# Deciding the finiteness of the number of simple permutations contained in a wreath-closed class is polynomial*

Frédérique Bassino
LIPN UMR 7030, Université Paris 13 and CNRS,
99, avenue J.- B. Clément, 93430 Villetaneuse, France.

Mathilde Bouvel
LaBRI UMR 5800, Université de Bordeaux and CNRS,
351, cours de la Libération, 33405 Talence cedex, France.

Adeline Pierrot
LIAFA UMR 7089, Université Paris Diderot and CNRS,
Case 7014, 75205 Paris cedex 13, France.

Dominique Rossin
LIX UMR 7161, Ecole Polytechnique and CNRS,
91128 Palaiseau, France.

**Abstract**

We present an algorithm running in time $\mathcal{O}(n \log n)$ which decides if a wreath-closed permutation class $Av(B)$ given by its finite basis $B$ contains a finite number of simple permutations. The method we use is based on an article of Brignall, Ruškuc and Vatter [9] which presents a decision procedure (of high complexity) for solving this question, without the assumption that $Av(B)$ is wreath-closed. Using combinatorial, algorithmic and language theoretic arguments together with one of our previous results on pin-permutations [6], we are able to transform the problem into a co-finiteness problem in a complete deterministic automaton.

## 1 Introduction

Permutation classes were first introduced in the literature by Knuth in [11], where the class of permutations sortable through one stack is characterized as the permutations avoiding the pattern 231. This result has been the starting point of the study of permutation classes and pattern-avoiding permutations in combinatorics. The study of permutation classes has been mostly interested in enumeration questions as testified by

*This work was completed with the support of the ANR project GAMMA number 07-2_195422

the survey [10] and its references. The predominance of the counting questions certainly finds an explanation in the Stanley-Wilf conjecture, stating that the enumeration sequence $(s_n)_n$ of any (non trivial) permutation class is at most simply exponential in the length $n$ of the permutations (as opposed to $n!$ in general). This conjecture has been proved by Marcus and Tardos in 2004 [12], and this result can be considered as one of the first general results on permutation classes, that is to say a result that deals with *all* permutation classes. More recently, some other general results dealing with wide families of permutation classes have been described [3, 4, 5, 8, 9, 13]. In particular, Albert and Atkinson [3] proved some sufficient conditions for the generating function $S(x) = \sum s_n x^n$ of a class to be algebraic. This is also the direction chosen in this article, where we are interested in describing an efficient algorithm to decide the finiteness of the number of simple permutations in a class, for *any* wreath-closed permutation class.

To be more precise, in a series of three articles [8, 7, 9] Brignall *et al.* prove that it is decidable to know if a permutation class of finite basis contains a finite number of simple permutations, which is a sufficient condition for the generating function to be algebraic. Every algorithm involved in this decision procedure is polynomial except the algorithm deciding if the class contains arbitrarily long proper pin-permutations.

In [6] a detailed study of pin-permutations is performed. We use some of the properties of the simple pin-permutations established in [6] to give a polynomial-time algorithm for the preceding question in the restricted case of wreath-closed permutation classes, that is to say the classes of permutations whose bases contain only simple permutations. More precisely, we give a $\mathcal{O}(n \log n)$ algorithm to decide if a finitely based wreath-closed class of permutations $Av(\pi^{(1)}, \dots, \pi^{(k)})$ contains a finite number of simple permutations where $n = \sum |\pi^{(i)}|$. A key ingredient of this procedure is the transformation of a containment relation involving permutations into a factor relation between words. As a consequence deciding the finiteness of the number of proper pin-permutations is changed into testing the co-finiteness of a regular language given by a complete deterministic automaton.

The paper is organized as follows. We first recall basic definitions and known results that will be used in the sequel. In Section 3 we establish, in the special case of simple patterns and proper pin-permutations, some links between pattern containment relation on permutations and factor relation between words. Finally Section 4 is devoted to the presentation of a polynomial algorithm deciding the finiteness of the number of proper pin-permutations contained in a wreath-closed permutation class.

# 2 Background

## 2.1 Definitions

We recall in this section a few definitions about permutations, pin representations and pin words. More details can be found in [8, 9, 6]. A permutation $\sigma \in S_n$ is a bijective function from $\{1, \dots, n\}$ onto $\{1, \dots, n\}$. We either represent a permutation by a word $\sigma = 2\,3\,1\,4$ or its *diagram* (see Figure 1). A permutation $\pi = \pi_1\pi_2 \dots \pi_k$ is a *pattern* of a permutation

$\sigma = \sigma_1\sigma_2\ldots\sigma_n$, and we write $\pi \leq \sigma$ if and only if there exist $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ such that $\sigma_{i_1}\ldots\sigma_{i_k}$ is order isomorphic to $\pi$. We also say that $\sigma$ *involves* or *contains* $\pi$. If $\pi$ is not a pattern of $\sigma$ we say that $\sigma$ *avoids* $\pi$. A permutation class $Av(B)$ – where $B$ is a finite or infinite antichain of permutations called the *basis* – is the set of all permutations avoiding every element of $B$. A permutation is called *simple* if it contains no block, *i.e.* no mapping from $\{i,\ldots,(i+l)\}$ to $\{j,\ldots,(j+l)\}$, except the trivial ones corresponding to $l = 0$ or $i = j = 1$ and $l = n-1$. Wreath-closed permutation classes have been introduced in [3] in terms of substitution- or wreath-product of permutations. This original definition is not crucial to our work, and we prefer to define them by the characterization proved in [3]: a permutation class $Av(B)$ is said to be *wreath-closed* when its basis $B$ contains only simple permutations.

