Permutree sorting

Vincent Pilaud, Viviane Pons, Daniel Tamayo Jiménez

PERMUTREE SORTING

VINCENT PILAUD, VIVIANE PONS, AND DANIEL TAMAYO JIMÉNEZ

ABSTRACT. Generalizing stack sorting and c-sorting for permutations, we define the permutree sorting algorithm. Given two disjoint subsets U and D of $\{2,\dots,n-1\}$, the (U,D)-permutree sorting tries to sort the permutation $\pi \in \mathfrak{S}_n$ and fails if and only if there are $1 \le i < j < k \le n$ such that π contains the subword jki if $j \in U$ and kij if $j \in D$. This algorithm is seen as a way to explore an automaton which either rejects all reduced expressions of π, or accepts those reduced expressions for π whose prefixes are all (U,D)-permutree sortable.

1. INTRODUCTION

The motivation of this paper is the classical family of *stack-sortable* permutations introduced by D. Knuth in his textbook [Knu69, Sect. 2.2.1] and characterized by the following equivalent conditions for a permutation $\pi \in \mathfrak{S}_n$:

(i) π is sent to the identity by the stack sorting S defined inductively by $S(\tau n \rho) := S(\tau)S(\rho)n$.
(ii) π avoids the pattern 231 (*i.e.* there is no $p < q < r$ such that $\pi_r < \pi_p < \pi_q$).
(iii) π is minimal among all linear extensions of a binary tree on n nodes (seen as a poset, where the nodes are labeled in inorder and the edges are oriented towards the leaves).
(iv) For $i < j < k$, the inversion set $\mathrm{inv}(\pi) := \{(\pi_p, \pi_q) \mid p < q \text{ and } \pi_p > \pi_q\}$ of π contains the inversion (k,j) as soon as it contains the inversion (k,i).
(v) π admits a reduced expression of the form $\pi = c_{I_1} \cdots c_{I_p}$ with nested subsets $I_1 \supseteq \cdots \supseteq I_p$, where $c_{\{i_1 < \cdots < i_j\}} := s_{i_j} \cdots s_{i_1}$ is a product of the simple transpositions $s_i := (i \;\; i+1)$.

It follows from (iii) that these permutations are counted by the Catalan number $C_n := \frac{1}{n+1}\binom{2n}{n}$.

In his seminal work on lattice congruences [Rea04, Rea06, Rea07a], N. Reading defined natural counterparts to conditions (iii), (iv), and (v) above, parametrized by the choice of a Coxeter element c in a finite Coxeter group W: the minimality in c-Cambrian classes, the c-alignment, and the c-sortability. (We skip the general definitions of these conditions here as we stick with the combinatorics of the symmetric group.) In the situation of the symmetric group $\mathfrak{S}_n$, we can think of a Coxeter element on $\mathfrak{S}_n$ as an orientation of an $(n-1)$-path, or equivalently as a partition of $\{2,\dots,n-1\}$ into two subsets U and D. The Cambrian analogues of the conditions (ii), (iii), (iv) and (v) above are the following equivalent conditions for a permutation $\pi \in \mathfrak{S}_n$:

(ii') For $i < j < k$, the permutation π does not contain the subword jki if $j \in U$ and kij if $j \in D$.
(iii') π is minimal among all linear extensions of a c-Cambrian tree on n nodes. A c-Cambrian tree is an oriented tree on $[n]$ where node j has one parent if $j \notin U$ and two parents if $j \in U$, and one child if $j \notin D$ and two children if $j \in D$, with an additional local condition at each node similar to the binary search tree condition [CP17].
(iv') For $i < j < k$, if $\mathrm{inv}(\pi)$ contains (k,i), then it also contains (k,j) if $j \in U$ and (j,i) if $j \in D$.
(v') π admits a reduced expression of the form $\pi = c_{I_1} \cdots c_{I_p}$ with nested subsets $I_1 \supseteq I_2 \supseteq I_p$, where $c_I := c_{i_1} \cdots c_{i_{|I|}}$ denotes the subword of $c := c_1 \cdots c_{n-1}$ indexed by $I := \{i_1 < \cdots < i_j\}$.

It turns out that for any Coxeter element c, the permutations satisfying these conditions are still counted by the Catalan number C_n.

These Cambrian combinatorics motivated the introduction of permutree combinatorics [PP18]. Permutrees generalize and interpolate between permutations, binary trees, and binary sequences, and explain the combinatorial, geometric, and algebraic similarities between them. The data is now given by two subsets U and D of $\{2,\dots,n-1\}$ that are not anymore required to form a partition

VP was supported by the French ANR (grants CAPPS 17 CE40 0018 and CHARMS 19 CE40 0017).

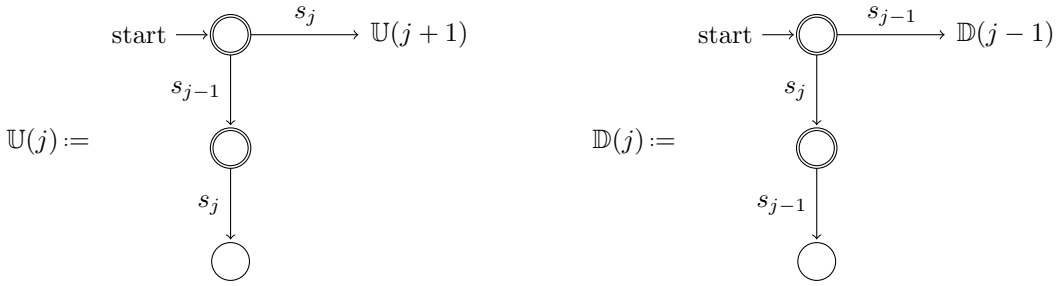

FIGURE 1. The automata $\mathbb{U}(j)$ (left) and $\mathbb{D}(j)$ (right) defined recursively.

of $\{2, \dots, n-1\}$ (they may intersect and may not cover all the set). It was proved in [PP18, CPP19] that the conditions (ii'), (iii'), and (iv') are still equivalent for a permutation $\pi \in \mathfrak{S}_n$. The number of permutations satisfying these conditions is called (U, D)-factorial-Catalan number and admits recursive formulae interpolating between the formulae for the factorial and for the Catalan number.

The objective of this paper is to discuss characterizations of permutree minimal permutations in terms of their reduced expressions. In other words, we aim at a condition playing the role of condition (v') and equivalent to conditions (ii'), (iii'), and (iv') for arbitrary subsets U and D of $\{2, \dots, n-1\}$. We first focus on the case where $U = \varnothing$ and $D = \{j\}$ for some $j \in \{2, \dots, n-1\}$, or the opposite. To characterize the permutree minimal permutations in terms of their reduced expressions in that situation, we use two automata $\mathbb{U}(j)$ and $\mathbb{D}(j)$ defined inductively as shown in Figure 1. The induction stops at $\mathbb{U}(n)$ and $\mathbb{D}(1)$, which are defined by deleting the transitions s_n and s_0 respectively in Figure 1. Figure 2 presents the complete automaton $\mathbb{U}(j)$ after all recursion is done, and Figure 3 shows the automata $\mathbb{U}(2)$, $\mathbb{D}(2)$, $\mathbb{U}(3)$, and $\mathbb{D}(3)$. In all these pictures the initial state is marked with "start", the accepting states are doubly circled, all transitions are labeled with simple transpositions s_i for $i \in [n-1]$, and all missing transitions are loops (we assume the reader familiar with basic automata theory, see for instance [HU79]). Our main tool is the following statement, proved in Section 2.

Theorem 1. *Fix $j \in \{2, \dots, n-1\}$. The following conditions are equivalent for $\pi \in \mathfrak{S}_n$:*

- *π admits a reduced expression accepted by the automaton $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$),*
- *π contains no subword jki (resp. kij) with $i < j < k$.*

Let us warn the reader on the fact that j is fixed in Theorem 1, while i and k are arbitrary such that $1 \le i < j < k \le n$. A priori, we should try all possible reduced expressions of π to decide if one is accepted by the automaton $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$). However, we can show that if π contains no subword jki (resp. kij) with $i < j < k$ and has a descent s_ℓ distinct from s_{j-1} (resp. s_j), then it has a reduced expression starting with s_ℓ and accepted by the automaton $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$). In other words, there is no loss of generality in starting constructing a reduced expression for π as long as we stay in the states of the top row of $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$). This yields a simple algorithm to construct a reduced expression accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$). It also yields natural tree structures on the permutations characterized by Theorem 1, which can be glanced upon in Figure 3. These algorithmic and combinatorial consequences of Theorem 1 are explored in Section 3. Most results of Sections 2 and 3 are stated with respect to both automata $\mathbb{U}(j)$ and $\mathbb{D}(j)$ but proved only for $\mathbb{U}(j)$ as all proofs for $\mathbb{D}(j)$ are symmetric.

Consider now arbitrary subsets U and D of $\{2, \dots, n-1\}$. It follows from Theorem 1 that a permutation is minimal in its (U, D)-permutree class if and only if it admits a reduced expression accepted by $\mathbb{U}(j)$ for each $j \in U$ and by $\mathbb{D}(j)$ for each $j \in D$. In general, the reduced expressions accepted by the automata $\mathbb{U}(j)$ for each $j \in U$ and by $\mathbb{D}(j)$ for each $j \in D$ are distinct. We prove however in Section 4 that there is a reduced expression simultaneously accepted by all these automata when U and D are disjoint.

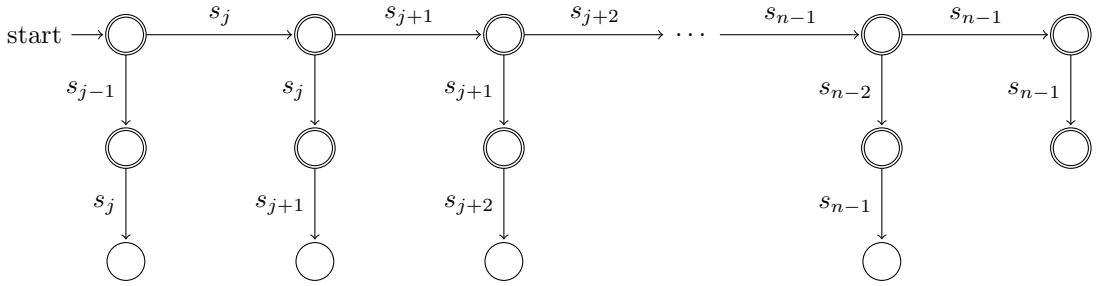

FIGURE 2. The complete automaton $\mathbb{U}(j)$.

Theorem 2. *Consider two disjoint subsets U and D of $\{2,\dots,n-1\}$. The following conditions are equivalent for $\pi \in \mathfrak{S}_n$:*

- *π admits a reduced expression accepted by all automata $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$,*
- *π contains no subword jki if $j \in U$ and kij if $j \in D$ for any $i < j < k$.*

Theorem 2 implies that given any permutation π avoiding jki if $j \in U$ and kij if $j \in D$, we can sort π while preserving these avoiding conditions. The resulting sorting procedures, that we call *(U,D)-permutree sorting*, are discussed in Section 4.4. For instance, stack sorting is a $(\{2,\dots,n-1\},\varnothing)$-permutree sorting.

