



**HAL**  
open science

## 2-stack pushall sortable permutations

Adeline Pierrot, Dominique Rossin

► **To cite this version:**

| Adeline Pierrot, Dominique Rossin. 2-stack pushall sortable permutations. 2013. hal-00801861

**HAL Id: hal-00801861**

**<https://hal.science/hal-00801861v1>**

Preprint submitted on 18 Mar 2013

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# 2-stack pushall sortable permutations \*

Adeline Pierrot

Dominique Rossin

March 18, 2013

## Abstract

In the 60's, Knuth introduced stack-sorting and serial compositions of stacks. In particular, one significant question arise out of the work of Knuth: how to decide efficiently if a given permutation is sortable with 2 stacks in series? Whether this problem is polynomial or NP-complete is still unanswered yet. In this article we introduce 2-stack pushall permutations which form a subclass of 2-stack sortable permutations and show that these two classes are closely related. Moreover, we give an optimal  $\mathcal{O}(n^2)$  algorithm to decide if a given permutation of size  $n$  is 2-stack pushall sortable and describe all its sortings. This result is a step to the solve the general 2-stack sorting problem in polynomial time.

## 1 Introduction

In the 60's, Knuth introduced the problem of stack-sorting [8] and then serial compositions of stacks [9]. To answer the one-stack case, he introduced both the pattern-containment relation on permutations and permutation classes, two new fields of combinatorics. Stack-sorting was further generalized to sorting networks by Tarjan [12] while several variants appear by either considering other types of combinatorial structures or by changing rules [11, 7, 1].

In this article, we focus on sorting with two stacks in series. More precisely, if  $\sigma$  is a permutation, we consider  $\sigma$  as a sequence of integers  $\sigma_1, \sigma_2, \dots, \sigma_n$  that we take as input and at each step we have three possibilities as described in Figure 2 (p.4):

$\rho$ : Get the next element of  $\sigma$  and push its value on top of the first stack denoted  $H$ .

$\lambda$ : Pop the topmost element of stack  $H$  and push this value on top of the second stack  $V$ .

$\mu$ : Pop the topmost element of  $V$  and write it to the output.

We iterate over these three possibilities until all elements have been output. If there is a sequence of operations that leads to identity on the output, then we say that the permutation is 2-stack sortable. Three natural questions among others arise:

1. Decision: what is the complexity of the problem consisting of deciding whether a given permutation is sortable or not?
2. Characterization: can one characterize permutations that are sortable?
3. Counting: establish the generating function of sortable permutations.

---

\*This work was completed with the support of the ANR project ANR BLAN-0204\_07 MAGNUM

For the one-stack case these three problems were solved by Knuth in [8]. A greedy algorithm allows to answer the decision problem in linear time. Moreover he characterized sortable permutations by introducing the 231-avoiding permutations class, whose generating function is the Catalan series. Since this article, the more general question of sorting with multiple stacks in series or in parallel has been widely studied. Knuth [8], Tarjan [12] and Pratt [11] noted that the permutations sortable by the various configurations could be described by forbidding certain patterns to occur in the permutations.

Regarding  $t$  parallel stacks, the decision problem can be answered in time  $\mathcal{O}(n \log n)$  for  $t = 1, 2, 3$ , while for  $t > 3$  this is NP-complete (this is proved by a reduction in [7] to a problem solved in [13]). The characterization problem is studied in [11]: for  $t > 1$ , the basis of the class of permutations sortable with  $t$  stacks in parallel is infinite. Finally, about the counting problem, when  $t = 2$  the generating function is described in [3], but by an infinite system of equations.

For stacks in series, it has been shown in [8] that every permutation of size  $n$  can be sorted by  $\log_2(n)$  stacks in series. But none of the above three questions has been answered for more than one stack in series. For two stacks, Murphy [10] proved that the basis of the class of sortable permutations is infinite. In his Phd thesis, he also studies the problem of deciding whether a given permutation is sortable with 2 stacks in series. He reduced this problem to a 3-SAT problem; at the same time he reduced a 2-SAT instance to the decision problem, and hoped that one of both reduction was actually an equivalence. But none of those results has been proved or disproved. In [5], Bóna gives an overview of advances in sorting networks and mentions this problem as possibly NP-complete. More surprising, both conjectures exist: in [4], the authors conjectured that the decidability problem is NP-complete, while Murphy in his PhD ([10] Conjecture 260) conjectured that it is in  $P$ . Several weaker variants of this problem have been studied. First, West considered permutations sortable with two consecutive greedy passes through a stack in [14, 15]. He conjectured the enumeration formula which was proved after by Zeilberger [16]. For more than two passes, few results are known [6, 17]. Another variant studied in [4] is to consider decreasing stacks (i.e. elements in the stack must be decreasing from bottom to top) instead of general stacks. In this article we define a new restriction of 2-stacks sorting, namely 2-stacks pushall sorting, and prove that the decidability problem in this case is polynomial.

Throughout this article we usually write permutations as  $\sigma = \sigma_1 \sigma_2 \dots \sigma_n$  where  $n$  is the size of  $\sigma$ , denoted by  $|\sigma|$ , and  $\sigma_i$  is the image of  $i$  for all  $i \leq n$ . A permutation  $\pi = \pi_1 \pi_2 \dots \pi_k$  is a *pattern* of  $\sigma$  if and only if there exist  $1 \leq i_1 \leq i_2 \leq i_3 \leq \dots \leq i_k \leq n$  such that  $\sigma_{i_1} \sigma_{i_2} \dots \sigma_{i_k}$  is order isomorphic to  $\pi$ . We note  $Av(B)$  the set of permutations avoiding  $B$ , i.e. not having any permutation of  $B$  as a pattern. A *permutation class*  $\mathcal{C}$  is a set of permutations downward-closed for the pattern relation: if  $\sigma$  belongs to  $\mathcal{C}$ , then every pattern of  $\sigma$  belongs to  $\mathcal{C}$ . Note that for any set  $B$ ,  $Av(B)$  is a class. A permutation class  $\mathcal{C}$  can be defined by its minimal set  $B$  such that  $\mathcal{C} = Av(B)$ . This minimal set is called the *basis* of the class. For example, Knuth proved that 1-stack sortable permutations are those that belong to the class  $Av(231)$ . Unfortunately, the basis can be infinite. For 2-stack sortable permutations, as stated above, it has been proved in [10] that the basis is infinite.

A permutation can also be represented by its *diagram*, consisting in the set of points at coordinates  $(i, \sigma_i)$  drawn in the plane (see two examples in Figure 1). An *interval* in a permutation is a consecutive range of elements, consecutive both in indices and values. For example in the permutation 479681325, the elements 7968 form an interval: they are

consecutive in the permutation and the values span the whole integer interval  $[6 \dots 9]$ . In the diagram, note that an interval is a square which is itself a diagram of a permutation (if translated to the origin). In particular, no point outside this square has the same x or y coordinate than any cell of the square (see the yellow stripes of Figure 1). A permutation where all intervals are trivial –either a singleton or the whole permutation– is called a *simple* permutation. For instance, 2413 and 3142 are the two simple permutations of size 4. An *inflation* of an element  $\sigma_i$  in  $\sigma$  by a permutation  $\pi$  is the permutation obtained by replacing  $\sigma_i$  by  $\pi$  and renormalizing the resulting permutation. For example if we inflate 3 in 2314 by the permutation 41523, we obtain the permutation **263745**18 (see the second diagram of Figure 1). Notice that in an inflation by  $\pi$ , elements corresponding to  $\pi$  form an interval in the resulting permutation.

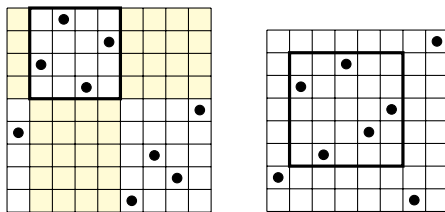


Figure 1: Diagram of 479681325 and the inflation of 3 in 2314 by 41523

We denote inflations by  $\sigma = \tau[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  where  $\tau$  is a permutation of size  $k$  and  $\tau_i$  is inflated by  $\pi^{(i)}$  for all  $i$ . When  $\tau$  is the identity  $12 \dots k$  (resp. the decreasing permutation  $k \dots 1$ ) we write  $\sigma = \oplus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  (resp.  $\ominus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$ ).

A permutation  $\sigma$  is  $\oplus$ -decomposable (resp.  $\ominus$ -decomposable) if it can be written  $\sigma = \oplus[\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \dots, \pi^{(k)}]$  (resp.  $\ominus[\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \dots, \pi^{(k)}]$ ) with  $k > 1$ . Otherwise  $\sigma$  is  $\oplus$ -indecomposable (resp.  $\ominus$ -indecomposable).

A decomposition theorem [2] states that any permutation  $\sigma \neq 1$  can be written in a unique way as either:

- $\sigma = \oplus[\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \dots, \pi^{(k)}]$  where  $k \geq 2$  and the  $\pi^{(i)}$  are  $\oplus$ -indecomposable.
- $\sigma = \ominus[\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \dots, \pi^{(k)}]$  where  $k \geq 2$  and the  $\pi^{(i)}$  are  $\ominus$ -indecomposable.
- $\sigma = \tau[\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \dots, \pi^{(k)}]$  where  $k \geq 4$  and  $\tau$  is simple.

In the next section we study 2-stack sorting and 2-stack pushall sorting and show the close correlation between these two models. This combinatorial study concludes on some partial characterization of both classes in terms of permutations they contain or permutations in the basis. The key idea is to use the block-decomposition of permutations given in the above theorem.

Then in section 3 we prove that 2-stack pushall sorting can be expressed as a 2-color problem on the diagram of permutations. Moreover we characterize diagram of permutations that can be colored. This characterization leads to a polynomial algorithm to check whether a permutation is 2-stack pushall sortable by finding all colorings for its diagram.

Section 4 refines the results of section 3 by limiting the number of colorings to test. This leads to an optimal algorithm computing in quadratic time a linear representation of all

pushall sortings of a given permutation, which thus decides whether a permutation is 2-stack pushall sortable.

To conclude, we give in section 5 some natural continuations of our work.

## 2 2-stack sorting vs 2-stack pushall sorting

In this section we define pushall sorting and point out the close link between 2-stack sorting and 2-stack pushall sorting. Moreover, for each of these sorting problems we exhibit some recursive necessary and sufficient conditions for a permutation to be sortable depending on the root of its block-decomposition.

In 2-stack sorting, three different operations are allowed as pictured in Figure 2. Each of this operation can be encoded with a letter (see Figure 2). For example, whenever an element is popped from stack  $H$  and pushed in stack  $V$ , we write  $\lambda$ . A sequence of operations is encoded by a word whose length is the number of operations performed.

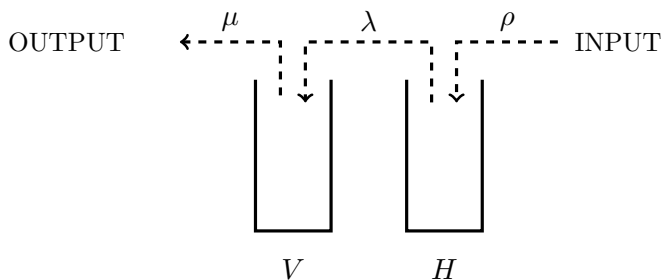


Figure 2: Sorting with two stacks in series

**Definition 2.1.** A stack word  $w$  is a word over the alphabet  $\{\rho, \lambda, \mu\}$  such that  $|w|_\rho = |w|_\lambda = |w|_\mu$  and for all prefix  $v$  of  $w$ ,  $|w|_\rho \geq |w|_\lambda \geq |w|_\mu$ . Intuitively it's a word which describes a sequence of appropriate stack operations which take a permutation through two stacks in series (without necessarily sorting it). A permutation  $\sigma = \sigma_1 \dots \sigma_n$  is 2-stack sortable if and only if there exists a stack word of length  $3n$  ( $n$  times each letter  $\rho, \lambda, \mu$ ) which leads to the identity in the output with  $\sigma$  as input. Such a word is called a valid stack word for  $\sigma$ .

There are several valid stack word for a given permutation: for example, permutation 2341 admits either  $\rho\rho\rho\rho\lambda\mu\lambda\lambda\lambda\mu\mu\mu$  or  $\rho\rho\rho\lambda\lambda\lambda\rho\lambda\mu\mu\mu$  as valid words. Note also that  $\rho$  and  $\mu$  commutes: if  $w$  is a valid stack word for  $\sigma$  an  $w'$  is obtained for  $w$  by exchanging adjacent letters  $\rho$  and  $\mu$ , then  $w'$  is a valid stack word for  $\sigma$ . In his Phd [10], Murphy studied 2-stack sorting by studying stack words. This presentation of 2-stack sorting allow us to define formally 2-stack pushall sorting.

**Definition 2.2.** A pushall stack word is a stack word such that the first occurrence of letter  $\mu$  is after the last occurrence of the letter  $\rho$ . A permutation  $\sigma$  of size  $n$  is 2-stack pushall sortable if and only it admits a valid pushall stack word.

More informally, 2-stack pushall sortable permutations are those which can be sorted by pushing all elements in the stacks before writing any element to the output.

For example 2431 is 2-stack pushall sortable as the word  $\rho\rho\rho\rho\lambda\mu\lambda\lambda\lambda\mu\mu\mu$  respects the required condition (as does  $\rho\rho\rho\lambda\lambda\lambda\rho\lambda\mu\mu\mu$ ).

**Remark 2.3.** A stack word  $w$  is a pushall stack word if and only if it can be written as  $w = uv$  with  $u \in \{\rho, \lambda\}^*$  and  $v \in \{\lambda, \mu\}^*$ . This decomposition is not unique. In the preceding example, the word  $w = \rho\rho\rho\lambda\mu\lambda\lambda\lambda\mu\mu\mu$  admits two decompositions:  $w = (\rho\rho\rho)(\lambda\mu\lambda\lambda\mu\mu\mu)$  and  $w = (\rho\rho\rho\lambda)(\mu\lambda\lambda\lambda\mu\mu\mu)$ .

The previous definition of 2-stack pushall sortable permutations implies that they form a subset of 2-stack sortable permutations. Moreover it is easy to check that 2-stack pushall sorting is stable by pattern relation: if  $\sigma$  is 2-stack pushall sortable then every pattern  $\pi$  of  $\sigma$  is 2-stack pushall sortable: choose an occurrence of  $\pi$  in  $\sigma$  and a valid pushall stack word  $w$  of  $\sigma$ . To obtain a valid pushall stack word of  $\pi$ , delete letters of  $w$  that correspond to elements of  $\sigma$  not involved in the occurrence of  $\pi$ . The same reasoning holds for general 2-stack sorting.

**Proposition 2.4.** 2-stack pushall permutations form a subclass of 2-stack sortable permutation class.

Although we do not know the ratio between these two classes, there exists a close correlation between them and solving 2-stack pushall sorting is a prerequisite for the more general case. We first study the possible configurations of the stacks during a sorting procedure. This will help us to obtain properties of stack sorting permutations thanks to their decomposition. In a last subsection, we study the basis of 2-stack sortable permutation class and show how it is correlated to the 2-stack pushall one.

## 2.1 Stack configurations

At each step of a sorting procedure, some elements of the permutation lie in the stacks. We call a *stack configuration* the position of these elements in stacks  $H$  and  $V$ . In this section, we exhibit a necessary condition on stack configurations to be part of a sorting procedure. First we define formally stack configurations.

**Definition 2.5.** A stack configuration is a pair of two vectors of positive integers  $(\vec{V}, \vec{H})$  of arbitrary (and maybe different) sizes, such that all coordinates are distincts. A stack configuration may be empty (if both vectors are of size zero). Vector  $\vec{V}$  (resp.  $\vec{H}$ ) represents elements that are in stack  $V$  (resp.  $H$ ) given from bottom to top, so we can apply to stack configurations moves  $\lambda$  and  $\mu$ , and move  $\rho$  if we know what is the next integer in the input.

Let  $\sigma$  be a permutation, a stack configuration of  $\sigma$  is a stack configuration in which coordinates are bounded by  $|\sigma|$ .

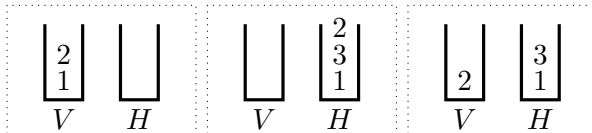
**Definition 2.6.** To each stack word  $w$  of size  $3n$  and permutation  $\sigma$  of size  $n$  we associate a sequence of  $3n + 1$  stack configurations  $(c_k(w, \sigma))$  describing how the sequence of moves  $w = w_1 \dots w_{3n}$  take  $\sigma$  through the stacks:  $c_1(w, \sigma)$  is empty and we obtain  $c_{k+1}(w, \sigma)$  from  $c_k(w, \sigma)$  by doing operation  $w_k$  with  $\sigma$  as input at the beginning.

**Definition 2.7.** Let  $\sigma$  be a permutation. A stack configuration  $c$  is reachable for  $\sigma$  if it exists a stack word  $w$  and an integer  $k$  such that  $c = c_k(w, \sigma)$ . A stack configuration  $c$  is total for  $\sigma$  if all integers from 1 to  $|\sigma|$  appear in  $c$  (this notion depends only on  $|\sigma|$ , we don't ask  $c$  to be reachable for  $\sigma$ ).

**Remark 2.8.** Let  $w$  be a stack word of size  $3n$  and  $\sigma$  a permutation of size  $n$ . Then  $w$  is a pushall stack word if and only if at least one of the stack configurations  $(c_k(w, \sigma))$  is total.

During a sorting procedure, stack configurations have constraints so that all elements can be popped out in increasing order. Recall that in one-stack sorting, the stack must be in decreasing order (from bottom to top). For two-stack sorting, we have the same decreasing constraint on stack  $V$  but other constraints appear that can be represented as stack patterns.

**Definition 2.9.** We call unsortable stack-patterns the following three patterns, denoted respectively  $|12|$ ,  $|132|$  and  $|2|13|$ :



More precisely pattern  $|12|$  means that there is in stack  $V$  one element which has a smaller element below it. Pattern  $|132|$  means that there is in stack  $H$  one element which has a greater element below it and a smaller element more below. Pattern  $|2|13|$  is somehow special as the pattern is divided in both stacks. It means that there are elements  $a, b, c$  such that  $b \in V$ ,  $a, c \in H$ ,  $a < b < c$  and  $c$  is above  $a$  in stack  $H$ .

**Theorem 2.10.** A stack configuration can be popped out in increasing order if and only if it avoids each unsortable stack-pattern.

*Proof.* Notice that if a stack configuration contains any of the 3 unsortable stack-patterns, then elements involved in the pattern cannot be popped out in increasing order.

For the converse, we prove by induction on the number of elements in the stacks that a configuration which avoids the 3 unsortable stack-patterns can be popped out in increasing order. Suppose that the result has been proved for all stack configurations with at most  $k$  elements. Note that the result is trivially true for  $k \leq 2$ . Let  $c$  be a stack configuration with  $k + 1$  elements avoiding the 3 unsortable stack-patterns and  $m$  the smallest element of this configuration. We show that  $m$  can be popped out so that the stack configuration of the  $k$  remaining elements still avoids the 3 unsortable stack-patterns. Without loss of generality assume  $m = 1$ .