In the following we study wreath-closed classes with finite basis. Note that it is not a restriction for our purpose: from [3], when the basis is infinite we know that the number of simple permutations in the class is infinite. Our goal is indeed to check whether a wreath-closed class contains a finite number of simple permutations, ensuring in this way that its generating function is algebraic [3]. As we shall see in the following, a class of particular permutations, called the pin-permutations, plays a central role in the decision procedure of this problem. For this reason, we record basic definitions and results related with these pin-permutations.

A pin in the plane is a point at integer coordinates. A pin $p$ *separates* - horizontally or vertically - the set of pins $P$ from the set of pins $Q$ if and only if a horizontal - resp. vertical - line drawn across $p$ separates the plane into two parts, one of which contains $P$ and the other one contains $Q$. A pin sequence is a sequence $(p_1,\ldots,p_k)$ of pins in the plane such that no two points lie in the same column or row and for all $i \geq 2$, $p_i$ lies outside the bounding box of $\{p_1,\ldots,p_{i-1}\}$ and respects one of the following conditions:

- $p_i$ separates $p_{i-1}$ from $\{p_1,\ldots,p_{i-2}\}$.
- $p_i$ is independent from $\{p_1,\ldots,p_{i-1}\}$, *i.e.*, it does not separate this set into two non empty sets.

A pin sequence represents a permutation $\sigma$ if and only if it is order isomorphic to its diagram. We say that a permutation $\sigma$ is a *pin-permutation* if it can be represented by a pin sequence, which is then called a *pin representation* of $\sigma$. Not all permutations are pin-permutations (see for example the permutation $\sigma$ of Figure 1).

A *proper* pin representation is a pin representation in which every pin $p_i$, for $i \geq 3$, separates $p_{i-1}$ from $\{p_1,\ldots,p_{i-2}\}$. A *proper* pin-permutation is a permutation that admits a proper pin representation.

**Remark 2.1.** *A pin representation of a simple pin-permutation is always proper as any independent pin $p_i$ with $i \geq 3$ creates a block corresponding to $\{p_1,\ldots,p_{i-1}\}$.*

Pin representations can be encoded by words on the alphabet $\{1,2,3,4,U,D,L,R\}$ called *pin words*. Consider a pin representation $(p_1,\ldots,p_n)$ and choose an arbitrary origin $p_0$ in the plane such that it extends the pin representation to a pin sequence

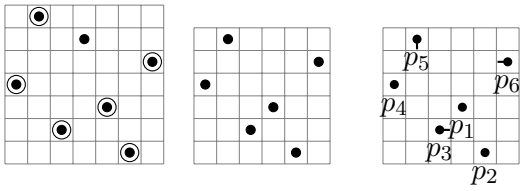



Figure 1: The permutation $\sigma = 4\,7\,2\,6\,3\,1\,5$, the pattern $\pi = 4\,6\,2\,3\,1\,5$ and a pin representation of $\pi$. $14L2UR$ (if we place $p_0$ between $p_3$ and $p_1$) and $3DL2UR$ are pin words corresponding to this pin representation.

$(p_0, p_1, \ldots, p_n)$. Then every pin $p_1, \ldots, p_n$ is encoded by a letter according to the following rules:

- The letter associated to $p_i$ is $U$ -resp. $D, L, R$- if and only if $p_i$ separates $p_{i-1}$ and $\{p_0, p_1, \ldots, p_{i-2}\}$ from the top -resp. bottom, left, right-.
- The letter associated to $p_i$ is 1 -resp. $2, 3, 4$- if and only if $p_i$ is independent from $\{p_0, p_1, \ldots, p_{i-1}\}$ and is situated in the up-right -resp. up-left, bottom-left, bottom-right- corner of the bounding box of $\{p_0, p_1, \ldots, p_{i-1}\}$.

This encoding is summarized by Figure 2. The region encoded by 1 is called the first *quadrant*. The same goes for $2, 3, 4$. The letters $L, R, U, D$ are called *directions*, while $1, 2, 3$ and $4$ are *numerals*. An important remark is that the definition of pin words implies that they do not contain any of the factors $UU, UD, DU, DD, LL, LR, RL$ and $RR$.

| | | |
|---|---|---|
| 2 | U | 1 |
| L | | R |
| 3 | D | 4 |

Figure 2: Encoding of pins by letters.

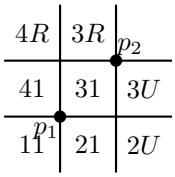



Figure 3: The two letters in each cell indicate the first two letters of the pin word encoding $(p_1, \ldots, p_n)$ when $p_0$ is taken in this cell.

To each pin word corresponds a unique pin representation, hence a unique permutation but each pin-permutation of length greater than 1 has at least 6 pin words associated to it. The reason is that for any pin representation, there are 8 possible placements of $p_0$ w.r.t. $p_1$ and $p_2$, among which at least 6 give a possible prefix of a pin word (see Figure 3 for an example). On Figure 3, the two prefixes $4R$ and $3R$ (resp.. $2U$ and $3U$) may be excluded, when $p_3$ is encoded by $R$ or $L$ (resp. $U$ or $D$).

