Zigzagging through acyclic orientations of chordal graphs and hypergraphs*

Jean Cardinal† Hung P. Hoang‡ Arturo Merino§ Torsten Mütze¶

Abstract

In 1993, Savage, Squire, and West described an inductive construction for generating every acyclic orientation of a chordal graph exactly once, flipping one arc at a time. We provide two generalizations of this result. Firstly, we describe Gray codes for acyclic orientations of hypergraphs that satisfy a simple ordering condition, which generalizes the notion of perfect elimination order of graphs. This unifies the Savage-Squire-West construction with a recent algorithm for generating elimination trees of chordal graphs (SODA 2022). Secondly, we consider quotients of lattices of acyclic orientations of chordal graphs, and we provide a Gray code for them, addressing a question raised by Pilaud (FPSAC 2022). This also generalizes a recent algorithm for generating lattice congruences of the weak order on the symmetric group (SODA 2020). Our algorithms are derived from the Hartung-Hoang-Mütze-Williams combinatorial generation framework, and they yield simple algorithms for computing Hamilton paths and cycles on large classes of polytopes, including chordal nestohedra and quotientopes. In particular, we derive an efficient implementation of the Savage-Squire-West construction. Along the way, we give an overview of old and recent results about the polyhedral and order-theoretic aspects of acyclic orientations of graphs and hypergraphs.

In 1953, Frank Gray registered a patent [Gra53] for a method to list all binary words of length n in such a way that any two consecutive words differ in exactly one bit, and he called it the *binary reflected code*. More generally, a *combinatorial Gray code* [Rus16] is a listing of all objects of a combinatorial class such that any two consecutive objects differ by a 'small local change', sometimes also called a 'flip'. Over the years, Gray codes have been designed for numerous classes of combinatorial objects, including permutations, combinations, integer and set partitions, Catalan objects (binary trees, triangulations etc.), linear extensions of a poset, spanning trees or matchings of a graph etc.; see the surveys [Sav97, Müt22]. This area has been the subject of intensive research combining ideas from combinatorics, algorithms, graph theory, order theory, algebra, and discrete geometry. This enabled recent exciting progress on long-standing problems in this area (see e.g. [SW18]), and the development of versatile general techniques for designing Gray codes [Wil13, RSW12, HHMW20]. One of the main applications of Gray codes is to efficiently generate a class of combinatorial objects (see e.g. [Wil09]), and many such algorithms are described in the most recent volume of Knuth's book 'The Art of Computer Programming' [Knu11].

1 The Steinhaus-Johnson-Trotter algorithm

The *Steinhaus-Johnson-Trotter algorithm*, also known as 'plain changes', is one of the classical Gray codes for generating permutations. Specifically, it lists all permutations of $[n] := \{1, 2, \ldots, n\}$ so that every pair of successive permutations differs by exactly one adjacent transposition, i.e., by swapping two neighboring entries of the permutation. Using suitable auxiliary arrays, this algorithm can be implemented in time $\mathcal{O}(1)$ per visited permutation.

The Steinhaus-Johnson-Trotter ordering of permutations can be defined inductively as follows: For $n = 1$ the listing consists only of a single permutation 1. To construct the listing for permutations of $[n]$ for $n \geq 2$, we consider

*Arturo Merino was supported by ANID Becas Chile 2019-72200522. Torsten Mütze was supported by Czech Science Foundation grant GA 22-15272S. Arturo Merino and Torsten Mütze were also supported by German Science Foundation grant 413902284.

†Computer Science Department, Université Libre de Bruxelles (ULB), Belgium

‡Department of Computer Science, ETH Zürich, Switzerland

§Department of Mathematics, TU Berlin, Germany

¶Department of Computer Science, University of Warwick, United Kingdom and Faculty of Mathematics and Physics, Charles University Prague, Czech Republic

the listing for permutations of $[n-1]$, and we replace every permutation π in it by the sequence of permutations obtained by inserting the new largest symbol n in all possible positions in π from right to left, or from left to right, alternatingly. It is easy to check that this indeed gives a listing of all permutations of $[n]$ by adjacent transpositions. Moreover, as $n!$ is even for $n \geq 2$, the listing is cyclic, i.e., the last and first permutation differ only in an adjacent transposition. For example, for $n = 2$ we get the listing $12, 21$, for $n = 3$ we get $123, 132, 312, 321, 231, 213$, and for $n = 4$ we get $1234, 1243, 1423, 4123, 4132, 1432, 1342, 1324, 3124, 3142, 3412, 4312, 4321, 3421, 3241, 3214, 2314, 2341, 2431, 4231, 4213, 2413, 2143, 2134$; see Figure 1. In those listings, the newly inserted symbol n is highlighted, which allows tracking its zigzag movement.

Williams [Wil13] found a strikingly simple equivalent description of the Steinhaus-Johnson-Trotter ordering via the following greedy algorithm: Start with the identity permutation, and repeatedly perform an adjacent transposition with the largest possible value that yields a previously unvisited permutation.

The results in this work can be seen as far-ranging generalizations of these two alternative descriptions of the same fundamental ordering.

2 Flip graphs, lattices, and polytopes

Any Gray code problem gives rise to a corresponding *flip graph*, which has as vertices the combinatorial objects of interest, and an edge between any two objects that differ by the specified flip operation. For example, the flip graph on binary words of length n under flips that change a single bit is the n-dimensional hypercube. Moreover, the flip graph for permutations under adjacent transpositions discussed in the previous section is the Cayley graph of the symmetric group generated by adjacent transpositions. Another heavily studied example is the flip graph on binary trees under tree rotations [STT88, Pou14].

Clearly, computing a Gray code for a set of combinatorial objects amounts to traversing a Hamilton path in the corresponding flip graph. In particular, Hamilton paths in the three aforementioned flip graphs can be computed by the binary reflected code, the Steinhaus-Johnson-Trotter algorithm, and by an algorithm due to Lucas, Roelants van Baronaigien, and Ruskey [LRvBR93], respectively.

It turns out that many flip graphs can be equipped with a poset structure and realized as polytopes, i.e., they are cover graphs of certain lattices, and 1-skeleta of certain high-dimensional polytopes. For example, the n-dimensional hypercube is the cover graph of the Boolean lattice and the skeleton of the Cartesian product $[0, 1]^n$. Similarly, the flip graph on permutations under adjacent transpositions is the cover graph of the weak order on the symmetric group, and the skeleton of the *permutohedron*; see Figure 1. Lastly, the flip graph on binary trees under rotations is the cover graph of the Tamari lattice and the skeleton of the *associahedron*.

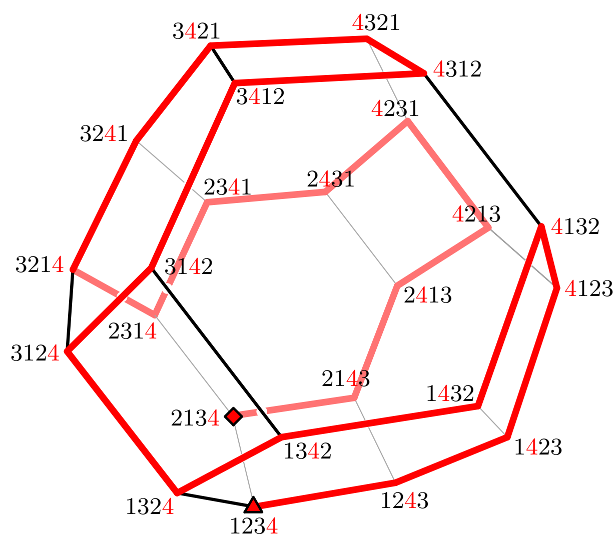

Figure 1: The 3-dimensional permutohedron with the Steinhaus-Johnson-Trotter Hamilton path. The start and end vertex are highlighted by a triangle and diamond, respectively, and can be joined to a Hamilton cycle.

Generalizations of these lattices and polytopes and the associated combinatorial structures have been the subject of intensive research in algebraic and polyhedral combinatorics; see Figure 4. The theory of *generalized permutohedra* [PRW08, Pos09, AA17] and of *lattice congruences* and their *quotients* [Rea12, Rea16], in particular, provides us with a rich framework that contains all previous three examples as very special cases of a much broader picture. Specifically, in the next three sections we discuss the generalizations shown one level above the bottom in Figure 4, namely acyclic orientations, elimination trees and lattice quotients.