Suppose that 1 lies in stack  $V$ . As  $c$  avoids pattern  $|12|$ ,  $V$  is in decreasing order so 1 is at the top of it. It can be popped out and there remains  $k$  elements still avoiding the 3 unsortable stack-patterns. Thus they can be all popped out in increasing order by induction.

Suppose now that 1 lies in stack  $H$ . As  $c$  avoids pattern  $|132|$  and 1 is the smallest element, all elements above 1 are in increasing order (from 1 to top). All these elements can be pushed onto stack  $V$  so that stack  $V$  remains in decreasing order. Indeed as  $c$  avoids pattern  $|2|13|$ , the top of stack  $V$  is greater than the top of stack  $H$ . When all elements greater than 1 and above 1 in stack  $H$  are transferred onto stack  $V$ , then 1 can be popped out both stacks  $H$  and  $V$  and the remaining configuration still avoids the 3 unsortable stack-patterns (as  $c$  avoids pattern  $|132|$ , no pattern  $|2|13|$  has been created) and we can apply the induction.  $\square$

**Remark 2.11.** There is at most one way to pop out in increasing order elements from a stack configuration. Indeed to pop out we only use moves  $\mu$  and  $\lambda$ , and if we want to pop out in increasing order we have to perform move  $\mu$  if and only if the smallest element lies in the top of  $V$ .

**Data:**  $\sigma$  a permutation and  $c$  a total stack configuration of  $\sigma$ .

**Result:** True if  $c$  can be popped out in increasing order.

```
1  $i \leftarrow 1$ ;  
2 while  $i \leq |\sigma|$  do  
3   if  $\text{top}(V) = i$  then  
4     | pop out  $\text{top}(V)$  from stack  $V$  and let  $i \leftarrow i + 1$   
5   else  
6     | if  $H$  is non empty and  $\text{top}(H) < \text{top}(V)$  then  
7       | pop  $\text{top}(H)$  from stack  $H$  and push it into  $V$ ;  
8     | else  
9       | Return false;  
10    | end  
11  end  
12 end  
13 Return true;
```

**Algorithm 1:** Pop out in increasing order

**Proposition 2.12.** *Let  $c$  be a total stack configuration of a permutation  $\sigma$ . Then Algorithm 1 applied to  $c$  returns true if and only if  $c$  can be popped out in increasing order. Moreover Algorithm 1 runs in linear time w.r.t.  $|\sigma|$ .*

*Proof:* At each step, Algorithm 1 performs either a move  $\mu$  or a move  $\lambda$ . As at most  $|\sigma|$  moves  $\mu$  and  $|\sigma|$  moves  $\lambda$  can be done, it runs in linear time w.r.t.  $|\sigma|$ . We conclude using Remark 2.11.  $\square$

Theorem 2.10 ensures that a stack configuration can be popped out in increasing order. Conditions of this theorem must be verified at each step of a sorting procedure. This is formalised in the following proposition:

**Proposition 2.13.** *If  $w$  is a valid stack word for the permutation  $\sigma$ , then each stack configuration of  $(c_k(w, \sigma))$  avoids the 3 unsortable stack-patterns.*

The converse is not true: let  $w = (\rho\lambda\mu)^n$  then for all permutation  $\sigma$  of size  $n$  each stack configuration of  $(c_k(w, \sigma))$  avoids the 3 unsortable stack-patterns (as it has at most one element in the stacks). But if  $\sigma$  is not the identity,  $w$  is not a valid stack word for  $\sigma$ .

For 2-stack pushall sorting, however, it is sufficient to check whether the stack configuration obtained just after the last element of  $\sigma$  has been pushed onto  $H$  avoids the 3 unsortable stack-patterns.

**Proposition 2.14.** *A permutation  $\sigma$  is 2-stack pushall sortable if and only if there is a way to put all its elements in the stacks so that the total stack configuration obtained avoids the three unsortable patterns.*

*Proof:* If  $\sigma$  is 2-stack pushall sortable we conclude using Proposition 2.13 and Remark 2.8. The converse is a consequence of Theorem 2.10.  $\square$

## 2.2 Decomposition and stack sorting

In this part we exhibit conditions for a permutation  $\sigma$  to be 2-stack sorted depending on its decomposition.



**$\ominus$ -decomposable permutations :**

**Proposition 2.15.** *A permutation  $\sigma = \ominus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  is 2-stack sortable if and only if every  $\pi^{(i)}$  for  $i \in \{1 \dots k-1\}$  is 2-stack pushall sortable and  $\pi^{(k)}$  is 2-stack sortable.*

*Proof.* Suppose that  $\sigma$  is 2-stack sortable. Let  $w_\sigma$  be a valid stack word of  $\sigma$ . For  $i \in \{1 \dots k\}$ , consider the subword  $w_{\pi^{(i)}}$  of  $w_\sigma$  by taking letters corresponding to an element of  $\pi^{(i)}$ . This word is of size  $3|\pi^{(i)}|$  and has equal number of occurrences of the letters  $\rho, \lambda, \mu$ . Moreover, it is a valid stack word for  $\pi^{(i)}$  as the relative order of elements of  $\pi^{(i)}$  under the action of  $w_{\pi^{(i)}}$  will be the same as the action of  $w_\sigma$  on  $\sigma$ . Furthermore, as the element 1 in  $\sigma$  belongs to the last block  $\pi^{(k)}$ , all elements of  $\pi^{(i)}$  are pushed into the stacks before the first pop. Hence  $\pi^{(i)}$  is 2-stack pushall sortable. 2-stack sortable permutations form a permutation class, so that  $\pi^{(k)}$  must be 2-stack sortable.

Conversely, if every  $\pi^{(i)}$  for  $i \in \{1 \dots k-1\}$  is 2-stack pushall sortable and  $\pi^{(k)}$  is 2-stack sortable, let  $w_i$  ( $1 \leq i \leq k-1$ ) be a pushall stack word for  $\pi^{(i)}$  and  $w_k$  be a stack word for  $\pi^{(k)}$ . Then each  $w_i$  ( $1 \leq i \leq k-1$ ) can be written as  $w'_i w''_i$  where  $w'_i$  contains no occurrence of  $\mu$  and  $w''_i$  no occurrence of  $\rho$ . It is easy to check that the word  $w'_1 w'_2 \dots w'_{k-1} w_k w''_{k-1} w''_{k-2} \dots w''_1$  is a valid stack word for  $\sigma$ , hence  $\sigma$  is 2-stack sortable.  $\square$

With a similar proof, we have the following result when restricting to 2-stack pushall sortable permutations:

**Proposition 2.16.** *A permutation  $\sigma = \ominus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  is 2-stack pushall sortable if and only if every  $\pi^{(i)}$  for  $i \in \{1 \dots k\}$  is 2-stack pushall sortable.*

**$\oplus$ -decomposable permutations** The case where  $\sigma$  is  $\oplus$ -decomposable is a bit different as each block of the decomposition can be popped out as soon as they are pushed into the stacks. So the only condition is given in the following proposition.

**Proposition 2.17.** *If  $\sigma = \oplus[\pi^{(1)}, \dots, \pi^{(k)}]$  then  $\sigma$  is 2-stack sortable if and only if each  $\pi^{(i)}$  is 2-stack sortable.*

For 2-stack pushall sortable permutations,  $\oplus$ -decomposable permutations are harder to handle. As no element can be popped out before all elements have been pushed, the element 1 which belongs to the first block must remain in the stacks until every element is pushed. This induces several constraints which are proved in the following propositions. All these propositions aim at proving Theorem 2.18 which fully characterizes  $\oplus$ -decomposable 2-stack pushall sortable permutations.

**Theorem 2.18.** *Let  $\sigma$  be a  $\oplus$ -decomposable permutation. Then  $\sigma$  is 2-stack pushall sortable if and only if  $\sigma$  avoids*

$$B_+ = \{132465, 135246, 142536, 142635, 143625, 153624, 213546, 214365, 214635, 215364, \\ 241365, 314265, 315246, 315426, 351426, 1354627, 1365724, 1436527, 1473526, 1546273, \\ 1573246, 1624357, 1627354, 1632547, 1632574, 1642573, 1657243, 2465137, 2631547, \\ 2635147, 3541627, 4621357, 4652137, 5136427, 5162437, 21687435, 54613287\}$$

The proof proceeds step by step in Propositions 2.19 to 2.26.

**Proposition 2.19.** *Let  $\sigma$  be a permutation such that either:*

- $\sigma \in Av(132)$
- $\sigma \in Av(213)$
- $\sigma \in \oplus[Av(132), Av(213)]$
- $\sigma \in \oplus[Av(213), Av(132)]$

Then  $\sigma$  is 2-stack pushall sortable.

*Proof.* We show that we can put all elements of  $\sigma$  in the stacks so that they avoid patterns of Theorem 2.10 (p.6). In the first case, just push every element in stack  $H$ . For the second case, we know from Knuth [9] that each permutation avoiding 231 can be sort in increasing order with one stack. So each permutation avoiding 213 can be sort in deacreasing order with one stack. Hence we can use stack  $H$  to push all elements of  $\sigma$  in decreasing order onto stack  $V$ . For the last two cases, we push the elements in corresponding stacks  $H$  for  $Av(132)$  and  $V$  for  $Av(213)$ . In each case, the stack configuration respect conditions of Theorem 2.10.  $\square$

Note that Proposition 2.19 give a sufficient condition which is not necessary: the permutation 143652 is 2-stack pushall sortable but do not belong to one of the preceding cases. In this proposition, an important role is given to classes  $Av(213)$  and  $Av(132)$ . These indeed are exactly the classes of permutations that can be pushall sorted with a stack configuration where all elements lie in one *single* stack ( $V$  for  $Av(213)$  and  $H$  for  $Av(132)$ ). Thus the only difficult case is whenever a permutation contains both pattern 132 and 213. This is characterized by the following proposition:

**Proposition 2.20.** *A permutation  $\sigma$  contains both patterns 213 and 132 if and only if it contains one of the following patterns: 1324, 2143, 2413, 3142, 465213 and 546132.*

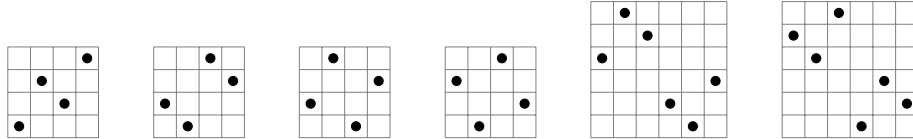


Figure 3: Minimal permutations containing patterns 132 and 213.

*Proof.* Minimal permutations that contain both 132 and 213 are exactly permutations of the basis of  $Av(132) \cup Av(213)$ . By minimality of the elements of the basis those permutations are at most of size 6 and a comprehensive study ends the proof.  $\square$

To prove a complete characterization of  $\oplus$ -decomposable 2-stack pushall sortable permutations, we deal first with permutations whose decomposition contains non-trivial block -i.e. blocks not reduced to a singleton-.

**Proposition 2.21.** *Suppose  $\sigma = \oplus[\alpha_1 \dots \alpha_r]$  with  $r \geq 2$ , each  $\alpha_i$   $\oplus$ -indecomposable and blocks  $\alpha_1$  and  $\alpha_r$  are non-trivial. Then  $\sigma$  is 2-stack pushall sortable if and only if  $\sigma$  avoids every pattern of  $B_1 = \{132465, 213546, 214365, 214635, 215364, 241365, 314265, 1657243, 4652137, 21687435, 54613287\}$ .*

*Proof.* We state by checking each pushall stack word of the right size that permutations of  $B_1$  are not 2-stack pushall sortable. Hence if  $\sigma$  is 2-stack pushall sortable it avoids  $B_1$ . Conversely, let  $\sigma$  be a permutation avoiding every pattern of  $B_1$ . As  $\alpha_1$  and  $\alpha_r$  are non-trivial and  $\oplus$ -indecomposable, they contain 21 as a pattern. But  $\sigma$  avoids 214365 so that blocks  $\alpha_i$  with  $2 \leq i \leq r-1$  are trivial. Let  $I = \{i \mid \alpha_i \text{ contains pattern } 132\}$  and  $J = \{j \mid \alpha_j \text{ contains pattern } 213\}$ . These sets are included in  $\{1, r\}$  and not equal to  $\{1, r\}$  as  $\sigma$  avoids 132465 and 213546.

- If  $I = J = \emptyset$ , then  $\alpha_1 \in Av(132)$  and  $\oplus[\alpha_2 \dots \alpha_r] \in Av(213)$  so  $\sigma \in \oplus[Av(132), Av(213)]$  and  $\sigma$  is 2-stack pushall sortable by Proposition 2.19.
- If  $I = \emptyset$  and  $J = \{j_0\}$ , then  $j_0 \in \{1, r\}$ . If  $j_0 = 1$  then  $\alpha_1 \in Av(132)$  and  $\oplus[\alpha_2 \dots \alpha_r] \in Av(213)$  hence  $\sigma \in \oplus[Av(132), Av(213)]$  and  $\sigma$  is 2-stack pushall sortable by Proposition 2.19. If  $j_0 = r$ , as  $\sigma$  avoids 213546 then  $r = 2$ , but  $\alpha_1 \in Av(213)$  and  $\alpha_r \in Av(132)$  hence  $\sigma \in \oplus[Av(213), Av(132)]$ . So  $\sigma$  is 2-stack pushall sortable by Proposition 2.19.
- If  $I = \{i_0\}$  and  $J = \emptyset$ , then  $i_0 \in \{1, r\}$ . If  $i_0 = 1$ , as  $\sigma$  avoids 132465 then  $r = 2$ , but  $\alpha_1 \in Av(213)$  and  $\alpha_r \in Av(132)$  hence  $\sigma \in \oplus[Av(213), Av(132)]$ . So  $\sigma$  is 2-stack pushall sortable by Proposition 2.19. If  $i_0 = r$  then  $\alpha_1 \in Av(132)$  and  $\oplus[\alpha_2 \dots \alpha_r] \in Av(213)$  hence  $\sigma \in \oplus[Av(132), Av(213)]$  and  $\sigma$  is 2-stack pushall sortable by Proposition 2.19.
- If  $I = \{i_0\} \neq J = \{j_0\}$ . If  $i_0 = 1$  then  $j_0 = r$  and  $r = 2$  as  $\sigma$  avoids 132465. But  $\alpha_1 \in Av(213)$  and  $\alpha_r \in Av(132)$  hence  $\sigma \in \oplus[Av(213), Av(132)]$  and  $\sigma$  is 2-stack pushall sortable by Proposition 2.19. If  $i_0 = r$  then  $j_0 = 1$ ,  $\alpha_1 \in Av(132)$  and  $\oplus[\alpha_2 \dots \alpha_r] \in Av(213)$  hence  $\sigma \in \oplus[Av(132), Av(213)]$ . So  $\sigma$  is 2-stack pushall sortable by Proposition 2.19.
- If  $I = J = \{i_0\}$ , then by Proposition 2.20,  $\alpha_{i_0}$  contains either 1324, 2143, 2413, 3142, 465213 or 546132. We prove that  $\sigma$  contains a pattern of  $B_1$ . If  $\alpha_{i_0}$  contains 1324, either  $i_0 < r$ , and  $\sigma$  would contain 132465 or  $i_0 = r$ , and  $\sigma$  would contain 213546. Similarly if  $\alpha_{i_0}$  contains 2143,  $\sigma$  would contain 214365. The same goes for  $\alpha_{i_0}$  containing 2413, 3142, 465213 or 546132. Hence the case  $I = J = \{i_0\}$  cannot occur.

□

Given two permutation classes  $\mathcal{C}$  and  $\mathcal{C}'$ , their horizontal juxtaposition  $[\mathcal{C} \ \mathcal{C}']$  consists of all permutations  $\sigma$  that can be written as a concatenation  $[\pi, \tau]$  where  $\pi$  is order isomorphic to a permutation in  $\mathcal{C}$  and  $\tau$  is order-isomorphic to a permutation in  $\mathcal{C}'$ . In other words, a diagram of a permutation  $\sigma \in [\mathcal{C} \ \mathcal{C}']$  can be divided by a vertical line into two parts, such that the left one is order-isomorphic to a permutation of  $\mathcal{C}$  and the right one to a permutation of  $\mathcal{C}'$ . We can similarly define the vertical juxtaposition  $\begin{bmatrix} \mathcal{C} \\ \mathcal{C}' \end{bmatrix}$  consisting of permutations having a diagram cut by a horizontal line.

**Proposition 2.22.** *A permutation  $\oplus[1, \sigma]$  is 2-stack pushall sortable if and only if  $\sigma \in [Av(213) \ Av(132)]$  and there exists an associated decomposition  $\sigma = [\pi, \tau]$  such that there are no pattern 213 in  $\sigma$  where 2 is in  $\pi$  and 13 is in  $\tau$ .*

*Proof.* If  $\sigma = [\pi, \tau]$  with this decomposition satisfying hypothesis of the proposition, then  $\oplus[1, \sigma]$  is 2-stack pushall sortable using the following algorithm. Put 1 in  $H$ . Then push elements of  $\pi$  in stack  $V$  in decreasing order. Then put 1 at top of  $V$  and finally push every element of  $\tau$  onto  $H$ . As there are no pattern 213 in  $\sigma$  with 2 in  $\pi$  and 13 in  $\tau$ , the stack configuration respects conditions of Theorem 2.10 hence can be popped out.

Conversely, suppose that  $\oplus[1, \sigma]$  is 2-stack pushall sortable and consider a stack word for this permutation. As 1 is the first element, it is pushed at the bottom of  $H$ . Then some elements are pushed onto 1 and into  $V$  before 1 is popped out from stack  $H$  to stack  $V$ . The remaining elements are pushed into  $H$  as they are greater than 1. We consider the moment where all elements have been pushed and 1 is at the top of  $V$ . This separates in two parts the elements of  $\sigma$  taking  $\tau$  as the elements in  $H$  and  $\pi$  the elements in  $V$  apart from 1. From Theorem 2.10 decomposition  $\sigma = [\pi, \tau]$  satisfies conditions of the statement.  $\square$

**Proposition 2.23.** *Let  $E = \{\sigma | \oplus[1, \sigma] \text{ is 2-stack pushall sortable}\}$ . Then  $E$  is a finitely based permutation class whose basis is  $B_2 = \{21354, 24135, 31425, 31524, 32514, 42513, 243516, 254613, 325416, 362415, 435162, 462135, 513246, 516243, 521436, 521463, 531462, 546132, 4652137\}$ .*

*Proof.* As 2-stack pushall sortable permutations is a permutation class, so does  $E$ . Let  $B_2$  be the basis of  $E$ . To prove that  $B_2$  is finite, we first prove that every permutation in  $B_2$  has size less than 9. Then an comprehensive computation gives the permutations in  $B_2$ .

By Proposition 2.22,  $E = \{\sigma = \pi\tau \mid \pi \in Av(213), \tau \in Av(132) \text{ and there are no pattern 213 in } \sigma \text{ where 2 is in } \pi \text{ and 13 is in } \tau\}$ . Let  $\sigma \in B_2$ . By definition  $\sigma \notin E$  so  $\sigma \notin Av(213)$  and  $\sigma \notin Av(132)$ . Let  $\sigma_i\sigma_j\sigma_k$  be a pattern 132 such that  $i$  is maximal and  $\sigma_r\sigma_s\sigma_t$  be a pattern 213 such that  $t$  is minimal, then  $r$  minimal (for  $t$  fixed) and finally  $s$  maximal (for  $t$  and  $r$  fixed).

