

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, Adeline Pierrot
LIAFA UMR 7089, Université Paris Diderot and CNRS,
Paris, France.

Dominique Rossin
LIX, UMR 7161, Ecole Polytechnique and CNRS,
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 [7] 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 [4], we are able to transform the problem into a co-finiteness problem in a complete deterministic automaton.

1 Introduction

In [8], Knuth introduced pattern avoiding permutation classes. These classes present nice combinatorial properties, for example 231 avoiding permutations are in one-to-one correspondence with Dyck words. It is then natural to extend this enumerative question to all classes and compute, given a set of permutations B , the generating function $S(x) = \sum s_n x^n$ where s_n is the number of permutations of length n avoiding every permutation of B . From the work of Albert and Atkinson [3], many criteria arise giving sufficient conditions to decide the nature of the generating function: rational, algebraic, P-recursive ...

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

In a series of three articles [6, 5, 7] Brignall *et al.* prove that it is decidable to know if a permutation class of finite basis contains a finite number of simple permutations, and hence has an algebraic generating function. Every algorithm involved in this decision procedure is polynomial except the algorithm deciding if the class contains arbitrarily long proper pin-permutations.

In [4] a complete characterization of pin-permutations is given. We use this characterization 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 basis 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 wreath-closed permutation classes, 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 by checking the co-finiteness of a regular language given by a complete deterministic automaton.

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 [6, 7, 4]. 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 $\sigma = 2314$ or its *diagram* (see Figure 1). A permutation $\pi = \pi_1\pi_2 \dots \pi_k$ is a *pattern* of a permutation $\sigma = \sigma_1\sigma_2 \dots \sigma_n$, and we write $\pi \leq \sigma$ if and only if there exist $1 \leq i_1 < i_2 < \dots < i_k \leq n$ such that π is order isomorphic to $\sigma_{i_1} \dots \sigma_{i_k}$. We also say that σ *involves* or *contains* π . If π is not a pattern of σ we say that σ *avoids* π . 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 $\{i, \dots, (i+l)\}$ to $\{j, \dots, (j+l)\}$, except the trivial ones corresponding to $l = 0$ or $i = j = 1$ and $l = n - 1$. When the basis B contains only simple permutations the permutation class $Av(B)$ is said to be *wreath-closed*. In [3] wreath-closed classes are defined in a different way but Albert and Atkinson prove that both definitions are equivalent.

In the following we study wreath-closed classes with finite basis. Our goal is to check whether such a 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, play 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, \dots, p_k) of pins in the plane such that no two points lie in the same column or line and for all $i \geq 2$, p_i lies outside the bounding box of $\{p_1, \dots, p_{i-1}\}$ and respects one of the following conditions:

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

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

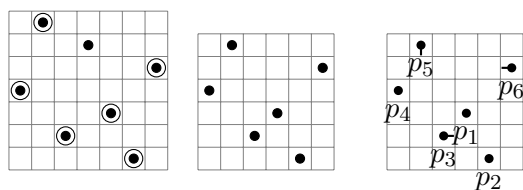


Figure 1: The permutation $\sigma = 4726315$, the pattern $\pi = 462315$ and a pin representation of π . $14L2UR$ (if we place p_0 between p_3 and p_1) and $3DL2UR$ are pin words corresponding to this pin representation.

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

Remark 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 $\{p_1, \dots, p_{i-1}\}$.*

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, \dots, p_n) and choose an arbitrary origin in the plane (we can think of the origin as extending the pin representation to the sequence (p_0, p_1, \dots, p_n) where p_0 is the origin). Then every pin p_1, \dots, 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 $\{p_0, p_1, \dots, p_{i-2}\}$ 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 $\{p_0, p_1, \dots, p_{i-1}\}$ and is situated in the up-right -resp. up-left, bottom-left, bottom-right- corner of the bounding box of $\{p_0, p_1, \dots, p_{i-1}\}$.

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.

To each pin word corresponds a unique pin representation, hence a unique permutation but each pin-permutation of size greater than 1 has at least 8 pin words associated to it -choice of the origin-.

A *strict* pin word is a pin word that begins by a numeral followed only by directions.

Remark 2. *Notice that strict pin words always encode proper pin representations, whereas a proper pin representation is encoded by both strict and non-strict pin words -again, choice of the origin-. Following this idea, another remark is that a pin-permutation is proper if and only if it admits a strict pin word.*

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

$$(1+2+3+4)\left((\epsilon+L+R)(U+D)((L+R)(U+D))^*+(\epsilon+U+D)(L+R)((U+D)(L+R))^*\right).$$

2.2 Some known results

In [7] 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 1. [7] *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 [7] 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 [7], where it is shown that it is easy to deal with this problem using the three following lemmas.