A *strict* (resp. *quasi-strict*) pin word is a pin word of length at least 2 that begins by a numeral (resp. two numerals) followed only by directions.

**Remark 2.2.** *Encodings of proper pin-permutations*

*a. Strict and quasi-strict pin words are the encodings of proper pin representations.*

*b. However a pin-permutation is proper if and only if it admits a strict pin word among its encodings.*

The language $\mathcal{SP}$ of strict pin words can be described by the following regular expression:

$$(1{+}2{+}3{+}4)\Big((\epsilon{+}L{+}R)(U{+}D)\big((L{+}R)(U{+}D)\big)^*{+}(\epsilon{+}U{+}D)(L{+}R)\big((U{+}D)(L{+}R)\big)^*\Big).$$

## 2.2 Some known results

In [9] Brignall *et al.* studied conditions for a class to contain an infinite number of simple permutations. Introducing three new kinds of permutations they show that this problem is equivalent to looking for an infinite number of permutations of one of these three simpler kinds.

**Theorem 2.3.** *[9] A permutation class $Av(B)$ contains an infinite number of simple permutations if and only if it contains either:*

- *An infinite number of wedge simple permutations.*
- *An infinite number of parallel alternations.*
- *An infinite number of proper pin-permutations.*

The definitions of the wedge simple permutations and the parallel alternations are not crucial to our work, hence we refer the reader to [9] for more details. What is however important for our purpose is to be able to test whether a class given by its finite basis contains an infinite number of permutations of these kinds. Alternations and wedge simple permutations are well characterized in [9], where it is shown that it is easy to deal with this problem using the three following lemmas.

**Lemma 2.4.** *[9] The permutation class $Av(B)$ contains only finitely many parallel alternations if and only if its basis $B$ contains an element of every symmetry of the class $Av(123, 2413, 3412)$.*

**Lemma 2.5.** *[9] The permutation class $Av(B)$ contains only finitely many wedge simple permutations of type 1 if and only if $B$ contains an element of every symmetry of the class $Av(1243, 1324, 1423, 1432, 2431, 3124, 4123, 4132, 4231, 4312)$.*

**Lemma 2.6.** *[9] The permutation class $Av(B)$ contains only finitely many wedge simple permutations of type 2 if and only if $B$ contains an element of every symmetry of the class $Av(2134, 2143,\ 3124, 3142, 3241, 3412, 4123, 4132, 4231, 4312)$.*

With these lemmas, it is possible to decide in polynomial time whether a class contains a finite number of wedge simple permutations or of parallel alternations. More precisely, we have:

**Lemma 2.7.** *Testing whether a finitely based class $Av(B)$ contains finitely many parallel alternations (resp. wedge simple permutations of type 1, resp. wedge simple permutations of type 2) can be done in $\mathcal{O}(n \log n)$ time, where $n = \sum_{\pi \in B} |\pi|$.*

*Proof.* By Lemmas 2.4, 2.5 and 2.6, deciding if a class $Av(B)$ contains a finite number of wedge simple permutations or parallel alternations is equivalent to checking if there exists an element of $B$ in every symmetric class of special pattern avoiding permutation classes, where the bases are composed only of permutations of length at most 4. From [2] checking whether a permutation $\pi$ avoids some patterns of length at most 4 can be done in $\mathcal{O}(|\pi| \log |\pi|)$. This leads to a $\mathcal{O}(n \log n)$ algorithm for deciding whether the numbers of parallel alternation and of wedge simple permutations in the class are finite. □

In [9] Brignall *et al.* also proved that it is decidable to know if a class contains a infinite number of proper pin-permutations using language theoretic arguments. Analyzing their procedure, we can prove that it has an exponential complexity due to the resolution of a co-finiteness problem for a regular language given by a non-deterministic automaton. As said before our goal in this paper is to solve this same problem in polynomial time for the wreath-closed classes.

# 3 Pattern containment and pin words

In this section we show how to transform into a factor relation between words the pattern containment relation of a simple permutation pattern in a proper pin-permutation. More precisely, let $Av(B)$ be a finitely based wreath-closed class of permutations, that is to say such that its basis $B$ is finite and contains only simple permutations. We prove that the set of strict pin words corresponding to permutations that contain an element of $B$ is characterized as the set of all strict pin words whose images by a particular bijection (denoted by $\phi$ in the sequel) contain some factors.

First recall the definition of the partial order $\preceq$ on pin words introduced in [9].

**Definition 3.1.** *Let $u$ and $w$ be two pin words. We decompose $u$ in terms of its strong numeral-led factors as $u = u^{(1)} \dots u^{(j)}$,* a strong numeral-led factor *being a sequence of contiguous letters beginning with a numeral and followed by any number of directions (but no numerals). We then write $u \preceq w$ if $w$ can be chopped into a sequence of factors $w = v^{(1)} w^{(1)} \dots v^{(j)} w^{(j)} v^{(j+1)}$ such that for all $i \in \{1, \dots, j\}$:*

- *if $w^{(i)}$ begins with a numeral then $w^{(i)} = u^{(i)}$, and*
- *if $w^{(i)}$ begins with a direction, then $v^{(i)}$ is nonempty, the first letter of $w^{(i)}$ corresponds to a point lying in the quadrant specified by the first letter of $u^{(i)}$, and all other letters in $u^{(i)}$ and $w^{(i)}$ agree.*

This order is closely related to the pattern containment order $\leq$ on permutations.