- If  $t < i$  then  $\pi = \sigma_r\sigma_s\sigma_t\sigma_i\sigma_j\sigma_k \notin E$ , hence by minimality of the basis  $\sigma = \pi$  so  $|\sigma| = 6$ .
- If  $t = i$  then  $\pi = \sigma_r\sigma_s\sigma_i\sigma_j\sigma_k \notin E$  and by minimality  $\sigma = \pi$  so  $|\sigma| = 5$ .
- If  $t > i$ , consider the pattern  $\sigma_r\sigma_s\sigma_t$  (shown in Figure 4). Minimality conditions for  $t$  and  $r$  and maximality condition for  $s$  imply that gray zones in the diagram of  $\sigma$  are empty. So  $s = t - 1$ . As  $\sigma \notin E$ , there is no possible cut  $\sigma = \pi\tau$  such that  $\pi \in Av(213)$ ,  $\tau \in Av(132)$  and there are no pattern 213 in  $\sigma$  where 2 is in  $\pi$  and 13 is in  $\tau$ . Hence, all cuts in  $\sigma$  are forbidden, either because they are to the left of a 132 pattern or to the right of a 213 pattern or between element 2 and 1 of a pattern 213. More specially the cut between  $t - 1$  and  $t$  is forbidden. This cut cannot be to the left of a pattern 132 by maximality of  $i$  ( $t > i$ ) and cannot be to the right of a pattern 213 by minimality of  $t$ . So this cut is between elements 2 and 1 of a pattern 213. We consider a pattern 213 denoted by  $\sigma_x\sigma_y\sigma_z$  such that  $x$  is minimal and  $y$  is minimal for  $x$  fixed among patterns 213 such that  $x \leq s = t - 1$  and  $y \geq t$ .

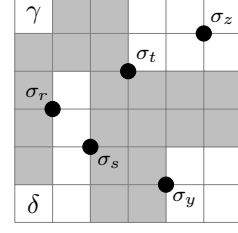
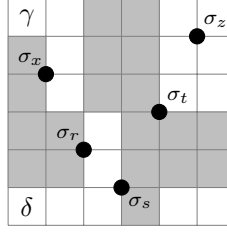
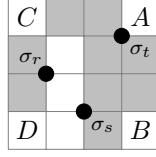
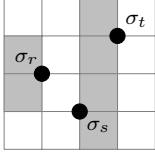


Figure 4:  $\sigma_r\sigma_s\sigma_t$     Figure 5: Cas  $r > i$     Figure 6:  $\sigma_x\sigma_r\sigma_s\sigma_t\sigma_z$     Figure 7:  $\sigma_r\sigma_s\sigma_t\sigma_y\sigma_z$

- If  $r \leq i$  then  $\pi = \{\sigma_r\sigma_s\sigma_t\sigma_i\sigma_j\sigma_k\sigma_x\sigma_y\sigma_z\} \notin E$ . Indeed all cuts are forbidden: those before  $r$  by  $\sigma_i\sigma_j\sigma_k$ , between  $r$  and  $s$  by  $\sigma_r\sigma_s\sigma_t$ , between  $s$  and  $t$  by  $\sigma_x\sigma_y\sigma_z$  and before  $t$  by  $\sigma_r\sigma_s\sigma_t$ . So by minimality of the basis  $|\sigma| \leq 9$ .
- If  $r > i$ , we want to prove that  $x \leq i$ . Then  $\pi = \{\sigma_r\sigma_s\sigma_t\sigma_i\sigma_j\sigma_k\sigma_x\sigma_y\sigma_z\} \notin E$  since all cuts are forbidden as before and  $|\sigma| \leq 9$ . As  $r > i$  and  $i$  maximal, gray zones added in Figure 5 are empty. As  $y \geq t$ ,  $\sigma_y$  and  $\sigma_z$  lie either both in  $A$ , or both in  $B$ , or  $\sigma_y$  lies in  $B$  and  $\sigma_z$  in  $A$ .
  - \* If  $\sigma_y$  and  $\sigma_z$  lie both in  $B$ , then  $\sigma_x$  lies in  $D$  and  $\sigma_x\sigma_t\sigma_z$  form the permutation 132 and as  $i$  is maximal,  $x \leq i$ .
  - \* If  $\sigma_y$  and  $\sigma_z$  lie both in  $A$ , then  $\sigma_x$  lies in  $C$  and by minimality of  $y$  we have  $y = t$ .  $x$  is minimal, so that gray zones added in Figure 6 are empty. Suppose that  $x > i$ . The cut between  $i$  and  $i + 1$  is forbidden as  $\sigma \notin E$ . As  $i$  is maximal the cut cannot be to the left of a pattern 132, neither to the right of a pattern 213 by minimality of  $t$ . Hence the cut lies between element 2 and 1 of a pattern 213. Let  $\sigma_a\sigma_b\sigma_c$  be such a pattern 213 such that  $a \leq i$  and  $b > i$ . Then  $a < x$  and  $\sigma_a$  lies in area  $\gamma$  or  $\delta$  and  $c \geq t$  by minimality of  $t$ . If  $\sigma_a$  lies in  $\gamma$  then  $\sigma_a\sigma_t\sigma_c$  is the pattern 213, which is forbidden by minimality of  $x$ . Hence  $\sigma_a$  lies in  $\delta$  and  $b \geq t$  otherwise  $\sigma_a\sigma_b\sigma_t$  is a pattern 213 with  $a \leq i < r$ , which is also forbidden by minimality of  $r$ . Hence  $\sigma_a\sigma_b\sigma_c$  is a pattern 213 with  $a \leq i < x \leq s$  and  $b \geq t$  which is impossible by minimality of  $x$ .
  - \* If  $\sigma_y$  lies in  $B$  and  $\sigma_z$  in  $A$ , by minimality of  $x$ ,  $x = r$  or  $\sigma_x$  lies in  $C$  or  $\sigma_x$  lies in  $D$ . If  $\sigma_x$  lies in  $C$  then  $\sigma_x\sigma_t\sigma_z$  is a pattern 213 which contradicts the minimality of  $y$ . If  $\sigma_x$  lies in  $D$ ,  $\sigma_x\sigma_r\sigma_s$  is a pattern 132 hence  $x \leq i$ . If  $x = r$ , by minimality of  $x$  then  $y$ , gray zones added in Figure 7 are empty. The cut between  $i$  and  $i + 1$  is forbidden as  $\sigma \notin E$ . As before the cut lies between elements 2 and 1 of a pattern 213. Let  $\sigma_a\sigma_b\sigma_c$  such a pattern 213 such that  $a \leq i$  and  $b > i$ . Then  $a < r$  and  $\sigma_a$  lies in  $\gamma$  or  $\delta$  and  $c \geq t$  by minimality of  $t$ . If  $\sigma_a$  lies in  $\gamma$  then  $\sigma_a\sigma_t\sigma_c$  is a pattern 213 and by minimality of  $x$ ,  $x \leq a \leq i$ . If  $\sigma_a$  lies in  $\delta$  then  $b \geq t$  otherwise  $\sigma_b\sigma_t\sigma_y$  is a pattern 132 with  $b > i$ , which is forbidden by maximality of  $i$ . But  $\sigma_a\sigma_b\sigma_c$  is a pattern 213 with  $a \leq i$  and  $b \geq t$ , so by minimality of  $x$ ,  $x \leq i$ .

□

**Proposition 2.24.** *A permutation  $\oplus[\sigma, 1]$  is 2-stack pushall sortable if and only if  $\sigma \in \left[ \begin{array}{c} Av(132) \\ Av(213) \end{array} \right]$  and there exists an associated decomposition  $\sigma = \left[ \begin{array}{c} \pi \\ \tau \end{array} \right]$  such that there is no pattern 132 in  $\sigma$  where element 3 is in  $\pi$  and elements 1 and 2 are in  $\tau$ .*

*Proof.* Let  $n = |\sigma| + 1$ . Consider a pushall sorting of  $\oplus[\sigma, 1]$ . This permutation has  $n$  as last element, so that we consider the configuration of the stacks just after the insertion of  $n$ . By Theorem 2.10, it must avoid the pattern  $|2|13|$ , so that all elements in  $H$ -under  $n$ - are greater than those of  $V$ . Hence we can write  $\sigma = \begin{bmatrix} \pi \\ \tau \end{bmatrix}$  where  $\tau$  contains elements of  $V$  and  $\pi$  those in  $H$ -except  $n$ -. Then from Theorem 2.10  $\pi \in Av(132)$  and  $\tau \in Av(213)$  and that there are no pattern  $132$  in  $\sigma$  where element 3 is in  $\pi$  and elements 1 and 2 are in  $\tau$ .

Conversely, suppose that there exists a decomposition  $\sigma = \begin{bmatrix} \pi \\ \tau \end{bmatrix}$  respecting the previous conditions then we have a pushall sorting of the permutation  $\oplus[\sigma, 1]$  using the following algorithm. While the input is not empty, if stack  $H$  is empty or if the top of  $H$  belongs to  $\pi$ , we push the next element of the input onto  $H$ . If  $\sigma_i$ , the top of  $H$  belongs to  $\tau$ , and if the next element of the input  $\sigma_j$  belongs to  $\tau$  and is greater than  $\sigma_i$ , we push  $\sigma_j$  onto  $H$ , otherwise we pop  $\sigma_i$  from  $H$  and push it onto  $V$ . At each step we verify conditions of Theorem 2.10 so that all elements can be popped out in increasing order at the end.  $\square$

**Proposition 2.25.** *Let  $F = \{\sigma | \oplus[\sigma, 1] \text{ is 2-stack pushall sortable}\}$ .  $F$  is a finitely based permutation class whose basis is  $B_3 = \{13524, 14253, 21354, 31524, 31542, 35142, 135462, 143652, 162435, 163254, 246513, 263154, 263514, 354162, 462135, 465213, 513642, 516243, 1657243\}$ .*

*Proof.* As the set of 2-stack pushall sortable permutations is a permutation class, so is  $F$ . By Proposition 2.24,  $F = \{\sigma \in \begin{bmatrix} Av(132) \\ Av(213) \end{bmatrix} \text{ such that there exists an associated decomposition } \sigma = \begin{bmatrix} \pi \\ \tau \end{bmatrix} \text{ such that there is no pattern } 132 \text{ in } \sigma \text{ where element 3 is in } \pi \text{ and elements 1 and 2 are in } \tau\}$ . Hence  $E$  and  $F$  are in one-to-one correspondence by taking an element of  $E$ , rotate its diagram by  $-\pi/2$  and apply the symmetry with respect to axis  $(Oy)$ . If elements are in one-to-one correspondence by rotation and symmetry so does the basis which proves the result.  $\square$

**Proposition 2.26.** *A permutation  $\oplus[1, \sigma, 1]$  is 2-stack pushall sortable if and only if  $\sigma \in \oplus[Av(213), Av(132)]$ .*

*Proof.* By Proposition 2.22,  $\oplus[1, \sigma, 1]$  is 2-stack pushall sortable if and only if  $\oplus[\sigma, 1] \in [Av(213), Av(132)]$  and there exists a corresponding decomposition  $\sigma = \pi\tau$  such that there is no pattern  $213$  in  $\sigma$  where element 2 is in  $\pi$  and 13 are in  $\tau$ , which is equivalent to  $\sigma \in [Av(213), Av(132)]$  and there exists a corresponding decomposition  $\sigma = \pi\tau$  such that there are no pattern  $21$  in  $\sigma$  where element 2 is in  $\pi$  and element 1 is in  $\tau$ , i.e.  $\sigma \in \oplus[Av(213), Av(132)]$ .  $\square$

We are now able to prove Theorem 2.18 (p.8).

*Proof.* Permutations of  $B_+$  are not 2-stack pushall sortable (check each pushall stack word of the right size), hence if  $\sigma$  is 2-stack pushall sortable it avoids  $B_+$ . Conversely suppose that  $\sigma$  avoids  $B_+$ . Let  $\sigma = \oplus[\alpha_1 \dots \alpha_r]$  be the  $\oplus$ -decomposition of  $\sigma$  with  $r \geq 2$  and  $\alpha_i$   $\oplus$ -indecomposable for all  $i$ .

- If  $\alpha_1$  and  $\alpha_r$  are non trivial then  $\sigma$  is 2-stack pushall sortable thanks to Proposition 2.21. Indeed  $\sigma$  avoids  $B_1 = \{132465, 213546, 214365, 214635, 215364, 241365, 314265, 1657243, 4652137, 21687435, 54613287\}$  as  $B_1 \subset B_+$ .

- If  $\alpha_1$  is trivial then  $\sigma = \oplus[1, \pi]$  and  $\pi$  avoids  $B_2 = \{21354, 24135, 31425, 31524, 32514, 42513, 243516, 254613, 325416, 362415, 435162, 462135, 513246, 516243, 521436, 521463, 531462, 546132, 4652137\}$  so that  $\sigma$  is 2-stack pushall sortable by Proposition 2.23.
- If  $\alpha_r$  is trivial then  $\sigma = \oplus[\pi, 1]$  and  $\pi$  avoids  $B_3 = \{13524, 14253, 21354, 31524, 31542, 35142, 135462, 143652, 162435, 163254, 246513, 263154, 263514, 354162, 462135, 465213, 513642, 516243, 1657243\}$  hence  $\sigma$  is 2-stack pushall sortable by Proposition 2.25.

□

We call *separable* permutations the class  $Av(2413, 3142)$ .

**Theorem 2.27.** *Let  $\sigma$  be a separable permutation.  $\sigma$  is 2-stack pushall sortable if and only if  $\sigma$  avoids  $B = \{132465, 213546, 214365, 1354627, 1436527, 1624357, 1632547, 1657243, 4652137, 21687435, 54613287\}$ .*

*Proof.* As permutations of  $B$  are not 2-stack pushall sortable, every 2-stack pushall sortable permutation avoids  $B$ . Conversely, suppose that  $\sigma$  avoids  $B$ . As  $\sigma$  is separable,  $\sigma$  is either  $\oplus$ -decomposable or  $\ominus$ -decomposable or trivial (i.e. of size 1), and  $\sigma$  avoids 2413 and 3142 which added to constraints of  $B$  gives that  $\sigma$  avoids  $B_+$ , the set defined in Theorem 2.18. If  $\sigma$  is  $\oplus$ -decomposable, then  $\sigma$  is 2-stack pushall sortable by Theorem 2.18. If  $\sigma$  is  $\ominus$ -decomposable, then  $\sigma = \ominus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  where each  $\pi^{(i)}$  is either trivial or  $\oplus$ -decomposable. So  $\sigma$  is 2-stack pushall sortable by Proposition 2.16 and Theorem 2.18. □

### 2.3 Basis of stack sorting class

In the previous section, we show that 2-stack pushall sortable separable permutations form a finitely based permutation class. This property does not hold for 2-stack pushall sortable permutations and we exhibit an infinite antichain in the following proposition.

**Proposition 2.28.** *The basis of 2-stack pushall sortable permutation is infinite.*

*Proof.* Consider permutations  $2n - 3 \ 2n - 1 \ 2n - 5 \ 2n \dots p \ p + 5 \dots 1 \ 6 \ 2 \ 4$  for  $n \geq 3$ . The first ones are depicted in Figure 8. These permutations are simple and incomparable. To complete the proof, straightforward though technical, just check that those permutations are not 2-stack pushall sortable and that every pattern of these permutations are 2-stack pushall sortable.

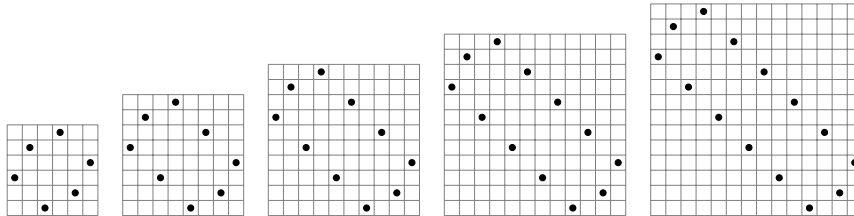


Figure 8: An antichain of the basis of 2-stack pushall sortable permutations class.

□

Note that the basis is infinite and contains a infinite number of simple permutations, and the 2-stack pushall sortable class contains also an infinite number of simple permutations.

**Proposition 2.29.** *If  $\sigma$  is in the basis of 2-stack pushall sortable permutations, then  $\sigma$  is 2-stack sortable.*

*Proof.* Let  $\sigma = \sigma_1\sigma_2\dots\sigma_n$  be in the basis of 2-stack pushall sortable permutations. By definition,  $\sigma_1\sigma_2\dots\sigma_{n-1}$  is 2-stack pushall sortable. We can sort  $\sigma$  (not pushall sort  $\sigma$ ) using the following algorithm. Push all elements  $\sigma_1$  to  $\sigma_{n-1}$  in the stacks following the 2-stack pushall sortable operations of  $\sigma_1\dots\sigma_{n-1}$ . Then pop elements  $1, 2, \dots, \sigma_n - 1$ , then push  $\sigma_n$  and pop it to the output and pop the remaining elements. It is easy to check that these operations are allowed.  $\square$

Those last two propositions give a partial characterization of the basis of 2-stack pushall sortable permutations class and 2-stack sortable permutations. A more accurate result can be given for certain type of permutations in the basis.



**Proposition 2.30.** *Let  $\pi$  be a  $\ominus$ -decomposable permutation. Then  $\pi$  belongs to the basis of 2-stack sortable permutations class if and only if  $\pi = \ominus[\sigma, 1]$  where  $\sigma$  belongs to the basis of 2-stack pushall sortable permutations class.*

*Proof.* Let  $\pi = \ominus[\sigma, 1]$  with  $\sigma$  a permutation of the basis of 2-stack pushall sortable permutations class. Proposition 2.15 ensures that  $\pi$  is not 2-stack sortable. Note also that every pattern of  $\pi$  is 2-stack sortable. To prove this result, suppose that you remove a point in the permutation. Suppose we delete element 1 then the obtained permutation is  $\sigma$ , hence it is 2-stack sortable by Proposition 2.29. Otherwise we delete an element of  $\sigma$  leading to  $\sigma'$  which is 2-stack pushall sortable by the definition of a permutation class basis. Then,  $\ominus[\sigma', 1]$  is 2-stack pushall sortable using Proposition 2.15.

Conversely, if  $\sigma = \ominus[\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(k)}]$  belongs to the basis of 2-stack sortable permutations class, then by Proposition 2.15, either  $\pi^{(k)}$  is not 2-stack sortable which contradicts the minimality of  $\sigma$  ( $\sigma$  is an element of the basis so that every pattern of  $\sigma$  must belong to the class) or there exists  $1 \leq i \leq k - 1$  such that  $\pi^{(i)}$  is not 2-stack pushall sortable. But in that case,  $\ominus[\pi^{(i)}, 1]$  is not 2-stack sortable by Proposition 2.15 hence  $\sigma = \ominus[\pi^{(i)}, 1]$  by minimality of basis elements. If  $\pi^{(i)}$  has a proper pattern  $\tau$  which is not 2-stack pushall sortable then  $\ominus[\tau, 1]$  is a proper pattern of  $\sigma$  which is not 2-stack sortable. This is impossible as  $\sigma$  belongs to the basis of 2-stack sortable permutations class. So  $\pi^{(i)}$  belongs to the basis of 2-stack pushall sortable permutations class, which concludes the proof.  $\square$

### 3 Sorting and bi-coloring

#### 3.1 A simple characterization

There is a natural relation between 2-stack pushall sorting and coloring of permutation diagram into two colors. The key idea is to look at the stack configuration once all elements of the permutation are pushed into the stacks. Then each element of the permutation belong either to stack  $H$  or to stack  $V$ . We assign a color to them depending in which stack they lie at this particular step of the sorting. In this article we color like  points that lie in stack  $H$  and like  points in stack  $V$ .