Finally, in the particular situation when the subsets U and D form a partition of $\{2,\dots,n-1\}$, we actually show that the reduced expression simultaneously accepted by the automata $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$ is the c-sorting word of π as defined in [Rea07a]. This yields in particular an alternative proof that condition (v') characterizes the Cambrian minimal permutations. This new perspective on c-sortability is explored in Section 5.

2. Automata for reduced expressions

2.1. **Reduced expressions, automata, and subword avoiding.** We start with properly fixing the few notations needed in this paper. We consider the *symmetric group* $\mathfrak{S}_n$ of permutations of the set $[n] := \{1,\dots,n\}$. It is generated by the *transpositions* $s_i := (i \;\; i+1)$ for $i \in [n-1]$ which are involutions $s_i^2 = id$ and satisfy the commutation relations $s_i \cdot s_j = s_j \cdot s_i$ if $|i-j| > 1$ and the braid relations $s_i \cdot s_{i+1} \cdot s_i = s_{i+1} \cdot s_i \cdot s_{i+1}$. Note that we multiply permutations as usual, so that the left multiplication by s_i exchanges the entries with values i and $i+1$, while the right multiplication by s_i exchanges the entries at positions i and $i+1$. Each permutation π decomposes into products of transpositions of the form $\pi = s_{i_1} \cdots s_{i_k}$ with $i_1,\dots,i_k \in [n-1]$. The minimal number of transpositions in such a decomposition is the *length* $\ell(\pi)$ of π and the decompositions of length $\ell(\pi)$ are the *reduced expressions* for π.

Consider now the automata $\mathbb{U}(j)$ and $\mathbb{D}(j)$ described in the introduction, see Figures 1 to 3. We call a state *healthy*, *ill*, or *dead* depending on whether it belongs to the top, middle, or bottom row of the automata. Each state has $n-1$ possible transitions, one for each s_i for $i \in [n-1]$, but we only explicitly indicate the ones between different states. The automata $\mathbb{U}(j)$ and $\mathbb{D}(j)$ take as entry a reduced expression $s_{i_1} \cdots s_{i_\ell}$ for a permutation of $\mathfrak{S}_n$ and read it from left to right. We start at the initial state (marked with "start"), and at step t we follow the transition marked by the letter s_{i_t} if any, or stay in the current state otherwise. After ℓ steps, the reduced expression $s_{i_1} \cdots s_{i_\ell}$ is declared accepted if the current state is accepting (doubly circled, healthy or ill states), and rejected otherwise (dead states).

For a fixed $j \in \{2,\dots,n-1\}$, we say that a permutation π *avoids* jki (resp. kij) if for any $i < j < k$, the word jki (resp. kij) does not appear as a subword of the one-line notation of π, or said differently if there are no positions $p < q < r$ such that $\pi(r) < \pi(p) = j < \pi(q)$ (resp. $\pi(q) < \pi(r) = j < \pi(p)$). We insist on the fact that while the value j is fixed, i and k take all possible values such that $1 \le i < j < k \le n$. This convenient notion here should not be mixed

up with the notion of pattern avoidance where j is not fixed. For instance, a permutation avoids the pattern 231 if and only if it avoids jki for all $j \in \{2, \dots, n-1\}$.

Example 3. *The permutation* 42135 *avoids* $2ki$*,* $3ki$*, and* $4ki$ *(and therefore the pattern* 231*), but contains* $ki3$ *(and therefore the pattern* 312*) because its one-line notation contains* 423*.*

2.2. **Behavior under left multiplication.** In the perspective of proving Theorem 1, we study the two properties "π admits a reduced expression accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$)" and "π avoids jki (resp. kij)". In this section, we study the behavior of these properties under left multiplication. We treat separately the cases when we multiply by a permutation commuting with both s_{j-1} and s_j (Lemma 4), by s_{j-1} (Lemma 6), and by s_j (Lemma 8).

Lemma 4. *If two permutations* $\sigma, \tau \in \mathfrak{S}_n$ *are such that* $\sigma([j-1]) = [j-1]$*,* $\sigma(j) = j$*, and* $\sigma([n] \smallsetminus [j]) = [n] \smallsetminus [j]$ *and* $\ell(\sigma \cdot \tau) = \ell(\sigma) + \ell(\tau)$*, then:*

(1) τ *admits a reduced expression accepted by* $\mathbb{U}(j)$ *(resp.* $\mathbb{D}(j)$*) if and only if* $\sigma \cdot \tau$ *admits a reduced expression accepted by* $\mathbb{U}(j)$ *(resp.* $\mathbb{D}(j)$*),*

(2) τ *avoids* jki *(resp.* kij*) if and only if* $\sigma \cdot \tau$ *avoids* jki *(resp.* kij*).*

Proof. We deal with the two statements separately:

(1) The conditions on σ imply that none of its reduced expressions contain the transpositions s_{j-1} or s_j. Therefore, while reading any reduced expression for σ, the automaton $\mathbb{U}(j)$ stays in the initial state. The result immediately follows.

(2) Since σ permutes only values smaller than j between themselves and values greater than j between themselves, we see a subword jki with $i < j < k$ in τ if and only if we see a subword $jk'i'$ with $i' < j < k'$ in $\sigma \cdot \tau$, where $i' = \sigma(i)$ and $k' = \sigma(k)$. □

Example 5. *Consider* $j \coloneqq 4$ *and the permutations* $\sigma \coloneqq 312465 = s_2 \cdot s_1 \cdot s_5$*,* $\tau_1 \coloneqq 143256 = s_3 \cdot s_2 \cdot s_3$*, and* $\tau_2 \coloneqq 124536 = s_3 \cdot s_4$*. Multiplying we obtain* $\sigma \cdot \tau_1 = 342165$ *and* $\sigma \cdot \tau_2 = 314625$*. Observe that*

(1) $\mathbb{U}(4)$ *accepts all reduced expressions of both* τ_1 *and* $\sigma \cdot \tau_1$ *on its first ill state, and rejects all reduced expressions of both* τ_2 *and* $\sigma \cdot \tau_2$*,*

(2) both τ_1 *and* $\sigma \cdot \tau_1$ *avoid* $4ki$*, while both* τ_2 *and* $\sigma \cdot \tau_2$ *contain* $4ki$*.*

Lemma 6. *If a permutation* $\tau \in \mathfrak{S}_n$ *has a reduced expression starting with* s_{j-1} *(resp.* s_j*) and accepted by* $\mathbb{U}(j)$ *(resp.* $\mathbb{D}(j)$*), then*

(1) τ *does not permute* j *and* $j+1$ *(resp.* $j-1$ *and* j*),*

(2) τ *avoids* jki *(resp.* kij*).*

Proof. Consider a reduced expression w starting with s_{j-1} and accepted by $\mathbb{U}(j)$. We deal with the two statements separately:

(1) Since w starts with s_{j-1}, the values $j-1$ and j are reversed in τ. If j and $j+1$ were also reversed in τ, we would obtain that $j-1$ and $j+1$ are reversed. It follows that w must contain a s_j at some point after the s_{j-1}. But this would lead to a dead state, contradicting the assumption that w is accepted.

(2) Let $\tau = s_{j-1} \cdot \rho$. Since any reduced expression of ρ cannot contain s_j (as w is be accepted by $\mathbb{U}(j)$), we have that $\rho([j]) = [j]$ and $\rho([n+1] \smallsetminus [j]) = [n+1] \smallsetminus [j]$ and find that ρ contains no subword ki with $i < j < k$. Therefore, τ avoids jki. □

Example 7. *Consider* $j \coloneqq 4$ *and the permutation* $\tau \coloneqq 413265$*, whose reduced expression* $s_3 \cdot s_5 \cdot s_2 \cdot s_1 \cdot s_3$ *is accepted by* $\mathbb{U}(4)$*. Observe that*

(1) τ *indeed does not permute the values* 4 *and* 5*,*

(2) τ *avoids* $4ki$*.*

Lemma 8. *If a permutation* $\tau \in \mathfrak{S}_n$ *does not permute* j *and* $j+1$ *(resp.* $j-1$ *and* j*), then*

(1) $s_j \cdot \tau$ *(resp.* $s_{j-1} \cdot \tau$*) admits a reduced expression accepted by* $\mathbb{U}(j)$ *(resp.* $\mathbb{D}(j)$*) if and only if* τ *admits a reduced expression accepted by* $\mathbb{U}(j+1)$ *(resp.* $\mathbb{D}(j-1)$*),*

(2) $s_j \cdot \tau$ *(resp.* $s_{j-1} \cdot \tau$*) avoids* jki *(resp.* kij*) if and only if* τ *avoids* $(j+1)ki$ *(resp.* $ki(j-1)$*).*

Proof. We deal with the two statements separately:

(1) Suppose that w is a reduced expression for τ accepted by $\mathbb{U}(j+1)$. Since τ does not permute j and $j+1$, we know that $s_j \cdot w$ is a reduced expression for $s_j \cdot \tau$, and it is accepted by $\mathbb{U}(j)$ by construction. Conversely assume that $s_j \cdot \tau$ admits a reduced expression w accepted by $\mathbb{U}(j)$. Since $s_j \cdot \tau$ permutes j and $j+1$, w must contain a s_j and cannot start by s_{j-1} by Lemma 6. Due to Lemma 4 we can also assume that w starts with s_j. Thus the suffix is a reduced expression for τ that is accepted by $\mathbb{U}(j+1)$.

(2) Observe that since j and $j+1$ are reversed in $s_j \cdot \tau$ and not in τ, the value $j+1$ cannot serve as k in a subword jki of $s_j \cdot \tau$ and the value j cannot serve as i in a subword $(j+1)ki$ in τ. The result thus immediately follow from the fact that the left multiplication by s_j only exchanges the values j and $j+1$. □

Example 9. *Consider $j := 4$ and the permutations $\tau_1 := 142536$ and $\tau_2 := 142563$ that do not permute 4 and 5. Multiplying we obtain $s_4 \cdot \tau_1 = 152436$ and $s_4 \cdot \tau_2 = 152463$. Observe that*

(1) the reduced expression $s_4 \cdot s_3 \cdot s_4 \cdot s_2$ of $s_4 \cdot \tau_1$ is accepted by $\mathbb{U}(4)$ and the reduced expression $s_3 \cdot s_4 \cdot s_2$ of τ_1 is accepted by $\mathbb{U}(5)$, while all reduced expressions of $s_4 \cdot \tau_2$ are rejected by $\mathbb{U}(4)$ and all reduced expressions of τ_2 are rejected by $\mathbb{U}(5)$,

(2) $s_4 \cdot \tau_1$ avoids $4ki$ and τ_1 avoids $5ki$, while $s_4 \cdot \tau_2$ contains 463 and τ_2 contains 463.

