Combinatorial generation via permutation languages

Elizabeth Hartung[1], Hung P. Hoang[2], Torsten Mütze[3], Aaron Williams[4]

Introduction

Goal: Exhaustively generate all objects of a combinatorial class efficiently, where consecutive objects differ only a 'little bit' (=Gray code), see [1,2].
This work: A versatile algorithmic framework for generating many different combinatorial objects, such as permutations, binary trees, triangulations, Dyck paths, set partitions, binary strings, rectangulations etc.
Idea: Encode the objects as a subset $L_n \subseteq S_n$, where S_n is the set of all permutations of $[n] := \{1, \ldots, n\}$, and use a greedy algorithm to generate the permutations from L_n by cyclic substring rotations.

Jump

A *jump* moves an entry in the permutation across some neighboring smaller entries left or right (=cyclic substring rotation by one position):

7 4 **5** 1 3 2 6 ⟶ 7 4 1 3 2 **5** 6

An invalid jump (across bigger entries):

7 4 **5** 1 3 2 **6**

Algorithm J (greedy jumps)

Greedily generate a set of permutations $L_n \subseteq S_n$ by minimal jumps, where a jump is *minimal* if any shorter jump of the same value yields a permutation not in L_n.
J1. Visit the initial permutation $\pi_0 \in L_n$
J2. Generate an unvisited permutation from L_n by performing a **minimal jump** of the **largest possible value** in the most recently visited permutation. Visit this permutation and repeat J2.

Zigzag languages

For $\pi \in S_n$, we let π^- denote the permutation in S_{n-1} obtained from π by removing the largest symbol n. Moreover, for $\pi \in S_{n-1}$, we let $n\pi$ and πn denote the permutations in S_n obtained by inserting n at the leftmost or rightmost position of π, respectively.
Zigzag language: A set of permutations $L_n \subseteq S_n$ satisfying (i) $n = 0$ and $L_0 = \{\varepsilon\}$, or (ii) $n \geq 1$ and $L_{n-1} := \{\pi^- \mid \pi \in L_n\}$ is a zigzag language, and for every $\pi \in L_{n-1}$, we have that $n\pi$ and πn are both in L_n.
Theorem: Algorithm J generates *any* zigzag language, using the identity permutation for initialization.

Tree of permutations

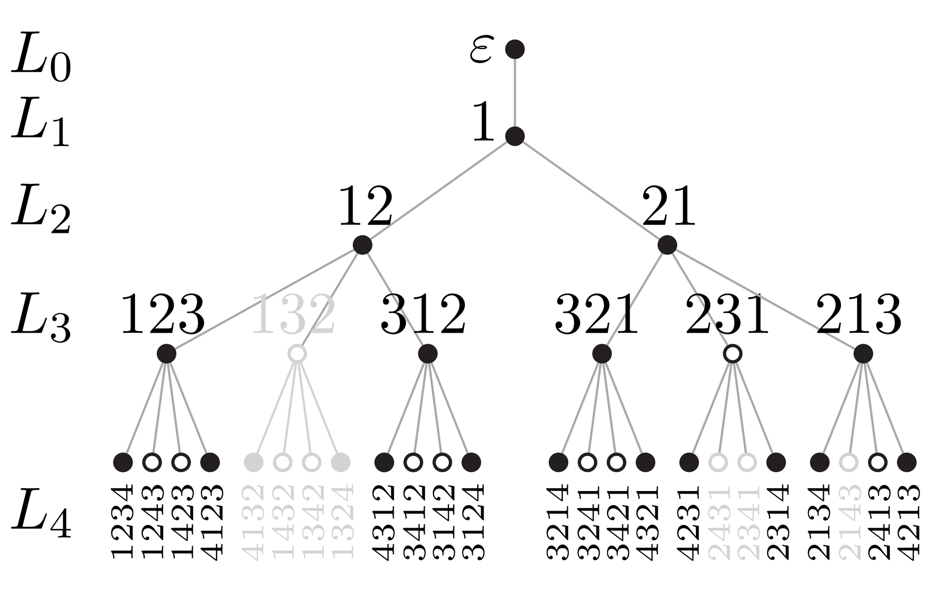

Figure 1: A zigzag language L_n can be interpreted as the set of nodes that remain in distance n from the root in the *tree of permutations* after pruning nodes that are not of the form $k\pi$ or πk for any $\pi \in S_{k-1}$ (filled nodes in the figure; pruned nodes are grayed out).