However by Remark 2.3, this stack configuration is not unique, and neither is the coloring.



**Definition 3.1.** A bicolored of a permutation  $\sigma$  is a coloring of the points of the diagram of  $\sigma$  with two colors  $G$  and  $R$ .

A valid coloring is a bicolored which avoids each of the four following colored pattern:

- pattern  $132$ : there is a pattern  $132$  in  $R$
- pattern  $213$ : there is a pattern  $213$  in  $G$
- pattern  $1X2$ : there is a point of  $R$  lying vertically between a pattern  $12$  of  $G$
- pattern  $2/13$ : there is a point of  $G$  lying horizontally between a pattern  $12$  of  $R$

**Definition 3.2.** Let  $\sigma$  be a permutation. To each total stack configuration of  $\sigma$  the map  $Col$  assigns the bicolored of  $\sigma$  such that elements of  $H$  are in  $R$  and elements of  $V$  are in  $G$ . To every bicolored of a permutation  $\sigma$  the map  $Conf$  associates the total stack configuration of  $\sigma$  such that elements of  $G$  lie in  $V$  in decreasing order of value from bottom to top and elements of  $R$  lie in  $H$  in increasing order of indices from bottom to top.

**Remark 3.3.** For any bicolored  $b$ ,  $Col(Conf(b)) = b$ . For any stack configuration  $c$  such that elements of  $V$  are in decreasing order of value from bottom to top and elements of  $H$  are in increasing order of indices from bottom to top,  $Conf(Col(c)) = c$ .

**Proposition 3.4.** Let  $b$  be a bicolored of a permutation  $\sigma$ . Then Algorithm 2 applied to  $b$  returns true if and only if  $Conf(b)$  is reachable for  $\sigma$ . In this case the stack configuration to which Algorithm 2 leads is  $Conf(b)$ .

To state this proposition we need the two following lemmas:

**Lemma 3.5.** At each step of Algorithm 2, the stack configuration we have is reachable for  $\sigma$ , elements of  $H$  are in increasing order of indices from bottom to top, elements of  $V$  are in decreasing order of value from bottom to top, there is no element of  $R$  in  $V$ , there is no element of  $R$  above an element of  $G$  in  $H$  and elements of  $G$  that lie in  $H$  are in increasing order of value from bottom to top.

Moreover index  $i$  verifies that if  $i \leq |\sigma|$  then  $\sigma_i$  is the next element of the input and if  $i > |\sigma|$  then there is no more element in the input.

*Proof:* The proof is by induction on the number of stack operations performed by the algorithm. Algorithm 2 begins with the empty stack configuration and  $\sigma$  as input and  $i = 1$  so the properties are true at the beginning. Algorithm 2 performs only appropriate stack operations so at each step the configuration obtained is reachable for  $\sigma$ . Moreover in a reachable configuration, elements of  $H$  are in increasing order of indices. When an element is put in  $V$  (this happens at line 9 or 21) then this element is in  $G$  (checked at line 6 or 19) and is smaller than the top of  $V$  (checked at line 8 or 20) so that elements of  $V$  remain in decreasing order of value from bottom to top and  $V$  contains no element of  $R$ . When we put an element in  $H$ , it can be at line 4 or 14. In the first case,  $H$  is empty or its top is in  $R$  (checked at line 3) so all its elements are in  $R$  by induction hypothesis. In the second case, the top of  $H$  is in  $G$  and the element we put in  $H$  is in  $G$  and greater than the top of  $H$ . This ensures that there is no element of  $R$  above an element of  $G$  in  $H$  and that elements of  $G$  that lie in  $H$  are in increasing order from bottom to top (using induction hypothesis). Finally  $i$  is increased exactly when  $\sigma_i$  is put into  $H$  so the last property remains true.  $\square$

```

Data:  $\sigma$  a permutation and  $b$  a bicoloring of  $\sigma$ .
Result: True if the stack configuration corresponding to  $b$  is reachable from  $\sigma$ .
1 Begin with the empty stack configuration and  $\sigma$  as input and  $i = 1$ ;
2 while  $i \leq |\sigma|$  do
3   if  $H$  is empty or  $\text{top}(H) \in R$  then
4     | push  $\sigma_i$  into  $H$ ;
5     |  $i \leftarrow i + 1$ ;
6   else /*  $\text{top}(H) \in G$  */
7     | if  $\sigma_i \in R$  or  $\sigma_i < \text{top}(H)$  then
8       | if  $V$  is empty or  $\text{top}(H) < \text{top}(V)$  then
9         | | pop  $\text{top}(H)$  from stack  $H$  and push it into  $V$ ;
10        | | else
11          | | | Return false;
12        | | end
13        | else /*  $\text{top}(H) \in G$ ,  $\sigma_i \in G$  and  $\sigma_i > \text{top}(H)$  */
14          | | push  $\sigma_i$  into  $H$ ;
15          | |  $i \leftarrow i + 1$ ;
16        | | end
17      | end
18    end
19 while  $H$  is nonempty and  $\text{top}(H) \in G$  do
20   | if  $\text{top}(H) < \text{top}(V)$  then
21     | | pop  $\text{top}(H)$  from stack  $H$  and push it into  $V$ ;
22     | | else
23       | | | Return false;
24     | | end
25   end
26 Return true;

```

**Algorithm 2:** Algorithm to obtain a reachable configuration compatible with a bicoloring

**Lemma 3.6.** *Algorithm 2 terminates in linear time w.r.t  $|\sigma|$ .*

*Proof:* At each step, Algorithm 2 performs either a legal move  $\rho$ , or a legal move  $\lambda$ , or return false or true (and stops). As at most  $|\sigma|$  legal moves  $\rho$  and  $|\sigma|$  legal moves  $\lambda$  can be done, Algorithm 2 terminates after at most  $2|\sigma| + 1$  steps. As each step is done in constant time, we have the result.  $\square$

We are now able to prove Proposition 3.4:

*Proof:* If Algorithm 2 applied to  $b$  returns true, then it reaches line 26. In particular the loop of line 19 stops so the top of  $H$  is not in  $G$ . Thus by Lemma 3.5 there is no element of  $G$  in  $H$ . In addition by the same lemma elements of  $H$  are in increasing order of indices from bottom to top, elements of  $V$  are in decreasing order of value from bottom to top and there is no element of  $R$  in  $V$ . So the stack configuration we have is  $Conf(b)$ . Moreover Lemma 3.5 states that the stack configuration we have is reachable for  $\sigma$ , so  $Conf(b)$  is reachable for  $\sigma$ .

Conversely if  $Conf(b)$  is reachable for  $\sigma$ , then there is a sequence  $w$  of appropriate stack operations so that the configuration obtained with  $\sigma$  as input is  $Conf(b)$ . Let us prove that the sequence of moves  $w'$  performed by Algorithm 2 applied to  $b$  is  $w$ . We prove by induction on  $k \leq |w|$  ( $k \geq 0$ ) that  $w$  and  $w'$  have the same prefix of length  $k$  (obvious for  $k = 0$ ). First notice that as  $Conf(b)$  is a total stack configuration, so  $w$  has no letter  $\mu$ , and that Algorithm 2 performs only moves  $\lambda$  and  $\rho$ , so  $w'$  has no letter  $\mu$ . Suppose that  $w$  and  $w'$  have the same prefix  $v$  of length  $k$  with  $k < |w|$ , let  $c'$  be the stack configuration obtained after performing moves of  $v$  with  $\sigma$  as input. We want to prove that  $w'_{k+1}$  exists and  $w'_{k+1} = w_{k+1}$ . By definition of  $w'$ ,  $w'_{k+1}$  is the move performed by Algorithm 2 in configuration  $c'$  (setting by extension  $w'_{k+1} = \alpha$  if Algorithm 2 terminates in configuration  $c'$ , i.e. if  $|w'| = k$ ), and by definition of  $w$ ,  $w_{k+1}$  is a move which allows to go from configuration  $c'$  to configuration  $Conf(b)$  (maybe with some additional moves).

We check the value of  $i$  after Algorithm 2 has performed moves  $v$ . We know that at this step stacks are in configuration  $c'$ .

If  $i > |\sigma|$ , then from Lemma 3.5 in configuration  $c'$  all elements of  $\sigma$  lie already in the stacks. As  $w$  is a sequence of appropriate stack operations, then  $w_{k+1} \neq \rho$  so  $w_{k+1} = \lambda$  ( $w$  has no letter  $\mu$ ). As  $w_{k+1}$  is a move which allows to go from configuration  $c'$  to configuration  $Conf(b)$  in which there is no elements of  $R$  in  $V$  and  $V$  is decreasing, then the top of  $H$  in  $c'$  is in  $G$  and smaller than the top of  $V$  (or  $V$  is empty). As  $i > |\sigma|$  and as the top of  $H$  in  $c'$  is in  $G$  and smaller than the top of  $V$  (or  $V$  is empty) then Algorithm 2 performs line 21 so  $w'_{k+1} = \lambda = w_{k+1}$ .

If  $i \leq |\sigma|$  then we are in the loop beginning at line 2 of the algorithm and from Lemma 3.5  $\sigma_i$  is the next element of the input. Suppose that  $w_{k+1} = \lambda$ . As  $w_{k+1}$  is a legal move which allows to go from configuration  $c'$  to configuration  $Conf(b)$  in which there is no elements of  $R$  in  $V$  and  $V$  is decreasing, then  $H$  is non empty, the top of  $H$  is in  $G$  and smaller than the top of  $V$ . Suppose in addition that  $\sigma_i \in G$ . As  $\sigma_i$  is still on the input after  $w_{k+1}$  and  $w_{k+1}$  is a move which allows to go to configuration  $Conf(b)$  in which  $V$  is decreasing, then  $\sigma_i$  is smaller than the top of  $H$  in  $c'$ . So either  $\sigma_i < \text{top}(H)$  or  $\sigma_i \in R$ . So from  $c'$  Algorithm 2 performs line 9 so  $w'_{k+1} = \lambda = w_{k+1}$ .

Suppose that  $w_{k+1} = \rho$ . If in configuration  $c'$  stack  $H$  is empty or  $\text{top}(H) \in R$  then Algorithm 2 performs line 4 so  $w'_{k+1} = \rho = w_{k+1}$ . Otherwise let  $\sigma_h$  be the top of  $H$  in  $c'$ , then  $\sigma_h \in G$ . So  $\sigma_h \in V$  in  $Conf(b)$ . But once  $w_{k+1} = \rho$  is performed  $\sigma_i$  is above  $\sigma_h$  in  $H$ . As  $w_{k+1}$  is a move which allows to go from configuration  $c'$  to configuration  $Conf(b)$  then  $\sigma_i$  is below  $\sigma_h$  in  $V$  in  $Conf(b)$  (indeed it is impossible that  $\sigma_h$  goes to stack  $V$  and  $\sigma_i$  remains in

stack  $H$ ). So  $\sigma_i \in G$  and as in  $Conf(b)$  elements of  $V$  are in decreasing order,  $\sigma_i > \sigma_h$ . So the test of line 7 of the algorithm is false and Algorithm 2 performs line 14 so  $w'_{k+1} = \rho = w_{k+1}$ .

This ends the induction. We have proved that  $w$  is a prefix of  $w'$ , so Algorithm 2 reaches configuration  $Conf(b)$ . We have now to prove that Algorithm 2 stops in this configuration and returns true.

When  $Conf(b)$  is reached then there is no element in the input anymore, so from Lemma 3.5  $i > |\sigma|$ , and  $top(H) \notin G$  in  $Conf(b)$ . So both loops while of Algorithm 2 are finished and the algorithm reaches line 27, returns true and terminates in configuration  $Conf(b)$ .  $\square$

**Lemma 3.7.** *Let  $b$  be a bicolored of a permutation  $\sigma$ . If Algorithm 2 applied to  $b$  returns false then  $b$  has a pattern  $1X2$  or a pattern  $213$ .*

*Proof:* We consider the stack configuration reached when Algorithm 2 returns *false*. We set  $\sigma_h = top(H)$  and  $\sigma_v = top(V)$ . By Lemma 3.5,  $\sigma_v \in G$ . Algorithm 2 returns *false* by reaching either line 11 or line 23. In both cases,  $\sigma_h \in G$  and  $\sigma_h > \sigma_v$ . Now we consider the step of the algorithm where  $\sigma_v$  was put in  $V$ , the indice  $i$  at this step of the algorithm, and the corresponding configuration  $c$  just before the move putting  $\sigma_v$  into  $V$  is done. So at this step  $\sigma_v$  is on the top of  $H$ , and  $i > v$ . If  $\sigma_h$  is in  $H$  in  $c$ , then it is below  $\sigma_v$ , contradicting Lemma 3.5 ( $\sigma_h > \sigma_v$  and both are in  $G$ ). As  $\sigma_h$  is in  $H$  when the algorithm ends, it cannot be in  $V$  in  $c$ . So  $\sigma_h$  is still in the input and  $i \leq h \leq |\sigma|$ . Recall that we consider the step of the algorithm where  $\sigma_v$  is put in  $V$ . This can happen at line 9 or 21 but  $i \leq |\sigma|$  so it is at line 9. So the test of line 8 is true, thus either  $\sigma_i \in R$  and then  $\sigma_v, \sigma_i, \sigma_h$  is a pattern  $1X2$  of  $b$ , or  $\sigma_i \in G$  but  $\sigma_i < \sigma_v$  and then  $\sigma_v, \sigma_i, \sigma_h$  is a pattern  $213$  of  $b$ .  $\square$

**Theorem 3.8.** *The map  $Col$  is a bijection from the set of reachable total stack configuration of  $\sigma$  avoiding the three unsortable patterns to the set of valid coloring of  $\sigma$ . Moreover the inverse of  $Col$  is the map  $Conf$ .*

*Proof:* Let  $c$  be a reachable total stack configuration of  $\sigma$  avoiding the three unsortable patterns and set  $b = Col(c)$ . We have to prove that  $b$  avoids every forbidden colored pattern of Definition 3.1.

If  $b$  has a pattern  $132$  in  $R$  then there are three element  $\sigma_i, \sigma_j$  and  $\sigma_k$  of  $R$  such that  $i < j < k$  and  $\sigma_i < \sigma_k < \sigma_j$ . By definition of  $Col$ ,  $\sigma_i, \sigma_j$  and  $\sigma_k$  lie in  $H$ . As  $c$  is reachable and  $i < j < k$ ,  $\sigma_i$  is below  $\sigma_j$  which is below  $\sigma_k$ . So we have a stack-pattern  $|132|$  in  $c$  which contradicts our hypothesis. So  $b$  has no pattern  $132$ .

If  $b$  has a pattern  $213$  in  $G$  then there are three element  $\sigma_i, \sigma_j$  and  $\sigma_k$  of  $G$  such that  $i < j < k$  and  $\sigma_j < \sigma_i < \sigma_k$ . By definition of  $Col$ ,  $\sigma_i, \sigma_j$  and  $\sigma_k$  lie in  $V$ . As  $c$  avoids stack-pattern  $|12|$ ,  $\sigma_k$  is below  $\sigma_i$  which is below  $\sigma_j$ . But then  $c$  is not reachable: as  $\sigma_k$  is below  $\sigma_i$  and  $\sigma_j$  in  $V$ ,  $\sigma_i$  and  $\sigma_j$  have to stay in stack  $H$  until  $\sigma_k$  enters stack  $H$ . But as  $i < j$ ,  $\sigma_i$  is below  $\sigma_j$  in stack  $H$  and cannot be below  $\sigma_j$  in stack  $V$  as going from stack  $H$  to stack  $V$  reverse the order. So  $b$  has no pattern  $213$ .

If  $b$  has a point of  $R$  lying vertically between a pattern  $12$  of  $G$  then there are elements  $\sigma_i$  and  $\sigma_j$  of  $G$  and  $\sigma_k$  of  $R$  such that  $i < k < j$  and  $\sigma_i < \sigma_j$ . By definition of  $Col$ ,  $\sigma_i$  and  $\sigma_j$  lie in  $V$  and  $\sigma_k$  lies in  $H$ . Configuration  $c$  is reachable. We consider a sequence of stack operations leading to  $c$ . As  $i < k$ ,  $\sigma_i$  is already in the stacks when  $\sigma_k$  enters  $H$ . As  $\sigma_k$  remains in  $H$  in  $c$  but  $\sigma_i$  is in  $V$  in  $c$ ,  $\sigma_i$  has to be already in  $V$  when  $\sigma_k$  enters stack  $H$ . As  $k < j$ , at this moment  $\sigma_j$  is not already in stack  $V$ , so  $\sigma_j$  will be above  $\sigma_i$  in  $V$  and they form a pattern  $|12|$  in  $c$ , which is excluded. So  $b$  has no pattern  $1X2$ .

If  $b$  has a point of  $G$  lying horizontally between a pattern  $12$  of  $R$  then in  $c$  these points form a pattern  $|2|13|$  which is excluded. So  $b$  has no pattern  $2/13$ .

Conversely let  $b$  be a valid coloring of  $\sigma$ . By definition  $Conf(b)$  is a total stack configuration of  $\sigma$ . We have to prove that  $Conf(b)$  is reachable for  $\sigma$  and avoids the three unsortable stack patterns. As  $b$  is a valid coloring, it avoids patterns  $1X2$  and  $213$ . So from Lemma 3.7, Algorithm 2 started with input  $b$  returns true. Thus from Proposition 3.4,  $c$  is reachable for  $\sigma$ . Moreover by definition of  $Conf$ ,  $Conf(b)$  avoids pattern  $|12|$ . Furthermore we know that in  $Conf(b)$ , elements of  $H$  are in increasing order of indices from bottom to top. So if  $Conf(b)$  has a pattern  $|132|$ , then  $b$  has a pattern  $132$ , and if  $Conf(b)$  has a pattern  $|2|13|$  then  $b$  has a pattern  $2/13$ . As  $b$  is a valid coloring, we conclude that  $Conf(b)$  avoids the three unsortable stack patterns.

