Pattern matching in (213, 231)-avoiding permutations

Neou Both Emerite, Romeo Rizzi, Stéphane Vialette

Pattern matching in (213, 231)-avoiding permutations

Both Emerite Neou*[1] Romeo Rizzi[2] and Stéphane Vialette[1]

[1] Université Paris-Est, LIGM (UMR 8049), CNRS, UPEM, ESIEE Paris, ENPC, F-77454, Marne-la-Valle, France
{neou,vialette}@univ-mlv.fr
[2] Department of Computer Science, Universit degli Studi di Verona, Italy
romeo.rizzi@univr.it

Abstract. Given permutations $\sigma \in S_k$ and $\pi \in S_n$ with $k < n$, the *pattern matching* problem is to decide whether σ occurs π as an order-isomorphic subsequence. We give a linear-time algorithm in case both π and σ avoid the two size-3 permutations 213 and 231. For the special case where only σ avoids 213 and 231, we present a $O(max(kn^2, n^2 \log(\log(n)))$ time algorithm. We extend our research to bivincular patterns that avoid 213 and 231 and present a $O(kn^4)$ time algorithm. Finally we look at the related problem of the longest subsequence which avoids 213 and 231.

1 Introduction

A permutation σ is said to occur in another permutation π, denoted $\sigma \preceq \pi$, if there exists a subsequence of elements of π that has the same relative order as σ. Otherwise, π is said to *avoid* the permutation σ. For example a permutation has an occurrence of the pattern 123 (resp. 321) if it has an increasing (resp. decreasing) subsequence of length 3. Similarly, 6152347 has an occurrence of 213 but not 231. During the last decade, the study of the pattern matching for permutations has become a very active area of research [13]. and a whole annual conference (Permutation Patterns) focuses on pattern in permutation.

We consider here the so-called *pattern matching* problem (also sometimes referred to as the *pattern involvement problem*): Given two permutations σ and π, this problem is to decide whether $\sigma \preceq \pi$ (the problem is attributed to Wilf in [5]). The pattern matching for permutations is known to be **NP**-hard [5]. It is, however, polynomial-time solvable by brute-force enumeration if σ has bounded size. Improvements to this algorithm were presented in [3] and [1], the latter describing a $O(|\pi|^{0.47|\sigma|+o(|\sigma|)})$ time algorithm. Bruner and Lackner [7] gave a fixed-parameter algorithm solving the pattern matching for permutations problem with an exponential worst-case runtime of $O(1.52^n.n.k)$, (This is an improvement upon the $O(2^{|\pi|})$ runtime required by brute-force search without imposing restrictions on σ and π.) A recent major step was taken by Marx

* On a Co-tutelle Agreement with the Department of Mathematics of the University of Trento

and Guillemot [10]. They showed that the pattern matching for permutations problem is fixed-parameter tractable (FPT) for parameter $|\sigma|$.

A few particular cases of the pattern matching for permutations problem have been attacked successfully. The case of increasing patterns is solvable in $O(|\pi| \log\log |\sigma|)$ time in the RAM model [8], improving the previous 30-year bound of $O(|\pi| \log |\sigma|)$. Furthermore, the patterns 132, 213, 231, 312 can all be handled in linear-time by stack sorting algorithms. Any pattern of length 4 can be detected in $O(|\pi| \log |\pi|)$ time [3]. Algorithmic issues for 321-avoiding patterns matching for permutations have been investigated in [11] and more recently in [2]. The pattern matching for permutation problem is also solvable in polynomial-time for separable patterns [12, 5] (see also [6] for Longest common subsequence problem and related one on separable permutations). Separable permutations are permutations that do not have an occurrence of 2413 nor 3142, and they are enumerated by the Schröder numbers. (Notice that the separable permutations include as a special case the stack-sortable permutations, which avoid the pattern 231.)

There exist many generalisations of patterns that are worth considering in the context of algorithmic issues in pattern matching (see [13] for an up-to-date survey). *Vincular patterns*, also called *generalized patterns*, resemble (classical) patterns with the additional constraint that some of the elements in a occurrence must be consecutive in postitions. Of particular importance in our context, Bruner and Lackner [7] proved that deciding whether a vincular pattern σ of length k can occur in a longer permutation π is $W[1]$-complete for parameter k; for an up-to-date survey of the $W[1]$ class and related material, see [9]. *Bivincular patterns* generalize classical patterns even further than vincular patterns by adding a constraint on values.

We focus in this paper on pattern matching issues for $(213, 231)$-avoiding permutations that we call wedge permutations. (*i.e.*, those permutations that avoid both 213 and 231). The number of n-wedge permutations is $t_0 = 1$ for $n = 0$ and $t_n = 2^{n-1}$ for $n \geq 1$ [15], as they are in bijection with binary word of size $n - 1$. On an individual basis, the permutations that do not have an occurrence of the permutation pattern 231 are exactly the *stack-sortable permutations* and they are counted by the Catalan numbers [14]. This paper is organized as follows. In Section 2 the needed definitions are presented. Section 3 is devoted to presenting an online linear-time algorithm in case both permutations are wedge permutations, whereas Section 4 focuses on the case where only the pattern is a wedge permutation. In Section 5 we give a polynomial-time algorithm for a bivincular wedge permutations pattern. In Section 6 we consider the problem of finding the longest wedge permutations pattern in permutations.

2 Definitions

A *permutation* of size n is a linear ordering of an ordered set of size n. When writing the permutation we omit the $<$ sign, thus writing the permutation as a word : $\pi = \pi_1\pi_2 \ldots \pi_n$, whose elements are distinct and usually consist of the

integers $12 \ldots n$. We let $\pi[i]$ stands for π_i. Conveniently, we let $\pi[i:j]$ stands for $\pi_i\pi_{i+1} \ldots \pi_j$, $\pi[:j]$ stands for $\pi[1:j]$ and $\pi[i:]$ stands for $\pi[i:n]$.

We need to consider both the natural order of the set and the linear order given by the permutation. When talking about an element, we refer to the relative position in the natural order of the set as its value and we refer to the relative position in the order given by the permutation as its position. We write $\pi[i]$ to indicate the value of the element at position i. For example, in the permutation $\pi = 51342$, The element 1 is at position 2 and the element at position 1 has for value 5.

It is convenient to use a geometric representation of permutation to ease the understanding of algorithms. The geometric representation corresponds to the set of points with coordinate $(i, \pi[i])$ (see Figure 1).

When an element e_1 is smaller (resp. bigger) than an element e_2 by value (in the natural order of the set), we say that e_1 is below (resp. above) e_2, as a reference to the figure of the permutation. Besides, the element with the biggest value is called the topmost element and the element with the smallest value is called the bottommost element. For example, in the permutation $\pi = 51342$, 2 is below 4, the topmost element is 5 and the bottommost element is 1.