**Lemma 3.2.** *[9] If the pin word $w$ corresponds to the permutation $\sigma$ and $\pi \leq \sigma$ then there is a pin word $u$ corresponding to $\pi$ with $u \preceq w$. Conversely if $u \preceq w$ then the permutation corresponding to $u$ is contained in the permutation corresponding to $w$.*

In what follows, $\sigma$ is a proper pin-permutation. So we can choose a strict pin word $w$ that encodes $\sigma$ (see Remark 2.2 b.). As a consequence of Lemma 3.2, checking whether a permutation $\pi$ is a pattern of $\sigma$ is equivalent to checking whether there exists a pin word $u$ corresponding to $\pi$ with $u \preceq w$. Additionally, we show that when $\pi$ is simple, we can associate to each strict (resp. quasi-strict) pin word $v = v_1 v_2 \dots v_n$ of $\pi$ a word $\phi(v)$ (resp. $\phi(v_2 \dots v_n)$) that does not contain numerals and such that the pattern involvement problem is equivalent to checking if $\phi(w)$ has a factor of the form $\phi(v)$ for $v$ strict or $\phi(v_2 \dots v_n)$ for $v$ quasi-strict encoding $\pi$.

**Definition 3.3.** *Let $M$ be the set of words of length greater than or equal to 3 over the alphabet $L, R, U, D$ such that $R, L$ is followed by $U, D$ and conversely.*

*We define a bijection $\phi$ from $\mathcal{SP}$ to $M$ as follows. For any strict pin word $u \in \mathcal{SP}$ such that $u = u'u''$ with $|u'| = 2$, we set $\phi(u) = \varphi(u')u''$ where $\varphi$ is given by:*

| | | | |
|---|---|---|---|
| $1R \mapsto RUR$ | $2R \mapsto LUR$ | $3R \mapsto LDR$ | $4R \mapsto RDR$ |
| $1L \mapsto RUL$ | $2L \mapsto LUL$ | $3L \mapsto LDL$ | $4L \mapsto RDL$ |
| $1U \mapsto URU$ | $2U \mapsto ULU$ | $3U \mapsto DLU$ | $4U \mapsto DRU$ |
| $1D \mapsto URD$ | $2D \mapsto ULD$ | $3D \mapsto DLD$ | $4D \mapsto DRD$ |

For any $n \geq 2$, the map $\phi$ is a bijection from the set $\mathcal{SP}_n$ of strict pin words of length $n$ to the set $M_{n+1}$ of words of $M$ of length $n+1$. Furthermore, it satisfies, for any $u \in \mathcal{SP}$, $u_i = \phi(u)_{i+1}$ for any $i \geq 2$.

In the above table, we can notice that, for any $u \in \mathcal{SP}$, the first two letters of $\phi(u)$ are sufficient to determine the first letter of $u$ (which is a numeral). Thus it is natural to extend the definition of $\phi$ to words of length 1 (that do not belong to $\mathcal{SP}$ by definition) by setting $\phi(1) = \{UR, RU\}, \phi(2) = \{UL, LU\}, \phi(3) = \{DL, LD\}$ and $\phi(4) = \{RD, DR\}$, and by defining consistently $\phi^{-1}(v) \in \{1, 2, 3, 4\}$ for any $v$ in $\{LU, LD, RU, RD, UL, UR, DL, DR\}$.

Notice that our bijection consists of replacing the only numeral in any strict pin word by two directions. Lemma 3.4 below shows that for each strict pin word $w$, we know in which quadrant lies every pin of the pin representation corresponding to $w$.

**Lemma 3.4.** *Let $w$ be a strict pin word and $p$ the pin representation corresponding to $w$. For any $i \geq 2$, set*

$$q(w_{i-1}, w_i) = \begin{cases} \phi^{-1}(w_{i-1}w_i) \text{ if } i \geq 3 \\ \phi^{-1}(BC) \text{ if } i = 2 \text{ and } \phi(w_1 w_2) = ABC \end{cases}$$

*Then for any $i \geq 2$, $q(w_{i-1}, w_i)$ is a numeral indicating the quadrant in which $p_i$ lies with respect to $\{p_0, \dots, p_{i-2}\}$.*

*Proof.* It is obvious that $q(w_{i-1}, w_i)$ is a numeral. The fact that it indicates the claimed quadrant is proved by case examination, distinguishing the case $i \geq 3$ from $i = 2$.

If $i \geq 3$, $w_{i-1}$ and $w_i$ are directions. For example if $w_{i-1} = L$ and $w_i = U$, then $p_i$ lies in the quadrant 2 and $\phi^{-1}(LU) = 2$.