Pattern-avoiding permutations

Preliminaries: A permutation π *contains a pattern* τ, if π contains a substring of entries in the same relative order as τ. Otherwise, π *avoids* τ. E.g., 635412 contains 231, but 654123 avoids it. If τ has one underlined pair of entries, we call it a *vincular pattern*, and a permutation π containing τ must have the two underlined entries appear consecutively in π. E.g., 3142 contains 231, but avoids $\underline{23}1$.
$S_n(\tau)$ is the set of all permutations of $[n]$ that avoid τ. Furthermore, $S_n(\tau \wedge \rho) := S_n(\tau) \cap S_n(\rho)$ and $S_n(\tau \vee \rho) := S_n(\tau) \cup S_n(\rho)$. A pattern τ is *tame*, if $S_n(\tau)$ is a zigzag language for all $n \geq 1$.
Lemma: If τ does not have the largest symbol at the leftmost or rightmost position, then τ is tame. If τ is a vincular pattern, then in addition the largest symbol must be part of the vincular pair for τ to be tame.
Theorem: Let F be a propositional formula made of logical ANDs $\wedge$, ORs $\vee$, and tame patterns as variables, then $S_n(F)$ is a zigzag language of permutations for all $n \geq 1$. Hence, it can be generated by Algorithm J.