Now using Remark 3.3 it's clear that  $Conf$  is the inverse of  $Col$ . □

**Theorem 3.9.** *A permutation  $\sigma$  is 2-stack pushall sortable if and only if its diagram admits a valid coloring.*

*Proof:* Consequence of Proposition 2.14 and Theorem 3.8. □

Now thank to Theorem 3.9 we have a naive algorithm to check if a permutation  $\sigma$  is 2-stack pushall sortable: for all bicoloring  $b$  of  $\sigma$ , we can test if  $b$  is valid by checking if  $b$  avoids patterns  $213$ ,  $1X2$ ,  $2/13$  and  $132$  of Definition 3.1. But first notice that we have a more efficient way to test if a bicoloring is valid:

**Proposition 3.10.** *Let  $b$  be a bicoloring of a permutation  $\sigma$ . We can check in linear time w.r.t.  $|\sigma|$  if  $b$  is a valid coloring. More precisely,  $b$  is a valid coloring if and only if Algorithm 2 applied to  $b$  returns true and Algorithm 1 applied to  $Conf(b)$  returns true.*

*Proof:* From Theorem 3.8,  $b$  is valid if and only if  $Conf(b)$  is reachable for  $\sigma$  and avoids the three unsortable patterns. We conclude using Lemma 3.6, Proposition 3.4, Theorem 2.10 and Proposition 2.12. □

Now even using this efficient way to test if a bicoloring is valid, the naive algorithm described above is unefficient. Indeed there is  $2^{|\sigma|}$  bicolorings of  $\sigma$ , leading to an exponential algorithm. Yet we will find a way to restrict the possible number of colorings to a polynomial number. The key idea is to look at increasing sequences in the permutation.

### 3.2 Increasing sequences in a valid coloring

First we reformulate the notion of valid coloring thanks to increasing and decreasing sequences.

**Proposition 3.11.** *Let  $c$  be a bicoloring of a permutation  $\sigma$ . Then  $c$  is a valid coloring if and only if  $c$  respects the following set of rules denoted  $\mathcal{R}_8$ :*

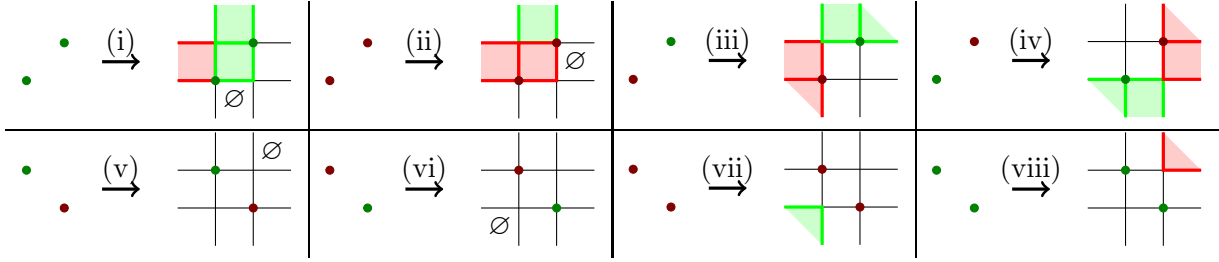


Figure 9: Rewriting rules  $\mathcal{R}_8$

For example rule (i) means that if two points  $(i, \sigma_i)$  and  $(j, \sigma_j)$  are in increasing order  $i < j$  and  $\sigma_i < \sigma_j$  and belong to  $G$  then every point  $(k, \sigma_k)$  of the permutation must respect:

- If  $i < k < j$  then  $\sigma_k > \sigma_i$  and  $(k, \sigma_k)$  belongs to  $G$ .
- If  $k < i$  and  $\sigma_i < \sigma_k < \sigma_j$  then  $(k, \sigma_k)$  belongs to  $R$ .

*Proof.* We prove that  $c$  is not valid if and only if  $c$  violates a rule of  $\mathcal{R}_8$ . Suppose that  $c$  is not valid then  $c$  has one of the four colored patterns of Definition 3.1. If  $c$  has a pattern **132** then  $c$  violate rule (ii) applied to elements 1 and 3 of the pattern **132**, as element 2 of the pattern lies in a zone that should be empty. If  $c$  has a pattern **213** then  $c$  violate rule (i) applied to elements 2 and 3 of the pattern **213**, as element 1 of the pattern lies in a zone that should be empty. If  $c$  has a pattern **1X2** then  $c$  violate rule (i) applied to elements of  $G$  of the pattern **1X2**. If  $c$  has a pattern **2/13** then  $c$  violate rule (ii) applied to elements of  $R$  of the pattern **2/13**. Conversely if  $c$  violates a rule of  $\mathcal{R}_8$  then a comprehensive study shows that  $c$  has one of the four colored patterns of Definition 3.1 and is not valid.  $\square$

We can use implication rules of  $\mathcal{R}_8$  to limit the number of bicoloring to test, using the following idea: knowing the coloring of some points in the permutation (either in  $R$  or in  $G$ ), the deduction rules given in Figure 9 can be applied until we obtain either a contradiction or no more rule can be applied. We can try the following algorithm: Set the color of two increasing points of  $\sigma$ , use implication rules to deduce the color of the other points and test whether the coloring obtained is right. Unfortunately, implication rules are not sufficient to ensure that given the color two points, the color of all other points is set. We may have to choose arbitrary the color of lots of points. To ensure that the number of bicoloring to test is polynomial, we have to study more precisely properties of increasing sequences in a valid bicoloring.

**Definition 3.12.** Let  $c$  be a bicoloring of a permutation  $\sigma$ . We call increasing sequence  $RG$  a pair of two points  $(\sigma_i, \sigma_j)$  such that  $i < j$ ,  $\sigma_i < \sigma_j$ ,  $\sigma_i \in R$  and  $\sigma_j \in G$ . We define in the same way increasing sequences  $GR$ ,  $RR$  or  $GG$ .

Rule (iii) of  $\mathcal{R}_8$  implies that every increasing sequence  $RG$  fixes the color of all points to the left of  $\sigma_i$  below  $\sigma_j$  (which are in  $R$ ) and to the right of  $\sigma_i$  above  $\sigma_j$  (which are in  $G$ ). The following theorem shows that when  $\sigma$  is  $\ominus$ -indecomposable, the color of points to the left of  $\sigma_i$  above  $\sigma_j$  is also fixed.

**Theorem 3.13.** Consider a valid coloring of a  $\ominus$ -decomposable permutation  $\sigma$ . If there exist two points  $\sigma_i < \sigma_j, i < j$  such that  $\sigma_i \in \mathbf{R}$  and  $\sigma_j \in \mathbf{G}$ , then the color of every point  $\sigma_k$  with  $k < i$  or  $\sigma_k > \sigma_j$  is determined by iterations of rules  $\mathcal{C}_8$  knowing only the color of  $\sigma_i$  and  $\sigma_j$  and can be represented as follows, the second diagram being a short representation of this alternance which will be used in the sequel. Furthermore, any increasing sequence  $(i, \sigma_i), (j, \sigma_j)$  of points located either to the left of  $i$  or to the top of  $\sigma_j$  is either monochromatic or colored  $RG$ . Moreover, knowing  $\sigma_i$  and  $\sigma_j$ , we can decide the color of the points whose indices are less than  $i$  or whose values are greater than  $\sigma_j$  in linear time.

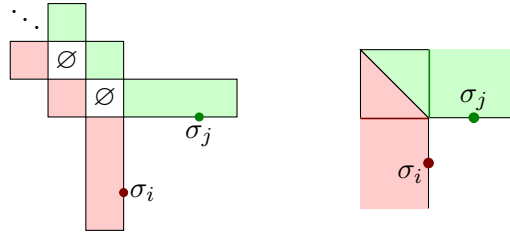


Figure 10: Zone  $A_{RG}$

*Proof:* The proof is by induction on the *assigned* border between the zone not yet assigned to a stack and the *assigned* zone containing  $\sigma_i$  and  $\sigma_j$  where points are forced to be in a specific stack. At first, the *assigned* border is reduced to the segments  $(i, 1) - (i, \sigma_j) - (n, \sigma_j)$  as well as the assigned zone (where  $n = |\sigma|$ ).

More formally we build sequences  $(\sigma_{i_k})$  and  $(\sigma_{j_k})$  such that  $\sigma_{i_k} \sigma_{j_k}$  in an increasing sequence  $RG$  and the color of all points lying in the set  $C_k = \{\sigma_\ell \mid i_k \leq \ell \leq i \text{ and } \sigma_j \leq \sigma_\ell \leq \sigma_{j_k}\}$  is determined and respects Figure 10. We set  $i_0 = i$  and  $j_0 = j$ . We prove that if  $(\sigma_{i_k} \neq 1$  or  $\sigma_{j_k} \neq n)$  then we can build  $\sigma_{i_{k+1}}$  and  $\sigma_{j_{k+1}}$  such that  $\sigma_{i_{k+1}} < \sigma_{i_k}$  or  $\sigma_{j_{k+1}} > \sigma_{j_k}$ .

We set  $H_k = \{\sigma_\ell \mid \ell \leq i_k \text{ and } \sigma_\ell \leq \sigma_{j_k}\}$  and  $V_k = \{\sigma_\ell \mid \ell \geq i_k \text{ and } \sigma_\ell \geq \sigma_{j_k}\}$  (see Figure 11). By rule (iii) applied to  $\sigma_{i_k}$  and  $\sigma_{j_k}$ ,  $H_k \subset H$  and  $V_k \subset V$ . Then, different situations may happen depending on whether areas  $H_k$  and  $V_k$  are empty:

**$H_k$  and  $V_k$  empty:** Then  $\sigma$  is  $\ominus$ -decomposable which is in contradiction with our hypothesis.

**$H_k$  and  $V_k$  both non empty:** If both of the colored zones  $H_k$  or  $V_k$  are non empty, we set  $i_{k+1} = \min\{\ell \mid \sigma_\ell \in H_k\}$  and  $\sigma_{j_{k+1}} = \max V_k$  (see Figure 11). Then  $C_{k+1} = C_k \cup H_k \cup V_k \cup Z$  is a partition of  $C_{k+1}$ , where  $Z = \{\sigma_\ell \mid i_{k+1} \leq \ell \leq i_k \text{ and } \sigma_{j_k} \leq \sigma_\ell \leq \sigma_{j_{k+1}}\}$  (see Figure 11). The only points of  $C_{k+1}$  whose color is not determined yet are those of  $Z$ . If  $Z$  is not empty consider a point  $\sigma_\ell$  of  $Z$ . If  $\sigma_\ell \in V$  then rule (i) applied to  $\sigma_\ell$  and  $\sigma_{j_{k+1}}$  is in contradiction with the existence of  $\sigma_{i_k}$ . Hence  $\sigma_\ell \in H$  but then rule (ii) applied to  $\sigma_{i_{k+1}}$  and  $\sigma_\ell$  is in contradiction with the existence of  $\sigma_{j_k}$ . So  $Z$  is empty and the color of all points of  $C_{k+1}$  is determined and respects Figure 10.

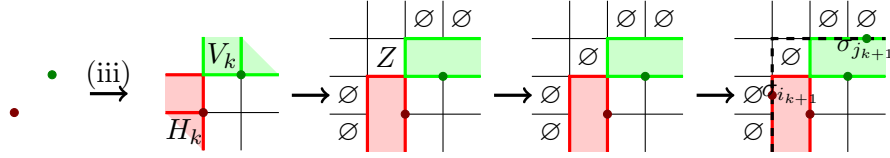
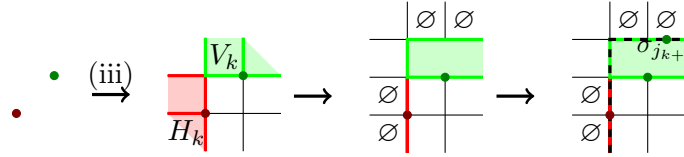


Figure 11: Transitive closure

**Only one area in  $H_k$  and  $V_k$  is empty:** The same proof as the preceding case allow us to define a new point  $\sigma_{j_{k+1}}$  or  $\sigma_{i_{k+1}}$  depending on which area is empty and we can extend the *assigned* border as shown in next figure.

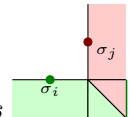


Transitive closure

Hence, the *assigned* zone keeps growing until all permutation points are assigned, proving Theorem 3.13.  $\square$

We also have a similar result extending rule (iv):

**Theorem 3.14.** Consider a valid coloring of a  $\ominus$ -decomposable permutation  $\sigma$ . If there exist two points  $\sigma_i < \sigma_j, i < j$  such that  $\sigma_i \in G$  and  $\sigma_j \in R$ , then the color of each point  $\sigma_k$  with

$k > j$  or  $\sigma_k < \sigma_i$  is determined. Such a zone will be represented as  in the sequel.

*Proof:* Notice that rules are symmetric so that the same proof as for Theorem 3.13 holds.  $\square$

Knowing Theorem 3.13 and Theorem 3.14, to set the color of as much points as possible, we better have to choose the lower right increasing sequence  $RG$  or the upper left increasing sequence  $GR$ . Let us now define properly these particular ascents.

We consider a valid bicoloring  $c$  of a permutation  $\sigma$ . We define  $A_{RG}$  as the set of increasing sequences  $RG$  of  $c$ .

**Lemma 3.15.** Suppose  $A_{RG} \neq \emptyset$ . Among increasing sequences  $RG$  of  $c$ , the pair  $(\sigma_i, \sigma_j)$  which maximizes  $i$  first then minimizes  $\sigma_j$  (for  $i$  fixed) is the same than the pair that minimizes  $\sigma_j$  first then maximizes  $i$  (for  $\sigma_j$  fixed).

*Proof.* Let  $(\sigma_{i_0}, \sigma_{j_0})$  be the pair that maximizes  $i_0$  first then minimizes  $\sigma_{j_0}$  and  $(\sigma_{i_1}, \sigma_{j_1})$  be the pair that minimizes  $\sigma_{j_1}$  first then maximizes  $i_1$ . Then by definition  $i_0 \geq i_1$  and  $\sigma_{j_1} \leq \sigma_{j_0}$ .

If  $j_1 < i_0$  then  $(\sigma_{j_1}, \sigma_{j_0})$  is an increasing sequence  $GG$  and rule (i) is in contradiction with  $\sigma_{i_0} \in H$  as  $j_1 < i_0 < j_0$ . If  $\sigma_{j_1} < \sigma_{i_0}$  then  $(\sigma_{i_1}, \sigma_{i_0})$  is an increasing sequence  $RR$  and rule (ii) is in contradiction with  $\sigma_{j_1} \in V$  as  $\sigma_{i_1} < \sigma_{j_1} < \sigma_{i_0}$ . Hence  $(\sigma_{i_0}, \sigma_{j_1})$  is an increasing sequence  $RG$ . Then by definition of  $j_0$ ,  $\sigma_{j_0} \leq \sigma_{j_1}$  and by definition of  $i_1$ ,  $i_1 \geq i_0$ . So  $(\sigma_{i_0}, \sigma_{j_0}) = (\sigma_{i_1}, \sigma_{j_1})$ .  $\square$



By the preceding lemma, when  $A_{RG} \neq \emptyset$  we can define  $i_{RG}, j_{RG}$  as the lower right increasing sequence  $RG$ . By symmetry, we can also define  $i_{GR}, j_{GR}$  the upper left increasing sequence  $GR$  when  $A_{RG} \neq \emptyset$ , where  $A_{GR}$  is the set similar to  $A_{RG}$  but for increasing sequences  $GR$ .

Now we have all the tools to prove that there are only a polynomial number of bicolourings to test. We just have to do a case study depending on  $A_{RG}$  or  $A_{GR}$  are empty.

### 3.3 Case study

Recall that from Proposition 2.16 if  $\sigma$  is  $\ominus$ -decomposable then  $\sigma$  is 2-stack pushall sortable if and only if each  $\ominus$ -indecomposable block of  $\sigma$  is 2-stack pushall sortable. Thus, we can assume that  $\sigma$  is  $\ominus$ -indecomposable.

In this section, we consider a valid coloring  $c$  of a  $\ominus$ -indecomposable permutation  $\sigma$ . We prove that knowing if there are ascents  $RG$  or  $GR$  in  $c$  and knowing  $i_{RG}, j_{RG}, i_{GR}$  and  $j_{GR}$  (if they exist), we can deduce the color of every point of  $\sigma$ .

We prove this considering 4 cases depending on whether there are ascents  $RG$  or  $GR$  in  $c$ .

#### 3.3.1 There is no bicolored ascents

If  $A_{RG}$  and  $A_{GR}$  are both empty, then the coloring is monochromatic:

**Proposition 3.16.** *Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that every pattern 12 of  $\sigma$  is monochromatic. Then all points of  $\sigma$  have the same color.*

*Proof:* Let  $\sigma_i$  and  $\sigma_j$  be two consecutive left-to-right minima of  $\sigma$ . By definition there are no point below  $\sigma_i$  and to the left of  $\sigma_j$  as shown by the empty sign in the following figure

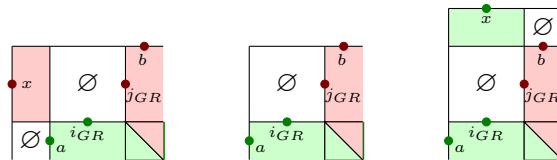


. As  $\sigma$  is  $\ominus$ -indecomposable, there exist a point  $\sigma_k$  above  $\sigma_j$  and to the right of  $\sigma_i$ . As increasing subsequences are monochromatic,  $\sigma_i$  and  $\sigma_k$  have the same color. The same goes for  $\sigma_j$  and  $\sigma_k$ . Thus  $\sigma_i$  and  $\sigma_j$  have the same color. So all left-to-right minima of  $\sigma$  have the same color. By definition of left-to-right minima, for every non-minimal point  $\sigma_l$  there exists a left-to-right minima  $\sigma_m$  such that  $(\sigma_m, \sigma_l)$  is a pattern 12 of  $\sigma$ . Thus  $\sigma_l$  has the same color as  $\sigma_m$ , and all points of  $\sigma$  have the same color.  $\square$

#### 3.3.2 There is no increasing sequence $RG$ but some increasing sequences $GR$

We suppose in this section that there exists at least one increasing sequence  $GR$  but no increasing sequence  $RG$ . As  $A_{GR}$  is non empty,  $i_{GR}$  and  $j_{GR}$  are defined. We prove that once  $i_{GR}$  and  $j_{GR}$  are determined, then it fixes the color of every other point of the permutation.

**Proposition 3.17.** *Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that there is no increasing subsequence  $RG$  in  $c$  and there is at least an increasing sequence  $GR$  in  $c$ . Then  $c$  has one of the following shapes (where maybe  $a = i_{GR}$  or  $b = j_{GR}$ ):*



**Remark 3.18.** Here and in all the following, when a zone of a diagram is colored with  $R$  (resp.  $G$ ), it means that if there are some points lying in this zone, they are in  $R$  (resp.  $G$ ). And when a zone of a diagram has an empty sign, it means that this zone is empty.

*Proof:* The color of every point  $\sigma_k$  such that  $k > j_{GR}$  or  $\sigma_k < \sigma_{i_{GR}}$  is determined by Theorem 3.14 (see the first diagram of Figure 12). Note that we denote by  $*$  the zone where the color of the points is unknown. By maximality of  $\sigma_{i_{GR}}$ , any point above  $\sigma_{i_{GR}}$  and lower left with respect to  $\sigma_{j_{GR}}$  is in  $R$ . By minimality of  $j_{GR}$ , any point to the left of  $\sigma_{j_{GR}}$  and top right with respect to  $\sigma_{i_{GR}}$  is in  $G$ . As no point can be both in  $R$  and in  $G$ , we know that the zone between  $\sigma_{i_{GR}}$  and  $\sigma_{j_{GR}}$  is empty, as shown in the second diagram of Figure 12.