If $i = 2$, $w_{i-1}$ is a numeral and $w_i$ is a direction. For example if $w_{i-1} = 1$ and $w_i = L$, then $p_i$ lies in the quadrant 2 and we have $\phi(1L) = RUL$ and $\phi^{-1}(UL) = 2$. □

By Remarks 2.1 and 2.2 a., pin words encoding simple permutations are either strict or quasi strict. We first show how to interpret $\preceq$ by a factor relation in the case of strict pin words.

**Lemma 3.5.** *For any strict pin words $u$ and $w$, $u \preceq w$ if and only if $\phi(u)$ is a factor of $\phi(w)$.*

*Proof.* If $u \preceq w$, as $u$ is a strict pin word, writing $u$ in terms of its strong numeral-led factors leads to $u = u^{(1)}$, thus $w$ can be decomposed into a sequence of factors $w = v^{(1)}w^{(1)}v^{(2)}$ as in Definition 3.1.

If $v^{(1)}$ is empty then $w^{(1)}$ begins with a numeral, $w^{(1)} = u^{(1)}$ and $u$ is a prefix of $w$. Consequently $\phi(u)$ is a prefix of $\phi(w)$.

Otherwise $i = |v^{(1)}| \geq 1$ and $w^{(1)}$ begins with a direction. By Definition 3.1, the first letter $w_{i+1}$ of $w^{(1)}$ corresponds to a point $p_{i+1}$ lying in the quadrant specified by $u_1$ (the first letter of $u^{(1)}$), and all other letters (which are directions) in $u^{(1)}$ and $w^{(1)}$ agree: $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$.

By Lemma 3.4, $q(w_i, w_{i+1})$ is the quadrant in which $p_{i+1}$ lies, *i.e.* $u_1 = q(w_i, w_{i+1})$. Since $|u| \geq 2$, by definition of $q$ we have that $\phi(u) = \phi(q(w_i, w_{i+1})w_{i+2} \dots w_{i+|u|})$ is a factor of $\phi(w)$.

Conversely if $\phi(u)$ is a factor of $\phi(w)$ then $\phi(w) = v\,\phi(u)\,v'$. If $v$ is empty then $\phi(u)$ is a prefix of $\phi(w)$ thus $u$ is a prefix of $w$ hence $u \preceq w$.

If $|v| = i \geq 1$ then by definition of $\phi$, $u_2 \dots u_{|u|}$ is a factor of $\phi(w)$ and more precisely appears in $\phi(w)$ for indices from $i + 3$ to $i + |u| + 1$. This means that $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$. Since $i \geq 1$, $w_{i+1}$ is a direction, and we are left to prove that the point $p_{i+1}$ corresponding to $w_{i+1}$ lies in the quadrant indicated by $u_1$. By Lemma 3.4, $p_{i+1}$ lies in quadrant $q(w_i, w_{i+1})$, and we easily check that $q(w_i, w_{i+1}) = \phi^{-1}(xy)$ where $xy$ are the first two letters of $\phi(u)$. Hence, we get that $q(w_i, w_{i+1}) = u_1$, concluding the proof. □

The second possible structure for a pin word corresponding to a simple permutation is to begin with two numerals.

**Lemma 3.6.** *Let $u$ be a quasi strict pin word and $w$ be a strict pin word. If $u \preceq w$ then $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$.*

*Proof.* Decompose $u$ into its strong numeral-led factors $u = u^{(1)}u^{(2)}$. Notice that $u^{(2)} = u_2 \dots u_{|u|}$. Since $u \preceq w$, $w$ can be decomposed into a sequence of factors $w = v^{(1)}w^{(1)}v^{(2)}w^{(2)}v^{(3)}$ satisfying Definition 3.1. Moreover $|w^{(1)}| = |u^{(1)}| = 1$ so $w^{(2)}$

contains no numerals thus $v^{(2)}$ is non-empty, the first letter of $w^{(2)}$ corresponds to a point lying in the quadrant specified by the first letter of $u^{(2)}$, and all other letters in $u^{(2)}$ and $w^{(2)}$ agree. Hence $w = v^{(1)}w^{(1)}v\phi(u^{(2)})v^{(3)}$ where $v$ is the prefix of $v^{(2)}$ of length $|v^{(2)}| - 1$. Then $\phi(u^{(2)})$ is a factor of $w$ which has no numeral thus $\phi(u^{(2)})$ is a factor of $\phi(w)$ which begin at position $p \geq 3$. □

**Lemma 3.7.** *Let $u$ be a quasi strict pin word corresponding to a permutation $\pi$ and $w$ be a strict pin word corresponding to a permutation $\sigma$. If $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ then $\pi$ is a pattern of $\sigma$.*

*Proof.* Set $u^{(2)} = u_2 \dots u_{|u|}$. Since $\phi(u^{(2)})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ then by Lemma 3.5, $u^{(2)} \preceq w$. Let $p_1 \dots p_n$ be a pin representation of $w$ (which corresponds to $\sigma$) and $\Gamma$ be the subset of points corresponding to $u^{(2)}$, then $\Gamma \subseteq \{p_3 \dots p_n\}$. Let $\pi'$ be the permutation corresponding to $\{p_1\} \cup \Gamma$, then $\pi' \leq \sigma$. We claim that $\pi' = \pi$. Let $i$ be the quadrant in which $p_1$ lies, and $v = i\,u^{(2)}$. Then $v$ is a pin word corresponding to $\pi'$. As $u$ begins with two numerals, there is $k \in \{1, \dots, 4\}$ such that $u = k\,u^{(2)}$. It is easy to see that $v$ and $u$ encode the same permutation, even if $i \neq k$. Hence $\pi' = \pi$. □

The set of pin words of any simple permutation $\pi$ contains at most 64 elements. Indeed by Lemma 4.6 of [6] there are at most 8 pin representations $p$ of $\pi$ –corresponding to the possible choices of $(p_1, p_2)$– and at most 8 pin words for each pin representation (see Figure 3) so at most 64 pin words for $\pi$.