When an element e_1 is smaller (resp. bigger) than an element e_2 by position (in the order given by the permutation), we say that e_1 is on the left of (resp. on the right of) e_2. Besides, the element with the biggest position is called the rightmost element and the element with the smallest position is called the leftmost element. For example, in the permutation $\pi = 51342$, 5 is on the left of 3, the leftmost element is 5 and the rightmost element is 2.

A permutation σ is said to *occur* in the permutation π, written $\sigma \preceq \pi$, if there exists a subsequence of (not necessarily consecutive) elements of π that has the same relative order as σ. Otherwise, π is said to *avoid* the permutation σ. For example, the permutation $\pi = 391867452$ has an occurrence of the pattern $\sigma = 51342$, as can be seen in the highlighted subsequence of $\pi = \mathbf{391867}\mathbf{4}52$ (or $\pi = \mathbf{391867}\mathbf{45}2$ or $\pi = \mathbf{391867}\mathbf{452}$ or $\pi = \mathbf{391867}\mathbf{452}$). Each subsequence 91674, 91675, 91672, 91452, in π is called an *occurrence* of σ. Since the permutation $\pi = 391867452$ contains no increasing subsequence of length four, π avoids 1234.

For an occurrence s of σ in π, we say that $\pi[j]$ is a matching of $\sigma[i]$ or that $\sigma[i]$ is matched to $\pi[j]$ if and only if $\pi[j] = s[i]$.

Geometrically, π has an occurrence of σ if there exists a set of points in π that is isomorphic to the set of points of σ. In other words, if there exists a set of points in π with the same disposition as the set of points of σ, regardless of the distance (see Figure 1).

An *ascent element* of an n-permutation $\pi \in S_n$ is any element $\pi[i]$ such that $i < n$ and $\pi[i] < \pi[i+1]$. For example, the permutation 3452167 has ascents 3, 4, 1 and 6. Similarly, a *descent* is any element $\pi[i]$ such that $i < n$ and $\pi[i] > \pi[i+1]$. So for every $1 \leq i < n$, $\pi[i]$ is either an ascent or a descent of π.

A *right to left maximum* (abbreviate RLMax) of π is an element that does not have any element above it at its right (see Figure 2). Formally, $\pi[i]$ is a RLMax if

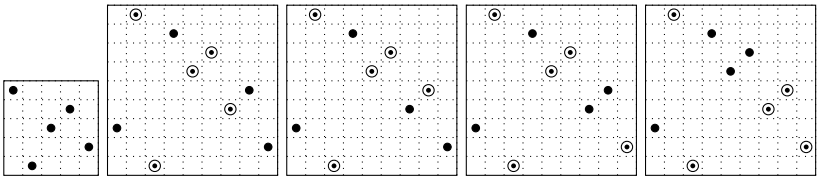

Fig. 1: The pattern $\sigma = 51342$ and four occurrences of σ in 391867452.

and only if $\pi[i]$ is topmost element of $\pi[i:]$. Similarly $\pi[i]$ is a *left to right minima* (abbreviate RLMin) if and only if $\pi[i]$ is the bottommost element of $\pi[i:]$.

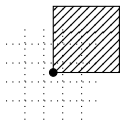

Fig. 2: The element is a RLMax if and only if the dashed area is empty.

A *bivincular pattern* σ of length k is a permutation in S_k written in two-line notation (the top row is $12 \dots k$ and the bottom row is a permutation $\sigma_1\sigma_2 \dots \sigma_k$) with overlined elements in the top row and underlined element in the bottom row. We have the following conditions on the top and bottom rows of σ, as see in [13] in Definition 1.4.1:

- If the bottom line of σ contains $\underline{\sigma_i\sigma_{i+1} \dots \sigma_j}$ then the elements corresponding to $\sigma_i\sigma_{i+1} \dots \sigma_j$ in a occurrence of σ in π must be adjacent, whereas there is no adjacency condition for non-underlined consecutive elements. Moreover if the bottom row of σ begins with $\llcorner\sigma_1$ then any occurrence of σ in a permutation π must begin with the leftmost element of π, and if the bottom row of σ begins with $\sigma_k\lrcorner$ then any occurrence of σ in a permutation π must end with the rightmost element of π.
- If the top line of σ contains $\overline{i\, i+1 \dots j}$ then the elements corresponding to $i, i+1, \dots, j$ in an occurrence of σ in π must be adjacent in values, whereas there is no value adjacency restriction for non-overlined elements. Moreover, if the top row of σ begins with $\ulcorner 1$ then any occurrence of σ is a permutation π must contain the bottommost element of π, and if top row of σ ends with $k\urcorner$ then any occurrence of σ is a permutation π must contain the topmost element of π.

For example, let $\sigma = \begin{smallmatrix} 1\overline{23}4\urcorner \\ \llcorner 21\underline{43} \end{smallmatrix}$. In 3217845, **32**17**84**5 is an occurrence of σ but 3**217**84**5** is not. The best general reference is [13].

Geometrically, We represent underlined and overlined elements by *forbidden areas*. If two elements are overlined then we draw a vertical forbidden areas between those two points. In an occurrence, we also draw the forbidden area

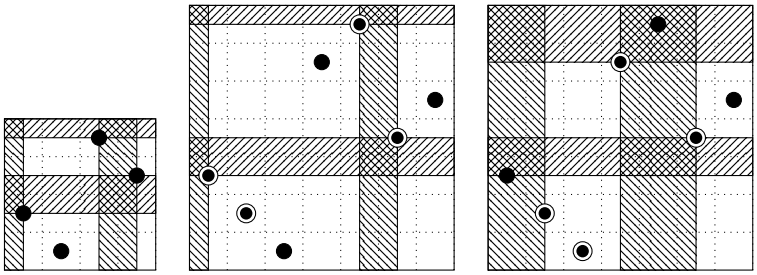

Fig. 3: From left to right, the bivincular pattern $\sigma = {}^{1\overline{23}4\urcorner}_{\llcorner 21\underline{43}}$, An occurrence of σ in 3216745, An occurrence of 2143 in 3216745 but not an occurrence of σ in 3216745 because the point $(1, 3)$ and $(5, 7)$ are in the forbidden area.

between the matching of two overlined elements. This area must be empty for an occurrence to be called so. If the area is empty then there is no element between them when reading the permutation from bottom to top, in other words the matching elements are consecutive in value. If two elements are underlined then we draw an horizontal forbidden area. (See figure 3).

3 Both π and σ are wedge permutation

This section is devoted to presenting a fast algorithm for deciding if $\sigma \preceq \pi$ in case both π and σ are wedge permutations. We begin with an easy but crucial structure lemma.