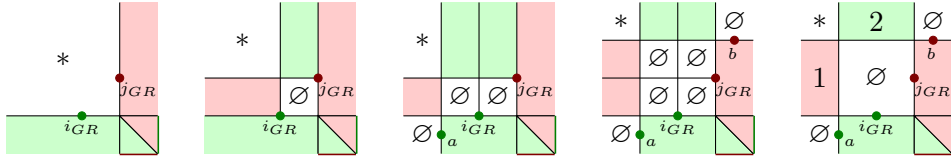


Figure 12: Only bicolored increasing subsequences GR exist

Let  $a$  be the leftmost point among points below  $i_{GR}$  (notice that  $a$  may be equal to  $i_{GR}$ ). Applying rule (i) to  $a$  and  $i_{GR}$ , we obtain the third diagram (note that if  $a = i_{GR}$  the column between  $i_{GR}$  and  $a$  does not exist). Let  $b$  be the topmost point to the right of  $j_{GR}$  ( $b$  may be equal to  $j_{GR}$ ). Applying rule (ii) to  $j_{GR}$  and  $b$ , we obtain the fourth diagram of Figure 12 (if  $b = j_{GR}$  the column between  $j_{GR}$  and  $b$  does not exist).

At last, we number two different areas and discuss about the different cases whether these zones are empty or not. These zones are pictured in the fifth diagram of Figure 12.

**Zone 1 is not empty** Let  $x$  be the leftmost point inside zone 1. Note that  $x$  may be above or below  $j_{GR}$ . First diagram of Figure 13 illustrates the position of point  $x$ . Applying rule (ii) to  $x$  and  $b$  we obtain the second diagram of Figure 13. By hypothesis, there are no increasing sequence RG, thus there are no points in  $G$  in the up-right quadrant of  $x$ . This leads to the third diagram. At last, if the zone  $*$  is not empty, then  $\sigma$  is  $\ominus$ -decomposable by cutting along the row of  $b$  and the column of  $x$ . Thus  $*$  is empty and all points have a determined color, as in the first diagram of Proposition 3.17.

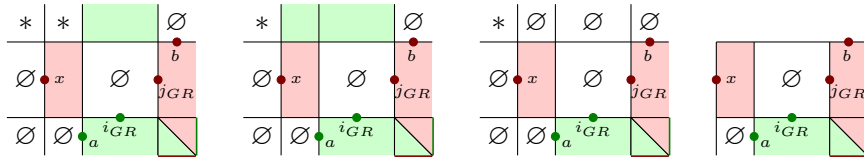


Figure 13: Zone 1 is not empty

**Zone 1 is empty** Suppose that zone 1 is empty. If zone 2 is also empty then as  $\sigma$  is  $\ominus$ -indecomposable, zone  $*$  is also empty and all points have a determined color, as in the second diagram of Proposition 3.17.

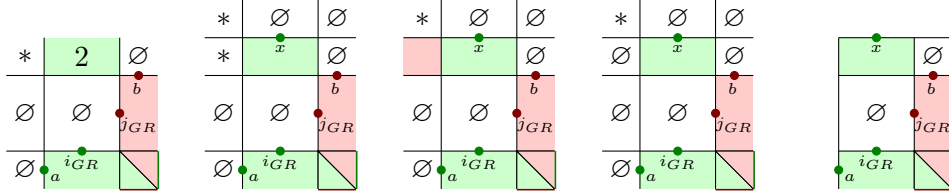
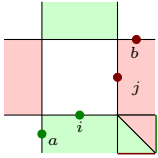


Figure 14: Zone 1 is empty

Otherwise, zone 2 is not empty and let  $x$  be the topmost point inside zone 2 ( $x$  may be to the left or to the right of  $i_{GR}$ ). This is depicted in the second diagram of Figure 14. We apply rule (i) to  $a$  and  $x$  to obtain the third diagram. As there is no increasing subsequence  $RG$ , there is no point of  $R$  in the lower left quadrant of  $x$  as depicted in the fourth diagram. Moreover,  $\sigma$  is  $\ominus$ -indecomposable, thus zone  $*$  is empty and each point has a determined color, as in the last diagram of Proposition 3.17.  $\square$

**Definition 3.19.** Let  $\sigma$  be a permutation and  $i$  and  $j$  two indices of  $\sigma$  such that  $\sigma_i\sigma_j$  is an ascent. Set  $a = \min\{k \mid \sigma_k \leq \sigma_i\}$  and  $b$  such that  $\sigma_b = \max\{\sigma_k \mid k \geq j\}$ . We define  $C_{GR}(\sigma, i, j)$  as the partial bicoloring of  $\sigma$  having the following shape:



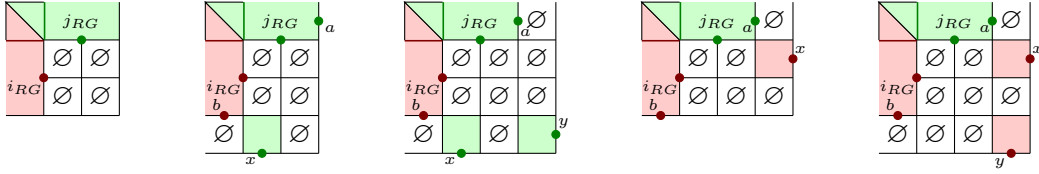
**Proposition 3.20.** Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that there is no increasing subsequence  $RG$  in  $c$  and there is at least an increasing sequence  $GR$  in  $c$ . Then  $c = C_{GR}(\sigma, i_{GR}, j_{GR})$ .

*Proof:* This is a direct consequence of Proposition 3.17 and Definition 3.19.  $\square$

### 3.3.3 All bicolored increasing sequences are labeled $RG$

We suppose in this section that there exists at least one increasing sequence  $RG$  but no increasing sequence  $GR$ . As  $A_{RG}$  is non empty,  $i_{RG}$  and  $j_{RG}$  are defined. We prove that once  $i_{RG}$  and  $j_{RG}$  are determined, then it fixes the color of every other point of the permutation.

**Proposition 3.21.** Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that there is no increasing subsequence  $GR$  in  $c$  and there is at least an increasing sequence  $RG$  in  $c$ . Then  $c$  has one of the following shapes (where maybe  $a = j_{RG}$  or  $b = i_{RG}$ ):



*Proof:* The color of every point  $\sigma_k$  such that  $k < i_{RG}$  or  $\sigma_k > \sigma_{j_{RG}}$  is determined by Theorem 3.13 (see the first diagram of Figure 15). We denote by  $*$  the zone where the color of the points is unknown. By maximality of  $i_{RG}$  and minimality of  $\sigma_{j_{GR}}$  we know the color of some other points, and as no point can be both in  $R$  and in  $G$ , we know that the zone between  $\sigma_{i_{RG}}$  and  $\sigma_{j_{RG}}$  must be empty, as shown in the second diagram of Figure 15.

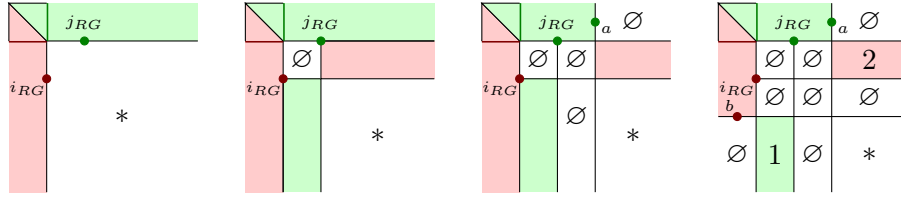


Figure 15: All bicolored increasing sequences are labeled  $RG$

Let  $a$  be the rightmost point among points above  $j_{RG}$  (maybe  $a = j_{RG}$ ). Rule (i) applied to points  $j_{RG}$  and  $a$  gives the third diagram of Figure 15 (note that if  $a = j_{RG}$  the column between  $j_{RG}$  and  $a$  does not exist). Similarly let  $b$  be the lowest point among points to the left of  $i_{RG}$  ( $b$  may be equal to  $i_{RG}$ ). Rule (ii) applied to  $b$  and  $i_{RG}$  leads to the fourth diagram of Figure 15. Note also that we numbered two specific zones in this diagram and we study now the different cases where they are empty or not.

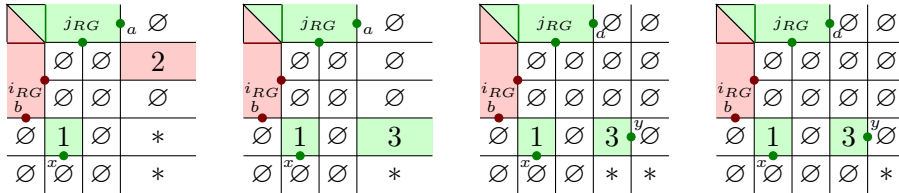


Figure 16: Zone 1 is non-empty

**Zone 1 is non-empty** If zone 1 is non-empty, let  $x$  be the lowest point inside this zone (see Figure 16). As there do not exist an increasing sequence  $GR$ , every point to the top-right of  $x$  is in  $G$  as shown in the second diagram of Figure 16, where we define a zone 3. If zone 3 is empty then zone  $*$  is empty as  $\sigma$  is  $\ominus$ -indecomposable, hence every point has a assigned color as in the first diagram of Proposition 3.21. If zone 3 is non empty, let  $y$  be the rightmost

point inside this zone as shown in the third diagram. Applying rule (i) to  $x$  and  $y$  add another empty zone, leading to the last diagram. As  $\sigma$  is  $\ominus$ -indecomposable, zone  $*$  is empty and all points have an assigned color as in the second diagram of Proposition 3.21.

**Zone 1 is empty** Suppose that zone 1 is empty. If zone 2 is also empty then as  $\sigma$  is  $\ominus$ -indecomposable, zone  $*$  is also empty and all points have a determined color, as in the third diagram of Proposition 3.21.

If zone 2 is non-empty, let  $x$  be the rightmost point of zone 2.

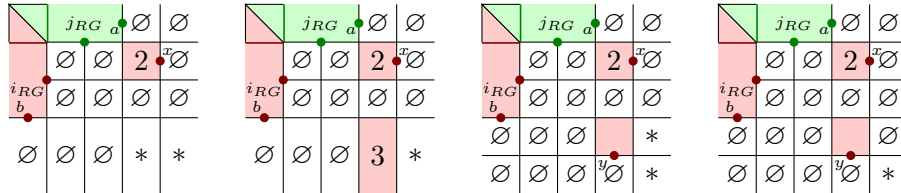
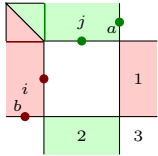


Figure 17: Zone 1 is empty

As there are no increasing subsequence GR, all points in the lower left quadrant of  $x$  lie in  $R$  as shown in the second diagram of Figure 17 where we define a zone 3. If zone 3 is empty then as  $\sigma$  is  $\ominus$ -indecomposable zone  $*$  is also empty and all points have a determined color, as in the fourth diagram of Proposition 3.21. Otherwise zone 3 is non-empty and let  $y$  the lowest point in zone 3 as depicted in the third diagram. We apply rule (ii) to  $x$  and  $y$  leading to the fourth diagram. As  $\sigma$  is  $\ominus$ -indecomposable, zone  $*$  is empty and all points have a determined color, as in the last diagram of Proposition 3.21.  $\square$

**Definition 3.22.** Let  $\sigma$  be a permutation and  $i$  and  $j$  two indices of  $\sigma$  such that  $\sigma_i\sigma_j$  is an ascent. Set  $a = \max\{k \mid \sigma_k \geq \sigma_j\}$  and  $b$  such that  $\sigma_b = \min\{\sigma_k \mid k \leq i\}$ . We define  $C_{RG}(\sigma, i, j)$  as the partial bicoloring of  $\sigma$  having the following shape:



where points of zone 3 are in  $G$  if zone 1 is empty and zone 2 is nonempty, in  $R$  if zone 1 is nonempty and zone 2 is empty, and have no color otherwise.

**Proposition 3.23.** Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that there is no increasing subsequence  $RG$  in  $c$  and there is at least an increasing sequence  $GR$  in  $c$ . Then  $c = C_{RG}(\sigma, i_{RG}, j_{RG})$ .

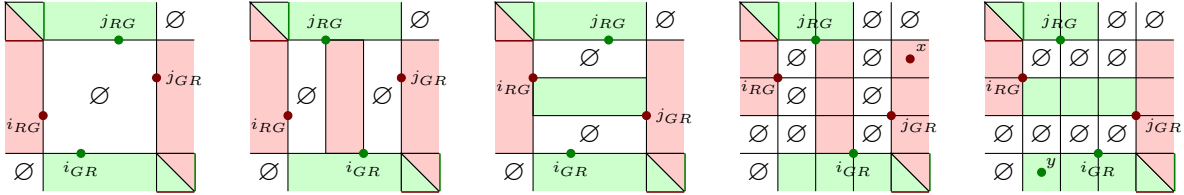
*Proof:* This is a direct consequence of Proposition 3.21 and Definition 3.22.  $\square$

### 3.3.4 There exist both increasing sequences labeled $GR$ and $RG$

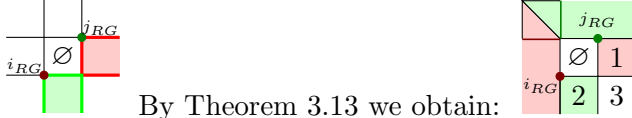
In this section we study the last case that remains to deal, *i.e.* there is at least one increasing sequence colored  $RG$  and at least one colored  $GR$ . As  $A_{GR}$  and  $A_{RG}$  are non empty,  $i_{GR}$ ,

$j_{GR}$ ,  $i_{RG}$  and  $j_{RG}$  are defined. We prove that once  $i_{GR}$ ,  $j_{GR}$ ,  $i_{RG}$  and  $j_{RG}$  are determined, then it fixes the color of every other point of the permutation.

**Proposition 3.24.** *Let  $\sigma$  be a permutation and  $c$  a valid coloring of  $\sigma$  such that there exists at least an increasing sequence colored  $GR$  and at least an increasing sequence colored  $RG$ . Then  $c$  has one of the following shapes:*

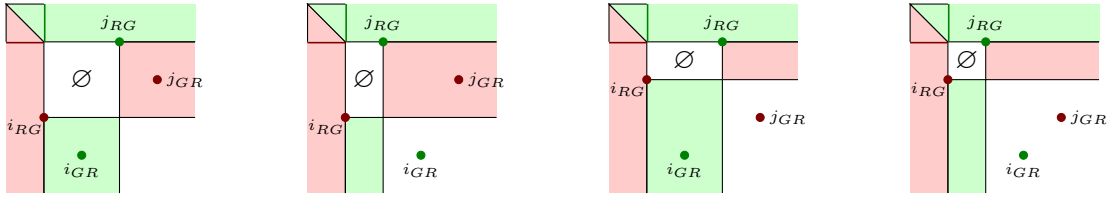


*Proof:* By maximality of  $i_{GR}$  and minimality of  $\sigma_{j_{GR}}$  we have:

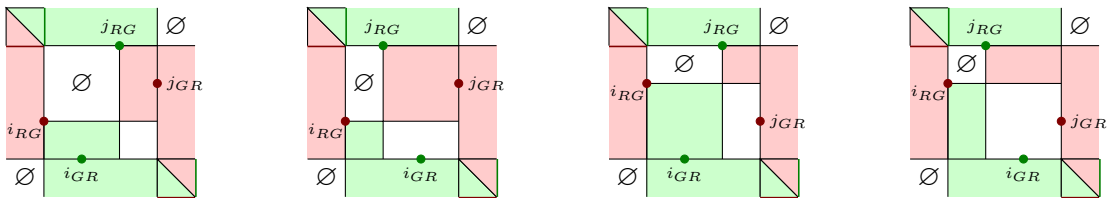


By Theorem 3.13 we obtain:

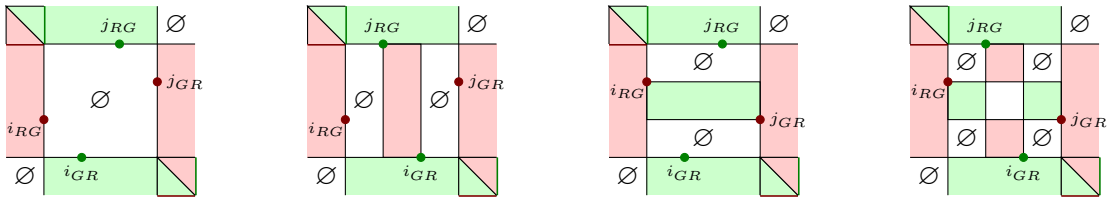
Recall that there exist an increasing sequence  $GR$ .  $i_{GR}$  lies in quadrant 2 or 3 and  $j_{GR}$  in quadrant 1 or 3. Hence the coloring  $c$  as either one of the 4 following shapes:



Applying Theorem 3.14 to  $i_{GR}$  and  $j_{GR}$  we obtain these new diagrams:



Finally, using maximality of  $\sigma_{i_{GR}}$  and minimality of  $j_{GR}$ , we obtain:



In the first 3 diagrams, the color of each point is determined – recall that upper-left and



last diagram where different zones are numbered. We now study different cases according whether zone 1 is empty or not, and we prove that both are excluded.

**Zone 1 is empty** Suppose that zone 1 is empty. As  $\sigma$  is  $\ominus$ -indecomposable then zone 2 must contain at least one point. Denote by  $x$  the rightmost point of this zone. Figure 20 illustrates the proof.

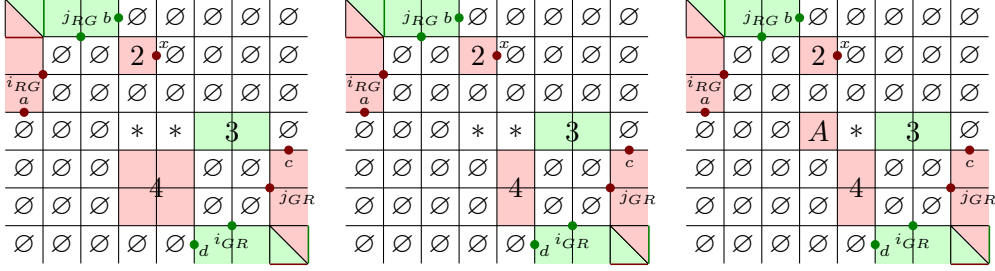


Figure 20: Increasing sequences RG and GR exist and 1 is empty.

Rule (vii) applied to  $x$  and  $c$  leads to the second diagram. Moreover as  $(i_{GR}, j_{GR})$  is the topmost and leftmost increasing sequence GR, all points to the lower left quadrant of  $x$  lie in  $R$ , leading to the third diagram where we define a zone  $A$ .

**Zone 4 is not empty** We prove that this case is not possible. If zone 4 is not empty, let  $y$  be its leftmost point (above or below  $j_{GR}$ ) as illustrated in the first diagram of Figure 21.