We define $E(\pi) = \{\phi(u)\,|u$ is a strict pin word corresponding to $\pi\} \cup \{v \in M\,|$ there is a quasi strict pin word $u$ corresponding to $\pi$ and $x \in \{LU, LD, RU, RD\} \uplus \{UL, UR, DL, DR\}$ such that $v = x\,\phi(u_2 \dots u_{|u|})\}$. For the second set, the first letter of $\phi(u_2 \dots u_{|u|})$ determines the set in which $x$ lies.

By Remarks 2.1 and 2.2 a., the pin words of $\pi$ are either strict or quasi strict, therefore $|E(\pi)| \leq 64 \times 4 = 256$.

**Theorem 3.8.** *Let $\pi$ be a simple permutation and $w$ be a strict pin word corresponding to a permutation $\sigma$. Then $\pi \not\leq \sigma$ if and only if $\phi(w)$ avoids the finite set of factors $E(\pi)$.*

Notice that it is enough to consider only one strict pin word corresponding to $\sigma$ rather than all of them.

*Proof.* If $\pi \leq \sigma$, then by Lemma 3.2, there is a pin word $u$ corresponding to $\pi$ with $u \preceq w$. By Remarks 2.1 and 2.2 a., $u$ is a strict pin word or a quasi strict pin word. If $u$ is a strict pin word then, by Lemma 3.5, $\phi(u)$ is a factor of $\phi(w)$ so $\phi(w)$ has a factor in $E(\pi)$. If $u$ is a quasi strict pin word then by Lemma 3.6, $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$. Let $x$ be the two letters preceding $\phi(u_2 \dots u_{|u|})$ in $\phi(w)$. As $\phi(w) \in M$, $x\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ that belongs to $E(\pi)$.

Conversely suppose that $\phi(w)$ has a factor $v$ in $E(\pi)$. If $v \in \{\phi(u)\,|u$ is a strict pin word corresponding to $\pi\}$ then by Lemma 3.5, there is a pin word $u$ corresponding to $\pi$ with $u \preceq w$ so by Lemma 3.2, $\pi \leq \sigma$. Otherwise there is a quasi strict pin word $u$ corresponding to $\pi$ and $x \in \{LU, LD, RU, RD, UL, UR, DL, DR\}$ such that

$v = x\,\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$. Thus $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ and by Lemma 3.7, $\pi \leq \sigma$. □

Returning to our motivation with respect to the number of proper pin-permutations in $Av(B)$, the links between pattern containment relation and pin words that we established yield Theorem 3.9.

**Theorem 3.9.** *A wreath-closed class $Av(B)$ has arbitrarily long proper pin-permutations if and only if there exist words of arbitrary length on the alphabet $\{L, R, U, D\}$ avoiding the set of factors $\cup_{\pi \in B} E(\pi) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$.*

*Proof.* The class $Av(B)$ contains arbitrarily long proper pin-permutations if and only if there exist arbitrarily long proper pin-permutations which have no pattern in $B$. That is –making use of Theorem 3.8 and Remark 2.2 b.–, if and only if there exist arbitrarily long strict pin words $w$ such that $\phi(w)$ avoids the set of factors $\cup_{\pi \in B} E(\pi)$, or equivalently if and only if there exist words of arbitrary length on the alphabet $\{L, R, U, D\}$ which avoid the set of factors $\cup_{\pi \in B} E(\pi) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$. □

# 4 From the finiteness problem to a co-finiteness problem

We are now able to give the general algorithm to decide if a wreath-closed permutation class given by its finite basis $B$ contains a finite number of proper pin-permutations (see Algorithm 1).

In this algorithm, $\mathcal{P}_B$ denotes the set of pin words that encode the permutations of $B$, $\mathcal{L}(\mathcal{P}_B)$ the language of words on the alphabet $\{L, R, U, D\}$ which contain as a factor a word of $\cup_{\pi \in B} E(\pi)$ or one of the 8 words $LL, LR, RR, RL, UU, UD, DD$ and $DU$, $\mathcal{A}$ the automaton recognizing $\mathcal{L}(\mathcal{P}_B)$ and $\mathcal{A}^c$ the automaton that recognizes the complementary language of $\mathcal{L}(\mathcal{P}_B)$ in $\{L, R, U, D\}^*$. Notice that $\mathcal{A}^c$ recognizes the words $\phi(w)$ for $w$ a strict pin word encoding a proper pin-permutation $\sigma \in Av(B)$.

