TRANSACTIONS OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 373, Number 5, May 2020, Pages 3085–3115
https://doi.org/10.1090/tran/7915
Article electronically published on February 20, 2020

ON THE COMPUTATIONAL COMPLEXITY OF ALGEBRAIC NUMBERS: THE HARTMANIS–STEARNS PROBLEM REVISITED

BORIS ADAMCZEWSKI, JULIEN CASSAIGNE, AND MARION LE GONIDEC

Abstract. We consider the complexity of integer base expansions of algebraic irrational numbers from a computational point of view. A major contribution in this area is that the base-b expansion of algebraic irrational real numbers cannot be generated by finite automata. Our aim is to provide two natural generalizations of this theorem. Our main result is that the base-b expansion of algebraic irrational real numbers cannot be generated by deterministic pushdown automata. Incidentally, this completely solves the Hartmanis–Stearns problem for the class of multistack machines. We also confirm an old claim of Cobham from 1968 proving that such real numbers cannot be generated by tag machines with dilation factor larger than one. In order to stick with the modern terminology, we also show that the latter generate the same class of real numbers as morphisms with exponential growth.

1. Introduction

An old source of frustration for mathematicians arises from the study of integer base expansions of classical constants like

$$\sqrt{2} = 1.414\,213\,562\,373\,095\,048\,801\,688\,724\,209\,698\,078\,569\cdots$$

or

$$\pi = 3.141\,592\,653\,589\,793\,238\,462\,643\,383\,279\,502\,884\,197\cdots.$$

While these numbers admit very simple geometric descriptions, a close look at their digital expansion suggests highly complex phenomena. Over the years, different ways have been envisaged to formalize this old problem. This recurring theme appeared in particular in three fundamental papers, using the language of probability according to É. Borel [17], the language of dynamical systems according to Morse and Hedlund [37], and the language of Turing machines according to Hartmanis and Stearns [30]. Each of these points of view leads to a different assortment of challenging conjectures. As the title of this paper suggests, the present work focuses on the latter approach. It is addressed to researchers interested both in number theory and theoretical computer science. In this respect, we took care to make the paper as self-contained as possible, and hopefully readable by members from these different communities.

Received by the editors October 3, 2017, and, in revised form, October 31, 2018, and March 13, 2019.

2010 *Mathematics Subject Classification.* Primary 11B85, 11J81, 68Q45, 68R15, 11J87.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under the Grant Agreement No 648132.

After the seminal work of Turing [47], real numbers can be coarsely divided into two classes. On one side we find the computable real numbers, those whose base-b expansion can be produced by a Turing machine, while on the other side lie uncomputable real numbers, which will remain forever beyond the ability of computers. Note that, though most real numbers belong to the second class, classical mathematical constants are usually computable. This is, in particular, the case for any algebraic number. However, among computable numbers, some are quite easy to compute, while others seem to have an inherent complexity that makes them difficult to compute. In 1965, Hartmanis and Stearns [30] investigated the fundamental question of how hard a real number may be to compute, introducing the now classical time complexity classes. The notion of time complexity takes into account the number $T(n)$ of operations needed by a multitape deterministic Turing machine to produce the first n digits of the expansion. In this regard, a real number is considered all the more simple as its base-b expansion can be produced very quickly by a Turing machine. At the end of their paper, Hartmanis and Stearns suggested the following problem.

Problem HS. Do there exist irrational algebraic numbers for which the first n binary digits can be computed in $O(n)$ operations by a multitape deterministic Turing machine?