Lemma 1. *The first element of any wedge permutations must be either the bottommost or the topmost element.*

Proof (of Lemma 1). Any other initial element would serve as a '2' in either a 231 or 213 with 1 and n as the '1' and '3' respectively. □

Corollary 1. *π is a wedge permutation if and only if for $1 \leq i \leq n$, $\pi[i]$ is a RLMax or a RLMin.*

Corollary 2. *Let π be a wedge permutation and $1 \leq i < n$. Then, (1) $\pi[i]$ is an ascent element if and only if $\pi[i]$ is a RLMin and (2) $\pi[i]$ is a descent element if and only if $\pi[i]$ is a RLMax*

Corollary 2 gives a bijection between the set of wedge permutations and the set of binary word of size $n-1$. The word w which corresponds to π is the word where each letter at position i represents whether $\pi[i]$ is an ascent or descent element (equivalently, is a RLMax or a RLMin). We call this bijection B.

A wedge permutation has a particular form that gives its name. If we take only the descent elements, they form a decreasing subsequence and the ascent elements form an increasing subsequence. Moreover the descent elements are above the ascent elements. This shapes the permutation as a $>$.

The following lemma is central to our algorithm.

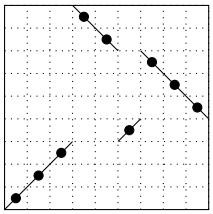

Fig. 4: The wedge permutation 123984765, every point which is on a north-west to south-east line represents a descent element and every point which is on a south-west to north-east line represents an ascent element.

Lemma 2. *Let π and σ be two wedge permutations, Then, π has an occurrence of σ if and only if there exists a subsequence t of π such $B(t) = B(\sigma)$.*

Proof (of Lemma 2). The forward direction is obvious. We prove the backward direction by induction on the size of the pattern σ. The base case is a pattern of size 2. Suppose that $\sigma = 12$ and thus $B(\sigma) = ascent$. Let $t = \pi_{i_1}\pi_{i_2}$, $i_1 < i_2$, be a subsequence of π such that $B(t) = ascent$, this reduces to saying that $\pi_{i_1} < \pi_{i_2}$, and hence that t is an occurrence of $\sigma = 12$ in π. A similar argument shows that the lemma holds for $\sigma = 21$. Now, assume that the lemma is true for all patterns up to size $k \geq 2$. Let σ be an k-wedge permutation and let t, be a subsequence of π of length $k+1$ such that $B(t) = B(\sigma)$. As $B(t)[2:] = B(\sigma)[2:]$ by the inductive hypothesis, it follows that $t[2:]$ is an occurrence of $\sigma[2:]$. Moreover $B(t)[1] = B(\sigma)[1]$ thus $t[1]$ and $\sigma[1]$ are both either the bottommost or the topmost element of their respective subsequences. Therefore, t is an occurrence of σ in π. □

We are now ready to solve the pattern matching problem in case both π and σ are wedge permutations.

Proposition 1. *Let π and σ be two wedge permutations. One can decide whether π has an occurrence of σ in linear time.*

Proof. According to Lemma 1 the problem reduces to deciding whether $B(\sigma)$ occurs as a subsequence in $B(\pi)$. A straightforward greedy approach solves this issue in linear-time, by reading both words from left to right and matching two letters when they are equal then passing to the next letters for both words or passing to the next letter in the word of the text. □

Thank to Corollary 2, we do not need to compute the word corresponding to the permutations before running the greedy algorithm. This computation can be done in the same time with the algorithm, thus, this gives on-line algorithm.

4 Only σ is a wedge permutation

This section focuses on the pattern matching problem in case only the pattern σ is a wedge permutation. We need to consider a specific decomposition of σ into

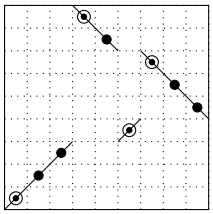

Fig. 5: The wedge permutation 123984765. Every line represents a factor, every circled point represents the leftmost element of each factor.

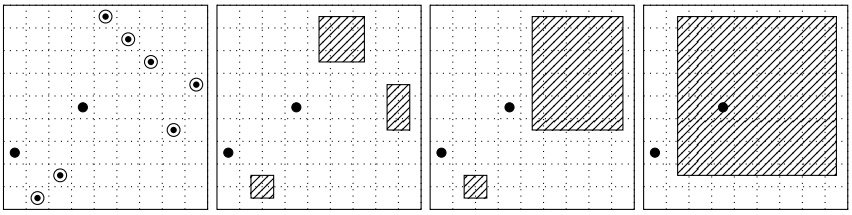

Fig. 6: The operation of merging the two rightmost rectangle.

factor : we split the permutation into maximal sequences of consecutive ascent and descent elements, respectively called an ascent factor and a descent factor. This corresponds to splitting the permutation between every pair of ascent-descent and descent-ascent element (see Figure 5). For the special case of a wedge permutation, this also corresponds to split the permutation into maximal sequences of element consecutive in value. We will label the factors from right to left. For example, $\sigma = 123984765$ is split as $123-98-4-765$. Hence $\sigma = \text{factor}(4)$ factor(3) factor(2) factor(1) with factor(4) = 123, factor(3) = 98, factor(2) = 4 and factor(1) = 765.

We introduce the notation LMEi(s) : Suppose that s is a subsequence of S, LMEi(s) is the position of the leftmost element of s in S. Thus for every factor, LMEi(factor(j)) stands for the position in σ of the leftmost element of factor(j). From the above example, LMEi(factor(4)) = 1, LMEi(factor(3)) = 4, LMEi(factor(2)) = 6 and LMEi(factor(1)) = 7.

To lighten the figure of a wedge permutation, we represent the elements of an ascent (resp. descent) factor by a rectangle which has the matching of the leftmost point of the factor as the left bottom (resp. top) corner and the matching of the rightmost point of the factor as the right top (resp. bottom) corner. We can merge the two rightmost rectangles and replace them by the smallest rectangle that contains both of them and repeat this operation to represent part of an occurrence (see Figure 6).

Remark 1. A factor is either an increasing or a decreasing sequence of element. Thus to find an occurrence for a factor, one must find an increasing or a decreasing subsequence of same size or bigger than the factor.

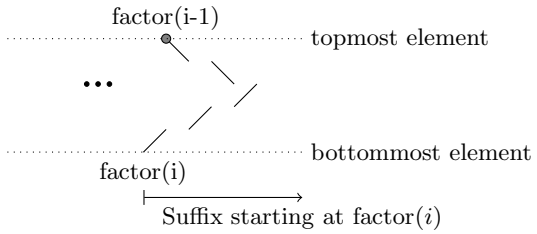

Fig. 7: The topmost element of the suffix starting at factor(i) is the leftmost element of factor($i-1$) (represented by the grey dot). $\mathrm{LM}^{\pi}_{\sigma}(i,j)$ is the smallest value of the matching of the topmost element (the grey dot) in all the occurrences of the suffix starting at LMEi(factor(i)) in $\pi[j:]$.