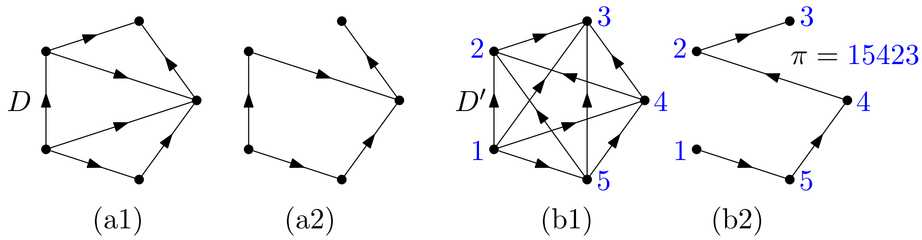

Figure 2: (a1) An acyclic orientation D of a graph; (a2) the transitive reduction of D, which contains precisely the flippable arcs in D; (b1) an acyclic orientation D' of the complete graph; (b2) the transitive reduction of D' and the corresponding permutation.

3 From permutations to acyclic orientations

The starting point of this work are Gray codes for acyclic orientations of a graph. Given a simple graph G, an *acyclic orientation* of G is a digraph D obtained by orienting every edge of G in one of two ways so that D does not contain any directed cycles. The goal is to list all acyclic orientations of G in such a way that any two consecutive orientations differ by reorienting a single arc, which we refer to as an *arc flip*.

It is easy to see that in an acyclic orientation D, the flippable arcs are precisely the arcs that are in the *transitive reduction* of D, which is the minimum subset of arcs that has the same reachability relations (i.e., the same transitive closure) as D; see Figure 2 (a1)+(a2). If G is the complete graph with vertex set $[n]$, then the transitive reduction of any of its acyclic orientations D is a path, directed from the source to the sink of the orientation. Consequently, we can interpret the vertex labels along this path as a permutation of $[n]$; see Figure 2 (b1)+(b2). Furthermore, an arc flip corresponds to an adjacent transposition in this permutation. Consequently, the flip graph on acyclic orientations of the complete graph is the skeleton of the permutohedron. In general, the flip graph on the acyclic orientations of a graph G is the skeleton of a polytope known as the *graphical zonotope* of G [Gre77, GZ83, Sta07].

In general, not all (skeletons of) graphical zonotopes admit a Hamilton path or cycle, and we do not know of any simple conditions on the graph G for this to hold. Clearly, the flip graph on acyclic orientations is bipartite for any graph G, and if the partition classes have sizes that differ by more than 1, then this rules out the existence of a Hamilton path, a phenomenon that occurs for example if G is a wheel graph with an even number of spokes [SSW93]. In this context, let us mention that counting the number of acyclic orientations of a graph is #P-complete [Lin86].

On the positive side, Savage, Squire and West [SSW93] showed that the flip graph on acyclic orientations of G has a Hamilton cycle if G is chordal. Their proof is a straightforward generalization of the Steinhaus-Johnson-Trotter construction, so we describe it here, with the goal of generalizing it even further subsequently. A graph is *chordal* if every induced cycle has length 3. It is well-known that every chordal graph G has a *simplicial* vertex v, i.e., a vertex whose neighborhood in the graph is a clique. We remove v from the graph, and by induction we obtain a Gray code for the acyclic orientations of $G-v$; see Figure 3 (a). Let k be the number of neighbors of v in G. To construct the listing of acyclic orientations of G, we replace every acyclic orientation in the listing for $G-v$ by the

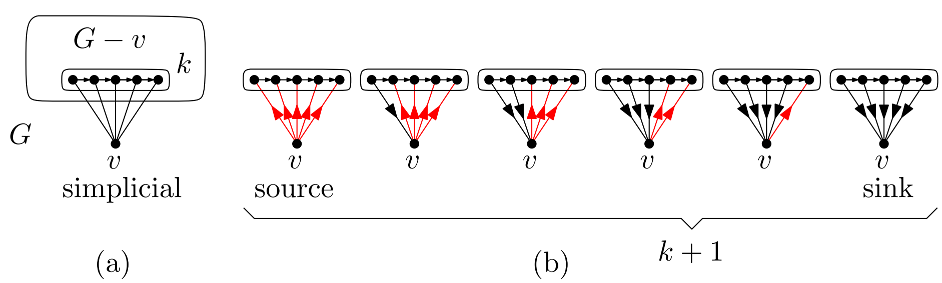

Figure 3: Illustration of the Savage-Squire-West proof. In the neighborhood of v, only the transitive reduction is shown, whereas transitive arcs are omitted for simplicity.

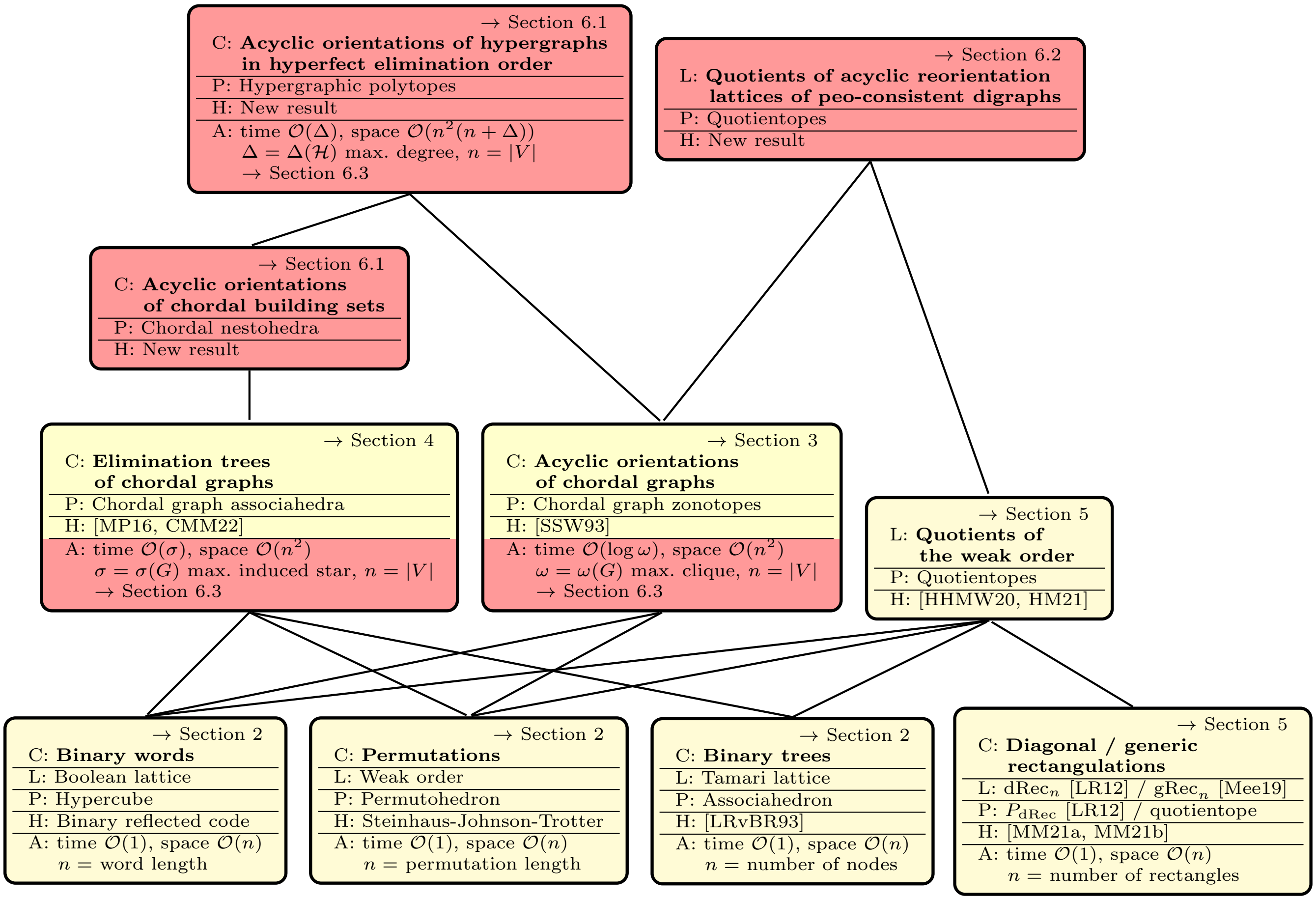

Figure 4: Inclusion diagram of combinatorial families (C), lattices (L), polytopes (P), Hamiltonicity results (H), and corresponding algorithmic results (A). More general objects are above their specialized counterparts. New results are highlighted red. Running times are per generated object, whereas the space refers to the total space needed (without storing previous objects). Section references indicate where those results are discussed in more detail.