Let us briefly recall why Problem HS is still open and likely difficult to solve. On the one hand, all known approaches to compute the base-b expansion of algebraic irrational numbers efficiently intimately rely on the cost of the multiplication $M(n)$ of two n-digit numbers (see, for instance, [18]). This operation is computable in quasilinear time, that is, computable in $O(n\log^j n)$ operations for some positive integer j. However, to determine whether $M(n) = O(n)$ remains a famous open problem in this area. On the other hand, a negative answer to Problem HS, which may be the less surprising issue according to [26], would contain a powerful statement about transcendence. A very special instance is the transcendence of the three simple irrational real-time computable numbers

$$\sum_{n=1}^{\infty}\frac{1}{2^{n!}}\,,\ \sum_{n=1}^{\infty}\frac{1}{2^{n^2}}\,,\ \text{and}\ \sum_{n=1}^{\infty}\frac{1}{2^{n^3}}\,\cdot$$

Of course, for the first one, Liouville's inequality [34] easily does the job. But the transcendence of the second number only dates back to 1996 [16, 28] and its proof requires the deep work of Nesterenko on the algebraic independence of values of Eisenstein series [39]. Finally, the transcendence of the third number remains unknown.

In 1968, Cobham [26] (also see [24, 25]) was the first to consider the restriction of the Hartmanis-Stearns problem to some classes of Turing machines. The model of computation he originally investigated is the so-called *tag machine*. Though this model has some historical interest, this terminology is not much used today by the computer science community. However, the class of sequences output by tag machines precisely corresponds to the class of *morphic sequences*, a well-known object of interest for both mathematicians and computer scientists. The latter are especially used in combinatorics on words and symbolic dynamics (see, for instance, [12, 42, 43]). The equivalence between the class of sequences output by tag machines and the class of morphic sequences follows from [26, 27]. See also, for instance, [12, Chapter 7] concerning the more modern terminology of morphic sequences. In what follows, we will present our result using the terminology of morphic sequences but, for the interested reader, we describe the connection with tag machines in Section 5.1.

In his paper, Cobham stated two main theorems without proof and only gave some hints that these statements should be deduced from a general transcendence method based on functional equations, now known as Mahler's method. His first claim was finally confirmed by the first author and Bugeaud [3], but using a totally different approach based on a p-adic version of the subspace theorem (see [3, 8]). Very recently, some advances in Mahler's method [10, 41] permitted completion of the proof originally envisaged by Cobham.

Theorem AB (Cobham's first claim)**.** *The base-b expansion of an algebraic irrational number cannot be generated by a uniform morphism or, equivalently, by a finite automaton.*

For definitions of real numbers generated by finite automata and by uniform morphisms, we refer the reader to Definition 2.3 and Section 2.2.

Remark 1.1. Theorem AB actually refers to two conceptually quite different models of computation: uniform morphisms and finite automata. There are two natural ways a multitape deterministic Turing machine can be used to define computable numbers. First, it can be considered as an *enumerator*, which means that the machine produces one by one all the digits of a real number ξ in a given base b on its output tape. Problem HS originally referred to the model of enumerators. In the other model, referred to as a *Turing transducer*, the machine is fed with some input representing a positive integer n and has to produce on its output tape the nth digit in the base-b expansion of ξ. In Theorem AB, uniform morphisms (or originally uniform tag machines) are enumerators while finite automata are used as transducers. Note that, used as enumerators, finite automata can only produce eventually periodic sequences of digits and thus rational numbers (see, for instance, [12, Theorem 5.7.1]). In contrast, used as transducers, finite automata output the interesting class of automatic sequences (see [12]). The fact that these two models are equivalent is due to Cobham [27].

Theorem AB is the main contribution to date toward a negative solution to Problem HS. In this paper, we show that the approach developed in [3, 8] leads to two interesting generalizations of this result. Our first generalization is concerned with enumerators. In this direction, we confirm the second claim of Cobham.

Theorem 1.2 (Cobham's second claim)**.** *The base-b expansion of an algebraic irrational number cannot be generated by a morphism with exponential growth.*

For the definition of a real number generated by a morphism with exponential growth, we refer the reader to Definition 2.6. We stress that, as stated above, Theorem 1.2 also appeared in the Thèse de Doctorat of Julien Albert [11]. In order to provide a self-contained proof of Cobham's second claim, which was originally formulated in terms of tag machines, we will complete and reprove with permission some content of [11] in Section 4.1. The fact that Theorem 1.2 is equivalent to Cobham's second claim will be proved in Section 5.1.