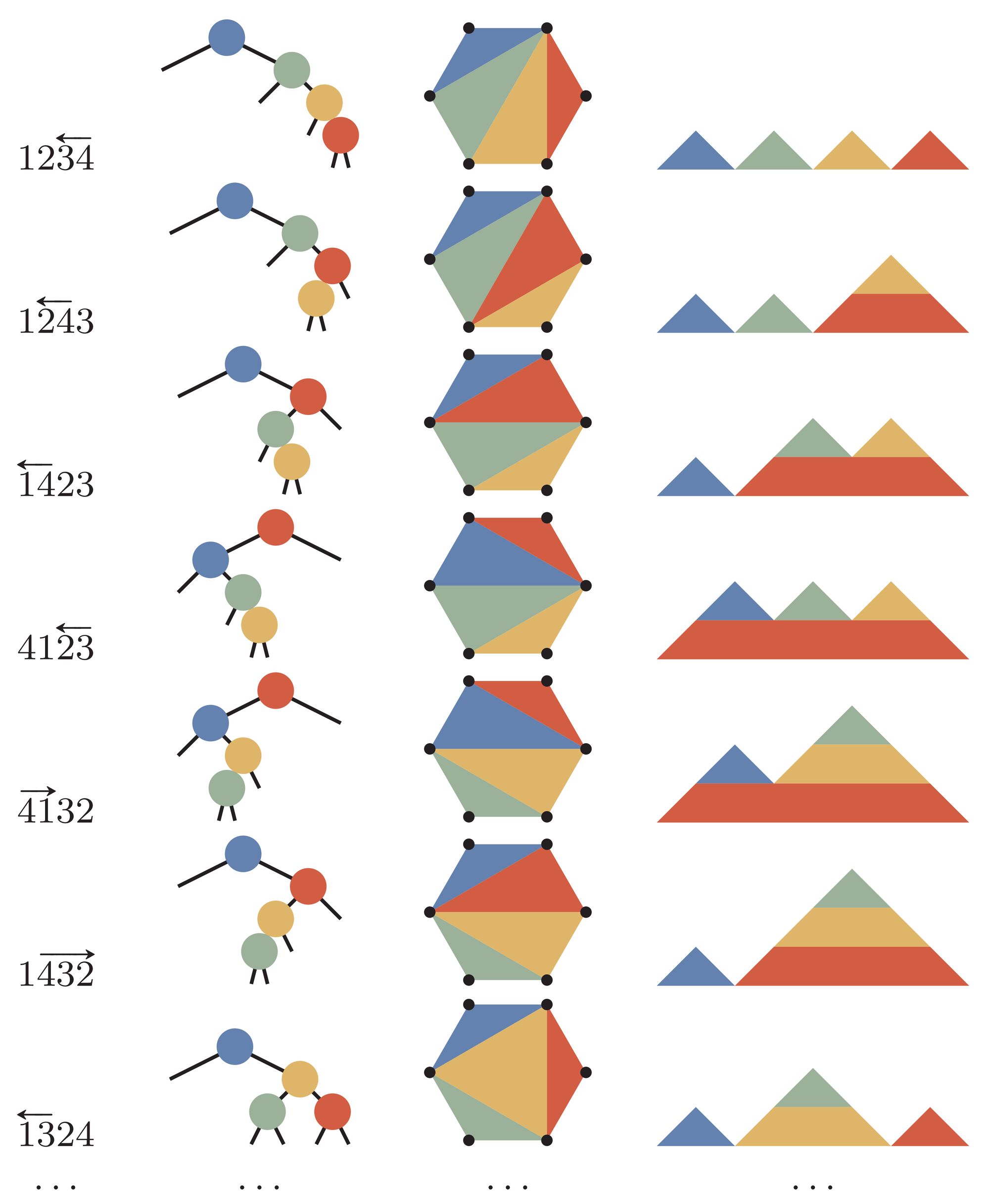

Figure 2: 231-avoiding permutations of length $n = 4$ generated by Algorithm J and resulting Gray codes for binary trees, triangulations and Dyck paths (only first 7 objects shown).