sequence of $k+1$ acyclic orientations obtained by adding v and orienting the edges incident with v in all possible ways (that yield an acyclic orientation). Specifically, since the neighborhood of v is a clique, whose transitive reduction is a path, there are precisely $k+1$ valid acyclic orientations obtained by adding v, and they differ in a sequence of arc flips of arcs incident with v, and this sequence starts and ends with v being a sink or a source; see Figure 3 (b). In the Gray code for the acyclic orientations of G, the vertex v alternates or 'zigzags' between being sink or source. As the number of acyclic orientations of any graph G with at least one edge is even (consider the involution on the set of all acyclic orientations that reorients every arc), the resulting ordering is cyclic, i.e., the last and first acyclic orientation differ only in an arc flip. For G being a complete graph, the resulting ordering of acyclic orientations and their corresponding permutations is exactly the Steinhaus-Johnson-Trotter ordering.

4 From permutations to elimination trees

An elimination tree T of a connected graph G is an unordered rooted tree obtained as follows: We remove a vertex v of G which becomes the root of T, and we recurse on the connected components of $G - v$, whose elimination trees become the subtrees of v in T; see Figure 5 (a1)+(a2). The goal is to list all elimination trees of G in such a way that any two consecutive trees differ by a *rotation*, which is the result of swapping the removal order of two vertices that form a parent-child relationship in the elimination tree; see Figure 10 (a)+(b).

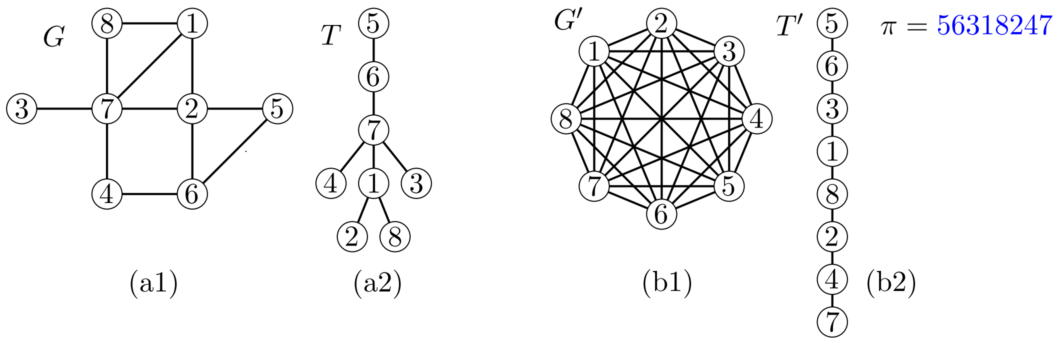

Figure 5: (a1) A graph G; (a2) an elimination tree T of G; (b1) the complete graph; (b2) an elimination tree of the complete graph and the corresponding permutation.

Clearly, every elimination tree of a complete graph with vertex set $[n]$ is a path, which can be interpreted as a permutation of $[n]$ by reading the labels of the path from the root to the leaf; see Figure 5 (b1)+(b2). Furthermore, a tree rotation corresponds to an adjacent transposition in this permutation. Consequently, the flip graph on elimination trees of the complete graph is the skeleton of the permutohedron. In general, the flip graph on elimination trees of a graph G is the skeleton of a polytope known as the *graph associahedron* of G [CD06, Dev09, Pos09].

Manneville and Pilaud [MP16] showed that the skeleton of the graph associahedron of G admits a Hamilton cycle for any graph G with at least two edges. In [CMM22], we present a simple algorithm for computing a Hamilton path on the graph associahedron for the case when G is a chordal graph. This algorithm visits each elimination tree along the Hamilton path in time $\mathcal{O}(m+n)$, where m and n are the number of edges and vertices of G, and this time can be improved to $\mathcal{O}(1)$ if G is a tree. Furthermore, if G is chordal and 2-connected, then the resulting Hamilton path is actually a Hamilton cycle, i.e., the first and last elimination tree differ only in a tree rotation. The proof in [CMM22] is an application of the Hartung-Hoang-Mütze-Williams generation framework [HHMW20, HHMW22], which generalizes the Steinhaus-Johnson-Trotter algorithm. Specifically, we consider a simplicial vertex v in G, we remove v from the graph, and by induction we obtain a rotation Gray code for the elimination trees of $G-v$; see Figure 6 (a). Let $N(v)$ be the set of neighbors of v in G. To construct the listing of elimination trees of G, we consider every elimination tree T in the listing for $G-v$. As the vertices in $N(v)$ form a clique in G, these vertices appear on a path P in T that starts at the root and ends at a vertex

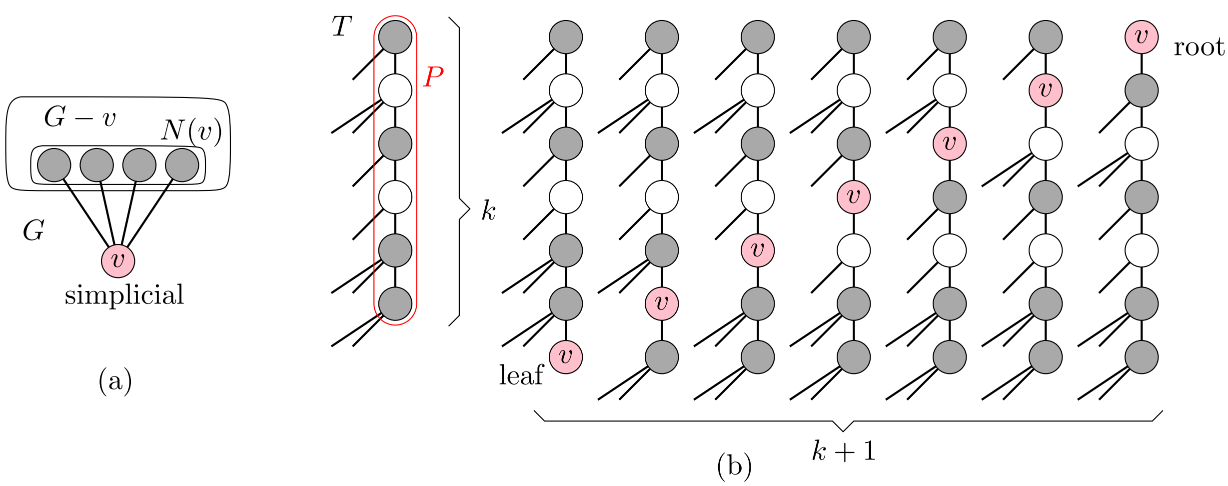

Figure 6: Illustration of the zigzag argument for elimination trees of a chordal graph. The vertices in the neighborhood $N(v)$ of v are shaded, whereas other vertices of G are white. The sloped edges in T connect to further subtrees (not shown).

from $N(v)$. We replace the elimination tree T for $G - v$ by the sequence of elimination trees of G obtained by inserting v in all possibly ways on the path P; see Figure 6 (b). In particular, if P has k vertices, then we obtain $k + 1$ elimination trees. This insertion is done alternatingly from leaf to root or root to leaf, i.e., in the resulting listing of elimination trees of G, the vertex v alternates or 'zigzags' between being root or leaf. In particular, for G being a complete graph, the resulting ordering of elimination trees and their corresponding permutations is exactly the Steinhaus-Johnson-Trotter ordering.

We will see that the appearance of chordal graphs in the aforementioned results on acyclic orientations and elimination trees is *not a coincidence*. In fact, our first main result gives a unified proof for both the results of [SSW93] and [CMM22] by introducing a suitable notion of chordality for hypergraphs (see Section 6.1).