**Algorithm 1**: Deciding the finiteness of the number of proper pin-permutations

**input** : a set $B$ of simple permutations
**output**: boolean : true if and only if $Av(B)$ contains only a finite number of proper pin-permutations

$\mathcal{P}_B \leftarrow$PinWords($B$) `// Determine the set of pin words associated to the elements of` $B$
$\mathcal{A} \leftarrow$ Automaton($\mathcal{L}(\mathcal{P}_B)$) `// Build a complete deterministic automaton recognizing` $\mathcal{L}(\mathcal{P}_B)$
**if** $\mathcal{A}^c$ *contains an accessible and co-accessible cycle* **then**
  **return** false
**else**
  **return** true

The first part of this algorithm relies on the function Pinwords (described by Algorithm 2) which computes the pin words associated to a simple permutation. It uses the

fact that the pin representations of a simple permutation, when they exist, are always proper (see Remark 2.1), and that from Lemma 4.3 in [6], the first two pins of a proper pin representation are in *knight position* (*i.e.*, in a configuration like  or one of its 3 symmetries under rotation and reflection). Next from two points in knight position, a proper pin representation, if it exists, can be efficiently computed using the separation condition. Finally it remains to encode the pin representation by pin words.

**Algorithm 2**: PINWORDS function

**input** : a simple permutation $\sigma$
**output**: The set $P$ of pin words encoding $\sigma$

`// Count the number of ordered pairs of points in knight position`
$E \leftarrow \varnothing$;
**foreach** $\sigma_i$ **do**
  $E \leftarrow E \bigcup \{(\sigma_i, \sigma_j)$ in knight position$\}$
`// If more than` 48 `pairs are found,` $\sigma$ `is not a pin-permutation`
**if** $|E| > 48$ **then**
  **return** $\varnothing$
`// Otherwise each knight may be the beginning of a pin representation of` $\sigma$
$P \leftarrow \varnothing$;
**foreach** $(\sigma_i, \sigma_j) \in E$ **do**
  $P \leftarrow P \bigcup \{$ pin words of the pin representation beginning with $(\sigma_i, \sigma_j)\}$
**return** $P$

**Lemma 4.1.** *Algorithm 2 computes the set of pin words encoding a simple permutation $\sigma$ in linear time with respect to the length $n$ of $\sigma$.*

*Proof.* Algorithm 2 can be decomposed into two parts. First, we count the number of ordered pairs of points in knight position that should be smaller than 48. Indeed from Lemma 4.4 of [6], if $\sigma$ is a simple pin-permutation of length $n$, in any of its pin representations $(p_1, \ldots, p_n)$, every unordered pair of points $\{p_i, p_j\}$ that is a knight contains at least one of the points $p_1, p_2$ or $p_n$. As only 8 points can be in knight position with a given point, the permutation $\sigma$ has at most 24 unordered pairs of points in knight position, hence at most 48 ordered pairs $(p_i, p_j)$ that are knights.

Therefore given a simple permutation $\sigma$, we count the number of ordered pairs of points in knight position. To do this, we take each point $p$ of the permutation and we check if another point is in knight position with $p$. As at most 8 cells can contain a point in knight position with $p$, this counting part runs in time $8n$.

If this number is greater than 48, $\sigma$ is not a pin-permutation. Otherwise, the second part of the algorithm computes, for each ordered pair of points in knight position, the pin representation beginning with it (if it exists) and its associated pin words. This can also be done in linear time as there is at most one pin representation of $\sigma$ beginning with a given ordered pair of points. Indeed, because $\sigma$ is simple, its pin representations

are always proper (see Remark 2.1). The pin representation starting with a given knight is then obtained as follows. If $(p_1, \ldots, p_i)$ has already been computed then, since the pin representation we look for is proper, $p_{i+1}$ separates $p_i = \sigma_k$ from previous points. It means that either it separates them vertically, and then $p_{i+1} = \sigma_{k+1}$ or $p_{i+1} = \sigma_{k-1}$, or it separates them horizontally and then its value must be $\sigma_k \pm 1$. Therefore, if we compute $\sigma^{-1}$ in advance (which is easily done by a linear-time precomputation) we are allowed to find the next point in a proper pin representation in constant time.

Finally as at most 8 pin words (choice of the origin, see Figure 3) correspond to a given pin representation, computing all pin words can easily be done in linear time from the pin representation. □

**Lemma 4.2.** *Algorithm 1 tests if a wreath-closed permutation class given by its finite basis $B$ contains a finite number of proper pin-permutations in linear time with respect to $n = \sum_{\pi \in B} |\pi|$.*

*Proof.* First according to Lemma 4.1 the PINWORDS function applied to the $|B|$ patterns of the basis runs in total time $\mathcal{O}(n)$, and produces a set $\mathcal{P}_B$, containing at most $|B| \cdot 48 \cdot 8$ words, whose lengths sum to $\mathcal{O}(n)$.