2.3. **Proof of Theorem 1.** With these lemmas in hand, we are now ready to show Theorem 1 that we repeat here for convenience.

Theorem 1. *Fix $j \in \{2, \dots, n-1\}$. The following conditions are equivalent for $\pi \in \mathfrak{S}_n$:*

- *π admits a reduced expression accepted by the automaton $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$),*
- *π contains no subword jki (resp. kij) with $i < j < k$.*

Proof of Theorem 1. We work by induction on the length of the permutations. Assume that a permutation π admits a reduced expression accepted by $\mathbb{U}(j)$. Let s_i be the first letter of this reduced expression and let τ be such that $\pi = s_i \cdot \tau$. We distinguish three cases:

- if $i = j-1$, then π avoids jki by Lemma 6 (2).
- if $i = j$, then τ admits a reduced expression accepted by $\mathbb{U}(j+1)$ by Lemma 8 (1). We obtain by induction that τ avoids $(j+1)ki$. Thus $\pi = s_j \cdot \tau$ avoids jki by Lemma 8 (2).
- otherwise, τ admits a reduced expression accepted by $\mathbb{U}(j)$ by Lemma 4 (1), so that τ avoids jki by induction. Thus $\pi = s_i \cdot \tau$ avoids jki by Lemma 4 (2).

In all three cases, we proved that π avoids jki.

Assume now that a permutation π avoids jki. Here, we have to be careful because not all reduced expressions for π will be accepted by $\mathbb{U}(j)$ *a priori*. So we have to construct a good reduced expression for π. We distinguish two cases:

- Assume first that there is $m > j$ such that π reverses j and m, and pick m minimal for this property. It follows that π reverses ℓ and m for all ℓ in $\{j, \dots, m-1\}$. In other words, π admits a reduced expression starting by the cyclic permutation $(j, j+1, ..., m) = s_{m-1} \cdot s_{m-2} \cdots s_{j+1} \cdot s_j$. Define $\sigma = s_{m-1} \cdot s_{m-2} \cdots s_{j+1}$ and τ such that $\pi = \sigma \cdot s_j \cdot \tau$ and so that this expression is reduced. By Lemmas 4 (2) and 8 (2), τ avoids $(j+1)ki$. By induction, we obtain that it admits a reduced expression accepted by $\mathbb{U}(j+1)$. By Lemmas 4 (1) and 8 (1), we conclude that π admits a reduced expression accepted by $\mathbb{U}(j)$.
- Assume now that j appears before all $m > j$ in π. Consider any reduced expression for π. If this expression is accepted by $\mathbb{U}(j)$, we are done. Otherwise, it first uses s_{j-1} (otherwise, j and some $m > j$ would be exchanged) and then s_j. Call i and k the two elements that are exchanged when the reduced expression first uses s_j. We have $i < j < k$ and jki in π (because j and k are not exchanged in π, and i and k are already exchanged so they will remain exchanged in π), a contradiction.

In both cases, we proved that π admits a reduced expression accepted by $\mathbb{U}(j)$. □

3. Structure of accepted reduced expressions

In this section, we explore some additional properties of the set of reduced expressions accepted by the automata $\mathbb{U}(j)$ and $\mathbb{D}(j)$ and derive relevant algorithmic and combinatorial consequences.

3.1. **The set of accepted reduced expressions.** Observe that a given permutation π may admit both accepted and rejected reduced expressions. For instance, the (non-simple) transposition $(j-1 \;\; j+1)$ has reduced expressions $s_j \cdot s_{j-1} \cdot s_j$ accepted by $\mathbb{U}(j)$ and $s_{j-1} \cdot s_j \cdot s_{j-1}$ rejected by $\mathbb{U}(j)$. However, Propositions 10 to 12 below show that the set of accepted reduced expressions satisfies the following three principles:

- **Who can do more can do less!** — The set of accepted reduced words is closed by prefix.
- **When health goes, everything goes!** — If π admits an accepted reduced expression, then π admits an accepted reduced expression starting with any descent that remains in the healthy states.
- **All roads lead to Rome!** — All accepted reduced expressions for π end at the same state.

Proposition 10. *The set of reduced words accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$) is closed by prefix.*

Proof. This immediately follows from the fact that the set of reduced words is closed by prefix, and that the set of accepting states of $\mathbb{U}(j)$ is connected and contains the initial state. □

Proposition 11. *Let $\ell \in [n-1]$ distinct from $j-1$ (resp. j). A permutation $\pi \in \mathfrak{S}_n$ that avoids jki (resp. kij) and reverses ℓ and $\ell+1$ admits a reduced expression starting with s_ℓ and accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$).*

Proof. Since π reverses ℓ and $\ell+1$, it admits a reduced expression of the form $\pi = s_\ell \cdot \tau$. Now consider two cases depending on the value of ℓ:

- If $\ell = j$, then τ avoids $(j+1)ki$ by Lemma 8 (2), thus τ has a reduced expression accepted by $\mathbb{U}(j+1)$ by Theorem 1, and we conclude by Lemma 8 (1).
- Otherwise, ℓ is neither $j-1$ nor j, so that τ avoids jki by Lemma 4 (2), thus τ has a reduced expression accepted by $\mathbb{U}(j)$ by Theorem 1, and we conclude by Lemma 4 (1). □

Proposition 12. *Given a permutation $\pi \in \mathfrak{S}_n$, all the reduced expressions for π accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$) end at the same state.*

To prove Proposition 12, it would be enough to check that any two reduced words accepted by $\mathbb{U}(j)$ that differ by a single commutation or a single braid relation indeed end at the same state. However, we prefer to prove instead the following stronger but more technical version of Proposition 12.

Proposition 13. *For a permutation $\pi \in \mathfrak{S}_n$, let $\operatorname{ninv}^j(\pi) = |\{(j,i) \mid i < j \text{ and } \pi^{-1}(i) > \pi^{-1}(j)\}|$ and $\operatorname{ninv}_j(\pi) = |\{(k,j) \mid j < k \text{ and } \pi^{-1}(j) > \pi^{-1}(k)\}|$.*

(i) if $\operatorname{ninv}^j(\pi) = 0$, then all reduced expressions for π end at the same healthy state of $\mathbb{U}(j)$,
(ii) if $\operatorname{ninv}_j(\pi) = 0$, then all reduced expressions for π end at the same state of $\mathbb{U}(j)$, which might be healthy if π avoids ji, ill if π contains ji but avoids jki, or dead if π contains jki,
(iii) if $\operatorname{ninv}^j(\pi) \neq 0 \neq \operatorname{ninv}_j(\pi)$, all accepted reduced expressions for π end at the same ill state of $\mathbb{U}(j)$ while the rejected reduced expressions may end at distinct dead states of $\mathbb{U}(j)$.

Moreover, all reduced expressions for π accepted by $\mathbb{U}(j)$ end in the $(\operatorname{ninv}_j(\pi)+1)$st column of $\mathbb{U}(j)$. A similar statement holds for $\mathbb{D}(j)$ by exchanging $\operatorname{ninv}_j(\pi)$ and $\operatorname{ninv}^j(\pi)$.

Proof. The proof works by induction on the length of π. Consider an arbitrary reduced expression w for π, starting with a transposition s_ℓ, and write $w = s_\ell \cdot w'$ and $\pi = s_\ell \cdot \tau$. Observe that:

- if $\ell \notin \{j-1, j\}$, then s_ℓ loops in $\mathbb{U}(j)$, $\operatorname{ninv}^j(\pi) = \operatorname{ninv}^j(\tau)$ and $\operatorname{ninv}_j(\pi) = \operatorname{ninv}_j(\tau)$,
- if $\ell = j$, then s_j goes to $\mathbb{U}(j+1)$, $\operatorname{ninv}^j(\pi) = \operatorname{ninv}^{j+1}(\tau)$ and $\operatorname{ninv}_j(\pi) = \operatorname{ninv}_{j+1}(\tau) + 1$,
- if $\ell = j-1$, then s_{j-1} goes to the first ill state of $\mathbb{U}(j)$, $\operatorname{ninv}^j(\pi) = \operatorname{ninv}^{j+1}(\tau) + 1$ and $\operatorname{ninv}_j(\pi) = \operatorname{ninv}_{j+1}(\tau)$.

By induction, we obtain that the reduced expression w' for τ ends as predicted in the statement. The previous observations ensure that the reduced expression w for π also does. □

Example 14. *We present an example of each case:*

(i) For $\pi := 4312$*, we have that* $\mathrm{ninv}^2(\pi) = 0$ *and all of its* 5 *reduced expressions end at the third healthy state of* $\mathbb{U}(2)$*.*

(ii) For $\pi := 32145$ *(resp.* $\pi := 43215$*, resp.* $\pi := 43251$*), we have* $\mathrm{ninv}_4(\pi) = 0$ *and all its* 2 *(resp.* 16*, resp.* 35*) reduced expressions end at the first healthy (resp. ill, resp. dead) state of* $\mathbb{U}(4)$*.*

(iii) For $\pi := 4321$*, we have* $\mathrm{ninv}^2(\pi) = |\{(2,1)\}| = 1$ *and* $\mathrm{ninv}_2 = |\{(3,2),(4,2)\}| = 2$*. Among the* 16 *reduced expressions of* π*, the automaton* $\mathbb{U}(2)$ *accepts* 7 *at its third ill state, rejects* 7 *at its first dead state, and rejects the other* 2 *at its second dead state.*

3.2. **Finding accepted reduced expressions.** Proposition 11 has a strong algorithmic consequence. Imagine we want to test whether a permutation $\pi \in \mathfrak{S}_n$ is minimal in its permutree class for $U = \{j\}$ and $D = \varnothing$. Of course, the quickest way is to check for all $i < j < k$ whether π contains the subword jki. But since this interpretation will be lost beyond type A, let us impose the use of reduced expressions for π to make this test. While it would be *a priori* necessary to check all reduced expressions on the automaton $\mathbb{U}(j)$, Proposition 11 enables us to construct without loss of generality a candidate reduced expression for π and we will just neeed to check that this one is accepted by $\mathbb{U}(j)$. Somewhat dually, one can also construct a reduced word accepted by $\mathbb{U}(j)$ that is a reduced expression for π if and only if π avoids jki. This is done in the following algorithm, that we call $(\{j\},\varnothing)$*-permutree sorting*. The reader is invited to write down the symmetric $(\varnothing,\{j\})$-permutree sorting. We will discuss further permutree sorting in Section 4.4.