5 From permutations to lattice quotients

The *inversion set* of a permutation $\pi = a_1 \cdots a_n$ is the set of pairs (a_i, a_j) that appear in the 'wrong' order, i.e., the set $\{(a_i, a_j) \mid 1 \le i < j \le n \text{ and } a_i > a_j\}$. If we order all permutations of $[n]$ by containment of their inversion sets, we obtain the *weak order* on permutations; see Figure 7 (a). The weak order forms a *lattice*, i.e., joins and meets are well-defined. Note that the cover relations in this lattice are precisely adjacent transpositions, i.e., the cover graph of this lattice is the skeleton of the permutohedron. Furthermore, the levels $0, \ldots, \binom{n}{2}$ correspond to the number of inversions.

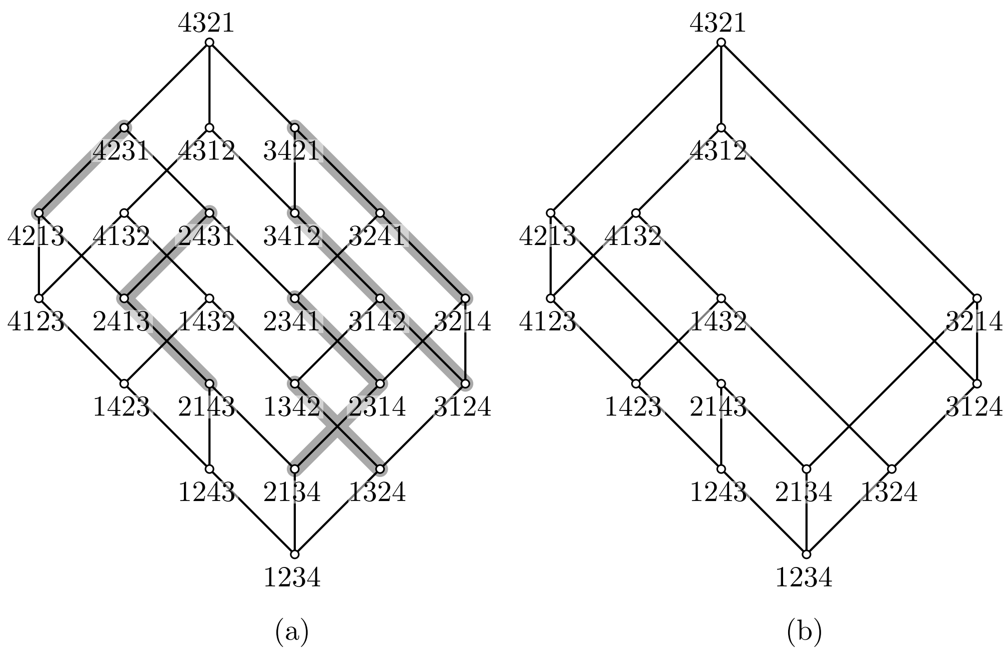

Figure 7: (a) Weak order on permutations with the sylvester congruence, with the equivalence classes drawn as gray bubbles; (b) its quotient is the Tamari lattice of 231-avoiding permutations.

A *lattice congruence* is an equivalence relation on a lattice that respects joins and meets, i.e., for any choice of representatives from two equivalence classes, their joins and meets must lie in the same equivalence class. A well-known example of a lattice congruence for the weak order is the *sylvester congruence*, defined as the transitive closure of the rewriting rule $\_b\_ca\_ \equiv \_b\_ac\_$, where $a < b < c$, i.e., whenever a permutation contains three symbols $a < b < c$ in the order b, c, a, with c and a directly next to each other, then this permutation belongs to the same equivalence class as the permutation obtained by transposing c and a. Figure 7 (a) shows the equivalence classes of this congruence, with 231-avoiding permutations as the minima of the equivalence classes. The *quotient* of some lattice congruence is the lattice obtained by 'contracting' each equivalence class to a single element; see Figure 7 (b). In this way, we obtain for example the Tamari lattice (Figure 7 (b)) and the Boolean lattice as quotients of suitable lattice congruences of the weak order on permutations. Let us also mention that the lattice of diagonal rectangulations [LR12] and the lattice of generic rectangulations [Mee19] arise as quotients of the weak order, and they have twisted Baxter permutations or 2-clumped permutations, respectively, as the minima of the equivalence classes.

The cover graphs of these quotient lattices are skeleta of polytopes known as *quotientopes* [PS19, PPR21]. We showed in [HHMW20, HM21] that the skeleton of any quotientope admits a Hamilton path, and this Hamilton path can be computed by a 'zigzag' strategy that generalizes the Steinhaus-Johnson-Trotter algorithm.

Pilaud [Pil22] generalized this notion of quotientopes as follows: He equipped the flip graph on acyclic orientations of a graph G with a poset structure. Specifically, he considers the containment order of the sets of reoriented arcs with respect to some acyclic reference orientation D of G. The cover relations are given by reorienting a single arc, and the levels of this poset correspond to the number of reoriented arcs; see Figure 8 (a). Pilaud characterized under which conditions on D this poset is a lattice, and he introduced lattice congruences and lattice quotients in this setting; see Figure 8 (b)+(c). Furthermore, he showed how to realize the cover graphs of those quotients as polytopes, generalizing the constructions from [PS19, PPR21]. We saw before that if G is a complete graph, then its acyclic orientations correspond to permutations, and arc flips correspond to adjacent transpositions, so in this special case Pilaud's lattice is precisely the weak order on permutations. In his paper, Pilaud raised the problem which of these generalized quotientopes (parametrized by a reference orientation D of some graph G) admit a Hamilton cycle. The second main result of our work addresses Pilaud's question, by showing that they all have a Hamilton path, which can be computed by a simple greedy algorithm, again following the 'zigzag' principle (see Section 6.2).

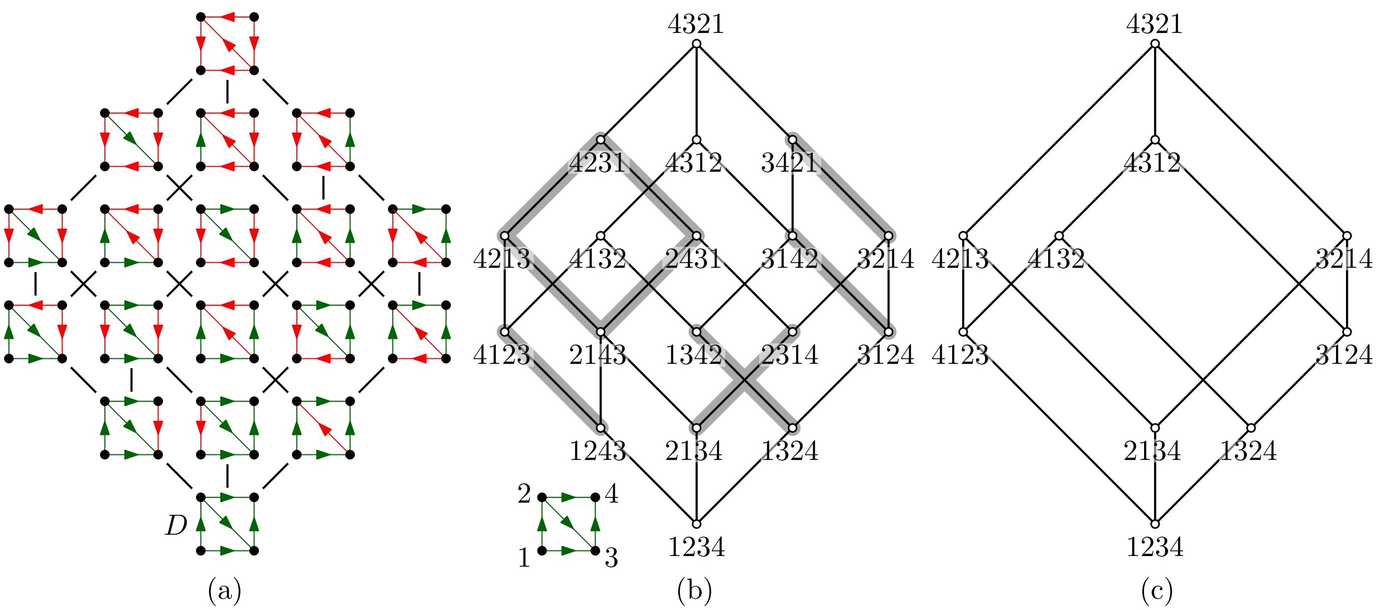

Figure 8: (a) Lattice of acyclic reorientations of a digraph D (reoriented arcs w.r.t. D are highlighted); (b) one of its lattice congruences, and encoding of acyclic orientations by permutations; (c) the resulting quotient lattice, and corresponding representative permutations.