Corollary 3. *Given a wedge permutation σ, if* factor(i) *is an ascent (respectively descent) factor then topmost (resp. bottommost) element of* σ[LMEi(factor($i+1$)) :] *is the leftmost element of* factor($i-1$).

Proof (of Corollary 3). This corollary states that, given a wedge permutation if the permutation starts with an ascent (respectively descent) elements then the topmost (resp. bottommost) element of this permutation is the first descent (resp. ascent) element (see Figure 7). This is easy to see from Corollary 2. The ascent elements are below the descent element, thus the topmost element must be the first descent element, which is the leftmost element of factor($i-1$). □

We define the set $\mathrm{S}^{\pi}_{\sigma}(i,j)$ as the set of every subsequence s of $\pi[j:]$ that starts at $\pi[j]$ and that is an occurrence of σ[LMEi(factor($i+1$)) :].

Lemma 3. *Let σ be a permutation,* factor(i) *be an ascent (respectively descent) factor, s be a subsequence of π such that $s \in \mathrm{S}^{\pi}_{\sigma}(i,j)$ and that minimizes (resp. maximizes) the matching of the leftmost element of* factor($i-1$). *For all subsequences $s' \in \mathrm{S}^{\pi}_{\sigma}(i,j)$ and for all subsequences t of π, such that $t = t's'$, if t is an occurrence of* σ[LMEi(factor($i+1$)) :] *such that the leftmost element of* factor(i) *is matched to $\pi[j]$ then the subsequence $t's$ is an occurrence of* σ[LMEi(factor($i+1$)) :] *such that the leftmost element of* factor(i) *is matched to $\pi[j]$.*

This lemma states that given any occurrence of factor($i+1$) factor(i) ... factor(1), where factor(i) is an ascent (resp. descent) factor, we can replace the part of the occurrence where factor(i) ... factor(1) occurs, by any occurrence that minimises (resp. maximises) the leftmost element of factor($i-1$). Indeed the leftmost element of factor($i-1$) is the topmost (resp. bottommost) element of factor(i) ... factor(1) (see Figure 7).

Proof (of Lemma 3). Let us consider the case where factor(i) is an ascent factor. By definition s is an occurrence of σ[LMEi(factor(i)) :]. To prove that $t's$ is an occurrence of σ[LMEi(factor($i+1$)) :] we need to prove that the elements of t' are

above the elements of s. If $t's'$ is an occurrence of $\sigma[\text{LMEi}(\text{factor}(i+1)) :]$ then the elements of t' are above the elements of s'. Moreover the topmost element of s is below (or equal to) the topmost element of s' thus the elements of s are below the elements of t. We use a similar argument if factor(i) is a descent factor. □

Corollary 4. *Let σ be a permutation,* factor(i) *be an ascent (respectively descent) factor and s be a subsequence of π such that $s \in \mathrm{S}^{\pi}_{\sigma}(i,j)$ and that minimizes (resp. maximizes) the matching of the leftmost element of* factor$(i-1)$. *These following statements are equivalent :*

- *There exists an occurrence of σ in π where the leftmost element of* factor(i) *is matched to $\pi[j]$.*
- *There exists an occurrence t of $\sigma[:\text{LMEi}(\text{factor}(i))-1]$ in $\pi[:j-1]$ such that ts is an occurrence of σ in π with the leftmost element of* factor(i) *is matched to $\pi[j]$.*

Proof (of Corollary 4). This corollary takes a step further from the previous one, as it states that if there is no occurrence using any occurrence that maximises (resp. minimises) the leftmost element of factor$(i-1)$ then there does not exist any occurrence at all. The backward direction is trivial and the forward direction is applying the Lemma 3. □

This corollary is central to the algorithm because it allows to test only the occurrence that maximises (resp. minimises) the leftmost element of factor$(i-1)$.

Proposition 2. *Let σ be a wedge permutation and $\pi \in S_n$. One can decide in $O(max(kn^2, n^2\log(\log(n)))$ time and $O(kn^2)$ space if π has an occurrence σ.*

Proof. We first introduce a set of values needed to our proof. Let $LIS_\pi(i,j,bound)$ (resp. $LDS_\pi(i,j,bound)$) be the longest increasing (resp. decreasing) sequence in $\pi[i:j]$ starting at i, with every element of this sequence being smaller (resp. bigger) than *bound*. LIS_π and LDS_π can be computed in $O(n^2\log(\log(n)))$ time (see [4]). As stated before, those values allow us to find an occurrence of a factor.

Given a factor i of σ and a position j of π, we want the optimal value of the matching of LMEi(factor$(i-1)$) in any occurrence of $\sigma[\text{LMEi}(\text{factor}(i)) :]$ in $\pi[j :]$ starting at $\pi[j]$ (see Figure 8), which we denote as:

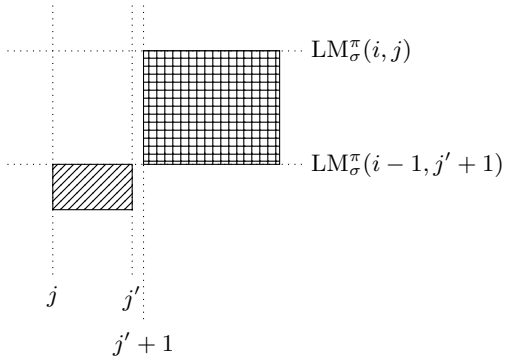

Fig. 8: When looking for an occurrence of σ[LMEi(factor(i)) :] (represented by the dashed and gridded rectangle) starting at $\pi[j:]$, (1) the dashed rectangle is an occurrence of factor(i) if and only if it contains an increasing subsequence of size equal or bigger than $|\,\text{factor}(i)|$. (2) the dashed and gridded rectangles have to be "compatible": their x-coordinates and y-coordinates have to be disjoint and the dashed rectangle has to be on the left and below the gridded rectangle. $\mathrm{AF}^{\pi}_{\sigma}(i,j)$ is the set of y-coordinates of every top edge of the gridded rectangle which is compatible. $\mathrm{LM}^{\pi}_{\sigma}(i,j)$ is the minimal element of $\mathrm{AF}^{\pi}_{\sigma}(i,j)$ if the set is not empty.

$$\mathrm{LM}^{\pi}_{\sigma}(i,j) = \begin{cases} \text{The minimal value of the matching} & \text{If factor}(i) \text{ is} \\ \text{of LMEi(factor}(i-1)) & \text{an ascent factor} \\ \text{in any occurrence of } \sigma[\text{LMEi(factor}(i)) :] & \\ \text{in } \pi[j:] \text{ strarting at } \pi[j] & \\ \text{Or } \infty \text{ if no occurrence exists} & \\ & \\ \text{The maximal value of the matching} & \text{If factor}(i) \text{ is} \\ \text{of LMEi(factor}(i-1)) & \text{an descent factor} \\ \text{in any occurrence of } \sigma[\text{LMEi(factor}(i)) :] & \\ \text{in } \pi[j:] \text{ strarting at } \pi[j] & \\ \text{Or 0 if no occurrence exists} & \end{cases}$$

There exists an occurrence of σ in π if and only if there exists a $1 \leq i \leq n$ such that $LM(n_{factors}, i) \neq 0$ and $LM(n_{factors}, i) \neq \infty$ with $n_{factors}$ the number of factor in σ. We show how to compute recursively those values.