Algorithm 1: $(\{j\},\varnothing)$-permutree sorting

Data: a permutation $\pi \in \mathfrak{S}_n$ and an integer $j \in [n]$
Result: a reduced word accepted by $\mathbb{U}(j)$, candidate reduced expression for π

$w := \varepsilon$
repeat
  if $\exists\, \ell \neq j-1$ *such that* ℓ *and* $\ell+1$ *are reversed in* π **then**
    $\pi := s_\ell \cdot \pi, \quad w := w \cdot s_\ell$
    if $\ell = j$ **then** $j := j+1$
if $j-1$ *and* j *are reversed in* π **then**
  $\pi := s_{j-1} \cdot \pi, \quad w := w \cdot s_{j-1}$
  $w := w \cdot w' \cdot w''$ where w' sorts $\pi_{[j]}$ and w'' sorts $\pi_{[n]\smallsetminus[j]}$
return w

Example 15. *Let us present the* $(2,\varnothing)$*-permutree sorting algorithm in action for the permutations* $\pi_1 := 3421$ *and* $\pi_2 := 4231$*. The steps of the algorithm are presented in Table 1. Each row contains the states of the permutation* π *and of the word* w *and the current values of* j *and* ℓ *in use at each step. Notice that for* $\pi_1 := 3421$ *the algorithm ends with the identity, which coincides with the fact that* $\pi_1 := 3421$ *avoids* $2ki$*. In contrast, for* $\pi_2 := 4231$ *the algorithm ends with the permutation* 1243*, meaning that* π_2 *is not* $(\{2\},\varnothing)$*-sortable, which coincides with the fact that* $\pi_2 := 4231$ *contains* $2ki$*.*

π_1	w_1	j_1	ℓ_1
3421	ε	2	2
2431	s_2	3	1
1432	$s_2 \cdot s_1$	3	3
1342	$s_2 \cdot s_1 \cdot s_3$	4	2
1243	$s_2 \cdot s_1 \cdot s_3 \cdot s_2$	4	3
1234	$s_2 \cdot s_1 \cdot s_3 \cdot s_2 \cdot s_3$	4	

π_2	w_2	j_2	ℓ_2
4231	ε	2	3
3241	s_3	2	2
2341	$s_3 \cdot s_2$	3	1
1342	$s_3 \cdot s_2 \cdot s_1$	3	2
1243	$s_3 \cdot s_2 \cdot s_1 \cdot s_2$	3	

TABLE 1. The $(\{2\},\varnothing)$-permutree sorting of $\pi_1 := 3421$ and $\pi_2 := 4231$.

Corollary 16. *For any permutation π and $j \in \{2, \dots, n-1\}$, Algorithm 1 returns a reduced word w accepted by $\mathbb{U}(j)$ with the property that w is a reduced expression for π if and only if π avoids jki.*

Proof. This algorithm constructs a candidate reduced word for π while following the automaton $\mathbb{U}(j)$ and prioritizing healthy states over ill states. Lines 2 to 5 start by all possible transitions s_ℓ that remain in healthy states, updating j to $j+1$ when $\ell = j$ according to Lemma 8 (if condition at line 5). When we have exhausted all these transitions, if we need to go to an ill state (if condition at line 6) applying s_{j-1}, then we are not anymore allowed to use s_j and we obtain a candidate reduced word by sorting independently the first j positions of π with a reduced expression in $\{s_1, \dots, s_{j-1}\}^*$ and the last $[n] \smallsetminus [j]$ positions of π with a reduced expression in $\{s_{j+1}, \dots, s_{n-1}\}^*$. The resulting reduced word w is clearly accepted by $\mathbb{U}(j)$ because we never allow the transition from an ill state to the corresponding dead state. If w is a reduced expression for π, then π avoids jki by Theorem 1. Conversely, if π avoids jki, then w must be a reduced expression for π since the choice to start with s_ℓ is valid in lines 2 to 5 by Proposition 11 and forced in lines 6 to 8 (since all reduced expressions of π then start by s_ℓ). □

Remark 17. *We really wrote Algorithm 1 as a sorting algorithm. It first tries to sort the permutation $\pi \in \mathfrak{S}_n$ while avoiding to swap $j-1$ and j for a certain token j (and changing the token when swapping j and $j+1$). Once it is forced to swap $j-1$ and j, it tries to sort the permutation π while avoiding to swap any value of $[j]$ with a value of $[n] \smallsetminus [j]$. If we were only interested in deciding whether the permutation π is $(\{j\}, \varnothing)$-sortable, then we could stop and accept the permutation as soon as we reach $j = n$, and we could just check at line 8 of the algorithm whether $\pi([j]) = [j]$ and $\pi([n] \smallsetminus [j]) = [n] \smallsetminus [j]$.*

3.3. **Generating trees on accepted reduced expressions.** Propositions 10 and 12 also have a relevant consequence, more combinatorial this time. Namely, they naturally define generating trees for the $(\{j\}, \varnothing)$-permutree minimal permutations, following certain special reduced expressions for them. To construct these trees, pick an arbitrary priority order $\prec$ on $\{s_1, \dots, s_{n-1}\}$. For a $(\{j\}, \varnothing)$-permutree minimal permutation $\pi \in \mathfrak{S}_n$, denote by $\pi(\{j\}, \varnothing, \prec)$ the $\prec$-lexicographic minimal reduced expression for π that is accepted by $\mathbb{U}(j)$. Denote by $\mathcal{R}(n, \{j\}, \varnothing, \prec)$ the set of reduced words of the form $\pi(\{j\}, \varnothing, \prec)$ for all $(\{j\}, \varnothing)$-permutree minimal permutations $\pi \in \mathfrak{S}_n$. The following statement is an analogue of Proposition 10.

Proposition 18. *The set $\mathcal{R}(n, \{j\}, \varnothing, \prec)$ is closed by prefix.*

Proof. Consider a reduced expression $w = u \cdot v$ where u is not in $\mathcal{R}(n, \{j\}, \varnothing, \prec)$. If u is not accepted by $\mathbb{U}(j)$, neither is w by Proposition 10. Otherwise, there exists a reduced expression u' representing the same permutation as u, accepted by $\mathbb{U}(j)$ and $\prec$-lexicographic smaller than u. By Proposition 12, the reduced expressions u and u' end at the same state of $\mathbb{U}(j)$. Therefore, if $w = u \cdot v$ is accepted by $\mathbb{U}(j)$, so is $u' \cdot v$. Since $u' \cdot v$ is $\prec$-lexicographically smaller than $u \cdot v$ and represents the same permutation, this ensures that w is not in $\mathcal{R}(n, \{j\}, \varnothing, \prec)$. □

Proposition 18 yields a natural generating tree for $\mathcal{R}(n, \{j\}, \varnothing, \prec)$ where the parent of a reduced word w is obtained by deleting its last letter. Replacing each reduced expression by the corresponding permutation, this provides a generating tree for the $(\{j\}, \varnothing)$-permutree minimal permutations of $\mathfrak{S}_n$. Of course there is a similar generating tree for the $(\varnothing, \{j\})$-permutree minimal permutations of $\mathfrak{S}_n$. Figure 3 presents these generating trees for $n = 4$ and $j = 2, 3$, with the priority order $s_1 \prec s_2 \prec s_3$. It is natural to draw these trees on top of the Hasse diagram of the right weak order on permutations, defined by inclusion of inversion sets. In other words, the cover relations in weak order correspond to the swap of the values at two consecutive positions in a permutation, *i.e.* to a right multiplication by a simple transposition. The edges of the trees corresponding to the right multiplications by s_1, s_2 and s_3 are colored by blue, red, and green respectively.

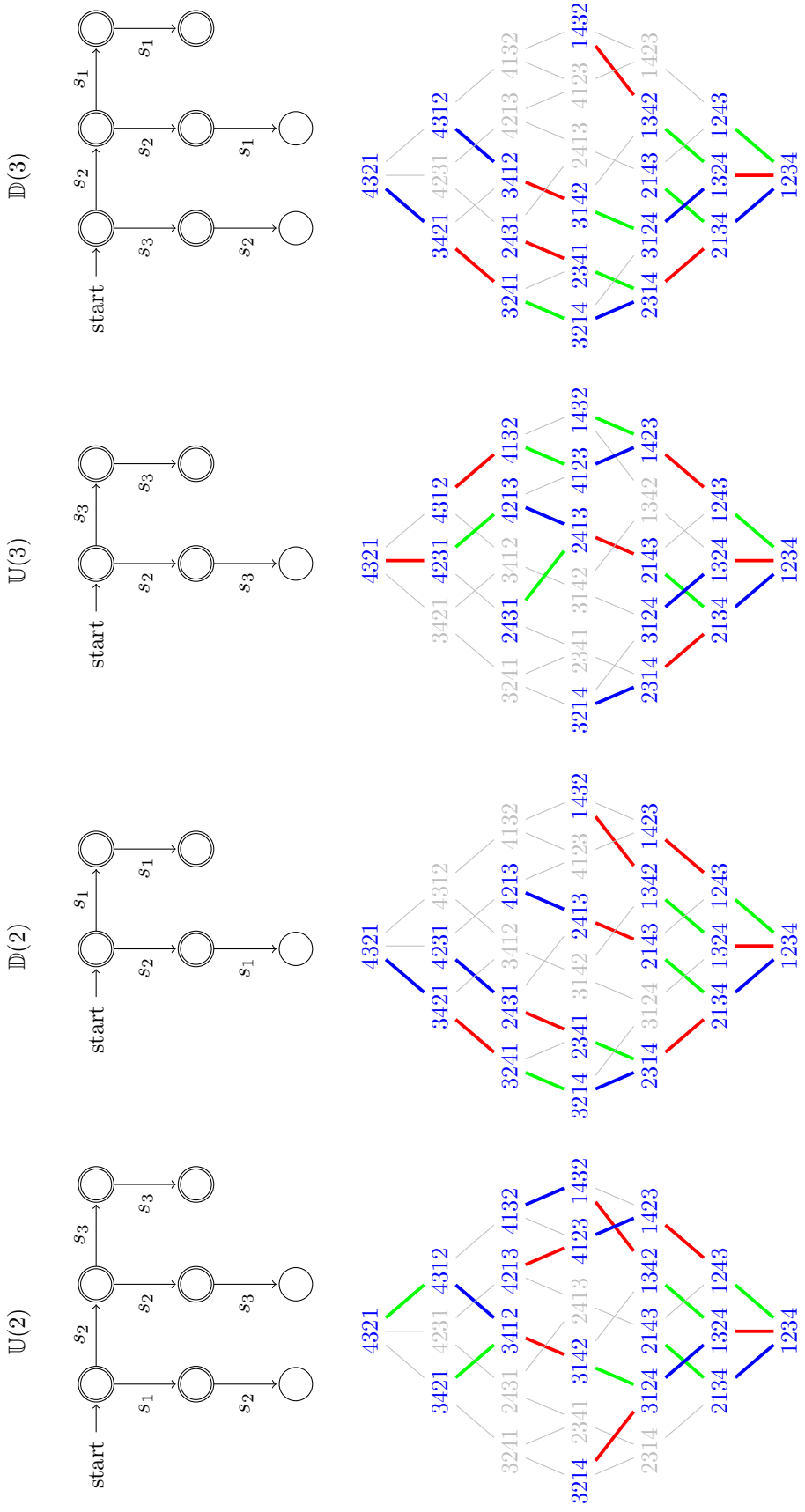

FIGURE 3. Generating trees for the $(\{j\}, \varnothing)$- and $(\varnothing, \{j\})$-permutree minimal permutations of $\mathfrak{S}_4$, with priority order $s_1 \prec s_2 \prec s_3$.