6 Our results

We proceed to give an overview of the results of this contribution, explaining the main statements and connections to previous work. These new results are highlighted red in Figure 4. In this extended abstract, no formal proofs of our results are given. They can be found in the full journal version.

6.1 Acyclic orientations of hypergraphs. Our first main contribution is the generalization of the Savage-Squire-West Gray code for acyclic orientations of chordal graphs to acyclic orientations of hypergraphs, by introducing a suitable notion of chordality for hypergraphs. Our construction yields Hamilton paths on the skeleta of certain *hypergraphic polytopes* [BBM19, AA17], and in particular on chordal nestohedra [PRW08].

Furthermore, this generalization subsumes the construction of Gray codes for elimination trees of chordal graphs presented in [CMM22].

Given a hypergraph $\mathcal{H} = (V, \mathcal{E})$, where $\mathcal{E} \subseteq 2^V$, an *orientation* is a mapping $h : \mathcal{E} \to V$ such that $h(A) \in A$ for every hyperedge A of $\mathcal{H}$; see Figure 9 (a). The letter h stands for 'head': Every hyperedge designates one of its vertices as head. This orientation is *acyclic* if the digraph formed by all arcs $u \to v$ for every pair of distinct vertices $u, v \in V$ with $u, v \in A$ and $v = h(A)$ for some hyperedge $A \in \mathcal{E}$ is acyclic; see Figure 9 (b). This definition clearly generalizes the notion of an acyclic digraph. It is a special case of a more general definition recently used in a similar context by Benedetti, Bergeron, and Machacek [BBM19].

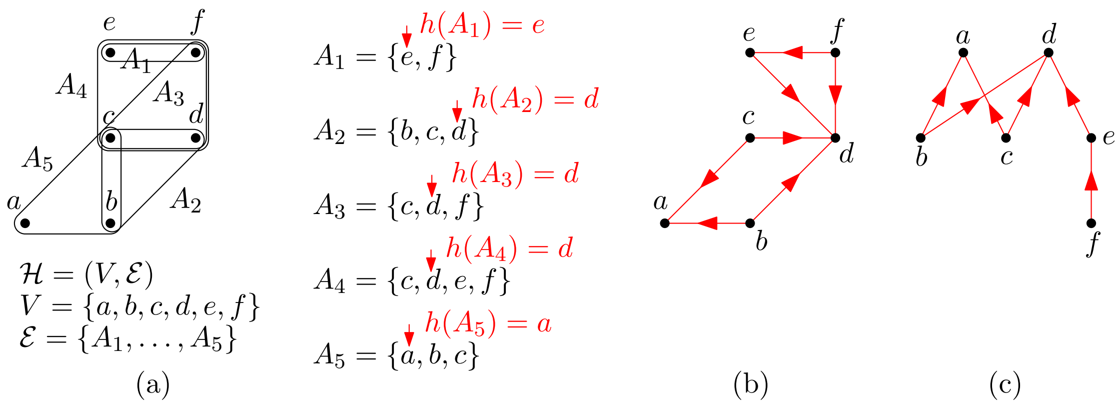

Figure 9: (a) Acyclic orientation of a hypergraph; (b) corresponding acyclic digraph; (c) corresponding poset (whose cover graph is the transitive reduction of (b)).

Given a chordal graph G, repeatedly removing one of its simplicial vertices yields a *perfect elimination ordering (peo)* of the graph. In fact, it is well-known that a graph G admits a perfect elimination ordering if and only if G is chordal [FG65]. We generalize the notion of perfect elimination order of chordal graphs to what we call *hyperfect elimination order* of hypergraphs. We then apply the aforementioned Hartung-Hoang-Mütze-Williams generation framework to obtain a Gray code for the acyclic orientations of a hypergraph in hyperfect elimination order, using the 'zigzag' idea common to Figures 3 and 6. The flip operation in an orientation h of a hypergraph $\mathcal{H} = (V, \mathcal{E})$ consists of picking two vertices $u, v \in V$, and for all hyperedges $A \in \mathcal{E}$ with $u, v \in A$ and $h(A) = u$ we instead define $h(A) := v$ (provided that the resulting orientation is acyclic); see Figure 10 (c).

The results from [CMM22] on elimination trees of chordal graphs can be recovered as a special case of our new results as follows; see Figure 10: Given a graph $G = (V, E)$, its *graphical building set* is the hypergraph $\mathcal{H} = (V, \mathcal{E})$ such that

$$\mathcal{E} := \{U \subseteq V \mid G[U] \text{ is connected}\},$$

where $G[U]$ is the subgraph of G induced by U, i.e., we consider all connected subgraphs of G as hyperedges. An elimination tree T of G with root v corresponds to the acyclic orientation of $\mathcal{H}$ in which every hyperedge $A \in \mathcal{E}$ with $v \in A$ satisfies $h(A) = v$, and this condition holds recursively for all subtrees. Consequently, acyclic orientations of $\mathcal{H}$ are in one-to-one correspondence with elimination trees of G, and the aforementioned flip operation on the hypergraph corresponds to a rotation in the elimination tree. Our new Gray code on acyclic orientations of hypergraphs in hyperfect elimination order thus yields as a special case the Gray code on elimination trees of chordal graphs presented in [CMM22].

The notions of building set and chordal building set have been defined abstractly without reference to a graph by Postnikov [Pos09], and Postnikov, Reiner, and Williams [PRW08], and the corresponding flip graphs arise as skeleta of *chordal nestohedra*, a term also coined in [PRW08]. Connections with acyclic orientations of hypergraphs have been investigated by Benedetti, Bergeron, and Machacek [BBM19]. We thus also obtain a simple constructive proof that chordal nestohedra admit a Hamilton path, directly yielding Gray codes for so-called $\mathcal{B}$-*forests* [PRW08].

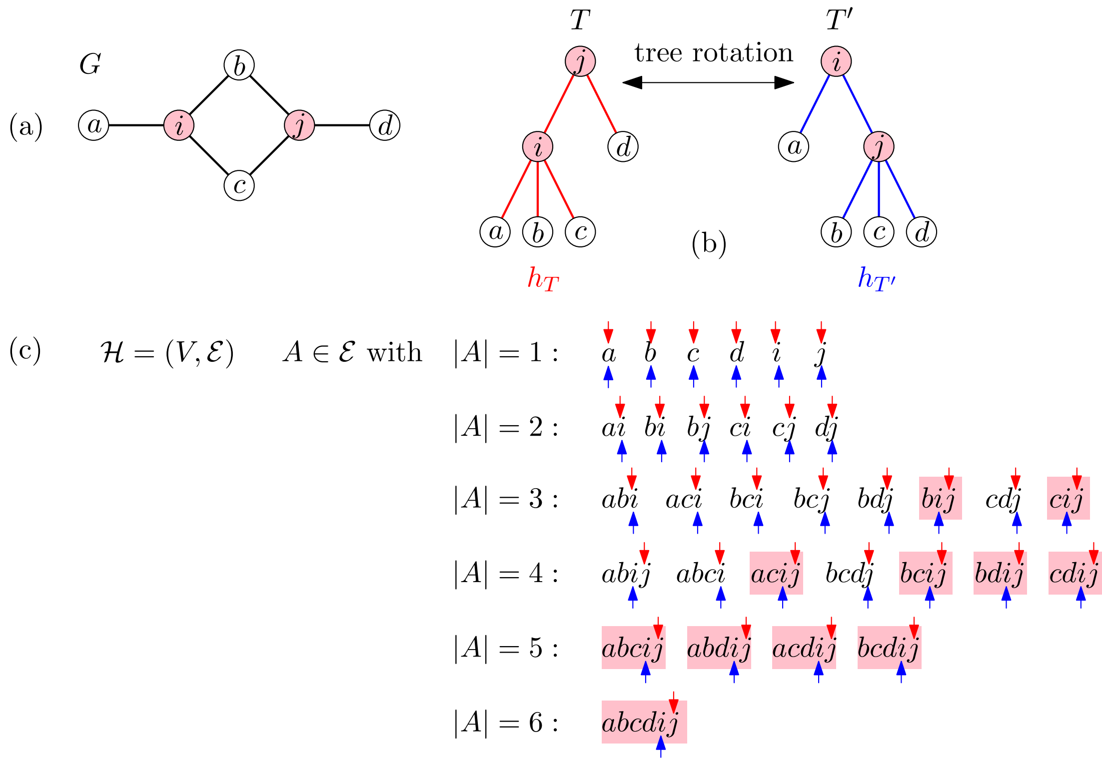

Figure 10: (a) A graph G; (b) two elimination trees T and T' of G that differ in a tree rotation; (c) corresponding two acyclic orientations h_T and $h_{T'}$ of the graphical building set of G, and flip operation between them. The orientations h_T and $h_{T'}$ differ only on the highlighted hyperedges that contain both i and j.

