$

where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu-1$ positive integers $\left\{1,2,3,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu-1\right\}$, $j=1,2,\ldots,k$, $k\geq 1$.

Definition 1. *Let $|s\left(n,x_1,x_2,\ldots,x_k\right)|$, $\sum_{j=1}^{k}x_j\leq n$, $x_j\geq 1$, $j=1,2\ldots,k$, $k\geq 1$, be the numbers given by the multiple sum*

$$|s\left(n,x_1,x_2,\ldots,x_k\right)|=\sum\prod_{j=1}^{k}m_{j,1}\cdots m_{j,n-\sum_{\nu=1}^{j}x_\nu},$$

where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu-1$ positive integers $\left\{1,2,3,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu-1\right\}$, $j=1,2,\ldots,k$, $k\geq 1$. The numbers $|s\left(n,x_1,x_2,\ldots,x_k\right)|$ are called multiparameter signless Stirling numbers of the first kind.

Remark 2. *For $k=1$, the numbers $|s(n,x_1)|$, $x_1=1,2,\ldots,n$, according to the previous definition, are given by*

$$|s(n,x_1)|=\sum m_{1,1}\cdots m_{1,n-x_1}$$

where the summation is extended over all $(n-x_1)$-combinations $\{m_{1,1},m_{1,2},\ldots,m_{1,n-x_1}\}$ of the $n-1$ positive integers $\{1,2,3,\ldots,n-1\}$. Therefore, $|s(n,x_1)|$ are the signless Stirling numbers of the first kind.

2.2 A Probability Model for Multiparameter Signless Noncentral Stirling Numbers of the First Kind

Next, we consider a chamber of k consecutive lined cells initially containing m_j balls of color $c_{1:j}$ and b_j balls of color $c_{2:j}$, $j=1,2,\ldots,k$ and a group of n batches of s red balls. Suppose that we apply the previous probability model of successive drawings described in Section 2.1. Let $A_{j,i}$ be the event of drawing a ball of color $c_{1:j}$, $j=1,2,\ldots,k$, $k\geq 1$, at the i-th trial, $i=1,2,\ldots,n-\sum_{\nu=0}^{j-1}x_j$. Then, setting $\theta_j=m_j/s$, $r_j=b_j/s$, we get

$$p_{j,i}=P\left(A_{j,i}\right)=\frac{\theta_j}{\theta_j+r_j+i},\ p'_{j,i}=P\left(A'_{j,i}\right)=\frac{r_j+i}{\theta_j+r_j+i},$$

$i=1,2,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu$, $x_j=0,1,2,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu$, where x_j are the number of balls of color $c_{1:j}$ drawn and placed back to the j-th cell in $n-\sum_{\nu=0}^{j-1}x_\nu$, trials $j=1,2,\ldots,k$.
Working analogously as in Section 2.1, we derive the following expansion of Definition 1.

Definition 3. *Let* $|s(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)|$, $\sum_{j=1}^{k} x_j \leq n$, $x_j \geq 1$, $j = 1,2\ldots,k$, $k \geq 1$, *be the numbers given by the multiple sum*

$$|s(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)| = \sum \prod_{j=1}^{k} (r_1+m_{j,1})\cdots\left(r_j+m_{j,n-\sum_{\nu=1}^{j} x_\nu}\right),$$

where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j} x_\nu\right)$*-combinations* $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j} x_\nu}\right\}$ *of the* $n-\sum_{\nu=0}^{j-1} x_\nu - 1$ *positive integers* $\left\{1,2,3,\ldots,n-\sum_{\nu=0}^{j-1} x_\nu - 1\right\}$, $j = 1,2,\ldots,k$, $k \geq 1$. *The numbers* $|s(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)|$ *are called multiparameter signless noncentral Stirling numbers of the first kind.*

Remark 4. *For* $k = 1$, *the numbers* $|s(n,x_1;r_1)|$, $x_1 = 1,2,\ldots,n$, *according to the previous definition, are given by*

$$|s(n,x_1;r_1)| = \sum (r_1+m_{1,1})\cdots(r_1+m_{1,n-x_1})$$

where the summation is extended over all $(n-x_1)$*-combinations* $\{m_{1,1},m_{1,2},\ldots,m_{1,n-x_1}\}$ *of the* $n-1$ *positive integers* $\{1,2,3,\ldots,n-1\}$. *Therefore,* $|s(n,x_1;r_1)|$ *are the signless noncentral Stirling numbers of the first kind (see Charalambides [2]).*

2.3 A Probability Model for a Class of Multiparameter Signless q-Stirling Numbers of the First Kind

The multivariate discrete q-distributions, $0 < q < 1$, are based on stochastic models of sequences of n independent Bernoulli trials with chain-composite failures (or successes), where the odds of success of a certain kind at a trial is assumed to vary geometrically, with rate q, with the number of previous trials or with the number of previous successes or both with the number of previous trials and successes (see Charalambides [6]).