4. Intersection of automata

We now consider arbitrary subsets U and D of $\{2, \dots, n-1\}$. We already know from [PP18] and Theorem 1 that the following conditions are equivalent for $\pi \in \mathfrak{S}_n$:

(i) the permutation π is minimal in its (U, D)-permutree class,
(ii) for $i < j < k$, the permutation π does not contain the subword jki if $j \in U$ and kij if $j \in D$,
(iii) for each $j \in U$ (each $j \in D$), there is a reduced expression for π accepted by $\mathbb{U}(j)$ (resp. by $\mathbb{D}(j)$).

A natural question is whether there is a reduced expression simultaneously accepted by all these automata. We start with an example showing that this is not always the case.

Example 19. *For $j \in \{2, \dots, n-1\}$, consider $U = \{j\} = D$, and $\pi = s_{j-1} \cdot s_j \cdot s_{j-1} = s_j \cdot s_{j-1} \cdot s_j$. Then, the expression $s_{j-1} \cdot s_j \cdot s_{j-1}$ is accepted by $\mathbb{D}(j)$ but not by $\mathbb{U}(j)$, while the expression $s_j \cdot s_{j-1} \cdot s_j$ is accepted by $\mathbb{U}(j)$ but not by $\mathbb{D}(j)$.*

This example clearly extends to all subsets U and D of $\{2, \dots, n-1\}$ with a non-empty intersection. In contrast, we will now show that this situation cannot occur when U and D are disjoint.

4.1. Proof of Theorem 2.

Example 19 motivates Theorem 2 that we repeat here for convenience.

Theorem 2. *Consider two disjoint subsets U and D of $\{2, \dots, n-1\}$. The following conditions are equivalent for $\pi \in \mathfrak{S}_n$:*

- *π admits a reduced expression accepted by all automata $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$,*
- *π contains no subword jki if $j \in U$ and kij if $j \in D$ for any $i < j < k$.*

Proof. The direct implication is immediate from Theorem 1. For the converse implication, consider a permutation π that avoids jki for $j \in U$ and kij for $j \in D$. Consider $\bar{U} := \{j \in U \mid \operatorname{ninv}_j(\pi) \neq 0\}$ and $\bar{D} := \{j \in D \mid \operatorname{ninv}^j(\pi) \neq 0\}$. By Proposition 13 (ii), any reduced expression for π is accepted by $\mathbb{U}(j)$ for $j \in U \smallsetminus \bar{U}$ and by $\mathbb{D}(j)$ for $j \in D \smallsetminus \bar{D}$. We can therefore assume that $\bar{U} = U$ and $\bar{D} = D$ and one of them is non-empty, say $\bar{U} = U \neq \varnothing$. Let $j_\circ := \max(U)$ and m be minimal such that $j_\circ < m$ and $\pi^{-1}(j_\circ) > \pi^{-1}(m)$. By minimality of m, we obtain that π contains the subword $mj_\circ\ell$ for any $j_\circ < \ell < m$. It implies that

- ℓ is neither in U by maximality of $j_\circ$, nor in D by assumption on π, for all $j_\circ < \ell < m$,
- π reverses ℓ and m for all ℓ in $\{j_\circ, \dots, m-1\}$, so that π admits a reduced expression of the form $\pi = s_{m-1} \cdots s_{j_\circ} \cdot \tau$.

Lemmas 4 (2) and 8 (2) ensure that

- τ avoids jki for all $j \in U \smallsetminus \{j_\circ\}$ and kij for all $j \in D \smallsetminus \{m\}$ (because $j_\circ, \dots, m-1$ are all distinct from $j-1$ and j in these cases by the second observation above),
- τ avoids $(j_\circ + 1)ki$,
- if $m \in D$, then τ avoids $kij_\circ$.

By induction, it implies that τ admits a reduced expression w simultaneously accepted by all automata $\mathbb{U}(j)$ for $j \in U \smallsetminus \{j_\circ\}$ and $j = j_\circ + 1$, and all automata $\mathbb{D}(j)$ for $j \in D \smallsetminus \{m\}$ and $j = j_\circ$ if $m \in D$. By Lemmas 4 (1) and 8 (1), we conclude that $s_{m-1} \cdots s_{j_\circ} \cdot w$ is a reduced expression for π simultaneously accepted by all $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$. □

4.2. Intersection of automata.

Theorem 2 can be rephrased in terms of intersection of automata. Recall that the intersection of some automata $\mathbb{A}_1, \dots, \mathbb{A}_p$ is the automaton $\mathbb{A} = \bigcap_{i \in [p]} \mathbb{A}_i$ such that a word is accepted by $\mathbb{A}$ if and only if it is accepted by all $\mathbb{A}_1, \dots, \mathbb{A}_p$. A state of the automaton $\mathbb{A}$ is p-tuple formed by states of the automata $\mathbb{A}_1, \dots, \mathbb{A}_p$, and a transition t simultaneously changes all entries of the p-tuple corresponding to states modified by t. See [HU79, p. 59–60] for details. We denote by $\mathbb{P}(U, D)$ the intersection of the automata $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$. We thus obtain the following statement.

Corollary 20. *When U and D are disjoint, the following conditions are equivalent for $\pi \in \mathfrak{S}_n$:*

- *π admits a reduced expression accepted by the automaton $\mathbb{P}(U, D)$,*
- *π contains no subword jki if $j \in U$ and kij if $j \in D$ with $i < j < k$.*

We say that a state of $\mathbb{P}(U, D)$ is *healthy* (resp. *ill*, resp. *dead*) when the corresponding states in $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$ are all healthy (resp. contain at least one ill state, but no dead one, resp. contains at least one dead state). Figure 4 illustrates the automata $\mathbb{P}(\{4\}, \{2\})$ when $n = 5$ (left), $\mathbb{P}(\{3\}, \{2\})$ for $n = 4$ (middle), and $\mathbb{P}(\{2\}, \{4\})$ for $n = 5$ (right). For the first two automata, we have drawn the complete automata on top, and their skeleta on the bottom. Here, we call skeleton a simplification of the automaton that recognizes the same reduced words. It is obtained using the fact that the word is rejected as soon as we reach a dead state, and that the automata $\mathbb{U}(n)$ and $\mathbb{D}(1)$ accept all reduced expressions. For the last automaton, the complete intersection is too big, so we only draw the reachable healthy states. In the picture, we color the transitions in red, blue, or purple depending on whether only $\mathbb{U}$, only $\mathbb{D}$, or both $\mathbb{U}$ and $\mathbb{D}$ change state.

4.3. **The set of accepted reduced expressions of $\mathbb{P}(U, D)$.** Applying the principles of Section 3.1 to each automaton $\mathbb{U}(j)$ for $j \in U$ and $\mathbb{D}(j)$ for $j \in D$, we derive similar principles for the automaton $\mathbb{P}(U, D)$. The following statements are direct consequences of Propositions 10 to 12.

Proposition 21. *The set of reduced words accepted by $\mathbb{P}(U, D)$ is closed by prefix.*

Proposition 22. *If a permutation π avoids jki for $j \in U$ and kij for $j \in D$, and admits a reduced expression starting with s_ℓ such that the transition s_ℓ leads to an healthy state of $\mathbb{P}(U, D)$, then it admits a reduced expression starting with s_ℓ and accepted by $\mathbb{P}(U, D)$.*

Proposition 23. *Given a permutation $\pi \in \mathfrak{S}_n$, all the reduced expressions for π accepted by $\mathbb{P}(U, D)$ end at the same state.*

4.4. **Permutree sorting.** We have seen that for any (U, D)-permutree minimal permutation, the set of reduced expressions for π that are accepted by $\mathbb{P}(U, D)$ is non-empty (by Corollary 20) and closed by prefix (by Proposition 21). Therefore, it is possible to sort π passing only through (U, D)-permutree minimal permutations along the way. This motivates the following definition.

Definition 24. *An (U, D)-permutree sorting algorithm is a sorting procedure such that*

- *applied to an (U, D)-permutree minimal permutation π, it only passes through (U, D)-permutree minimal permutations and arrives to the identity permutation,*
- *it fails to sort a non (U, D)-permutree minimal permutation π.*

Example 25. *The stack sorting algorithm is a $(\{2, \dots, n-1\}, \varnothing)$-permutree sorting algorithm.*

Said differently, any procedure that looks for a reduced expression accepted by $\mathbb{P}(U, D)$ gives a (U, D)-permutree sorting algorithm. For instance, Algorithm 1 is a $(\{j\}, \varnothing)$-permutree sorting algorithm. We generalize it in the following algorithm, where we opted for a recursive style. As in Algorithm 1, the algorithm will read the automaton $\mathbb{P}(U, D)$ without actually constructing it. To virtually follow the edges of the automaton $\mathbb{P}(U, D)$, we use the following two operations on our sets U and D:

$$\mathrm{moveU}(U, \ell) = \begin{cases} U & \text{if } \ell \notin U, \\ (U \smallsetminus \{\ell\}) \cup \{\ell + 1\} & \text{if } \ell \in U, \end{cases}$$

$$\mathrm{moveD}(D, \ell) = \begin{cases} D & \text{if } \ell + 1 \notin D, \\ (D \smallsetminus \{\ell + 1\}) \cup \{\ell\} & \text{if } \ell + 1 \in D. \end{cases}$$

Algorithm 2: (U, D)-permutree sorting

Data: a permutation $\pi \in \mathfrak{S}_n$ and two disjoint subsets U and D of $[n]$
Result: a reduced word accepted by $\mathbb{P}(U, D)$, candidate reduced expression for π

if $\exists\, \ell \in [n-1]$ *such that* ℓ *and* $\ell + 1$ *are reversed in* π, *and* $\ell + 1 \notin U$ *and* $\ell \notin D$ **then**
  return $s_\ell \cdot \mathrm{permutreeSort}(s_\ell \cdot \pi,\ \mathrm{moveU}(U, \ell),\ \mathrm{moveD}(D, \ell))$
if $\exists\, \ell \in [n-1]$ *such that* ℓ *and* $\ell + 1$ *are reversed in* π,
and ($\ell + 1 \notin U$ *or* $\pi([\ell + 1]) = [\ell + 1]$) *and* ($\ell \notin D$ *or* $\pi([\ell - 1]) = [\ell - 1]$) **then**
  return $s_\ell \cdot \mathrm{permutreeSort}(s_\ell \cdot \pi,\ \mathrm{moveU}(U \smallsetminus \{\ell + 1\}, \ell),\ \mathrm{moveD}(D \smallsetminus \{\ell\}, \ell))$
return ε