6.2 Quotients of acyclic reorientation lattices. Recall from Section 5 the definition of the poset of acyclic orientations of a graph G with respect to a reference orientation D of G. Pilaud [Pil22] characterized when this poset is a lattice. The following definitions are illustrated in Figure 11. Specifically, a digraph D is called *vertebrate* if the transitive reduction of every induced subgraph of D is a forest. It is easy to see that vertebrate implies acyclic. Furthermore, D is called *filled* if for any directed path $v_1 \to \cdots \to v_k$ in D, if the arc $v_1 \to v_k$ belongs to D, then all arcs $v_i \to v_j$, $1 \le i < j \le k$, also belong to D. A digraph is called *skeletal* if it is both vertebrate and filled. Pilaud [Pil22, Thm. 1+Thm. 3] showed that the acyclic reorientation poset of D is a lattice if and only if D is vertebrate, and that this lattice is semidistributive if and only if D is filled. He also raised the following question in his paper.

PROBLEM 1. ([PIL22, PROBLEM 51]) *Given a skeletal (i.e., vertebrate and filled) digraph D, do all cover graphs of lattice quotients of the acyclic reorientation lattice of D admit a Hamilton cycle?*

Our second main contribution is to address Pilaud's question, by showing that those cover graphs all have a Hamilton path, which can be computed by a simple greedy algorithm, generalizing Steinhaus-Johnson-Trotter. This also yields an algorithmic proof that the corresponding generalized quotientopes admit a Hamilton path. Furthermore, our result encompasses all earlier results on quotients of the weak order on permutations [HHMW20, HM21], which are obtained as special case when D is an acyclic orientation of a complete graph.

In fact, our results hold not only for skeletal digraphs D, but for a slightly larger class. Specifically, a *peo-consistent* digraph D has a source or sink v (i.e., all arcs incident with v are either outgoing or incoming, respectively) whose neighborhood is a clique, and $D - v$ is also peo-consistent or empty. A straightforward induction shows that peo-consistent implies vertebrate. The key fact we establish in our work is that skeletal implies peo-consistent. We thus have the following inclusions among classes of digraphs, which are strict (see

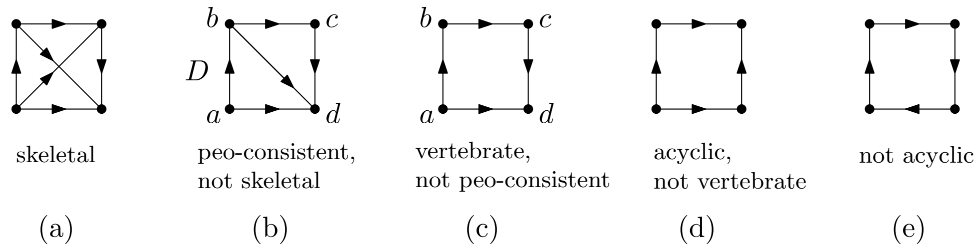

Figure 11: Illustration of various classes of digraphs: (b) is peo-consistent, as the neighborhood of the source a is a clique, and $D - a$ is also peo-consistent; it is not skeletal, as the path $a \to b \to c \to d$ is not filled; (c) is vertebrate, as the transitive reduction is the path $a \to b \to c \to d$; it is not peo-consistent, as a is a source and d is a sink, but none of their neighborhoods is a clique; (d) is not vertebrate, as the transitive reduction is the full graph, which is not a forest; (e) is a directed cycle.

Figure 11):

$$\text{skeletal} \subset \text{peo-consistent} \subset \text{vertebrate} \subset \text{acyclic}. \tag{6.1}$$

It is not difficult to see that an undirected graph has a peo-consistent orientation if and only if it is chordal. We also observe that an undirected graph admits a skeletal orientation only if it is *strongly chordal* [Far83].

We emphasize that our aforementioned results only guarantee a Hamilton path, whereas Pilaud's question asks for a Hamilton cycle. On the other hand, our results hold for a slightly larger class of graphs, and they yield a simple algorithm. As mentioned before, in the special case of elimination trees of chordal graphs G, there are interesting cases where the Hamilton path computed by our algorithm is actually a Hamilton cycle, i.e., the first and last elimination tree differ only in a tree rotation. Specifically, this happens if G is chordal and 2-connected; see [CMM22]. A special case of this is when G is a complete graph, in which case our algorithm specializes to the Steinhaus-Johnson-Trotter algorithm for permutations, which produces a cyclic Gray code.

6.3 Efficient generation algorithms. We briefly discuss the computational efficiency of the Gray code algorithms derived from our work. Those are summarized in the bottom part of the boxes in Figure 4. All four algorithms mentioned at the bottom level in Figure 4 can be implemented to output each new object in time $\mathcal{O}(1)$.

Our first algorithmic contribution is to turn the Savage-Squire-West Gray code for acyclic orientations of a chordal graph G into an algorithm that generates each acyclic orientation in time $\mathcal{O}(\log \omega)$ on average, where $\omega = \omega(G)$ is the clique number of G. For this algorithm, we represent each acyclic orientation as an adjacency matrix, which allows constant-time orientation queries for each arc. In the following we write m and n for the number of edges and vertices of G, respectively. Clearly, we have $\omega \leq n$, which yields the more generous bound $\mathcal{O}(\log n)$ for our Gray code algorithm. For comparison, Barbosa and Szwarcfiter [BS99] described an algorithm to generate each acyclic orientation of an arbitrary graph in time $\mathcal{O}(m + n)$ on average. Moreover, Conte, Grossi, Marino, and Rizzi [CGMR18] provided an algorithm that generates each acyclic orientation in time $\mathcal{O}(m)$, and this bound holds in every iteration. Their approach generalizes to the setting where some vertices can be prescribed as sources, at the cost of a higher running time. However, none of these algorithms produces a Gray code listing, i.e., they do not yield Hamilton paths on the corresponding graphical zonotopes, unlike our Gray code.

Secondly, we improve the running time of the algorithm for generating the elimination trees of a chordal graph G presented in [CMM22] from $\mathcal{O}(m + n)$ to $\mathcal{O}(\sigma)$ time on average per generated elimination tree, where $\sigma = \sigma(G)$ is the number of edges of the maximum induced star of G. Note that if G is a complete graph on n vertices, then this improvement brings the running time from $\mathcal{O}(n^2)$ down to $\mathcal{O}(1)$, i.e., we recover the constant-time feature of Steinhaus-Johnson-Trotter.

Generalizing these algorithmic techniques, we can implement our Gray code for all acyclic orientations of a hypergraph $\mathcal{H} = (V, \mathcal{E})$ in hyperfect elimination order in time $\mathcal{O}(\Delta)$ per generated acyclic orientation, where $\Delta = \Delta(\mathcal{H}) := \max_{v \in V} |\{A \in \mathcal{E} \mid v \in A\}|$ denotes the maximum degree.

For lattice congruences and their quotients, it is difficult to provide meaningful statements about running times because of representation issues. Specifically, for the weak order on permutations of $[n]$, there are double-exponentially in n many different lattice congruences [HM21, Thm. 18]. Specifying the congruence as input of an algorithm therefore takes exponential space in general. Consequently, one cannot improve much upon a naive specification of the congruence as a full list of equivalence classes as input of the algorithm. However, given such a specification as input, there is no value in generating it efficiently.