BASE :

$$\mathrm{LM}_\sigma^\pi(1,j) = \begin{cases} \min_{j<j'}\{\infty\} \cup \{\pi[j'] |\ j' \text{ such that } |\,\text{factor}(1)| \leq LIS(j,j',\pi[j']+1)\} & \text{If factor(1) is an ascent factor} \\ \max_{j<j'}\{0\} \cup \{\pi[j'] |\ j' \text{ such that } |\,\text{factor}(1)| \leq LDS(j,j',\pi[j']-1)\} & \text{If factor(1) is a descent factor} \end{cases}$$

In the base case, one is looking for an occurrence of the first factor.

STEP :

$$\mathrm{LM}_\sigma^\pi(i,j) = \begin{cases} \min\{\infty\} \cup \mathrm{AF}_\sigma^\pi(i,j) & \text{If factor}(i) \text{ is an ascent factor} \\ \max\{0\} \cup \mathrm{DF}_\sigma^\pi(i,j) & \text{If factor}(i) \text{ is a descent factor} \end{cases}$$

where $\mathrm{AF}_\sigma^\pi(i,j)$ and $\mathrm{DF}_\sigma^\pi(i,j)$ are the sets of elements matching the leftmost element of factor$(i-1)$ in an occurrence of σ[LMEi(factor(i)) :] starting at $\pi[j:]$. $\mathrm{LM}_\sigma^\pi(i,j)$ has a solution if and only if $\pi[j:j']$ contains an occurrence of factor(i) "compatibles" with an occurrence in $\pi[j'+1:]$ of σ[LMEi(factor$(i-1)$) :]. It is enough to ensure that every element of the occurrence of factor(i) is below (resp. above) the elements of the occurrence of σ[LMEi(factor$(i-1)$) :].

Thus we can compute $\mathrm{AF}_\sigma^\pi(i,j)$ and $\mathrm{DF}_\sigma^\pi(i,j)$ as follows:

$$\begin{aligned} \mathrm{AF}_\sigma^\pi(i,j) = \{\pi[j'+1] \mid j < j' < n \text{ and } \mathrm{LM}_\sigma^\pi(i-1,j'+1) \neq 0 \text{ and} \\ |\,\text{factor}(i)| \leq LIS_\pi(j,j',\mathrm{LM}_\sigma^\pi(i-1,j'+1))\} \\ \mathrm{DF}_\sigma^\pi(i,j) = \{\pi[j'+1] \mid j < j' < n \text{ and } \mathrm{LM}_\sigma^\pi(i-1,j'+1) \neq \infty \text{ and} \\ |\,\text{factor}(i)| \leq LDS_\pi(j,j',\mathrm{LM}_\sigma^\pi(i-1,j'+1))\} \end{aligned}$$

The number of factors is bound by k. Every instance of LIS_π and LDS_π can be computed in $O(n^2 \log(\log(n))$. There are n base cases that can be computed in $O(n)$ time, thus computing every base case takes $O(n^2)$ time. There are kn different instances of AF and each one of them takes $O(n)$ time to compute, thus computing every instance of AF takes $O(kn^2)$ time. There are kn different instances of LM and each one of them takes $O(n)$, thus computing every LM takes $O(kn^2)$ time. Thus computing all the values takes $O(max(kn^2, n^2\log(\log(n)))$ time. Every value takes $O(1)$ space, thus the whole problem takes $O(kn^2)$ space. □

5 Bivincular wedge permutation patterns

This section is devoted to the pattern matching problem with bivincular wedge permutation pattern. Recall that a bivincular pattern generalises a permutation

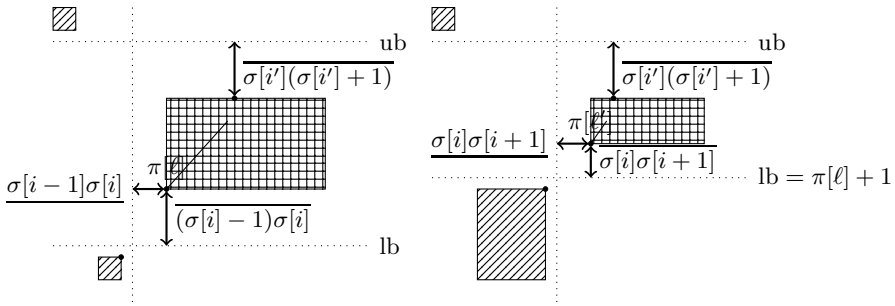

Fig. 9: When solving $\text{PM}_\sigma^{\pi,\text{lb},\text{ub}}(i,j)$, with $\sigma[i]$ an ascent element, the recursive calls of $\text{PM}_*^{*,*,*}(*,*)$ will have the same ub value, and the lb will be equal to the matching of $\pi[i]$ plus one.

pattern by being able to force elements to be consecutive in value or/and in position. Hence, bivincular pattern adds more restrictions on what the occurrence must look like. As a consequence it is easier to avoid a bivincular permutation pattern than the same permutation pattern. Intuitively we cannot use the previous algorithm, as the restrictions on position and value are not managed. As in a wedge permutation, we can describe the structural property of a bivincular wedge permutation pattern.

Lemma 4. *Given σ a bivincular wedge permutation pattern, if $\overline{m(m+1)}$ such that $\sigma[i] = m$ and $\sigma[j] = m+1$, if $\sigma[i]$ is an ascent (resp. decent) element and if $\sigma[i]+1$ is an ascent (resp. decent) element then :*

- *$i < j$ (resp. $j > i$)*
- *For every ℓ, $i < \ell < j$ (resp. $j > \ell > i$), $\sigma[\ell]$ is a descent (resp. ascent) element.*

This lemma states that if two ascent (resp. descent) elements need to be matched to consecutive elements in value then every element between those two elements (if any) is a descent (resp. ascent) element.