Next, we consider a sequence of independent Bernoulli trials with chain-composite failures, where the probability of success of the j-th kind at the i-th trial is given by

$$p_{j,i} = \frac{1}{[r_j+i]_q},\ 0 \leq r_j < \infty,\ j = 1,2,\ldots,k, i = 1,2,\ldots \tag{1}$$

Let $A_{j,i}$ be the event of success of the j-th kind at the i-th trial with probability of success given by Eq. (1) for $j = 1,\ldots,k$, $k \geq 1$, $i = 1,\ldots$, and consider a permutation $(i_{j,1},\ldots,i_{j,x_j},i_{j,x_{j+1}},\ldots,i_{j,n-\sum_{\nu=0}^{j-1} x_\nu})$ of $\{1,2,\ldots,n-\sum_{\nu=0}^{j-1} x_\nu\}$. Then, we get

$$P\left(\prod_{j=1}^{k} A_{j,i_{j,1}}\ldots A_{j,i_{j,x_j}} A'_{j,i_{j,x_j+1}}\cdots A'_{j,i_{j,n-\sum_{\nu=0}^{j-1} x_\nu}}\right) = \prod_{j=1}^{k} P\left(A_{j,i_{j,1}}\right)\ldots P\left(A_{j,i_{j,x_j}}\right) P\left(A'_{j,i_{j,x_j+1}}\right)\cdots P\left(A'_{j,i_{j,n-\sum_{\nu=0}^{j-1} x_\nu}}\right)$$

and the following theorem holds.

Theorem 5. *Let X_j be the number of successes of the j-th kind in a sequence of n independent Bernoulli trials with chain-composite failures, where the probability of success of the j-th kind at the i-th trial is given by Eq.* (1)*, for $j = 1,2,\ldots,k$. Then the probability function of the r.v. $(X_1,X_2,\ldots,X_k)$ is given by*

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_k = x_k) = \frac{q^{nk-\sum_{j=1}^{k}(k-j+1)x_j}}{\prod_{j=1}^{k}[r_j+n-\sum_{\nu=0}^{j-1}x_\nu]_{n-\sum_{\nu=0}^{j-1}x_\nu,q}} \cdot \sum\prod_{j=1}^{k}[r_j+m_{j,1}]_q[r_j+m_{j,2}]_q\cdots[r_j+m_{j,n-\sum_{\nu=1}^{j}x_\nu}]_q, x_j = 0,1,\ldots,n,$$

with $\sum_{j=1}^{k}x_j \le n$, $x_0 = 0$, where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu$ nonnegative integers $\left\{0,1,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu-1\right\}$, $j = 1,2,\ldots,k$, $k \ge 1$.

Definition 6. *Let $|s_q(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)|$, $\sum_{j=1}^{k}x_j \le n$, $x_j \ge 1$, $0 \le r_j < \infty$, $j = 1,2\ldots,k$, $k \ge 1$, be the numbers given by the multiple sum*

$$|s_q(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)| = q^{nk-\sum_{j=1}^{k}(k-j+1)x_j} \cdot \sum\prod_{j=1}^{k}[r_j+m_{j,1}]_q[r_j+m_{j,2}]_q\cdots[r_j+m_{j,n-\sum_{\nu=1}^{j}x_\nu}]_q,$$

where the multiple sum is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu$ nonnegative integers $\left\{0,1,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu-1\right\}$, $j = 1,2,\ldots,k$, $k \ge 1$. The numbers $|s_q(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)|$ are called multiparameter signless noncentral q-Stirling numbers of the first kind.

Remark 7. *For $k = 1$, the numbers $|s_q(n,x_1;r_1)|$, $x_1 = 1,2,\ldots,n$, $n = 1,2,\ldots$, $0 \le r_1 < \infty$, according to the previous definition, are given by*

$$|s_q(n,x_1;r_1)| = q^{n-x_1}\sum[r_1+m_{1,1}]_q[r_1+m_{1,2}]_q\cdots[r_1+m_{1,n-x_1}]_q, \tag{2}$$

where the summation is extended over all $(n-x_1)$-combinations $\{m_{1,1},m_{1,2},\ldots,m_{1,n-x_1}\}$ of the n nonnegative integers $\{0,1,\ldots,n-1\}$. Therefore, $|s_q(n,x_1;r_1)|$ are the signless noncentral q-Stirling numbers of the first kind (see Charalambides [3]).