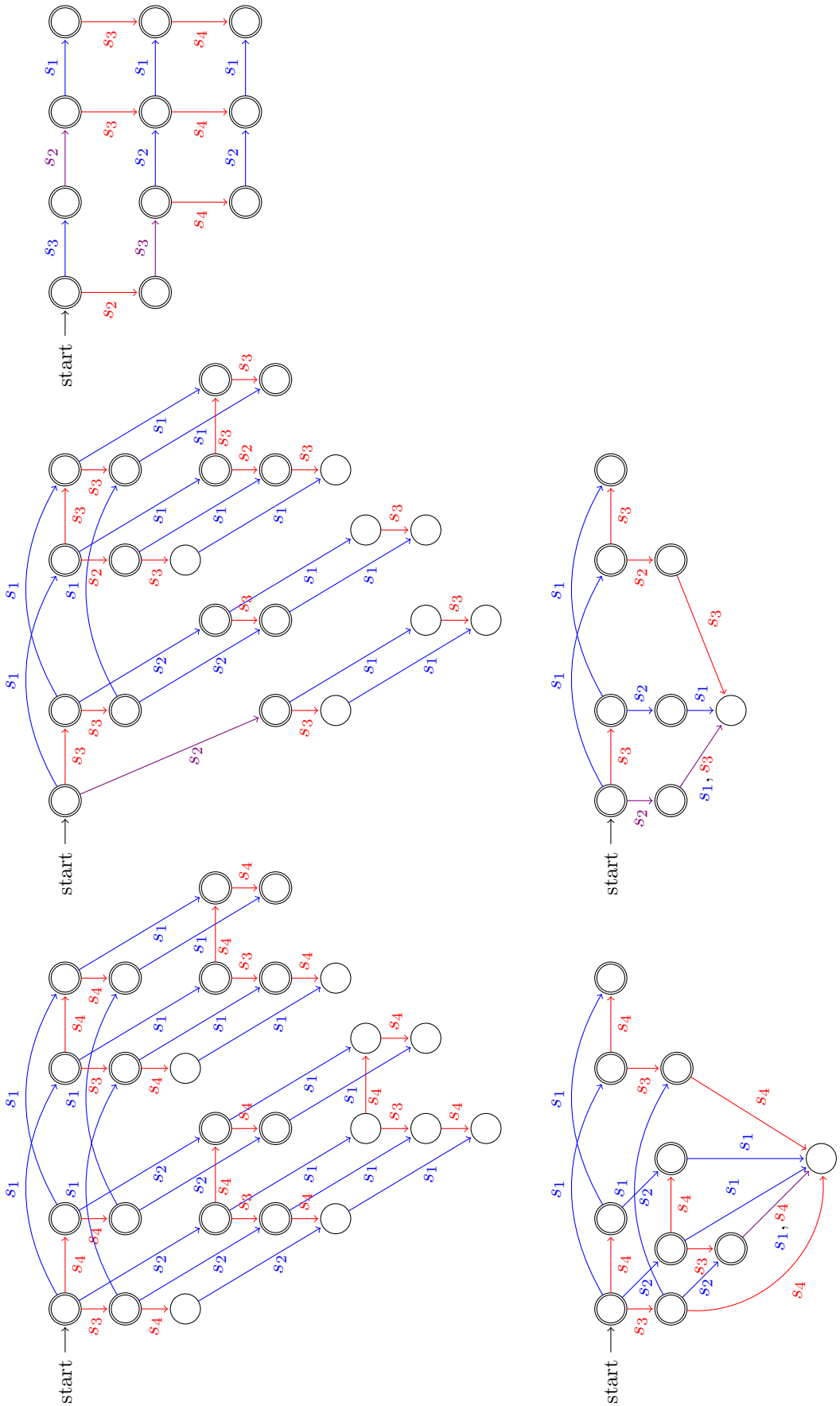

FIGURE 4. The automaton $\mathbb{P}(\{4\},\{2\})$ for $n=5$ and its skeleton (left), the automaton $\mathbb{P}(\{3\},\{2\})$ for $n=4$ and its skeleton (middle), and the healthy states of the automaton $\mathbb{P}(\{2\},\{4\})$ for $n=5$ (right).

Note that in Algorithm 2, we could ignore n in the list U (resp. 1 in the list D) since $\mathbb{U}(n)$ (resp. $\mathbb{D}(1)$) accepts all reduced words. We have decided not to do it to be coherent with our recursive definition of $\mathbb{U}(j)$ and $\mathbb{D}(j)$.

Example 26. *We present in Table 2 the $(\{3\},\{2\})$-permutree sorting algorithm in action for the permutations $\pi_1 := 3214$, $\pi_2 := 1324$ and $\pi_3 := 1342$, and in Table 3 the $(\{2\},\{4\})$-permutree sorting algorithm in action for the permutations $\pi_4 := 54213$ and $\pi_5 := 15342$. The corresponding automata $\mathbb{P}(\{3\},\{2\})$ and $\mathbb{P}(\{2\},\{4\})$ are represented in Figure 4. Each row in these tables contains the states of the permutation π and of the word w, the current values of j and ℓ in use at each step, and the values of k for which we have to check that $\pi([k]) = [k]$, crossed in red when it fails. These tables show that π_1 and π_2 are $(\{3\},\{2\})$-permutree sortable while π_3 is not, and that π_4 is $(\{2\},\{4\})$-permutree sortable while π_5 is not.*

π_1	w_1	U_1	D_1	ℓ_1	k_1
3214	ε	$\{3\}$	$\{2\}$	1	.
3124	s_1	$\{3\}$	$\{1\}$	2	3
2134	$s_1 \cdot s_2$	$\varnothing$	$\{1\}$	1	0
1234	$s_1 \cdot s_2 \cdot s_1$				

π_2	w_2	U_2	D_2	ℓ_2	k_2
1324	ε	$\{3\}$	$\{2\}$	2	1, 3
1234	s_2				

π_3	w_3	U_3	D_3	ℓ_3	k_3
1342	ε	$\{3\}$	$\{2\}$	2	1, ~~3~~

TABLE 2. The $(\{3\},\{2\})$-permutree sorting of $\pi_1 := 3214$, $\pi_2 := 1324$ and $\pi_3 := 1342$.

π_4	w_4	U_4	D_4	ℓ_4	k_4
54213	ε	$\{2\}$	$\{4\}$	3	.
53214	s_3	$\{2\}$	$\{3\}$	2	.
52314	$s_3 \cdot s_2$	$\{3\}$	$\{2\}$	1	.
51324	$s_3 \cdot s_2 \cdot s_1$	$\{3\}$	$\{1\}$	4	.
41325	$s_3 \cdot s_2 \cdot s_1 \cdot s_4$	$\{3\}$	$\{1\}$	3	.
31425	$s_3 \cdot s_2 \cdot s_1 \cdot s_4 \cdot s_3$	$\{4\}$	$\{1\}$	2	.
21435	$s_3 \cdot s_2 \cdot s_1 \cdot s_4 \cdot s_3 \cdot s_2$	$\{4\}$	$\{1\}$	1	.
12435	$s_3 \cdot s_2 \cdot s_1 \cdot s_4 \cdot s_3 \cdot s_2 \cdot s_1$	$\{4\}$	$\{1\}$	3	4
12345	$s_3 \cdot s_2 \cdot s_1 \cdot s_4 \cdot s_3 \cdot s_2 \cdot s_1 \cdot s_3$	$\{4\}$	$\{1\}$		

π_5	w_5	U_5	D_5	ℓ_5	k_5
15342	ε	$\{2\}$	$\{4\}$	2	.
15243	s_2	$\{3\}$	$\{4\}$	3	.
15234	$s_2 \cdot s_3$	$\{4\}$	$\{3\}$	4	.
14235	$s_2 \cdot s_3 \cdot s_5$	$\{5\}$	$\{3\}$	3	~~2~~

TABLE 3. The $(\{2\},\{4\})$-permutree sorting of $\pi_4 := 54213$ and $\pi_5 := 15342$.

Corollary 27. *For any permutation π and any disjoint subsets U and D of $\{2,\dots,n-1\}$, Algorithm 2 returns a reduced word w accepted by $\mathbb{P}(U,D)$ with the property that w is a reduced expression for π if and only if π avoids jki for $j \in U$ and kij for $j \in D$.*

Proof. This algorithm constructs a candidate reduced word for π following the automaton $\mathbb{P}(U,D)$ and prioritizing healthy states over ill states. It begins by checking all possible transitions s_ℓ that keep $\mathbb{P}(U,D)$ in healthy states following Lemma 8 (if condition in line 1). Doing this in the intersection of automata translates to updating ℓ to $\ell+1$ when $\ell \in U$ and $\ell+1$ to ℓ when $\ell+1 \in D$ (line 2). When we have exhausted all these transitions, we need to go to an ill state of $\mathbb{P}(U,D)$, *i.e.* to apply a transposition that sends at least one automaton of the intersection to an ill state. If there is $\ell+1 \in U$ (resp. $\ell \in D$) such that s_ℓ is a descent of π and $\pi([\ell+1]) = [\ell+1]$ (resp. $\pi([\ell-1]) = [\ell-1]$), then any reduced expression for π is accepted by the automaton $\mathbb{U}(\ell)$ (resp. $\mathbb{D}(\ell)$) by Proposition 13 (i). We can thus start with s_ℓ and forget about the automaton $\mathbb{U}(\ell)$ (resp. $\mathbb{D}(\ell)$) (lines 3, 4 and 5). Finally, if none of these options are possible, any reduced expression for π will lead to a dead state in at least one of the automata, so that π is not (U,D)-sortable. We thus return the empty reduced expression (line 6). □

4.5. **Generating trees.** As in Section 3.3, we can define natural generating trees for the (U, D)-permutree minimal permutations. Namely, fix an arbitrary priority order $\prec$ on $\{s_1, \dots, s_{n-1}\}$. For an (U, D)-permutree minimal permutation π, we denote by $\pi(U, D, \prec)$ the $\prec$-lexicographic minimal reduced expression for π that is accepted by $\mathbb{P}(U, D)$. We denote by $\mathcal{R}(n, U, D, \prec)$ the set of reduced words of the form $\pi(U, D, \prec)$ for all (U, D)-permutree minimal permutations $\pi \in \mathfrak{S}_n$. The same proof as that of Proposition 18 shows that $\mathcal{R}(n, U, D, \prec)$ is closed by prefix. This yields a natural generating tree on $\mathcal{R}(n, U, D, \prec)$ where the parent of a reduced word w is obtained by deleting its last letter. Replacing each reduced expression by the corresponding permutation, this provides a generating tree for the (U, D)-permutree minimal permutations of $\mathfrak{S}_n$. Figure 5 presents these generating trees for different values of U and D.

5. Permutree sorting versus Coxeter sorting

In this section, we discuss the particular case when U and D form a partition of $\{2, \dots, n-1\}$. In that situation, we connect the (U, D)-permutree sorting with the c-sorting of N. Reading [Rea07b].