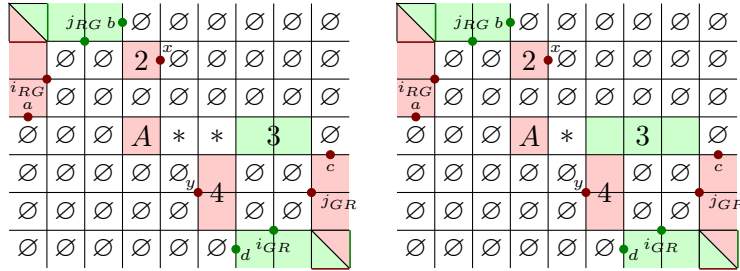


Figure 21: Zone 1 is empty and zone 4 is not empty

We apply rule (ii) to  $y$  and  $c$  and obtain the second diagram. But  $(i_{RG}, j_{RG})$  is lowest-right increasing sequence RG, hence there is no point labeled  $G$  in the above-right quadrant of  $y$ . Hence zone 3 is empty which is forbidden as  $\sigma$  is  $\ominus$ -indecomposable.

**Zone 4 is empty** We prove that this case is also not possible. Suppose that zone 4 is empty as illustrated in the first diagram of Figure 22.



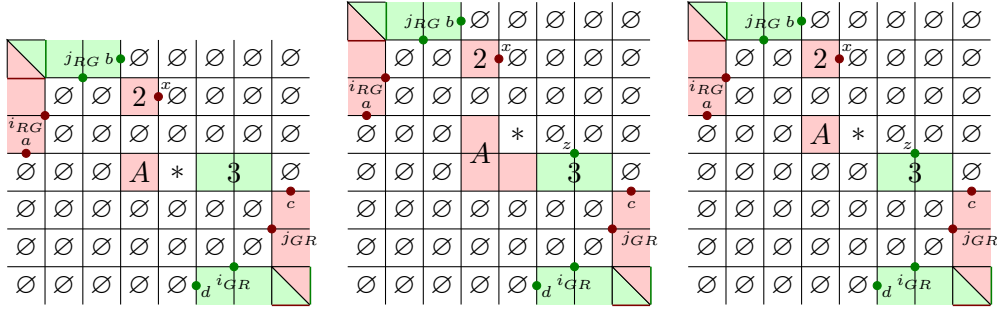


Figure 22: Zone 1 is empty and zone 4 is empty.

As  $\sigma$  is  $\ominus$ -indecomposable, zone 3 is non-empty. Let  $z$  be the topmost point of zone 3 (it may be to the left or to the right of  $i_{GR}$ ). Applying rule (i) to  $z$  and  $d$  we obtain the second diagram. But  $(i_{RG}, j_{RG})$  is the lowest right increasing sequence labeled RG, hence there are no point labeled  $R$  in the below-left quadrant of  $z$  – see diagram 3 –. But then  $\sigma$  is  $\ominus$ -decomposable which is forbidden.

**Zone 1 is not empty** Suppose that zone 1 of Figure 19 is non empty. Define  $x$  as the lowest point of this zone as shown in the first diagram of Figure 23.

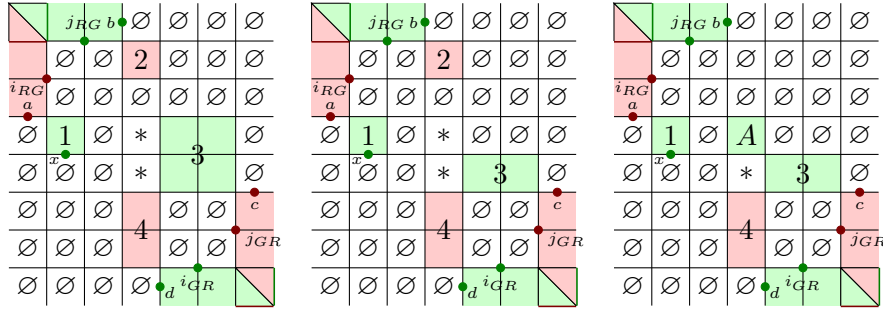


Figure 23: Zone 1 is not empty.

Rule (viii) applied to  $x$  and  $d$  implies the second diagram. Moreover, as  $(i_{GR}, j_{GR})$  is the leftmost-top increasing sequence labeled GR, all points to the top right of  $x$  are in  $G$ , leading to the last diagram.

**Zone 3 is not empty** If zone 3 is not empty, let  $y$  be its topmost point ( $y$  may be to the left or to the right of  $i_{GR}$ ) as pictured in Figure 24.

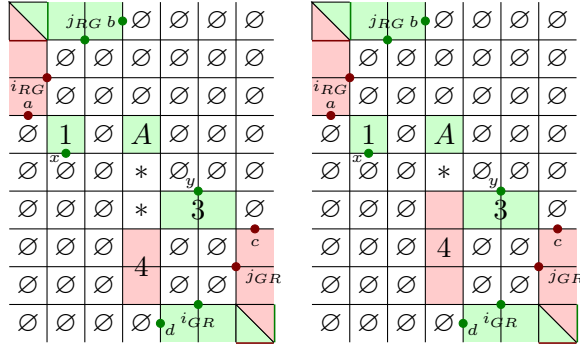


Figure 24: Zone 1 is not empty and zone 3 is not empty.

Rule (i) applied to  $d$  and  $y$  gives the second diagram. But  $(i_{RG}, j_{RG})$  is the bottom-rightmost increasing sequence  $RG$ , hence no point in the lower left quadrant of  $y$  lies in  $R$ . Hence zone 4 is empty and  $\sigma$  is  $\ominus$ -decomposable which is forbidden.

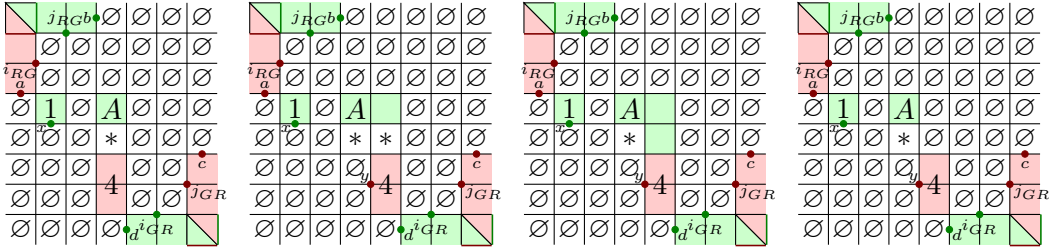
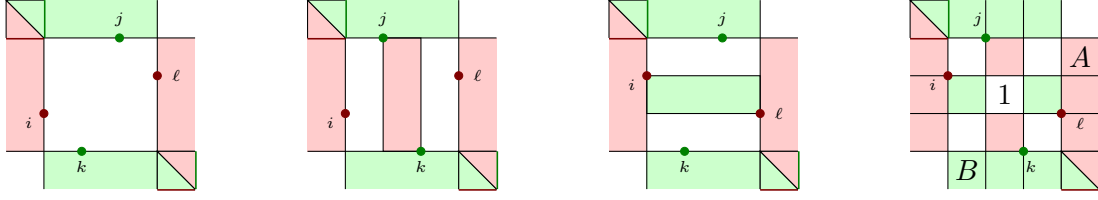


Figure 25: Zone 1 is not empty and zone 3 is empty

**Zone 3 is empty** Figure 25 illustrates the proof. As  $\sigma$  is  $\ominus$ -indecomposable, zone 4 is not empty. Let  $y$  be the leftmost point inside zone 4 – either above or under  $j_{GR}$  – as depicted in the second diagram. Rule (ii) applied to  $y$  and  $c$  leads to the third diagram. But  $(i_{RG}, j_{RG})$  is the bottom-rightmost increasing sequence  $RG$ , hence no point of  $G$  lies in the top-right quadrant of  $y$  leading to the fourth diagram. So  $\sigma$  is  $\ominus$ -decomposable which is forbidden.

This ends the cases study, proving that zone  $A$  and  $B$  cannot be both empty.  $\square$

**Definition 3.25.** Let  $\sigma$  be a permutation and  $i, j, k, \ell$  four indices of  $\sigma$  such that  $\sigma_i \sigma_j$  and  $\sigma_k \sigma_\ell$  are ascents. We define the partial bicoloring  $C_*(\sigma, i, j, k, \ell)$  of  $\sigma$  as follows.



If  $\sigma_i, \sigma_\ell, \sigma_k$  and  $\sigma_j$  have a relative position corresponding to one of the above diagrams, then we define  $C_*(\sigma, i, j, k, \ell)$  as the partial bicolored of  $\sigma$  having the corresponding shape, where in the first diagram points of zone 1 are in  $R$  if zone  $A$  is nonempty, in  $G$  if zones  $B$  is nonempty, and have no color otherwise.

Otherwise  $C_*(\sigma, i, j, k, \ell)$  is the partial coloring with no point colored.

**Proposition 3.26.** Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  such that there exist increasing sequences  $RG$  and increasing sequences  $GR$  in  $c$ . Then  $c = C_*(\sigma, i_{RG}, j_{RG}, i_{GR}, j_{GR})$ .

*Proof:* This is a consequence of Proposition 3.24 and Definition 3.25, noticing that if there exists a point  $x$  in zone  $A$ , then applying rule (ii) to  $\ell$  and  $x$ , all points in zone 1 belong to  $R$ , and if there exists a point  $y$  in zone  $B$ , then applying rule (i) to  $y$  and  $k$ , all points in zone 1 belong to  $G$ .  $\square$

### 3.4 A first polynomial algorithm

**Data:**  $\sigma$  a  $\ominus$ -indecomposable permutation (whose size is denoted  $n$ ).

**Result:** The set  $E$  of valid colorings of  $\sigma$

**for**  $c$  bicolored of  $\sigma$  being one of

- $c$  is unicolor  $R$
- $c$  is unicolor  $G$
- $c = C_{GR}(\sigma, i, j)$  or  $C_{RG}(\sigma, i, j)$  for  $i \in [1..n]$  and  $j \in [i..n]$  s.t.  $\sigma_j > \sigma_i$
- $c = C_*(\sigma, i, j, k, \ell)$  for  $i \in [1..n]$  and  $j \in [i..n]$  s.t.  $\sigma_j > \sigma_i$  and for  $k \in [1..n]$  and  $\ell \in [k..n]$  s.t.  $\sigma_\ell > \sigma_k$

**do**

- | If all points of  $\sigma$  are colored and  $c$  is valid then add  $c$  to  $E$ ;

**end**

**Algorithm 3:** ColoringIndecomposable1( $\sigma$ )

**Proposition 3.27.** Algorithm 3 compute in time  $\mathcal{O}(n^5)$  the set of valid colorings of any  $\ominus$ -indecomposable permutation  $\sigma$ .

*Proof:* Let  $\sigma$  be a  $\ominus$ -indecomposable permutation of size  $n$  and  $c$  a valid coloring of  $\sigma$ . Then from Propositions 3.16, 3.20, 3.23 and 3.26,  $c$  is either monochromatic, or  $C_{GR}(\sigma, i, j)$  or  $C_{RG}(\sigma, i, j)$  for some  $i \in [1..n]$  and some  $j \in [i..n]$  such that  $\sigma_j > \sigma_i$ , or  $c = C_*(\sigma, i, j, k, \ell)$  for some  $i \in [1..n]$ , some  $j \in [i..n]$  such that  $\sigma_j > \sigma_i$ , some  $k \in [1..n]$  and some  $\ell \in [k..n]$  such that  $\sigma_\ell > \sigma_k$ . Thus  $c$  is computed by Algorithm 3 and added to  $E$  as it is valid. Conversely, each coloring added to  $E$  is a valid bicolored of  $\sigma$ .

Now consider the complexity of Algorithm 3. There are  $\mathcal{O}(n^4)$  colorings computed. Indeed there are two monochromatic colorings,  $\mathcal{O}(n^2)$  colorings  $C_{GR}(\sigma, i, j)$  or  $C_{RG}(\sigma, i, j)$  and  $\mathcal{O}(n^4)$

colorings  $C_*(\sigma, i, j, k, \ell)$ . Moreover the coloring is computed in linear time and checking if the coloring is valid is done in linear time using Proposition 3.10. Hence Algorithm 3 runs in time  $\mathcal{O}(n^5)$ .  $\square$

## 4 An optimal algorithm

### 4.1 Rooting colorings

In this section we show how each diagram of Propositions 3.17, 3.21 and 3.24 can be rooted in a given point such that each point  $i_{GR}, i_{RG}, j_{GR}$  and  $j_{RG}$  can be deduced from this one. Moreover, given a diagram we show how we can assign colors to points of the permutations lying in a colored zone of the diagram in linear time.

**Definition 4.1.** *Let  $\sigma$  be a permutation and  $s \in [1..|\sigma|]$ . We set*

$$C_1(\sigma, s) = C_{GR}(\sigma, s, t) \text{ where } t = \min\{k \mid k > s \text{ and } \sigma_k > \sigma_s\}$$

$$C_2(\sigma, s) = C_{GR}(\sigma, t, s) \text{ where } t \text{ is such that } \sigma_t = \max\{\sigma_k \mid k < s \text{ and } \sigma_k < \sigma_s\}$$

$$C_3(\sigma, s) = C_{RG}(\sigma, s, t) \text{ where } t \text{ is such that } \sigma_t = \min\{\sigma_k \mid k > s \text{ and } \sigma_k > \sigma_s\}$$

$$C_4(\sigma, s) = C_{RG}(\sigma, t, s) \text{ where } t = \max\{k \mid k < s \text{ and } \sigma_k < \sigma_s\}$$

$$C_5(\sigma, s) = C_*(\sigma, p, q, t, s) \text{ with}$$

$$t = \max\{k \mid k < u \text{ and } \sigma_k < \sigma_s\}$$

$$\text{with } u = \max\{k \mid k < s \text{ and } \sigma_k > \sigma_s\},$$

$$p = \max\{k \mid k < t \text{ and } \sigma_t < \sigma_k < \sigma_s\} \text{ and}$$

$$q \text{ such that } \sigma_q = \min\{\sigma_k \mid t < k \leq u\}$$

$$C_6(\sigma, s) = C_*(\sigma, p, q, s, t) \text{ with}$$

$$t = \min\{k \mid k > s \text{ and } \sigma_k > \sigma_s\},$$

$$u = \max\{k \mid k < t \text{ and } \sigma_k > \sigma_t\},$$

$$p = \max\{k \mid k < u \text{ and } \sigma_s < \sigma_k < \sigma_t\} \text{ and}$$

$$q \text{ such that } \sigma_q = \min\{\sigma_k \mid \sigma_k > \sigma_t \text{ and } p < k \leq u\}$$

$$C_7(\sigma, s) = C_*(\sigma, q, p, t, s) \text{ with}$$

$$t \text{ such that } \sigma_t = \max\{\sigma_k \mid k < s \text{ and } \sigma_k < \sigma_s\},$$

$$u \text{ such that } \sigma_u = \min\{\sigma_k \mid k < t \text{ and } \sigma_k > \sigma_t\},$$

$$p \text{ such that } \sigma_p = \min\{\sigma_k \mid \sigma_k > \sigma_u \text{ and } t < k < s\} \text{ and}$$

$$q = \max\{k \mid k < p \text{ and } \sigma_u \leq \sigma_k < \sigma_p\}$$

$$C_8(\sigma, s) = C_*(\sigma, p, q, s, t) \text{ with}$$

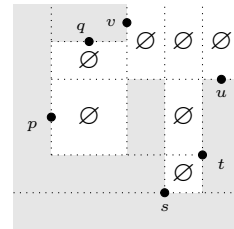
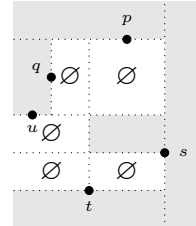
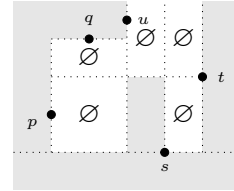
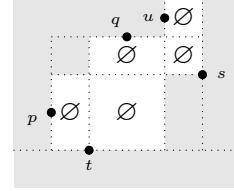
$$t = \min\{k \mid k > s \text{ and } \sigma_k > \sigma_s\},$$

$$u \text{ such that } \sigma_u = \max\{\sigma_k \mid \sigma_k > \sigma_t \text{ and } k > t\},$$

$$v = \max\{k \mid k < u \text{ and } \sigma_k > \sigma_u\},$$

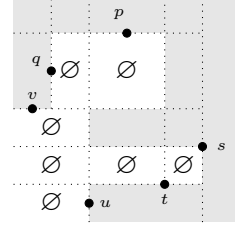
$$p = \max\{k \mid k < v \text{ and } \sigma_t < \sigma_k < \sigma_u\} \text{ and}$$

$$q \text{ such that } \sigma_q = \min\{\sigma_k \mid \sigma_k > \sigma_u \text{ and } p < k \leq v\}$$



$C_9(\sigma, s) = C_*(\sigma, q, p, t, s)$  with

- $t$  such that  $\sigma_t = \max\{\sigma_k \mid k < s \text{ and } \sigma_k < \sigma_s\}$ ,
- $u = \min\{k \mid k < t \text{ and } \sigma_k < \sigma_t\}$ ,
- $v$  such that  $\sigma_v = \min\{\sigma_k \mid k < u \text{ and } \sigma_k > \sigma_u\}$ ,
- $p$  such that  $\sigma_p = \min\{\sigma_k \mid \sigma_k > \sigma_v \text{ and } u < k < t\}$  and
- $q = \max\{k \mid k < p \text{ and } \sigma_v \leq \sigma_k < \sigma_p\}$



**Proposition 4.2.** *Let  $\sigma$  be a  $\ominus$ -indecomposable permutation and  $c$  a valid coloring of  $\sigma$  which is not monochromatic. Then there exists  $s \in [1..|\sigma|]$  and  $m \in [1..9]$  such that  $c = C_m(\sigma, s)$ .*

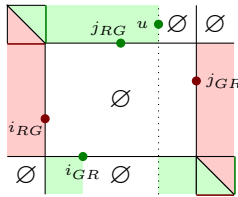
*Proof:* As  $c$  is not monochromatic, then from Proposition 3.16  $\sigma$  has at least a pattern 12 which is not monochromatic.

If there is no increasing subsequence RG in  $c$  then there is at least an increasing sequence GR in  $c$ . Thus from Proposition 3.20,  $c = C_{GR}(\sigma, i_{GR}, j_{GR})$ . Moreover,  $c$  has one of the  $u$  shapes described in Proposition 3.17. If the shape of  $c$  is one of the two first shapes, then  $j_{GR}$  is the leftmost point in the upper-right quadrant of  $i_{GR}$  and  $c = C_1(\sigma, i_{GR})$ . Otherwise the shape of  $c$  is the third one and  $i_{GR}$  is the topmost point in the bottom-left quadrant of  $j_{GR}$  thus  $c = C_2(\sigma, j_{GR})$ .

If there is an increasing subsequence RG in  $c$  but no increasing sequence GR, then from Proposition 3.23  $c = C_{RG}(\sigma, i_{RG}, j_{RG})$ . Moreover  $c$  has one of the 5 shapes described in Proposition 3.21. If the shape of  $c$  is one of the three first shapes, then  $j_{GR}$  is the lowest point in the upper-right quadrant of  $i_{GR}$  and  $c = C_3(\sigma, i_{GR})$ . Otherwise the shape of  $c$  is one of the two last shapes and  $i_{GR}$  is the rightmost point in the bottom-left quadrant of  $j_{GR}$  thus  $c = C_4(\sigma, j_{GR})$ .