Proof (of Lemma 4). The first statement is explained by the fact that ascent elements are increasing, as $\sigma[i] < \sigma[j]$ then $\sigma[i]$ is at the left of $\sigma[j]$ thus $i < j$. Suppose that there exists ℓ, $i < \ell < j$, such that $\sigma[\ell]$ is ascent. Ascent elements are increasing so $\sigma[i] < \sigma[\ell] < \sigma[j]$ which is in contradiction with $\sigma[j] = \sigma[i]+1$. We use a similar argument if $\sigma[i]$ is a descent element. □

Proposition 3. *Let σ be a bivincular wedge permutation pattern of length k and π an n-sized permutation. One can decide in $O(kn^4)$ time and $O(kn^3)$ space if π has an occurrence σ.*

Proof. Given a lower bound lb, a upper bound ub, a position i of σ and a position j of π, we want to know if there exists an occurrence of $\sigma[i:]$ in $\pi[j:]$ with every element of the occurrence is in $[\text{lb}, \text{ub}]$ and starting at $\pi[j]$, which we denote as:

$$\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(i,j) = \begin{cases} true & \text{If } \pi[j:] \text{ has an occurrance of } \sigma[i:] \\ & \text{with every element of the occurrence is in } [\mathrm{lb},\mathrm{ub}] \\ & \text{and starting at } \pi[j] \\ false & otherwise \end{cases}$$

We now show how to compute recursively those values (see Figure 9).

BASE:

$$\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(k,j) = \begin{cases} true & \text{if } \pi[j] \in [\mathrm{lb},\mathrm{ub}] \\ & \text{and if } \sigma[k]_{\lrcorner} \text{ then } j = n \\ & \text{and if } \sigma[k]^{\urcorner} \text{ then } \pi[j] = \mathrm{ub} = n \\ & \text{and if } {}^{\ulcorner}\sigma[k] \text{ then } \pi[j] = \mathrm{lb} = 1 \\ & \text{and if } \overline{(\sigma[k]-1)\sigma[k]} \text{ then } \pi[j] = \mathrm{lb} \\ & \text{and if } \overline{(\sigma[k]\sigma[k]+1)} \text{ then } \pi[j] = \mathrm{ub} \\ false & \text{otherwise} \end{cases}$$

The base case finds an occurrence for the rightmost element of the pattern. If the rightmost element does not have any restriction on positions and on values, then $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(k,j)$ is true if and only if $\sigma[k]$ is matched to $\pi[j]$. This is true if $\pi[j] \in [\mathrm{lb},\mathrm{ub}]$. If $\sigma[k]_{\lrcorner}$ then $\sigma[k]$ must be matched to the rightmost element of π thus j must be n. If $\sigma[k]^{\urcorner}$ then $\sigma[k]$ must be matched to the topmost element which is n. If ${}^{\ulcorner}\sigma[k]$ then $\sigma[k]$ must be matched to the bottommost element which is 1. If $\overline{(\sigma[k]-1)\sigma[k]}$ then the matching element of $\sigma[k]$ and $\sigma[k]-1$ must be consecutive in value, by recursion the value of the element matching $\sigma[k]-1$ will be recorded in lb and by adding 1 to it thus $\sigma[k]$ must be matched to lb. If $\overline{(\sigma[k]\sigma[k]+1)}$ then the element matching $\sigma[k]$ and $\sigma[k]+1$ must be consecutive in value, by recursion the value of the element matching $\sigma[k]+1$ will be recorded in ub and by removing 1 to it thus $\sigma[k]$ must be matched to ub.

STEP:

We need to consider 3 cases for the problem $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(i,j)$:

– If $\pi[j] \notin [\mathrm{lb},\mathrm{ub}]$ then:

$$\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(i,j) = false$$

which is immediate from the definition. If $\pi[j] \notin [\mathrm{lb},\mathrm{ub}]$ then it cannot be part of an occurrence of $\sigma[i:]$ in $\pi[j:]$ with every element of the occurrence in $[\mathrm{lb},\mathrm{ub}]$.

– If $\pi[j] \in [\mathrm{lb}, \mathrm{ub}]$ and $\sigma[i]$ is an ascent element then :

$$\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(i,j) = \begin{cases} \bigcup_{\ell>j} \mathrm{PM}_\sigma^{\pi,\pi[j]+1,\mathrm{ub}}(i+1,\ell) & \text{if } \sigma[i] \text{ is not underlined} \\ & \text{and } \sigma[i] \text{ is not overlined} \\ \bigcup_{\ell>j} \mathrm{PM}_\sigma^{\pi,\pi[j]+1,\mathrm{ub}}(i+1,\ell) & \text{if } \sigma[i] \text{ is not underlined} \\ & \text{and } \overline{(\sigma[i]-1)\sigma[i]} \text{ or } \ulcorner\sigma[i] \\ & \text{and } \pi[j] = \mathrm{lb} \\ \mathrm{PM}_\sigma^{\pi,\pi[j]+1,\mathrm{ub}}(i+1,j+1) & \text{if } \underline{\sigma[i]\sigma[i+1]} \\ & \text{and } \overline{(\sigma[i]-1)\sigma[i]} \text{ or } \ulcorner\sigma[i] \\ & \text{and } \pi[j] = \mathrm{lb} \\ \mathrm{PM}_\sigma^{\pi,\pi[j]+1,\mathrm{ub}}(i+1,j+1) & \text{if } \underline{\sigma[i]\sigma[i+1]} \\ & \text{and } \sigma[i] \text{ is not overlined} \\ false & \text{otherwise} \end{cases}$$

Remark that $\sigma[i]$ can be matched to $\pi[i]$ because $\pi[j] \in [\mathrm{lb}, \mathrm{ub}]$. Thus if $\pi[j+1:]$ has an occurrence of $\sigma[i+1:]$ with every element of the occurrence in $[\pi[i]+1, \mathrm{ub}]$ then $\pi[j:]$ has an occurrence $\sigma[i:]$. The last condition corresponds to know if there exists ℓ, $j < \ell$ such that $\mathrm{PM}_\sigma^{\pi,\pi[j]+1,\mathrm{ub}}(i+1,\ell)$ is true. The first case corresponds to an occurrence without restriction on position and on value. The second case asks for the matching of $\sigma[i]-1$ and $\sigma[i]$ to be consecutive in value, but the matching of $\sigma[i]-1$ is $lb-1$ thus we want $\pi[j] = \mathrm{lb}$. The fourth case asks for the matching of $\sigma[i]$ and $\sigma[i+1]$ to be consecutive in positions, thus the matching of $\sigma[i+1]$ must be $\pi[j+1]$. The third case is a union of the second and fourth case. Note that we do not have to consider the case $\overline{\sigma[i](\sigma[i]+1)}$ as the element $(\sigma[i]+1)$ is on the right of $\sigma[i]$ and thus will be taken care of later on.