7 The permutation language framework

In a recent line of work, Hartung, Hoang, Mütze, and Williams [HHMW20] introduced a far-ranging generalization of the Steinhaus-Johnson-Trotter algorithm, which yields efficient Gray code algorithms for a large variety of combinatorial objects, based on encoding them as permutations. So far, the framework has been applied successfully to obtain Gray codes for pattern-avoiding permutations [HHMW22], lattice congruences of the weak order on permutations [HM21], different families of rectangulations [MM21a, MM21b] (a rectangulation is a subdivision of a rectangle into smaller rectangles), and elimination trees of chordal graphs [CMM22]. The methods described in this work extend the reach of this framework, and make it applicable to generate even more general classes of objects, namely acyclic orientations of hypergraphs, which in particular subsumes the earlier results in [HM21] and [CMM22]. In the following we summarize the key methods and results provided by this generation framework.

7.1 Jumps in permutations. We use S_n to denote the set of all permutations of $[n]$. Furthermore, we use $\mathrm{id}_n = 12\cdots n$ to denote the identity permutation, and $\varepsilon \in S_0$ to denote the empty permutation. A permutation $\pi = a_1 \cdots a_n$ is *peak-free* if it does not contain any triple $a_{i-1} < a_i > a_{i+1}$. For any $\pi \in S_{n-1}$ and any $1 \leq i \leq n$, we write $c_i(\pi) \in S_n$ for the permutation obtained from π by inserting the new largest value n at position i of π, i.e., if $\pi = a_1 \cdots a_{n-1}$ then $c_i(\pi) = a_1 \cdots a_{i-1}\, n\, a_i \cdots a_{n-1}$. Moreover, for $\pi \in S_n$, we write $p(\pi) \in S_{n-1}$ for the permutation obtained from π by removing the largest entry n. Given a permutation $\pi = a_1 \cdots a_n$ with a substring $a_i \cdots a_{i+d}$ with $d > 0$ and $a_i > a_{i+1}, \ldots, a_{i+d}$, a *right jump of the value a_i by d steps* is a cyclic left rotation of this substring by one position to $a_{i+1} \cdots a_{i+d} a_i$. Similarly, given a substring $a_{i-d} \cdots a_i$ with $d > 0$ and $a_i > a_{i-d}, \ldots, a_{i-1}$, a *left jump of the value a_i by d steps* is a cyclic right rotation of this substring to $a_i a_{i-d} \cdots a_{i-1}$. For example, a right jump of the value 5 in the permutation 265134 by 2 steps yields 261354.

7.1.1 A simple greedy algorithm. The main ingredient of the framework is the following simple greedy algorithm to generate a set of permutations $L_n \subseteq S_n$. We say that a jump is *minimal* with respect to L_n, if every jump of the same value in the same direction by fewer steps creates a permutation that is not in L_n.

Algorithm J *(Greedy minimal jumps).* This algorithm attempts to greedily generate a set of permutations $L_n \subseteq S_n$ using minimal jumps starting from an initial permutation $\pi_0 \in L_n$.

J1. [Initialize] Visit the initial permutation π_0.

J2. [Jump] Generate an unvisited permutation from L_n by performing a minimal jump of the largest possible value in the most recently visited permutation. If no such jump exists, or the jump direction is ambiguous, then terminate. Otherwise visit this permutation and repeat J2.

Note that Algorithm J is a generalization of Williams' greedy description of the Steinhaus-Johnson-Trotter algorithm given in Section 1. Indeed, if $L_n = S_n$ is the set of all permutations of $[n]$, then minimal jumps correspond to adjacent transpositions.

Note that by the definition of step J2, Algorithm J never visits any permutation twice. The following key result provides a sufficient condition on the set L_n to guarantee that Algorithm J succeeds to list all permutations from L_n. This condition is captured by the following closure property of the set L_n. A set of permutations $L_n \subseteq S_n$ is called a *zigzag language*, if either $n = 0$ and $L_0 = \{\varepsilon\}$, or if $n \geq 1$ and $L_{n-1} := \{p(\pi) \mid \pi \in L_n\}$ is a zigzag language satisfying either one of the following conditions:

(z1) For every $\pi \in L_{n-1}$ we have $c_1(\pi) \in L_n$ and $c_n(\pi) \in L_n$.

(z2) We have $L_n = \{c_n(\pi) \mid \pi \in L_{n-1}\}$.

THEOREM 7.1. ([HHMW22]) *Given any zigzag language of permutations L_n and initial permutation $\pi_0 = \mathrm{id}_n$,* *Algorithm J visits every permutation from L_n exactly once.*

It was already argued in [HHMW22] that more generally, any peak-free permutation can be used as initial permutation π_0 for Algorithm J.

We emphasize that Algorithm J can be made *history-free*, i.e., by introducing suitable auxiliary arrays, step J2 can be performed without maintaining any previously visited permutations in order to decide which jump to perform; for details see [MM21a, Sec. 5.1+8.7]. The running time of this algorithm is then only determined by the time it takes to decide membership of a permutation in the zigzag language L_n. We should think of L_n as a set of permutations defined by some property, such as for example 'permutations that avoid the pattern 231' or 'permutations that encode acyclic orientations of some graph' (recall Figures 7 (b) and 8 (b), respectively), rather than an explicitly given set. After all, if the set was already provided explicitly as input, then there would be no point in generating it; recall the discussion in Section 6.3.

7.1.2 Inductive description of the same ordering. In the same way that the Steinhaus-Johnson-Trotter ordering can be defined both greedily or inductively, the ordering produced by Algorithm J can also be defined inductively, as we show next.

Specifically, given a zigzag language L_n, we write $J(L_n)$ for the ordering of all permutations from L_n produced by Algorithm J when initialized with $\pi_0 = \mathrm{id}_n$. For any $\pi \in L_{n-1}$ we let $\overrightarrow{c}(\pi)$ be the sequence of all $c_i(\pi) \in L_n$ for $i = 1, 2, \ldots, n$, starting with $c_1(\pi)$ and ending with $c_n(\pi)$, and we let $\overleftarrow{c}(\pi)$ denote the reverse sequence, i.e., it starts with $c_n(\pi)$ and ends with $c_1(\pi)$. In words, those sequences are obtained by inserting into π the new largest value n from left to right, or from right to left, respectively, in all possible positions that yield a permutation from L_n, skipping the positions that yield a permutation that is not in L_n. It was shown in [HHMW22] that the sequence $J(L_n)$ can be described inductively as follows: If $n = 0$ then we have $J(L_0) = \varepsilon$, and if $n \geq 1$ then we consider the finite sequence $J(L_{n-1}) =: \pi_1, \pi_2, \ldots$ and we have

$$J(L_n) = \overleftarrow{c}(\pi_1), \overrightarrow{c}(\pi_2), \overleftarrow{c}(\pi_3), \overrightarrow{c}(\pi_4), \ldots$$

if condition (z1) holds, and

$$J(L_n) = c_n(\pi_1), c_n(\pi_2), c_n(\pi_3), c_n(\pi_4), \ldots$$

if condition (z2) holds. In words, if condition (z1) holds then this sequence is obtained from the previous sequence by inserting the new largest value n in all possible positions alternatingly from right to left, or from left to right, in a 'zigzag' fashion. The case where condition (z2) holds is exceptional, as we only append n to each permutation of the previous sequence.

7.1.3 How we apply Algorithm J in this work. To prove our results on acyclic orientations of hypergraphs discussed in Section 6.1, we proceed as follows: Given a hypergraph $\mathcal{H}$ in hyperfect elimination order, we label its vertices with $1, 2, \ldots, n$ according to this ordering, and we encode any of its acyclic orientations as a permutation on $[n]$, in such a way that the set of permutations obtained for all acyclic orientations of $\mathcal{H}$ is a zigzag language. By Theorem 7.1 we can thus apply Algorithm J to generate this zigzag language in Gray code order, and in a final step we interpret the jumps in permutations performed by Algorithm J in terms of flip operations on acyclic orientations of $\mathcal{H}$. The key insight that makes this work is that when vertices are in hyperfect elimination order, the posets defined by the acyclic orientations have the *unique parent-child* property, namely that every vertex has at most one parent and one child in the poset.