5.1. **Coxeter sorting word and Coxeter sortable permutations.** We first recall the theory of c-sorting developed by N. Reading in [Rea07b]. While it was defined in arbitrary finite Coxeter groups, we focus on the symmetric group in this presentation.

We consider a *Coxeter element* c of $\mathfrak{S}_n$, *i.e.* the product of all simple transpositions $\{s_1, \dots, s_{n-1}\}$ in an arbitrary order. For a permutation $\pi \in \mathfrak{S}_n$, the *c-sorting word* $\pi(c)$ is the lexicographically smallest reduced expression for π in the infinite word $c^\infty = c \cdot c \cdot c \cdot c \cdots$. Note that strictly speaking, $\pi(c)$ depends on a reduced expression for c, not only on the Coxeter element c. Here, we assume that we have chosen a reduced expression and hide this dependence. We let $I_1, \dots, I_p$ denote the subsets of $[n-1]$ such that $\pi(c) = c_{I_1} \cdot c_{I_2} \cdots c_{I_p}$ where c_I is the subword of c obtained by keeping only the letters s_i for $i \in I$. The permutation π is *c-sortable* if $I_1 \supseteq I_2 \supseteq \cdots \supseteq I_p$. Note that this does not depend on the choice of the reduced expression c, only on the Coxeter element c.

For our proofs we will need some simple yet useful facts from [Rea07b] on how prefixes of words influence sortability.

Lemma 28. *Consider a Coxeter element of the form $c = s_\ell \cdot d$ and let $\pi \in \mathfrak{S}_n$. Then*

- *if $\pi = s_\ell \cdot \tau$ with $\ell(\pi) = \ell(\tau) + 1$, then $\pi(c) = s_\ell \cdot \tau(d \cdot s_\ell)$,*
- *otherwise, $\pi(c) = \pi(d \cdot s_\ell)$.*

Lemma 29. *Let $s_i \neq s_j$ be two letters that appear in the c-sorting word $\pi(c)$ of a c-sortable permutation π. Then*

(1) if s_i appears before s_j in c, then s_i appears before s_j in $\pi(c)$,
(2) if s_j does not appear in between two occurrences of s_i in $\pi(c)$, then it does not appear after these occurrences either.

Proof. We deal with the two statements separately:

(1) Immediate from the definition since both s_i and s_j appear in $\pi(c)$.
(2) Since π is c-sortable, $\pi(c)$ is formed by a succession of subwords that are nested. Thus if s_j were to appear after the occurrences of s_i it would have to appear between them as well. □

5.2. **Coxeter sorting via permutree automata.** A Coxeter element c of $\mathfrak{S}_n$ defines a partition $\{2, \dots, n-1\} = U_c \sqcup D_c$, where U_c (resp. D_c) consists of the elements $j \in \{2, \dots, n-1\}$ such that s_j appears before (resp. after) s_{j-1} in c. For instance, when $c = s_2 \cdot s_5 \cdot s_4 \cdot s_3 \cdot s_1 \cdot s_6$, we obtain $U_c = \{2, 4, 5\}$ and $D_c = \{3, 6\}$. Said differently, $j \in U$ (resp. $j \in D$) if c is accepted by $\mathbb{U}(j)$ but not by $\mathbb{D}(j)$ (resp. by $\mathbb{D}(j)$ but not by $\mathbb{U}(j)$). The goal of this section is the following connection between the c-sorting of Section 5.1 and the (U_c, D_c)-permutree sorting of Section 4.

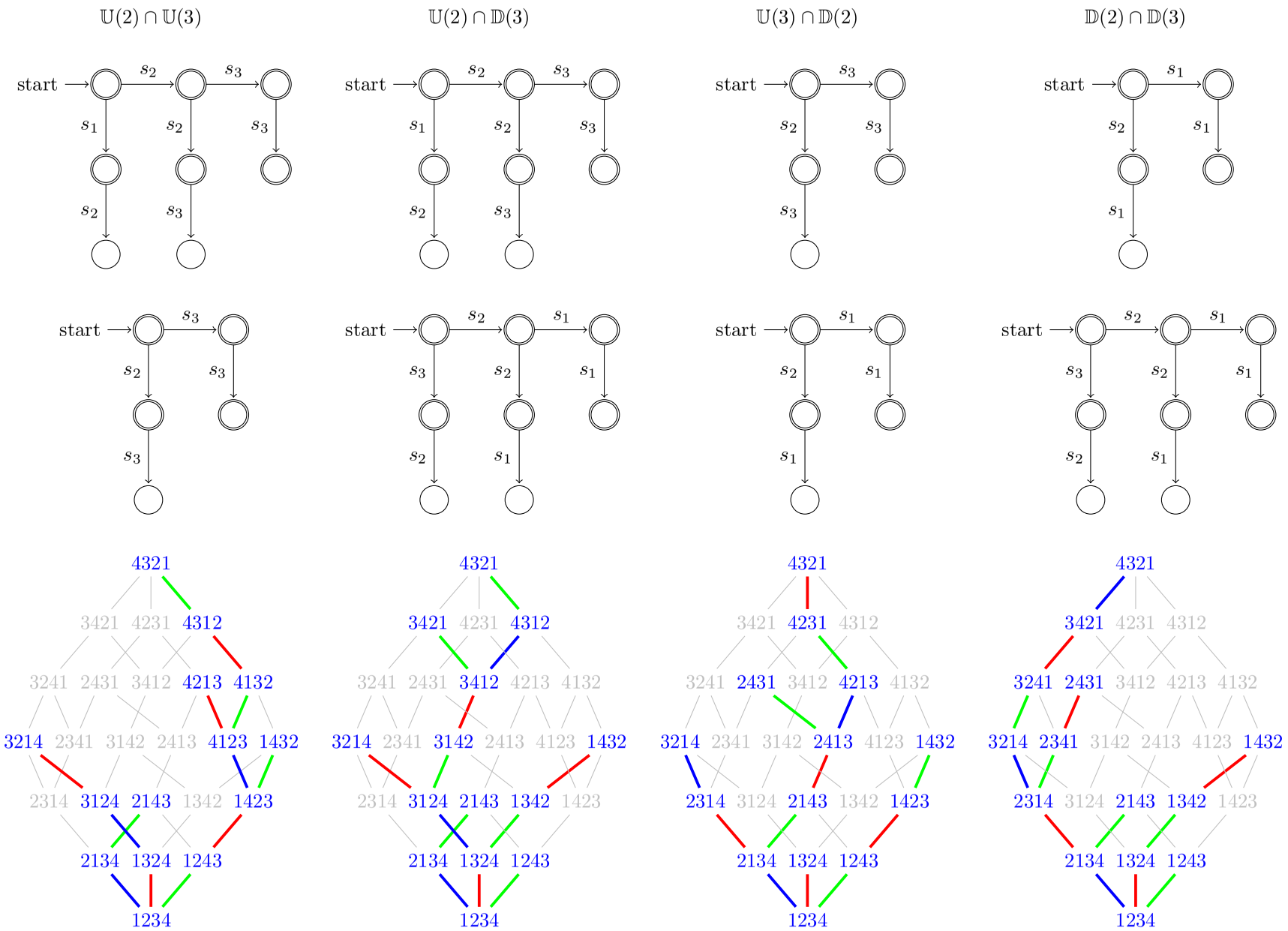

FIGURE 5. Generating trees for the (U, D)-permutree minimal permutations of $\mathfrak{S}_4$, with priority order $s_1 \prec s_2 \prec s_3$.

Theorem 30. *For any Coxeter element c and any permutation π, the following assertions are equivalent:*

(i) *π is c-sortable,*
(ii) *the c-sorting word $\pi(c)$ is accepted by the automaton $\mathbb{P}(U_c, D_c)$,*
(iii) *there exists a reduced expression for π accepted by the automaton $\mathbb{P}(U_c, D_c)$,*
(iv) *for each $j \in \{2, \dots, n-1\}$, there exists a reduced expression for π that is accepted by the automaton $\mathbb{U}(j)$ if $j \in U_c$ and $\mathbb{D}(j)$ if $j \in D_c$,*
(v) *π avoids jki for $j \in U_c$ and kij for $j \in D_c$.*

The equivalences (iii) $\iff$ (iv) $\iff$ (v) were already established earlier. Here, we aim at identifying the c-sorting word as a reduced expression for π accepted by $\mathbb{P}(U_c, D_c)$. We split the proof of the equivalences (i) $\iff$ (ii) $\iff$ (iii) into the following few lemmas.

Lemma 31. *The c-sorting word of a c-sortable permutation is recognized by $\mathbb{U}(j)$ for $j \in U_c$ and by $\mathbb{D}(j)$ for $j \in D_c$.*

Proof. Consider $j \in U_c$ (the proof for $j \in D_c$ is symmetric). We distinguish two possible cases:

- If $\pi(c)$ contains no s_{j-1}, then $\pi(c)$ either remains in the first healthy state or ends in the first ill state of $\mathbb{U}(j)$.
- If $\pi(c)$ contains s_{j-1}, then by Lemma 29 (1) s_{j-1} appears before s_j in $\pi(c)$ and $\pi(c)$ leads to the second healthy state of $\mathbb{U}(j)$. From here on out, notice that $\pi(c)$ cannot end at a dead state because of Lemma 29 (2).

In both cases, $\pi(c)$ is accepted by $\mathbb{U}(j)$. □

Lemma 32. *A permutation $\pi \in \mathfrak{S}_n$ whose c-sorting word $\pi(c)$ is accepted by $\mathbb{P}(U_c, D_c)$ is c-sortable.*

Proof. Suppose that π is not c-sortable. We will find an automaton that rejects $\pi(c)$ among the automata $\mathbb{U}(j)$ for $j \in U_c$ and $\mathbb{D}(j)$ for $j \in D_c$. Once again, we work by induction on the length of π and the size of c. Let s_ℓ be the first letter of c and write $c = s_\ell \cdot d$. Since s_ℓ appears before $s_{\ell-1}$ and $s_{\ell+1}$ in c, we have $\ell \in U_c$ and $\ell + 1 \in D_c$. Moreover, the letter s_ℓ yields to the next healthy state in both automata $\mathbb{U}(\ell)$ and $\mathbb{D}(\ell+1)$, and remains in the initial state for all other automata $\mathbb{U}(j)$ for $j \in U_c \smallsetminus \{\ell\}$ and $\mathbb{D}(j)$ for $j \in D_c \smallsetminus \{\ell+1\}$. We now distinguish two cases, depending on whether ℓ and $\ell+1$ are reversed in π.

Assume first that ℓ and $\ell+1$ are reversed in π and write $\pi = s_\ell \cdot \tau$. We then have $\pi(c) = s_\ell \cdot \tau(d \cdot s_\ell)$ by Lemma 28, so that τ is not $d \cdot s_\ell$-sortable. By induction hypothesis, $\tau(d \cdot s_\ell)$ is rejected by one of the automata $\mathbb{U}(j)$ for $j \in U_{d \cdot s_\ell}$ and $\mathbb{D}(j)$ for $j \in D_{d \cdot s_\ell}$. Since $U_{d \cdot s_\ell} = U_c \triangle \{\ell, \ell+1\}$ and $D_{d \cdot s_\ell} = D_c \triangle \{\ell, \ell+1\}$, Lemmas 4 and 8 ensure that $\pi(c) = s_\ell \cdot \tau(d \cdot s_\ell)$ is rejected by one of the automata $\mathbb{U}(j)$ for $j \in U_c$ and $\mathbb{D}(j)$ for $j \in D_c$.