– If $\pi[j] \in [\mathrm{lb}, \mathrm{ub}]$ and $\sigma[i]$ is a descent element then :

$$\mathrm{PM}_\sigma^{\pi,\mathrm{lb},\mathrm{ub}}(i,j) = \begin{cases} \bigcup_{\ell>j} \mathrm{PM}_\sigma^{\pi,\mathrm{lb},\pi[j]-1}(i+1,\ell) & \text{if } \sigma[i] \text{ is not underlined} \\ & \text{and } \sigma[i] \text{ is not overlined} \\ \bigcup_{\ell>j} \mathrm{PM}_\sigma^{\pi,\mathrm{lb},\pi[j]-1}(i+1,\ell) & \text{if } \sigma[i] \text{ is not underlined} \\ & \text{and } \overline{\sigma[i](\sigma[i]+1)} \text{ or } \sigma[i]^\urcorner \\ & \text{and } \pi[j] = \mathrm{ub} \\ \mathrm{PM}_\sigma^{\pi,\mathrm{lb},\pi[j]-1}(i+1,j+1) & \text{if } \underline{\sigma[i]\sigma[i+1]} \\ & \text{and } \overline{\sigma[i](\sigma[i]+1)} \text{ or } \sigma[i]^\urcorner \\ & \text{and } \pi[j] = \mathrm{ub} \\ \mathrm{PM}_\sigma^{\pi,\mathrm{lb},\pi[j]-1}(i+1,j+1) & \text{if } \underline{\sigma[i]\sigma[i+1]} \\ & \text{and } \sigma[i] \text{ is not overlined} \\ false & \text{otherwise} \end{cases}$$

The same remark as the last case holds.

Clearly if $\bigcup_{0<j} \mathrm{PM}_\sigma^{\pi,1,n}(1,j)$ is true then π has an occurrence σ. We now discuss the position and value constraints.

Position Constraint. There are 3 types of position constraints that can be added by underlined elements.

- If $\llcorner\sigma[1]$ then the leftmost element of σ must be matched to the leftmost element of π ($\sigma[1]$ is matched to $\pi[1]$ on a occurrence of σ in π). This constraint is satisfied by requiring that the occurrence starts at the leftmost element of π : if $\mathrm{PM}_\sigma^{\pi,1,n}(1,1)$ is true.
- If $\sigma[k]\lrcorner$ then the rightmost element σ must be matched the rightmost element of π ($\sigma[k]$ is matched to $\pi[n]$ on a occurrence of σ in π). This constraint is checked in the base case.
- If $\underline{\sigma[i]\sigma[i+1]}$ then the positions of the element matching $\sigma[i]$ and $\sigma[i+1]$ must be consecutive. In other words, if $\sigma[i]$ is matched to $\pi[j]$ then $\sigma[i+1]$ must be matched to $\pi[j+1]$. We ensure this restriction by recursion by requiring that the matching of $\sigma[i+1:]$ starts at position $j+1$ (see Figure 9).

Value Constraint. There are 3 types of position constraints that can be added by overlined elements.

- If $\ulcorner\sigma[i]$ (and thus $\sigma[i]=1$) then the bottommost element of σ must be matched to the bottommost element of π.
  - If $\sigma[i]$ is an ascent element, then remark that every problem $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i,*)$ is true if $\sigma[i]$ is matched to the element with value lb (by recursion) thus it is enough to require that $\mathrm{lb}=1$. Now remark that $\sigma[i]$ is the leftmost ascent element, indeed if not, then there exists an ascent element $\sigma[i']$, $i'<i$ and by definition $\sigma[i']<\sigma[i]$ which is not possible as $\sigma[i]$ must be the bottommost element. As a consequence $\sigma[1]$, ..., $\sigma[i-1]$ are descent elements. Moreover the recursive calls from a descent element do not change the lower bound thus for every $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i,*)$, $\mathrm{lb}=1$ (see Figure 9).
  - If $\sigma[i]$ is a descent element then $i=k$ ($\sigma[i]$ is the rightmost element). Thus every $\mathrm{PM}_\sigma^{\pi,*,*}(i,*)$ is a base case and is true if $\sigma[i]$ is matched to 1.
- If $\sigma[i]\urcorner$ (and thus $\sigma[i]=k$) then the topmost element of σ must be matched to the topmost element of π.
  - If $\sigma[i]$ is an descent element, then remark that every recursive call $\mathrm{PM}_\sigma^{\pi,*,\mathrm{ub}}(i,*)$ is true if $\sigma[i]$ is matched to element with value ub (by recursion) thus it is enough to require that $\mathrm{ub}=n_\pi$. Now remark that $\sigma[i]$ is the leftmost descent element, indeed if not, then there exists an descent element $\sigma[i']$, $i'<i$ and by definition $\sigma[i']>\sigma[i]$ which is not possible as $\sigma[i]$ must be the topmost element. As a consequence $\sigma[1]$, ..., $\sigma[i-1]$ are ascent elements. Moreover the recursive calls from a ascent element do not change the upper bound thus for every $\mathrm{PM}_\sigma^{\pi,*,\mathrm{ub}}(i,*)$, $\mathrm{ub}=n$ (see Figure 9).

- If $\sigma[i]$ is an ascent element then $i = k$ ($\sigma[i]$ is the rightmost element). Thus every $\mathrm{PM}_\sigma^{\pi,*,*}(i,*)$ is a base case and is true if $\sigma[i]$ is matched to n_π.

– If $\overline{\sigma[i]\sigma[i']}$, (which implies that $\sigma[i'] = \sigma[i]+1$) then if $\sigma[i]$ is matched to $\pi[j]$ then $\sigma[i']$ must be matched to $\pi[j]+1$.
  - The case $\sigma[i]$ is a descent element, $\sigma[i']$ is an ascent element and $i < i'$ (remark that this case is equivalent to the case where $\sigma[i]$ is an ascent element, $\sigma[i']$ is a descent element and $i' < i$) is not possible. Indeed $\sigma[i]$ is the topmost element of $\sigma[i:]$ thus $\sigma[i] > \sigma[i']$ which is in contradiction with $\sigma[i'] = \sigma[i]+1$.
  - If $\sigma[i]$ is an ascent element, $\sigma[i']$ is a descent element and $i < i'$ (remark that this case is symmetric to the case where $\sigma[i]$ is a descent element, $\sigma[i']$ is an ascent element and $i' < i$), then remark that every recursive call $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i',*)$ is true if $\sigma[i']$ is matched to the element with lb (by recursion) thus it is enough to require that $\mathrm{lb} = \pi[j]+1$. Now remark that $\sigma[i]$ is the rightmost ascent element and $\sigma[i']$ is the rightmost element (or $\sigma[i'] \neq \sigma[i]+1$). As a consequence $\sigma[i+1]$, $\sigma[i+2]$, …, $\sigma[i'-1]$ are descent elements. Moreover the recursive calls from a descent element do not change the lower bound and $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i,*)$ will put the lower bound to $\pi[j]+1$ thus for every $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i',*)$, $\mathrm{lb} = \pi[j]+1$ (see Figure 9).
  - If $\sigma[i]$ is an ascent element and $\sigma[i']$ is an ascent element then first remark that every recursive call $\mathrm{PM}_\sigma^{\pi,*,\mathrm{ub}}(i',*)$ is true if $\sigma[i']$ is matched to element with value lb. Now remark that
    $i < i'$ and there is no ascent element between $\sigma[i]$ and $\sigma[i']$ (lemma 4), As a consequence $\sigma[i+1]$, $\sigma[i+2]$, …, $\sigma[i'-1]$ are descent elements. Moreover the recursive calls from a descent element do not change the lower bound and $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i,*)$ will put the lower bound to $\pi[j]+1$ thus for every $\mathrm{PM}_\sigma^{\pi,\mathrm{lb},*}(i',*)$, $\mathrm{lb} = \pi[j]+1$ (see Figure 9).