Lemma 1. [7] *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. [7] *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 3. [7] *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 $Av(2134, 2143, 3124, 3142, 3241, 3412, 4123, 4132, 4231, 4312)$.*

Therefore deciding if a class contains an infinite number of wedge simple permutations or parallel alternations is equivalent to checking if elements of the basis involve patterns of size at most 4. Using [2], this test can be made efficient and its exact complexity is given in the proof of Theorem 3 in Section 4.

In [7] Brignall *et al.* also proved that it is decidable to know if a class contains an 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.

In the sequel making use of a characterization we established in [4], we give a polynomial algorithm for deciding if a wreath-closed class of permutations contains finitely many proper pin-permutations.

3 Pattern containment and pin words

In this section we show how to transform the pattern containment relation on permutations into a factor relation between words. 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 ϕ in the sequel) contain some factors.

First recall the definition of the partial order \preceq on pin words introduced in [7]. 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 4. [7] *If the pin word w corresponds to the permutation σ and $\pi \leq \sigma$ then there is a pin word u corresponding to π with $u \preceq w$. Conversely if $u \preceq w$ then the permutation corresponding to u is contained in the permutation corresponding to w .*

As a consequence, if π and σ are permutations and w is a pin word corresponding to σ , then $\pi \leq \sigma$ if and only if there is a pin word u corresponding to π with $u \preceq w$. We apply this result to the set $P(\pi)$ of pin words encoding a simple permutation π , and in the case of a strict pin word w encoding σ . Additionally we show that in this special case, we can associate to each strict (resp. quasi-strict – see definition in Remark 3) pin word $v = v_1v_2 \dots v_n$ 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 check if $\phi(w)$ has a factor in $\phi(P(\pi))$.

Definition 1. *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 ϕ from \mathcal{SP} to M as follows. For any strict pin word $u \in \mathcal{SP}$ such that $u = u' \cdot u''$ with $|u'| = 2$, we set $\phi(u) = \varphi(u') \cdot u''$ where φ 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 ϕ 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 $n + 1$. More precisely, for any $u \in \mathcal{SP}$,

- $u_i = \phi(u)_{i+1}$ for any $i \geq 2$,
- the two first letters of $\phi(u)$ are sufficient to determine the first letter of u (which is a numeral). Thus for a word v of $\{LU, LD, RU, RD, UL, UR, DL, DR\}$ we can by abuse of notation define $\phi^{-1}(v)$.

Notice that our bijection consists in replacing numbers in strict pin words by letters. This implies that we know in which quadrant lies every pin of a pin representation corresponding to the pin word.

Lemma 5. *Let w be a strict pin word and p the pin representation corresponding to w . Then for any $i \geq 2$, w_{i-1} and w_i determine the quadrant in which lies p_i with respect to $\{p_0, \dots, p_{i-2}\}$:*

- if $i \geq 3$, p_i lies in the quadrant $\phi^{-1}(w_{i-1}w_i)$.
- if $i = 2$, p_i lies in the quadrant $\phi^{-1}(BC)$ where $\phi(w_1w_2) = ABC$.

Proof. (Sketch) If $i \geq 3$, w_{i-1} and w_i are letters. 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 letter. 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 $\phi^{-1}(UL) = 2$. \square

Remark 3. *A pin word corresponding to a simple permutation is not always a strict pin word. But by Remark 1, the only other possibility is that it begins with two numerals followed only by directions. We call quasi strict pin words these pin words.*

We first study the case of strict pin words.

Lemma 6. *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 $w = v^{(1)}w^{(1)}v^{(2)}$.

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

Otherwise $w^{(1)}$ begins with a direction hence the first letter of $w^{(1)}$ corresponds to a point 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 : $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$ where $i = |v^{(1)}|$.

If $|v^{(1)}| = i \geq 2$ then according to Lemma 5 we have $u_1 = \phi^{-1}(w_i.w_{i+1})$. We then have $\phi(u) = w_i w_{i+1} \dots w_{i+|u|}$ and $\phi(u)$ is a factor of w which do not contain numeral thus $\phi(u)$ is a factor of $\phi(w)$.