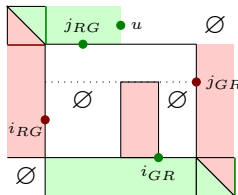
If there is an increasing subsequence RG and an increasing sequence GR in  $c$ , then from Proposition 3.26  $c = C_*(\sigma, i_{RG}, j_{RG}, i_{GR}, j_{GR})$ . Moreover  $c$  has one of the 5 shapes described in Proposition 3.24.

If the shape of  $c$  is the first one, let  $u$  be the rightmost point in the top left quadrant of  $j_{GR}$  (maybe  $u = j_{RG}$ ). Then applying rule (ii) to  $i_{GR}$  and  $u$ ,  $c$  has the following shape:



Thus  $i_{GR}$  is the rightmost point on the left of  $u$  below  $j_{GR}$ . Moreover  $i_{RG}$  is the rightmost point on the topleft quadrant of  $i_{GR}$  below  $j_{GR}$ . Finally  $j_{RG}$  is the lowest point on the right of  $i_{GR}$  and on the left of  $u$ . Hence  $c = C_5(\sigma, j_{GR})$ .

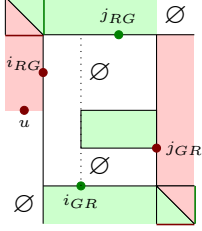
If the shape of  $c$  is the second one of Proposition 3.24, let  $u$  be the rightmost point in the top right quadrant of  $j_{RG}$  (maybe  $u = j_{RG}$ ). From rule (viii) applied to  $j_{RG}$  and  $i_{GR}$ ,  $u < i_{GR}$ . Then applying rule (ii) to  $i_{RG}$  and  $j_{GR}$  and applying rule (i) to  $j_{RG}$  and  $u$  if  $u \neq j_{RG}$ ,  $c$  has the following shape:



Thus  $j_{GR}$  is the leftmost point in the upper right quadrant of  $i_{GR}$  and  $u$  is the rightmost point in the upper left quadrant of  $j_{GR}$ . Moreover  $i_{RG}$  is the rightmost point to the left of  $u$ , below  $j_{GR}$  and above  $i_{GR}$ . Finally,  $j_{RG}$  is the lowest point in the upper left quadrant of  $j_{GR}$  and to the right of  $i_{RG}$ . Hence  $c = C_6(\sigma, i_{GR})$ .

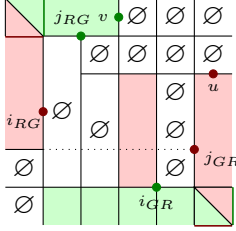
If the shape of  $c$  is the third one of Proposition 3.24, let  $u$  be the lowest point in the lower left quadrant of  $i_{RG}$  (maybe  $u = i_{RG}$ ). From rule (vii) applied to  $i_{RG}$  and  $j_{GR}$ ,  $\sigma_u > \sigma_{j_{GR}}$ .

Then applying rule (i) to  $i_{GR}$  and  $j_{RG}$  and applying rule (ii) to  $u$  and  $i_{RG}$  if  $u \neq i_{RG}$ ,  $c$  has the following shape:



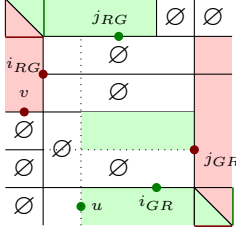
Thus  $i_{GR}$  is the topmost point in the lower left quadrant of  $j_{GR}$  and  $u$  is the lowest point in the upper left quadrant of  $i_{GR}$ . Moreover  $j_{RG}$  is the lowest point above  $u$ , to the right of  $i_{GR}$  and to the left of  $j_{GR}$ . Finally,  $i_{RG}$  is the rightmost point to the lower left of  $j_{RG}$  and above  $u$ . Hence  $c = C_7(\sigma, j_{GR})$

If the shape of  $c$  is the fourth one of Proposition 3.24, let  $u$  be the topmost point to the upright quadrant of  $j_{GR}$  and  $v$  be the rightmost point to the top-right quadrant of  $j_{RG}$  (maybe  $v = j_{RG}$ ). Note that  $u$  is above  $i_{RG}$  as  $u$  is above  $x$  ( $u$  is the topmost point) which is above  $i_{RG}$ . Then applying rule (ii) to  $i_{RG}$  and  $u$  and applying rule (iii) to  $j_{RG}$  and  $v$  if  $v \neq j_{RG}$ ,  $c$  has the following shape:



Thus  $j_{GR}$  is the leftmost point in the up right quadrant of  $i_{GR}$ . Point  $u$  is the topmost point in the upper right quadrant of  $j_{GR}$ . Point  $v$  is the rightmost point in the upper left quadrant of  $u$ . Then  $i_{RG}$  is the rightmost point to the left of  $v$ , below  $u$  and above  $i_{GR}$ . At last,  $j_{RG}$  is the lowest point above  $u$ , to the right of  $i_{RG}$  and to the left of  $v$ . Hence  $c = C_8(\sigma, i_{GR})$

If the shape of  $c$  is the last one of Proposition 3.24, let  $u$  be the leftmost point in the lower left quadrant of  $i_{GR}$  and  $v$  be the lowest point in the lower left quadrant of  $i_{RG}$  (maybe  $v = i_{RG}$ ). Note that  $u$  is to the left of  $j_{RG}$  as it is to the left of  $y$  ( $u$  is the leftmost point) and  $y$  is to the left of  $j_{RG}$ . Then applying rule (i) to  $u$  and  $j_{RG}$  and applying rule (ii) to  $u$  and  $i_{RG}$  if  $v \neq i_{RG}$ ,  $c$  has the following shape:



Thus  $i_{GR}$  is the topmost point in the lower left quadrant of  $j_{GR}$  and  $u$  is the leftmost point in the lower left quadrant of  $i_{GR}$ . Moreover  $v$  is the lowest point in the upper left quadrant of  $u$  and  $j_{RG}$  is the lowest point above  $v$  and to the right of  $u$  and to the left of  $i_{GR}$ . Finally,  $i_{RG}$  is the rightmost point in the lower left quadrant of  $j_{RG}$  and above  $v$ . Hence  $c = C_9(\sigma, j_{GR})$

□

**Proposition 4.3.** *Let  $\sigma$  be a permutation,  $s \in [1..|\sigma|]$  and  $m \in [1..9]$ . Then we can compute  $C_m(\sigma, s)$ , test whether all points of  $\sigma$  are colored and check whether  $C_m(\sigma, s)$  is valid in linear time w.r.t.  $|\sigma|$ .*

*Proof:* Theorems 3.13 and 3.14 and 3.10

□

## 4.2 Algorithm and linear number of sortings for $\ominus$ -indecomposable permutations

**Data:**  $\sigma$  a  $\ominus$ -indecomposable permutation  
**Result:** The set  $E$  of valid colorings of  $\sigma$   
**for**  $c$  *bicoloring of  $\sigma$  unicolor  $R$  or unicolor  $G$*  **do**  
    | If  $c$  is valid then add  $c$  to  $E$ ;  
**end**  
**for**  $s$  *from 1 to  $|\sigma|$*  **do**  
    | **for**  $m$  *from 1 to 9* **do**  
        |  $c = C_m(\sigma, s)$ ;  
        | If all points of  $\sigma$  are colored and  $c$  is valid then add  $c$  to  $E$ ;  
    | **end**  
**end**

**Algorithm 4:** ColoringIndecOptimal( $\sigma$ )

Given any point  $s$  in the permutation the Algorithm decides if the permutation can be colored in each possible case depicted in Propositions 3.17,3.21 and 3.24. Note that diagrams of Propositions 3.17,3.21 and 3.24 depend on  $v$  points  $i_{GR}, i_{RG}, j_{GR}, j_{RG}$ . Indeed, we prove in section 4.1 that any diagram can be rooted in one point – say  $i_{RG}$  for example – and from this points, we can find in linear time any other points –  $i_{GR}, j_{GR}, j_{RG}$  for instance –. Then, we color the permutations with respect to the different zones defined in the diagram. In this process, some points may be uncolored, meaning that they lie in empty zone of the diagram hence have to be rejected. At last, we have a coloring according to diagram and we have to check that this coloring is valid.

**Theorem 4.4.** *A  $\ominus$ -indecomposable permutation of size  $n$  has at most  $9n + 2$  valid colorings. Those colorings can be computed using Algorithm 4 in time  $\mathcal{O}(n^2)$  which is optimal.*

*Proof:* This is a direct consequence of Propositions 4.2 and 4.3, except for the optimality. Proposition 4.5 below implies that the size of the set of valid colorings of the identity of size  $n$  is  $2n^2$ , proving the optimality.  $\square$

**Proposition 4.5.** *For all  $n$  the identity of size  $n$  has exactly  $2n$  valid colorings.*

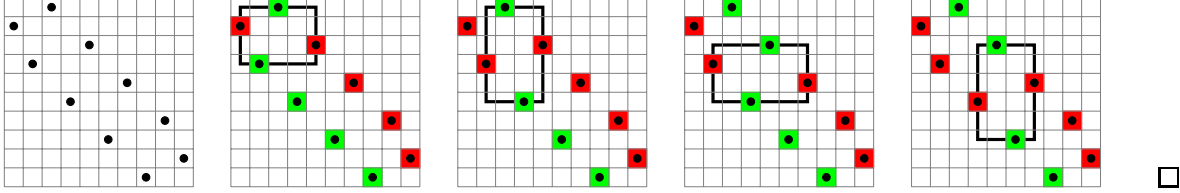
*Proof:* Let  $\sigma$  be the identity of size  $n$ . For all  $k$  between 1 and  $n$  let  $C_{RG}^k$  (resp.  $C_{GR}^k$ ) be the coloring of  $\sigma$  such that for all  $i$ ,  $\sigma$  is in  $R$  (resp.  $G$ ) if  $i \leq k$  and in  $G$  (resp.  $R$ ) otherwise. Then it is straightforward to check using Proposition 3.11 that  $C_{RG}^k$  (resp.  $C_{GR}^k$ ) is a valid coloring of  $\sigma$ . Conversely if  $c$  is a valid coloring of the identity, rules (iii) and (iv) of  $\mathcal{R}_8$  imply that there are at most one pair of consecutive points whose colors are different. So  $c$  is some  $C_{RG}^k$  or some  $C_{GR}^k$ .  $\square$

The property of having a linear number of sortings is not a special case of the identity. Indeed there are some simple permutations that also have a linear number of sortings, as shown in the next proposition.

**Proposition 4.6.** *Permutations  $\sigma^{(n)} = (2n - 1)(2n - 3)(2n)(2n - 5)(2n - 2)(2n - 7)(2n - 4) \dots 5836142$  of size  $2n$  have at least  $2n - 3$  valid colorings.*

*Proof.* To prove the result, we exhibit  $2n - 3$  colorings. We look at set of four points of  $\sigma$  whose indices (resp. values) are consecutive and which form a pattern 2413 (resp. 3142).

Notice that they can be taken to be  $\{i_{RG}, i_{GR}, j_{RG}, j_{GR}\}$  in a valid coloring of  $\sigma$  respecting to the last (resp. third) diagram of Proposition 3.24, as shown in the figure below. This way we obtain  $2n - 3$  valid colorings of  $\sigma$ .



### 4.3 Final algorithm

Recall first that if a permutation is  $\ominus$ -decomposable, then it is 2-stack pushall sortable if and only if each of the block of its decomposition is 2-stack pushall sortable and that we can just push elements of the first block according to any sorting procedure of it, then elements of the second and so on, before popping out all the elements. This means that the different colorings for a  $\ominus$ -decomposable permutation is the product of all colorings for each block.

**Proposition 4.7.** *Let  $\sigma$  be a permutation and  $Col(\sigma)$  the set of valid colorings of  $\sigma$ . If  $\sigma = \ominus[\pi_1, \dots, \pi_k]$  then the map  $c \rightarrow (c|_{\pi_1}, \dots, c|_{\pi_k})$  is a bijection from  $Col(\sigma)$  into  $Col(\pi_1) \times \dots \times Col(\pi_k)$ .*

*Proof:* Let  $c$  be a valid coloring of  $\sigma$ , then  $c$  avoids patterns **132**, **213**, **2/13** and **1X2**. Thus for all  $i$ ,  $c|_{\pi_i}$  avoids patterns **132**, **213**, **2/13** and **1X2** hence is a valid coloring of  $\pi_i$ . Conversely let  $c_i \in Col(\pi_i)$  for all  $i$ . Then coloring points of  $\sigma$  according to  $(c_1, \dots, c_k)$  (*i.e.* according to  $c_1$  for the  $|\pi_1|$  first points of  $\sigma$ , according to  $c_2$  for the  $|\pi_2|$  following points and so on) leads to a coloring  $c$  of  $\sigma$  which is valid. Indeed assume that  $c$  is not valid. Then  $c$  has a pattern **132**, **213**, **2/13** or **1X2**. Let  $p$  be such a pattern. Then  $p$  is not inside a block  $\pi_i$  as  $c_i$  is a valid coloring for all  $i$ . If all points of  $p$  are in different blocks  $\pi_i$  then  $p$  is 321 which is excluded. Thus there are one point of  $p$  in a block  $\pi_i$  and two points of  $p$  in a block  $\pi_j$ . If  $i < j$  then  $p$  begins with its greatest point, which is excluded as  $p$  is **132**, **213**, **2/13** or **1X2**. If  $i > j$  then  $p$  ends with its smallest point, which is excluded as  $p$  is **132**, **213**, **2/13** or **1X2**. As a consequence such a pattern  $p$  does not exist and  $c \in Col(\sigma)$ , concluding the proof.  $\square$

**Data:**  $\sigma$  a permutation  
**Result:** A linear description of the set  $Col(\sigma)$  of valid colorings of  $\sigma$   
 Compute the  $\ominus$ -decomposition of  $\sigma$ :  $\sigma = \ominus[\pi_1, \dots, \pi_k]$  with  $\pi_i$   $\ominus$ -indecomposable;  
**for**  $i$  from 1 to  $k$  **do**  
 | Compute  $Col(\pi_i)$  thanks to Algorithm 4;  
**end**  
 Return  $(Col(\pi_1), \dots, Col(\pi_k))$ ;

**Algorithm 5:** Colorings( $\sigma$ )

**Proposition 4.8.** *Let  $\sigma$  be a permutation of size  $n$ . Then Algorithm 5 gives a linear description of  $Col(\sigma)$  in time  $\mathcal{O}(n^2)$ .*

*Proof.* The algorithm computes the  $\ominus$ -decomposition of  $\sigma$ :  $\sigma = \ominus[\pi_1, \dots, \pi_k]$  with  $\pi_i$   $\ominus$ -indecomposable. This is done in linear time. If  $k = 1$  then  $\sigma$  is  $\ominus$ -indecomposable and  $Col(\sigma) = Col(\pi_1)$ . We conclude thanks to Theorem 4.4. If  $k > 1$  then from Proposition 4.7,



$Col(\sigma) \approx Col(\pi_1) \times \cdots \times Col(\pi_k)$ . For all  $i$ ,  $Col(\pi_i)$  has a size is smaller than  $9|\pi_i|$  and is computed in  $\mathcal{O}(|\pi|^2)$ . We concludes the proof noticing that  $9|\pi_1| + \cdots + 9|\pi_k| = 9|\sigma|$  and  $|\pi_1|^2 + \cdots + |\pi_k|^2 \leq |\sigma|^2$ .  $\square$

**Theorem 4.9.** *Using Algorithm 5, we can decide in time  $\mathcal{O}(n^2)$  whether a permutation  $\sigma$  of size  $n$  is 2-stack pushall sortable.*

*Proof.* By Theorem 3.9, a permutation  $\sigma$  is 2-stack pushall sortable if and only if it admits a valid coloring. Thus all we need is to test whether each set  $Col(\pi_i)$  returned by Algorithm 5 is non-empty with  $\sigma = \ominus[\pi_1, \dots, \pi_k]$  being the  $\ominus$ -decomposition of  $\sigma$ , and we conclude using Proposition 4.8.  $\square$

## 5 Conclusion

This article defines a new restriction of 2-stacks sorting, namely 2-stacks pushall sorting. We characterize every possible pushall sorting of a permutation by means of a bi-coloring of the permutation. Then we give an  $\mathcal{O}(n^2)$  algorithm which computes a linear representation of all pushall sortings of a given permutation, which thus decides if a permutation is 2-stack pushall sortable. We prove that this complexity is optimal.

More studies remain to be done on 2-stacks pushall sorting. First, a simpler mathematical characterization of 2-stack pushall sortable permutations would be interesting. Then, we could study more in depth the number of pushall sortings of a given permutation. More generally it would be nice to compute the generating function of 2-stack pushall sortable permutations, or at least asymptotic bounds on this function. But most importantly, this result is a step to the solve the general 2-stack sorting, which we do in a forthcoming article.

## References

- [1] Michael Albert, Mike Atkinson, and Steve Linton. Permutations generated by stacks and dequeues. *Annals of Combinatorics*, 14:3–16, 2010.
- [2] Michael H. Albert and Mike D. Atkinson. Simple permutations and pattern restricted permutations. *Discrete Mathematics*, 300(1-3):1–15, 2005.
- [3] Michael H. Albert and Mireille Bousquet-Melou. Permutations sortable by two stacks in parallel. In preparation.
- [4] M. D. Atkinson, M. M. Murphy, and N. Ruskuc. Sorting with two ordered stacks in series. *Theor. Comput. Sci.*, 289:205–223, October 2002.
- [5] Miklós Bóna. A survey of stack-sorting disciplines. *Electr. J. Comb.*, on(2), 2002.
- [6] Mireille Bousquet-Mélou. Sorted and/or sortable permutations. *Discrete Mathematics*, 225(1-3):25–50, 2000.
- [7] S. Even and A. Itai. Queues, stacks, and graphs. In *Theory of Machines and Computations*, pages 71–86. Academic Press, 1971.
- [8] Donald E. Knuth. *The Art of Computer Programming, Volume I: Fundamental Algorithms*. Addison-Wesley, 1968.

- [9] Donald E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching*. Addison-Wesley, 1973.
- [10] Maximillian M. Murphy. *Restricted permutations, anti chains, atomic classes and stack sorting*. Phd thesis, University of St Andrews, 2002.
- [11] Vaughan R. Pratt. Computing permutations with double-ended queues, parallel stacks and parallel queues. In Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard M. Karp, and H. Raymond Strong, editors, *STOC*, pages 268–277. ACM, 1973.
- [12] Robert Endre Tarjan. Sorting using networks of queues and stacks. *J. ACM*, 19(2):341–346, 1972.
- [13] Walter Unger. The complexity of colouring circle graphs (extended abstract). In Alain Finkel and Matthias Jantzen, editors, *STACS*, volume 577 of *Lecture Notes in Computer Science*, pages 389–400. Springer, 1992.
- [14] Julian West. *Permutations with forbidden subsequences and Stack sortable permutations*. Phd thesis, Massachusetts Institute of Technology, 1990.
- [15] Julian West. Sorting twice through a stack. *Theor. Comput. Sci.*, 117(1&2):303–313, 1993.
- [16] Doron Zeilberger. A proof of julian west’s conjecture that the number of two-stack-sortable permutations of length  $n$  is  $2(3n)!/((n + 1)!(2n + 1)!)$ . *Discrete Mathematics*, 102(1):85–93, 1992.
- [17] Henning Úlfarsson. Describing west-3-stack-sortable permutations with permutation patterns, 2011.