Remark 8. *The multiparameter noncentral q-Stirling numbers of the first kind are defined by the multiple sum as follows*

$$s_q(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k) = \sum\prod_{j=1}^{k}[-1]_q^{n-\sum_{\nu=1}^{j}x_\nu}[r_j+m_{j,1}]_q[r_j+m_{j,2}]_q\cdots[r_j+m_{j,n-\sum_{\nu=1}^{j}x_\nu}]_q,$$

where the summation is extended over all $\left(n-\sum_{\nu=1}^{j}x_\nu\right)$-combinations $\left\{m_{j,1},m_{j,2},\ldots,m_{j,n-\sum_{\nu=1}^{j}x_\nu}\right\}$ of the $n-\sum_{\nu=0}^{j-1}x_\nu$ nonnegative integers $\left\{0,1,\ldots,n-\sum_{\nu=0}^{j-1}x_\nu-1\right\}$, $j = 1,2,\ldots,k$, $k \ge 1$.

Note that if we change suitably the probability of success the following theorem also holds.

Theorem 9. *Let X_j be the number of successes of the j-th kind in a a sequence of n independent Bernoulli trials with chain-composite failures, where the probability of success of the j-th kind at the i-th trial, is given by*

$$p_{j,i} = q^{r_j+i-1},\ 0 \le r_j < \infty,\ j = 1,2,\ldots,k, i = 1,2,\ldots,\ 0 < q < 1. \tag{3}$$

Then the probability function of the r.v. $(X_1, X_2, \ldots, X_k)$ is given by

$$\begin{aligned} P(X_1 = x_1, X_2 = x_2, \ldots, X_k = x_k) &= q^{\sum_{j=1}^{k}\binom{n-\sum_{v=0}^{j-1}x_v}{2}+\sum_{j=1}^{k}\left(n-\sum_{v=0}^{j-1}x_v\right)r_j} \\ &\quad \cdot(1-q)^{\sum_{j=1}^{k}\left(n-\sum_{v=1}^{j}x_v\right)}\left|s_{q^{-1}}(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)\right| \end{aligned}$$

with $x_j = 0,1,\ldots,n$, $\sum_{j=1}^{k} x_j \le n$, $x_0 = 0$, where

$$\begin{aligned} \left|s_{q^{-1}}(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k)\right| &= (-1)^{\sum_{j=1}^{k}\left(n-\sum_{v=1}^{j}x_v\right)} \\ &\quad \cdot q^{-\sum_{j=1}^{k}\left(n-\sum_{v=1}^{j}x_v\right)}s_{q^{-1}}(n,x_1,x_2,\ldots,x_k;r_1,r_2,\ldots,r_k). \end{aligned}$$

References

[1] Theophilos Cacoullos & Haralambos Papageorgiou (1984): *Multiparameter Stirling and C-Numbers: Recurrences and Applications*. Fibonacci Quarterly 22(2), pp. 119–133, doi:10.1080/00150517.1984.12429903.

[2] Charalambos A. Charalambides (2002): *Enumerative Combinatorics*. Chapman & Hall/CRC, Boca Raton, Florida, doi:10.1201/9781315273112.

[3] Charalambos A. Charalambides (2016): *Discrete q-Distributions*. John Wiley & Sons, Hoboken, NJ, doi:10.1002/9781119119128.

[4] Charalambos A. Charalambides (2021): *q-Multinomial and Negative q-Multinomial Distributions*. Communications in Statistics - Theory and Methods 50, pp. 5673–5898, doi:10.1080/03610926.2020.1737711.

[5] Charalambos A. Charalambides (2022): *Multivariate q-Pólya and Inverse q-Pólya Distributions*. Communications in Statistics - Theory and Methods 51, pp. 4854–4876, doi:10.1080/03610926.2020.1825740.

[6] Charalambos A. Charalambides (2024): *Multivariate Discrete q-Distributions*. Springer Nature, Switzerland AG, doi:10.1007/978-3-031-43713-7.

[7] Markos V. Koutras (1990): *Two classes of numbers appearing in the convolution of binomial-truncated Poisson and Poisson-truncated binomial variables*. Fibonacci Quarterly 28(4), pp. 321–333, doi:10.1080/00150517.1990.12429470.

[8] Andreas Kyriakoussis & Malvina Vamvakari (2017): *Heine Process as a q-Analog of the Poisson Process: Waiting and Interarrival Times*. Communications in Statistics - Theory and Methods 46, pp. 4088–4102, doi:10.1080/03610926.2015.1078476.

[9] Malvina Vamvakari (2020): *On Multivariate Discrete q-Distributions: A Multivariate q-Cauchy's Formula*. Communications in Statistics - Theory and Methods 49, pp. 6080–6095, doi:10.1080/03610926.2019.1626427.