If $|v^{(1)}| = 1$ then from Lemma 5 we have $u_1 = \phi^{-1}(BC)$ where $\phi(w_1w_2) = ABC$ thus $\phi(u) = BCu_2 \dots u_{|u|}$ and $\phi(w) = ABCw_3 \dots w_{|w|}$. As $u_2 \dots u_{|u|} = w_3 \dots w_{|u|+1}$, $\phi(u)$ 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 $|v| = i \geq 2$ then $w = \phi^{-1}(v).\phi(u).v'$. Thus $\phi(u) = w_i \dots w_{i+|\phi(u)|-1} = w_i \dots w_{i+|u|}$ and $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$, $u_1 = \phi^{-1}(w_i w_{i+1})$. u_1 is the quadrant in which w_{i+1} lies hence $u \preceq w$.

If $|v| = 1$ then $\phi(w_1w_2) = v\phi(u)_1\phi(u)_2$ thus according to Lemma 5, w_2 lies in the quadrant $\phi^{-1}(\phi(u)_1\phi(u)_2) = u_1$. As

$$w_3 \dots w_{|u|+1} = \phi(w)_4 \dots \phi(w)_{|u|+2} = \phi(u)_3 \dots \phi(u)_{|u|+1} = u_2 \dots u_{|u|},$$

$u \preceq w$, concluding the proof. \square

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

Lemma 7. *Let u be a quasi strict pin word corresponding to a permutation π and w be a strict pin word corresponding to a permutation σ . 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 $u = u^{(1)}u^{(2)}$. Since $u \preceq w$, w can be decomposed into a sequence of factors $w = v^{(1)}w^{(1)}v^{(2)}w^{(2)}v^{(3)}$. Moreover $|w^{(1)}| = |u^{(1)}| = 1$ so $w^{(2)}$ contains no numerals thus $v^{(2)}$ is nonempty, 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 have no numeral thus $\phi(u^{(2)})$ is a factor of $\phi(w)$ which begin at position $p \geq 3$. \square

Lemma 8. *Let u be a quasi strict pin word corresponding to a permutation π and w be a strict pin word corresponding to a permutation σ . If $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ then π is a pattern of σ .*

Proof. Since $\phi(u^{(2)})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ then by Lemma 6, $u^{(2)} \preceq w$. Let $p_1 \dots p_n$ be a pin representation of w (which corresponds to σ) and Γ be the subset of points corresponding to $u^{(2)}$, then $\Gamma \subseteq \{p_3 \dots p_n\}$. Let π' 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 lies p_1 , and $v = iu^{(2)}$. Then v is a pin word corresponding to π' . As u begins with two numerals, there is $k \in \{1, \dots, 4\}$ such that $u = ku^{(2)}$ and u corresponds to π but $v = iu^{(2)}$ thus v corresponds to π hence $\pi' = \pi$. \square

To any simple permutation π we associate the set of its pin words. From Lemma 4.3, Fact 4.1 and Lemma 4.5 of [4], this set contains at most 64 elements. We define $E(\pi) = \{\phi(u) \mid u \text{ is a strict pin word corresponding to } \pi\} \cup \{v \in M \mid \text{there is a quasi strict pin word } u \text{ corresponding to } \pi \text{ and } x \in \{LU, LD, RU, RD\} \uplus \{UL, UR, DL, DR\} \text{ such that } v = x\phi(u_2 \dots u_{|u|})\}$. For the second set, the first letter of $\phi(u_2 \dots u_{|u|})$ determines the set in which x lies.

It is then immediate to check that $|E(\pi)| \leq 320$.

Theorem 2. *Let π be a simple permutation and w be a strict pin word corresponding to a permutation σ . 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 σ rather than all of them.

Proof. If $\pi \leq \sigma$, then by Lemma 4, there is a pin word u corresponding to π with $u \preceq w$. By Remark 3, u is a strict pin word or a quasi strict pin word. If u is a strict pin word then, by Lemma 6, $\phi(u)$ is a factor of $\phi(w)$ and $\phi(w)$ has a factor in $E(\pi)$. If u is a quasi strict pin word then by Lemma 7, $\phi(u_2 \dots u_{|u|})$ 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 $v \in \{\phi(u) \mid u \text{ is a strict pin word corresponding to } \pi\}$ then by Lemma 6, there is a pin word u corresponding to π with $u \preceq w$ so by Lemma 4, $\pi \leq \sigma$. Otherwise there is a quasi strict pin word u corresponding to π and $x \in \{LU, LD, RU, RD, UL, UR, DL, DR\}$ such that $v = x\phi(u_2 \dots u_{|u|}) \in M$. Thus $\phi(u_2 \dots u_{|u|})$ is a factor of $\phi(w)$ which begins at position $p \geq 3$ and by Lemma 8, $\pi \leq \sigma$. \square

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 the following lemma:

Lemma 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 2 and Remark 2–, 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 $\cup_{\pi \in B} E(\pi) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$. Note that we translate the finiteness result for proper pin-permutations into a finiteness result for strict pin words. The key idea is that for every proper pin-permutation, the number of corresponding strict pin words is bounded. \square

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 $\{L, R, U, D\}^*$.

```

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

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

```

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

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 1), and that from Lemma 4.1 in [4], two first pins of a proper pin representation are in *knight position* (i.e., in a configuration like $\begin{smallmatrix} \square & \square \\ \square & \square \end{smallmatrix}$ or one of its 7 symmetric under rotation and mirror). 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.

```

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

// Count the number of pairs of points in knight position;
 $E \leftarrow \emptyset$ ;
foreach  $\sigma_i$  do
  |  $E \leftarrow E \cup \{(\sigma_i, \sigma_j) \text{ in knight position}\}$ 
end
// If more than 24 pairs are found,  $\sigma$  is not a pin-permutation;
if  $|E| > 24$  then
  | return  $\emptyset$ 
end
// Otherwise each knight may be the beginning of a pin representation of  $\sigma$ ;
 $P \leftarrow \emptyset$ ;
foreach  $(\sigma_i, \sigma_j) \in E$  do
  |  $P \leftarrow P \cup \{\text{pin words of the pin representation beginning with } (\sigma_i, \sigma_j)\}$ 
end
return  $P$ 

```

Algorithm 2: PINWORDS function

Lemma 10. *Algorithm 2 computes the set of pin words encoding a simple permutation in linear time.*

Proof. The algorithm above can be decomposed into two parts. First, we count the number of points in knight position that should be smaller than 24. Indeed from Lemma 4.2 of [4], if σ is a simple pin-permutation, in any of its pin representations (p_1, \dots, p_n) , every knight (p_i, p_j) 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 σ has at most 24 knights.

Therefore given a simple permutation σ , we count the number of 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 24, σ is not a pin-permutation. Otherwise, we compute for each pair of points in knight position the potential associated pin representation.

The second part of the algorithm builds the pin representation beginning with a given knight, 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 σ beginning with a given pair of points. Indeed, because σ is simple, its pin representations are always proper (see Remark 1). The pin representation starting with a given knight is then obtained as follows. If (p_1, \dots, 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 knowing σ^{-1} allows us to find the next point in a proper pin representation in constant time.

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

Lemma 11. *Algorithm 1 tests in linear time if a wreath-closed permutation class given by its finite basis contains a finite number of proper pin-permutations.*

Proof. First according to Lemma 10 the PINWORDS function runs in linear time, and produces a set \mathcal{P}_B , containing at most $|B| \cdot 24 \cdot 8$ words, whose lengths sum to $\mathcal{O}(n)$, with $n = \sum_{\pi \in B} |\pi|$.

Next a complete deterministic automaton \mathcal{A} recognizing 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. 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 sizes of the factors – concluding the proof. \square

The precedent 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 [7], 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 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 1, we can look separately at parallel alternations, wedge simple permutations and proper pin-permutations. For parallel alternations and wedge simple permutations, Lemmas 1, 2 and 3 show that the problem can be solved by 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 pattern avoidance of patterns of length at most 4 can be done in $\mathcal{O}(n \log n)$, 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 is finite.

The case of proper pin-permutations can be solved with Algorithm 1. From Lemma 11 checking if there exist arbitrarily long proper pin-permutations in a wreath-closed permutation class can be done in linear time – linear in the sum of the lengths of the elements of the basis of the class – concluding the proof. \square

Conjecture We highly believe that Theorem 3 can be generalized to all finitely based permutation classes (and not only wreath-closed classes). The methods used here seem to extend to the general case using the results of [4]. This is a work in progress.

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.

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] Frédérique Bassino, Mathilde Bouvel, and Dominique Rossin. Enumeration of pin-permutations. Technical report, Université Paris Diderot, Université Paris Nord, 2009.
- [5] Robert Brignall, Sophie Huczynska, and Vincent Vatter. Decomposing simple permutations, with enumerative consequences. *Combinatorica*, 28(4):385–400, jul 2008.
- [6] Robert Brignall, Sophie Huczynska, and Vincent Vatter. Simple permutations and algebraic generating functions. *J. Combin. Theory Ser. A*, 115(3):423–441, 2008.
- [7] Robert Brignall, Nik Ruškuc, and Vincent Vatter. Simple permutations: decidability and unavoidable substructures. *Theoret. Comput. Sci.*, 391(1-2):150–163, 2008.
- [8] Donald E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*. Addison-Wesley, Reading MA, 3rd edition, 1973.