Assume now that ℓ and $\ell+1$ are not reversed in π. Then $\pi(c)$ does not use s_ℓ and π is not d-sortable in $W_{\langle s_\ell \rangle}$. By induction hypothesis, $\pi(c)$ is rejected by one of the automata $\mathbb{U}(j)$ for $j \in U_d$ and $\mathbb{D}(j)$ for $j \in D_d$. This concludes the proof since $U_d \subseteq U_c$ and $D_d \subseteq D_c$. □

Lemma 33. *If a permutation $\pi \in \mathfrak{S}_n$ admits a reduced expression accepted by $\mathbb{P}(U_c, D_c)$, then its c-sorting word $\pi(c)$ is accepted by $\mathbb{P}(U_c, D_c)$.*

Proof. Once agin, we work by induction on the length of π. Let s_ℓ be the first letter of c and write $c = s_\ell \cdot d$. Since s_ℓ appears before $s_{\ell-1}$ and $s_{\ell+1}$ in c, we have $\ell \in U_c$ and $\ell + 1 \in D_c$. Moreover, the letter s_ℓ yields to the next healthy state in both automata $\mathbb{U}(\ell)$ and $\mathbb{D}(\ell+1)$, and remains in the initial state for all other automata $\mathbb{U}(j)$ for $j \in U_c \smallsetminus \{\ell\}$ and $\mathbb{D}(j)$ for $j \in D_c \smallsetminus \{\ell+1\}$. We now distinguish two cases, depending on whether ℓ and $\ell+1$ are reversed in π.

Assume first that ℓ and $\ell+1$ are reversed in π and write $\pi = s_\ell \cdot \tau$. Using Lemma 28 it suffices now to show that after s_ℓ, there is a reduced expression for τ accepted by the automata. For each j, since the automaton $\mathbb{D}(j)$ or $\mathbb{U}(j)$ that we see accepts at least one reduced expression for π and s_ℓ does not lead to a ill state, it also accepts a reduced expression for π starting with s_ℓ by Proposition 11. Observe moreover that:

- τ admits a reduced expression accepted by $\mathbb{U}(j)$ for each $j \notin \{\ell, \ell+1\}$ by Lemma 4. This lines up with the fact that the order of s_{j-1} and s_j has not changed from $c = s_\ell \cdot d$ to $d \cdot s_\ell$.
- τ admits a reduced expression accepted by $\mathbb{U}(\ell+1)$ and a reduced expression accepted by $\mathbb{D}(\ell)$ by Lemma 8. This fits the fact that ℓ now appears after $\ell-1$ and $\ell+1$ in $d \cdot s_\ell$.

By induction, we obtain that $\tau(d \cdot s_\ell)$ is accepted by $\mathbb{P}(U_{d\cdot s_\ell}, D_{d\cdot s_\ell})$, so that $\pi(c) = s_\ell \cdot \tau(d \cdot s_\ell)$ is accepted $\mathbb{P}(U_c, D_c)$.

Assume now that ℓ and $\ell+1$ are not reversed in π. We want to show that ℓ never appears in the reduced expressions for π, ie. that $\pi([\ell]) = [\ell]$ and $\pi([n] \smallsetminus [\ell]) = [n] \smallsetminus [\ell]$. Otherwise, the reduced expression w_ℓ accepted by $\mathbb{U}(\ell)$ would see first a $s_{\ell+1}$, and then a s_ℓ before it sees any $s_{\ell-1}$, so that we would have an inversion $k\ell$ in π for some $\ell < k$. Similarly, the reduced expression $w_{\ell+1}$ accepted by $\mathbb{D}(\ell+1)$ should see first a $s_{\ell-1}$ and then a s_ℓ before it sees any $s_{\ell+1}$, so that we would have an inversion $(\ell+1)i$ in π for some $i < \ell+1$. Since ℓ and $\ell+1$ are not reversed, we see $k\ell(\ell+1)i$ which contradicts twice Theorem 1. We conclude that this case never happens, so that we can work in the parabolic subgroup of permutations that never use s_ℓ in their reduced expressions. □

5.3. **Some negative observations.** We conclude this paper with some negative observations and warnings about the connection between c-sorting and (U_c, D_c)-permutree sorting. First, we want to underline that using c-sorting words to test whether a permutation avoids jki or kij for a fixed j is dangerous for the following two reasons.

Remark 34. *Even if a permutation π avoids jki (resp. kij) for a given j, there might be no Coxeter element c for which π is c-sortable and $j \in U_c$ (resp. $j \in D_c$). For instance, the permutation $41325 \in \mathfrak{S}_5$ avoids $2ki$ and $ki4$, but contains 352 and 413, so it is not c-sortable for any Coxeter element c.*

Remark 35. *When a permutation π is not c-sortable, there might exist $j \in U_c$ (resp. $j \in D_c$) for which the c-sorting word $\pi(c)$ is not accepted by $\mathbb{U}(j)$ (resp. $\mathbb{D}(j)$) even if π avoids jki (resp. kij). For instance, consider $c = s_2 \cdot s_1 \cdot s_3$ and $\pi = 4213 = s_3 \cdot s_1 \cdot s_2 \cdot s_1 = s_3 \cdot s_2 \cdot s_1 \cdot s_2 = s_1 \cdot s_3 \cdot s_2 \cdot s_1$. Then $2 \in U_c$, and the c-sorting word $\pi(c) = s_1 \cdot s_3 \cdot s_2 \cdot s_1$ is rejected by $\mathbb{U}(2)$ while π contains no $2ki$ (and indeed $s_3 \cdot s_2 \cdot s_1 \cdot s_2$ is accepted by $\mathbb{U}(2)$).*

We conclude the paper with an observation about sorting networks and permutree sorting.

Remark 36. *Given a Coxeter element c, the word c^∞ which is used to compute $\pi(c)$ is a sorting network. This means that we decide* beforehand *a list of transpositions to apply if appropriate. On the other hand, the permutree sorting given in Algorithm 2 is* not *a sorting network. Indeed, the order on transpositions depends on the permutation and more specifically on the state of the automaton we are at. A natural question then occurs: can we replace the permutree sorting algorithm by a sorting network? Or said differently, when U and D are disjoint but do not cover $\{2, \dots, n-1\}$, can we find a word $\tilde{c}$ which plays the role of c^∞ in the sense that looking at $\pi(\tilde{c})$ would be enough to check whether π is accepted by $\mathbb{P}(U, D)$?*

The answer is negative in general. A counter-example is found for $n = 5$, $U = \{2\}$, and $D = \{4\}$. In this case one can check through computer exploration that no reduced word $\tilde{c}$ of the maximal permutation 54321 can be used as a sorting network. Namely, for all choices of $\tilde{c}$, there exist a permutation π which is accepted by $\mathbb{P}(U, D)$ whereas the reduced expression $\pi(\tilde{c})$ is rejected. The healthy states of $\mathbb{P}(\{2\}, \{4\})$ are shown in Figure 4. We see that accepted reduced expressions can start with either s_2 or s_3. For some permutations, such as 54213 shown in Example 26, all accepted reduced expressions start with s_3 whereas for some other permutations such as 35421, all accepted reduced expressions start with s_2. This eventually leads to an empty intersection for the choice of $\tilde{c}$.

Nevertheless, it seems interesting to study in which case the answer is positive. The Cambrien case with the c-sorting word when U and D form a partition of $\{2, \dots, n-1\}$ is one example. The case where $|U| + |D| = 1$ is another one. This is the case corresponding to Theorem 1 where we have only one automaton. In this case, we can construct a word $\tilde{c}$ by reading the healthy states of the automaton linearly, adding at each state the word $(s_{i_1} \cdots s_{i_k})^k s_j$ where $s_{i_1}, \dots, s_{i_k}$ are the

looping transitions and s_j is the transition going to the next healthy state. This process gives a prefix that can be extended in any way to obtain a proper sorting word $\tilde{c}$. For example, if $U = \{2\}$, we obtain the prefix $s_3 \cdot s_2 \cdot s_1 \cdot s_3$ and indeed $s_3 \cdot s_2 \cdot s_1 \cdot s_3 \cdot s_2 \cdot s_1$ acts as a sorting network equivalent to the $(\{2\}, \varnothing)$-permutree sorting. This process actually seems to extend to all cases where, at each healthy state of the intersection automaton, the choices for the healthy transitions commute. For example, in the case where $n = 5$, $U = \{4\}$ and $D = \{2\}$ as illustrated in Figure 4, the word $s_1 \cdot s_2 \cdot s_4 \cdot s_3 \cdot s_2 \cdot s_1 \cdot s_4 \cdot s_3 \cdot s_2$ gives a proper sorting network.

References

[CP17] Grégory Chatel and Vincent Pilaud. Cambrian Hopf Algebras. *Adv. Math.*, 311:598–633, 2017.

[CPP19] Grégory Chatel, Vincent Pilaud, and Viviane Pons. The weak order on integer posets. *Algebraic Combinatorics*, 2(1):1–48, 2019.

[HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation.* Addison-Wesley Publishing Co., Reading, Mass., 1979. Addison-Wesley Series in Computer Science.

[Knu69] Donald E. Knuth. *The art of computer programming. Vol. 1: Fundamental algorithms.* Second printing. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont, 1969.

[PP18] Vincent Pilaud and Viviane Pons. Permutrees. *Algebraic Combinatorics*, 1(2):173–224, 2018.

[Rea04] Nathan Reading. Lattice congruences of the weak order. *Order*, 21(4):315–344, 2004.

[Rea06] Nathan Reading. Cambrian lattices. *Adv. Math.*, 205(2):313–353, 2006.

[Rea07a] Nathan Reading. Clusters, Coxeter-sortable elements and noncrossing partitions. *Trans. Amer. Math. Soc.*, 359(12):5931–5958, 2007.

[Rea07b] Nathan Reading. Sortable elements and Cambrian lattices. *Algebra Universalis*, 56(3-4):411–437, 2007.

(VPi) CNRS & LIX, École Polytechnique, Palaiseau
E-mail address: vincent.pilaud@lix.polytechnique.fr
URL: http://www.lix.polytechnique.fr/~pilaud/

(VPo) Université Paris-Saclay, CNRS, Laboratoire de recherche en informatique, Orsay, France.
E-mail address: viviane.pons@lri.fr
URL: https://www.lri.fr/~pons/

(DTJ) Université Paris-Saclay, CNRS, Laboratoire de recherche en informatique, Orsay, France.
E-mail address: daniel.tamayo-jimenez@lri.fr