To prove our results on quotients of acyclic reorientations lattices discussed in Section 6.2, we proceed as follows: Given a peo-consistent digraph D, we label its vertices with $1, \ldots, n$ according to this ordering, and we encode any of its acyclic orientations as a permutation on $[n]$; see Figure 8 (b). For a given lattice congruence on the reorientation lattice of D, we select a set of representatives, one permutation from each equivalence class, such that those representative permutations form a zigzag language; see Figure 8 (c). We show that for peo-consistent digraphs, the equivalence classes of any lattice congruence have a simple projection property that enables selecting representatives in an inductive 'zigzag'-like way. It then follows that the jumps in permutations performed by Algorithm J correspond to steps along cover edges of the lattice quotient.

References

[AA17] M. Aguiar and F. Ardila. Hopf monoids and generalized permutahedra. To appear in *Mem. Amer. Math. Soc.*; preprint available at `https://arxiv.org/abs/1709.07504`, 2017.

[BBM19] C. Benedetti, N. Bergeron, and J. Machacek. Hypergraphic polytopes: combinatorial properties and antipode. *J. Comb.*, 10(3):515–544, 2019.

[BS99] V. C. Barbosa and J. L. Szwarcfiter. Generating all the acyclic orientations of an undirected graph. *Inform. Process. Lett.*, 72(1-2):71–74, 1999.

[CD06] M. P. Carr and S. L. Devadoss. Coxeter complexes and graph-associahedra. *Topology Appl.*, 153(12):2155–2168, 2006.

[CGMR18] A. Conte, R. Grossi, A. Marino, and R. Rizzi. Efficient enumeration of graph orientations with sources. *Discrete Appl. Math.*, 246:22–37, 2018.

[CMM22] J. Cardinal, A. I. Merino, and T. Mütze. Efficient generation of elimination trees and graph associahedra. In J. Naor and N. Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9–12, 2022*, pages 2128–2140. SIAM, 2022.

[Dev09] S. L. Devadoss. A realization of graph associahedra. *Discrete Math.*, 309(1):271–276, 2009.

[Far83] M. Farber. Characterizations of strongly chordal graphs. *Discrete Math.*, 43(2-3):173–189, 1983.

[FG65] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15:835–855, 1965.

[Gra53] F. Gray. Pulse code communication, 1953. March 17, 1953 (filed Nov. 1947). U.S. Patent 2,632,058.

[Gre77] C. Greene. Acyclic orientations. In *Proceedings of the NATO Advanced Study Institute held in Berlin (West Germany), September 1-10, 1976*, 1977.

[GZ83] C. Greene and T. Zaslavsky. On the interpretation of Whitney numbers through arrangements of hyperplanes, zonotopes, non-Radon partitions, and orientations of graphs. *Trans. Amer. Math. Soc.*, 280(1):97–126, 1983.

[HHMW20] E. Hartung, H. P. Hoang, T. Mütze, and A. Williams. Combinatorial generation via permutation languages. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms*, pages 1214–1225. SIAM, Philadelphia, PA, 2020.

[HHMW22] E. Hartung, H. P. Hoang, T. Mütze, and A. Williams. Combinatorial generation via permutation languages. I. Fundamentals. *Trans. Amer. Math. Soc.*, 375(4):2255–2291, 2022.

[HM21] H. P. Hoang and T. Mütze. Combinatorial generation via permutation languages. II. Lattice congruences. *Israel J. Math.*, 244(1):359–417, 2021.

[Knu11] D. E. Knuth. *The Art of Computer Programming. Vol. 4A. Combinatorial algorithms. Part 1.* Addison-Wesley, Upper Saddle River, NJ, 2011.

[Lin86] N. Linial. Hard enumeration problems in geometry and combinatorics. *SIAM J. Algebraic Discrete Methods*, 7(2):331–335, 1986.

[LR12] S. Law and N. Reading. The Hopf algebra of diagonal rectangulations. *J. Combin. Theory Ser. A*, 119(3):788–824, 2012.

[LRvBR93] J. M. Lucas, D. Roelants van Baronaigien, and F. Ruskey. On rotations and the generation of binary trees. *J. Algorithms*, 15(3):343–366, 1993.

[Mee19] E. Meehan. The Hopf algebra of generic rectangulations. `https://arxiv.org/abs/1903.09874`, 2019.

[MM21a] A. Merino and T. Mütze. Combinatorial generation via permutation languages. III. Rectangulations. To appear in *Discrete Comput. Geom.*; preprint available at `https://arxiv.org/abs/2103.09333`, 2021.

[MM21b] A. Merino and T. Mütze. Efficient generation of rectangulations via permutation languages. In *37th International Symposium on Computational Geometry*, volume 189 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. 54, 18. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021.

[MP16] T. Manneville and V. Pilaud. Graph properties of graph associahedra. *Sém. Lothar. Combin.*, 73:Art. B73d, 31, [2014–2016].

[Müt22] T. Mütze. Combinatorial Gray codes—an updated survey. `https://arxiv.org/abs/2202.01280`, 2022.

[Pil22] V. Pilaud. Acyclic reorientation lattices and their lattice quotients. To appear in *Proceedings of the 34th International Conference on Formal Power Series and Algebraic Combinatorics (FPSAC 2022)*; preprint available at `https://arxiv.org/abs/2111.12387`, 2022.

[Pos09] A. Postnikov. Permutohedra, associahedra, and beyond. *Int. Math. Res. Not. IMRN*, 2009(6):1026–1106, 2009.

[Pou14] L. Pournin. The diameter of associahedra. *Adv. Math.*, 259:13–42, 2014.

[PPR21] A. Padrol, V. Pilaud, and J. Ritter. Shard polytopes. *Sém. Lothar. Combin.*, 85B:Art. 11, 12, 2021.

[PRW08] A. Postnikov, V. Reiner, and L. Williams. Faces of generalized permutohedra. *Doc. Math.*, 13:207–273, 2008.

[PS19] V. Pilaud and F. Santos. Quotientopes. *Bull. Lond. Math. Soc.*, 51(3):406–420, 2019.

[Rea12] N. Reading. From the Tamari lattice to Cambrian lattices and beyond. In *Associahedra, Tamari lattices and related structures*, volume 299 of *Prog. Math. Phys.*, pages 293–322. Birkhäuser/Springer, Basel, 2012.

[Rea16] 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.

[RSW12] F. Ruskey, J. Sawada, and A. Williams. Binary bubble languages and cool-lex order. *J. Combin. Theory Ser. A*, 119(1):155–169, 2012.

[Rus16] F. Ruskey. *Combinatorial Gray Code*, pages 342–347. Springer New York, New York, NY, 2016.

[Sav97] C. D. Savage. A survey of combinatorial Gray codes. *SIAM Rev.*, 39(4):605–629, 1997.

[SSW93] C. D. Savage, M. B. Squire, and D. B. West. Gray code results for acyclic orientations. In *Proceedings of the Twenty-fourth Southeastern International Conference on Combinatorics, Graph Theory, and Computing (Boca Raton, FL, 1993)*, volume 96, pages 185–204, 1993.

[Sta07] R. P. Stanley. An introduction to hyperplane arrangements. In *Geometric combinatorics*, volume 13 of *IAS/Park City Math. Ser.*, pages 389–496. Amer. Math. Soc., Providence, RI, 2007.

[STT88] D. D. Sleator, R. E. Tarjan, and W. P. Thurston. Rotation distance, triangulations, and hyperbolic geometry. *J. Amer. Math. Soc.*, 1(3):647–681, 1988.

[SW18] J. Sawada and A. Williams. A Hamilton path for the sigma-tau problem. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 568–575. SIAM, Philadelphia, PA, 2018.

[Wil09] A. Williams. Loopless generation of multiset permutations using a constant number of variables by prefix shifts. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 987–996. SIAM, Philadelphia, PA, 2009.

[Wil13] A. Williams. The greedy Gray code algorithm. In *Algorithms and data structures*, volume 8037 of *Lecture Notes in Comput. Sci.*, pages 525–536. Springer, Heidelberg, 2013.