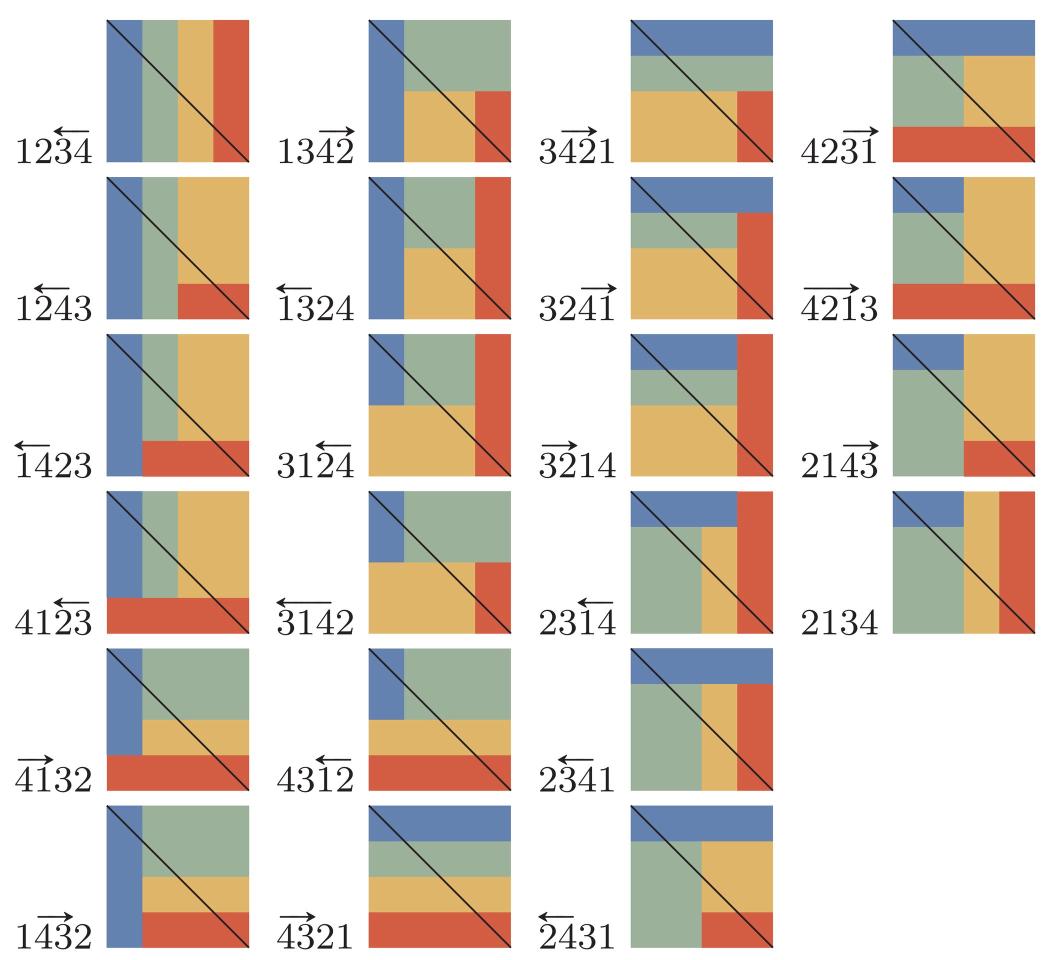

Figure 3: Twisted Baxter permutations ($2\underline{41}3 \wedge 3\underline{41}2$-avoiding) of length $n = 4$ generated by Algorithm J and resulting Gray code for diagonal rectangulations.

Lattice congruences

Preliminaries: The *inversion set* of a permutation π is the set all pairs $(\pi(i), \pi(j))$ for $i < j$ with $\pi(i) > \pi(j)$. The *weak order on* S_n is the lattice obtained by ordering all permutations by containment of their inversion sets. The cover relations are adjacent transpositions.
A *lattice congruence* is an equivalent relation on the weak order that is compatible with taking joins and meets. The corresponding *lattice quotient* is obtained by contracting the equivalence classes and by inheriting all comparabilities (see [4]).