There are n^3 base cases that can be computed in constant time. There are kn^3 different cases. Each case takes up to $O(n)$ time to compute. Thus computing all the cases take $O(kn^4)$ time. Each case take $O(1)$ space, thus we need $O(kn^3)$ space. □

6 Computing the longest wedge permutation pattern

This section is focused on a problem related to the pattern matching problem, finding the longest wedge permutation occurring in permutations: Given a set of permutations, find a longest wedge permutation that occurs in each input permutations. This problem is known to be NP-Hard for an arbitrary number of permutations and we do not hope it to be solvable in polynomial time even with the constraint that the subsequence must avoid $(213, 231)$, as for a fixed number of permutations (in our case, we have two permutations) the number of permutation appear in the exponent of the complexity of the algorithm. Thus we focus on the cases where only one or two permutations are given in input.

Conveniently, we say that a subsequence is a wedge subsequence if and only if the permutation represented by the subsequence is a wedge permutation.

We start with the easiest case where we are given just one input permutation. We need the set of descent elements and the set of ascent elements. $A(\pi) = \{i|\pi[i] \text{ is an ascent element}\} \cup \{n\}$ and $D(\pi) = \{i|\pi[i] \text{ is a descent element}\} \cup \{n\}$.

Proposition 4. *If s_i is the longest increasing subsequence with last element at position f in π and s_d is the longest decreasing subsequence with last element at position f in π then $s_i \cup s_d$ is a longest wedge subsequence with last element at position f in π.*

Proof. Let us first prove that we have a wedge subsequence. s_i is an increasing subsequence with values below or equal $\pi[f]$ and s_d is an decreasing subsequence with values above or equal $\pi[f]$, so $s_i \cup s_d$ is a wedge subsequence. Let us prove that this is a longest. Let s be a wedge subsequence with its rightmost element at position f in π such that $|s| > |s_i \cup s_d|$, first note that $A(s)$ is also an increasing subsequence with its rightmost at position f in π and $D(s)$ is also an decreasing subsequence with its rightmost at position f in π, then as $|s| > |s_i \cup s_d|$ then either $|A(s)| > |s_i|$ or $|D(s)| > |s_d|$, which is in contradiction with the definition of s_i and s_d.

Proposition 5. *Let π be a permutation. One can compute the longest $(213, 231)$-avoiding subsequence that can occur in π in $O(n \log(\log(n)))$ time and in $O(n)$ space.*

Proof (of Proposition 5). The Proposition 4 leads to an algorithm where one has to compute longest increasing and decreasing subsequence endings at every position possible. Then finding the maximum sum of longest increasing and decreasing subsequence endings at the same position. Computing the longest increasing subsequence and the longest decreasing subsequence can be done in $O(n \log(\log(n)))$ time and $O(n)$ space (see [4]), then finding the maximal can be done in linear time. □

We now consider the case where the input is composed of two permutations.

Proposition 6. *Given two permutations π_1 and π_2, one can compute the longest common $(213, 231)$-avoiding subsequence in $O(|\pi_1|^3|\pi_2|^3)$ time and space.*

Proof. Consider the following problem that computes the longest wedge subsequence common to π_1 and π_2: Given two permutations π_1 and π_2, we define $\mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1, i_2)$

$= \max \{|s| \mid s \text{ occurs } \pi_1[i_1 :] \text{ with every element of the occurrence in } [\mathrm{lb}_1, \mathrm{ub}_1]$
and s occurs $\pi_2[i_2 :]$ with every element of the occurrence in $[\mathrm{lb}_2, \mathrm{ub}_2]$ $\}$

We show how to solve this problem by dynamic programming.

BASE:

$$\mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(|\pi_1|,|\pi_2|) = \begin{cases} 1 & \text{if } \mathrm{lb}_1 \leq \pi_1[j] \leq \mathrm{ub}_1 \\ & \text{and } \mathrm{lb}_2 \leq \pi_2[j] \leq \mathrm{ub}_2 \\ 0 & \textit{otherwise} \end{cases}$$

STEP:

$$\mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1,i_2) = max \begin{cases} \mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1,i_2+1) \\ \\ \mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1+1,i_2) \\ \\ \mathrm{M}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1,i_2) \end{cases}$$

with

$$\mathrm{M}^{\pi_2,\mathrm{lb}_2,\mathrm{ub}_2}_{\pi_1,\mathrm{lb}_1,\mathrm{ub}_1}(i_1,i_2) = \max \begin{cases} 1+\mathrm{LCS}^{\pi_2,\pi_2[i_2]+1,\mathrm{ub}_2}_{\pi_1,\pi_1[i_1]+1,\mathrm{ub}_1}(i_1+1,i_2+1) & \pi_1[i_1] < \mathrm{lb}_1 \\ & \text{and } \pi_2[i_2] < \mathrm{lb}_2 \\ \\ 1+\mathrm{LCS}^{\pi_2,\mathrm{lb}_2,\pi_2[i_2]-1}_{\pi_1,\mathrm{lb}_1,\pi_1[i_1]-1}(i_1+1,i_2+1) & \pi_1[i_1] > \mathrm{ub}_1 \\ & \text{and } \pi_2[i_2] > \mathrm{ub}_2 \\ \\ 0 & \text{otherwise} \end{cases}$$

The solution to the problem relies on the fact that the longest wedge subsequence is found either by considering the problem with $\pi_1[i_1:]$ and $\pi_2[i_2+1:]$ or by considering the problem with $\pi_1[i_1+1:]$ and $\pi_2[i_2:]$ or by matching $\pi_1[i_1]$ and $\pi_2[i_2]$ and adding to the solution the LCS for $\pi_1[i_1+1:]$ and $\pi_2[i_2+1:]$ which is compatible, meaning that if the matching element is an ascent (resp. descent) element then we consider only the solution with elements above (below) $\pi_1[i_1]$ for the occurrence in $\pi_1[i_1+1:]$ and $\pi_2[i_2]$ for the occurrence in $\pi_2[i_2+1:]$.

These relations lead to a $O(|\pi_1|^3|\pi_2|^3)$ time and $O(|\pi_1|^3|\pi_2|^3)$ space algorithm. Indeed there are $|\pi_1|^3|\pi_2|^3$ cases possible for the problem and each case is solved in constant time. □

References