Next a complete deterministic automaton $\mathcal{A}$ recognizing $\mathcal{L}(\mathcal{P}_B)$ the set of words having a factor in $\cup_{\pi \in B} E(\pi) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$ can be built in linear time (w.r.t. $n$) using Aho-Corasick algorithm [1]. With this construction the number of states of the resulting automaton is also linear. The automaton $\mathcal{A}^c$ that recognizes the complementary language of $\mathcal{L}(\mathcal{P}_B)$ in $\{L, R, U, D\}^*$ is obtained by exchanging final and non-final states of the initial automaton $\mathcal{A}$ which is complete and deterministic. Then it remains to test in the complete deterministic automaton $\mathcal{A}^c$ whether there exists an accessible cycle from which a path leads to a final state (*i.e.*, that is co-accessible). Making use of a depth-first traversal, this step takes a linear time. Hence checking if there exist arbitrarily long words on $\{L, R, U, D\}$ which avoid a finite set of factors can be done in linear time – linear in the sum of the lengths of the factors. Together with Theorem 3.9 this concludes the proof. □

The preceding results allow us to decide in linear time if a wreath-closed permutation class given by its finite basis contains arbitrarily long proper pin-permutations. To end our proof, following the same steps as [9], we must deal with wedge simple permutations and parallel alternations in order to decide if the permutation class contains a finite number of simple permutations. These results are summarized in the following theorem:

**Theorem 4.3.** *Let $Av(B)$ be a finitely based wreath-closed class of permutations. Then there exists an algorithm to decide in time $\mathcal{O}(n \log n)$ where $n = \sum_{\pi \in B} |\pi|$ whether this class contains finitely many simple permutations.*

*Proof.* From Theorem 2.3, we can look separately at parallel alternations, wedge simple permutations and proper pin-permutations. For parallel alternations and wedge simple permutations, Lemma 2.7 shows that testing if their number in $Av(B)$ is finite can be done in $\mathcal{O}(n \log n)$ time. The case of proper pin-permutations can be solved with Algorithm 1. From Lemma 4.2 checking if there exist arbitrarily long proper pin-permutations

in a wreath-closed permutation class can done in linear time – linear in the sum of the lengths of the elements of the basis of the class – concluding the proof. □

**Conjecture** We strongly believe that Theorem 4.3 has a generalization to all finitely based permutation classes (and not only wreath-closed classes): namely, we expect that the complexity of deciding whether a finitely based permutation class contains a finite number of simple permutations is polynomial, however of higher degree.

This complexity gap we foresee for this generalization and the growing importance of wreath-closed classes in the permutation patterns field justify to our eyes the interest of the result proved in this article. Furthermore, in the general case the pattern relation on permutations cannot be translated into a factor relation on the pin words that encode these permutations. However the substitution decomposition of pin-permutations described in [6] should allow us to obtain an efficient recursive algorithm.

**Open problem** By [3], containing a finite number of simple permutations is a sufficient condition for a permutation class to have an algebraic generating function. Our work allows to decide efficiently whether the number of simple permutations in the class is finite, but does not allow the computation of the *set* of simple permutations in the class. Describing an efficient (polynomial?) procedure solving this question, and thereafter being able to compute algorithmically the algebraic generating function associated to the class, would be natural continuations of our work.

**Acknowledgement** The authors wish to thank the referees for their helpful comments and suggestions, that brought our work to a clarified presentation.

## References

[1] Alfred V. Aho and Margaret J. Corasick. Efficient string matching: An aid to bibliographic search. *Communications of the ACM*, 18(6), June 1975.

[2] Michael H. Albert, Robert E. L. Aldred, Mike D. Atkinson, and Derek A. Holton. Algorithms for pattern involvement in permutations. In *ISAAC '01: Proceedings of the 12th International Symposium on Algorithms and Computation*, volume 2223 of *Lecture Notes in Computer Science*, pages 355–366, London, UK, 2001. Springer-Verlag.

[3] Michael H. Albert and Mike D. Atkinson. Simple permutations and pattern restricted permutations. *Discrete Mathematics*, 300(1-3):1–15, 2005.

[4] Michael H. Albert, Steve Linton, and Nik Ruškuc. The insertion encoding of permutations. *Electron. J. Combin.*, 12:Research Paper 47, 31 pp. (electronic), 2005.

[5] Mike D. Atkinson, Nik Ruškuc, and Rebecca Smith. Substitution-closed pattern classes. To appear in *J. Combin. Theory Ser. A*.

[6] Frédérique Bassino, Mathilde Bouvel, and Dominique Rossin. Enumeration of pin-permutations. Technical report, Université Paris Diderot, Université Paris Nord, 2009.

[7] Robert Brignall, Sophie Huczynska, and Vincent Vatter. Decomposing simple permutations, with enumerative consequences. *Combinatorica*, 28(4):385–400, jul 2008.

[8] Robert Brignall, Sophie Huczynska, and Vincent Vatter. Simple permutations and algebraic generating functions. *J. Combin. Theory Ser. A*, 115(3):423–441, 2008.

[9] Robert Brignall, Nik Ruškuc, and Vincent Vatter. Simple permutations: decidability and unavoidable substructures. *Theoret. Comput. Sci.*, 391(1-2):150–163, 2008.

[10] Sergey Kitaev and Toufik Mansour. A survey on certain pattern problems. Preprint available at http://www.ru.is/kennarar/sergey/publications.html, 2003.

[11] Donald E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*. Addison-Wesley, Reading MA, 3rd edition, 1973.

[12] Adam Marcus and Gábor Tardos. Excluded permutation matrices and the Stanley-Wilf conjecture. *J. Comb. Theory, Ser. A*, 107(1):153–160, 2004.

[13] Vincent Vatter. Enumeration schemes for restricted permutations. *Comb. Probab. Comput.*, 17(1):137–159, 2008.