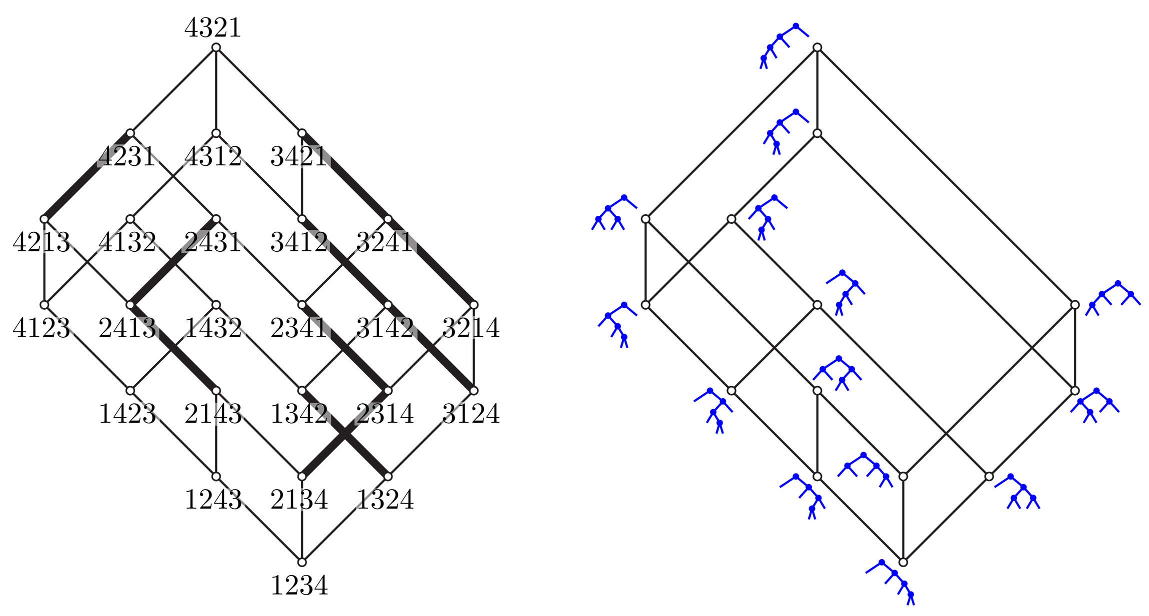

Figure 4: The weak order on S_4 (left) with a lattice congruence (bold edges) and the resulting lattice quotient (=Tamari lattice) with corresponding binary trees (right).

Theorem: For any lattice congruence of the weak order on S_n, there is a set of representative permutations, exactly one for each equivalence class, that forms a zigzag language. The resulting jump order forms a Hamilton path in the cover graph of the lattice quotient.

Pilaud and Santos [3] realized each cover graph of a lattice quotient as the skeleton of a polytope, and they called these polytopes *quotientopes*.
Corollary: Every quotientope has a Hamilton path.

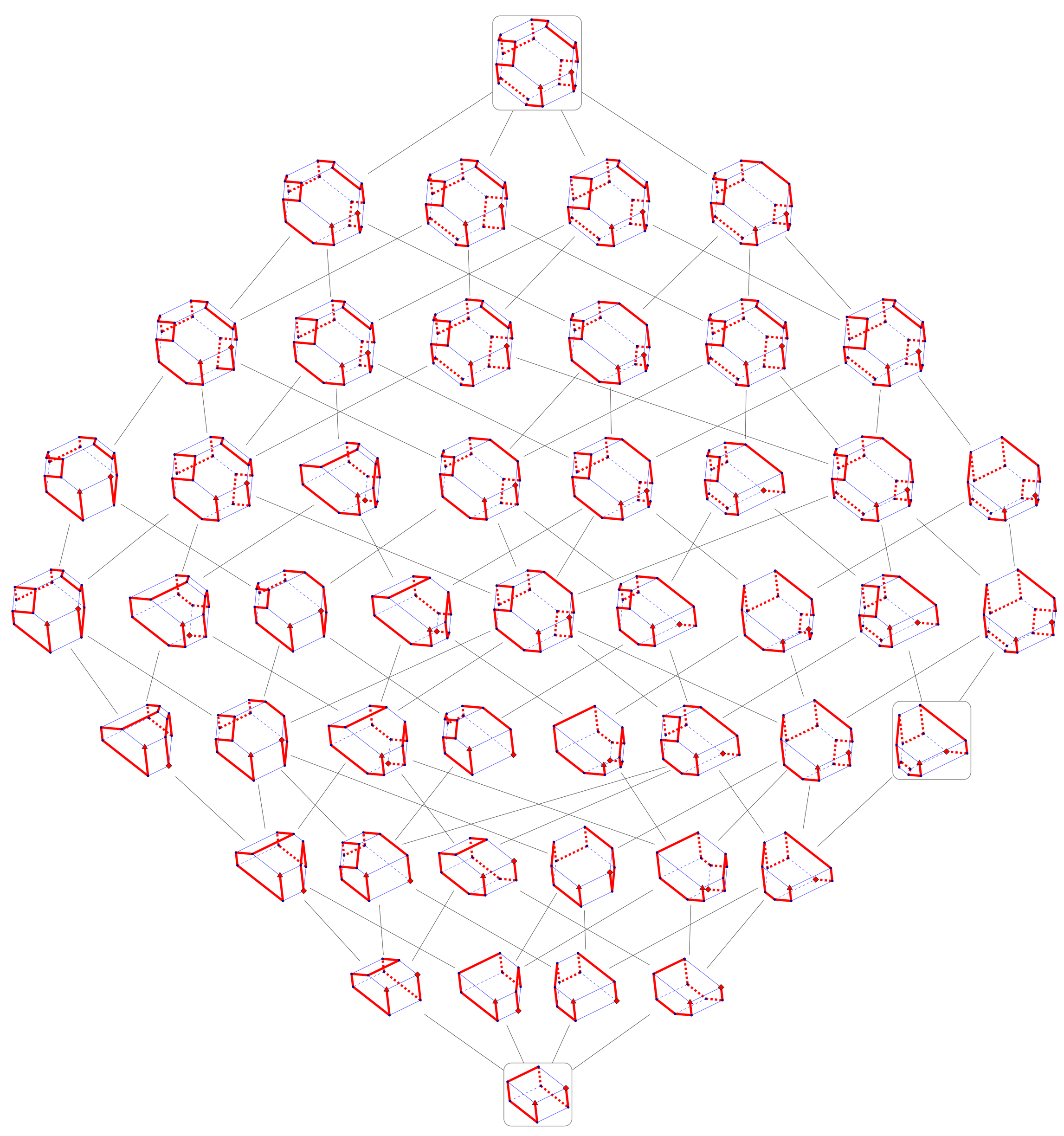

Figure 5: All quotientopes for $n = 4$, with a Hamilton path generated by Algorithm J (end vertices marked). Permutahedron, associahedron and hypercube are highlighted.

References

[1] E. Barcucci, A. Del Lungo, E. Pergola, and R. Pinzani. ECO: a methodology for the enumeration of combinatorial objects. *J. Differ. Equations Appl.*, 5(4-5):435–490, 1999.

[2] D. E. Knuth. *The art of computer programming. Vol. 4A. Combinatorial algorithms. Part 1.* Addison-Wesley, Upper Saddle River, NJ, 2011.

[3] V. Pilaud and F. Santos. Quotientopes. `https://arxiv.org/abs/1711.05353`. To appear in *Bulletin of the London Mathematical Society*, 2018.

[4] N. Reading. Finite Coxeter groups and the weak order. In *Lattice theory: special topics and applications. Vol. 2*, pages 489–561. Birkhäuser/Springer, Cham, 2016.

[1] `e.hartung@mcla.edu`, Massachusetts College of Liberal Arts, United States

[2] `hung.hoang@inf.ethz.ch`, Department of Computer Science, ETH Zürich, Switzerland

[3] `muetze@math.tu-berlin.de`, Institut für Mathematik, Technische Universität Berlin, Germany

[4] `awilliams@simons-rock.edu`, Bard College at Simon's Rock, United States

Tame permutation patterns

Patterns	Combinatorial objects and ordering
none	permutations by adjacent transpositions → plain change order
$231 = 2\underline{31}$	*Catalan families:* binary trees by rotations, triangulations by edge flips, Dyck paths by hill flips → Lucas-van Baronaigien-Ruskey's Gray code order
$\underline{23}1$	set partitions by exchanges → Kaye's Gray code order
$132 \wedge 231 = \underline{13}2 \wedge 2\underline{31}$	binary strings by bitflips → reflected Gray code order (BRGC)
2143: vexillary permutations	
2143 ∧ 3412: skew-merged permutations	
2143 ∧ 2413 ∧ 3142 ∧ 3412: X-shaped permutations	
2413 ∧ 3142: separable permutations	slicing floorplans (=guillotine partitions) by flips
$2\underline{41}3 \wedge 3\underline{14}2$: Baxter permutations $2\underline{41}3 \wedge 3\underline{41}2$: twisted Baxter permutations	mosaic floorplans (=diagonal rectangulations=R-equivalent rectangulations) by flips
$2\underline{14}3 \wedge 3\underline{41}2$	S-equivalent rectangulations by flips
$2\underline{14}3 \wedge 3\underline{41}2 \wedge 2413 \wedge 3142$	S-equivalent guillotine rectangulations by flips
$3\underline{51}24 \wedge 3\underline{51}42 \wedge 24\underline{51}3 \wedge 42\underline{51}3$: 2-clumped permutations	generic rectangulations (=rectangular drawings) by flips and wall slides