Our second and main generalization of Theorem AB is concerned with transducers. We consider a classical computation model called the *deterministic pushdown* *automaton*. It is of great importance on the one hand for theoretical aspects because of Chomsky's hierarchy [23] in formal language theory, and on the other for practical applications, especially in parsing (see for instance [31,46]). Roughly, such a device is a finite automaton with an unbounded memory organized as a stack, that is, as an LIFO data structure. The acronym LIFO stands for Last-In-First-Out and reflects the way symbols can be stored and removed from the stack. In what follows, all stacks are LIFO. Our main result is the following.

Theorem 1.3. *The base-b expansion of an algebraic irrational number cannot be generated by a deterministic pushdown automaton.*

For the definition of a real number generated by a deterministic pushdown automaton, we refer the reader to Definition 2.12. In Section 5.2, we use Theorem 1.3 to revisit the Hartmanis–Stearns problem as follows: instead of a time constraint, we impose a restriction based on the way the memory may be stored by Turing machines. This leads us to consider a classical computation model called a *multistack machine*. It corresponds to a version of the deterministic Turing machine where the memory is simply organized as stacks. It is as general as the Turing machine if one allows two or more stacks. Furthermore the one-stack machine turns out to be equivalent to the deterministic pushdown automaton, while a zero-stack machine is just the finite automaton of Theorem AB, that is, a machine with a strictly finite memory only stored in the finite-state control. Incidentally, Theorems AB and 1.3 turn out to completely solve the Hartmanis–Stearns problem for multistack machines. Our approach also provides a method to prove the transcendence of some real numbers generated by linearly bounded Turing machines (see the example in Section 5.3).

This paper is organized as follows. Definitions related to finite automata, morphisms, and pushdown automata are given in Section 2. The useful combinatorial transcendence criterion of [8], on which our results are based, is recalled in Section 3. Section 4 is devoted to the proofs of Theorems 1.2 and 1.3. In connection with these results, two models of computation are discussed in Section 5: the tag machine and the multistack machine. Finally, Section 6 is devoted to concluding remarks regarding factor complexity, some quantitative aspects of this method, and continued fractions.

2. Finite automata, morphic sequences, and pushdown automata

In this section, we give definitions for a real number to be generated by a finite automaton, a morphism, and a deterministic pushdown automaton. This provides a precise meaning to Theorems AB, 1.2, and 1.3.

Throughout this paper, we will use the following notation. An alphabet A is a finite set of symbols, also called letters. A finite word over A is a finite sequence of letters in A or, equivalently, an element of A^*, the free monoid generated by A. The length of a finite word W, that is, the number of symbols in W, is denoted by $|W|$. We let ϵ denote the empty word, the neutral element of A^*. We let A^+ denote the set of finite words of positive length over A. If a is a letter and W a finite word, then $|W|_a$ stands for the number of occurrences of the letter a in W. Let $k \geq 2$ be a natural number. We let Σ_k denote the alphabet $\{0, 1, \dots, k-1\}$. Given a positive integer n, we set $\langle n \rangle_k := w_r w_{r-1} \cdots w_0$ for the canonical base-k expansion of n (written from most to least significant digit), which means that

$n = \sum_{i=0}^{r} w_i k^i$ with $w_i \in \Sigma_k$ and $w_r \neq 0$. Note that by convention $\langle 0 \rangle_k := \epsilon$. Let $\mathcal{R}_k := (\Sigma_k \setminus \{0\})\,\Sigma_k^*$ denote the language of all canonical base-k expansion of positive integers. For every positive integer n, there is a unique word w in $\mathcal{R}_k$ such that $\langle n \rangle_k = w$. Conversely, if $w := w_0 \cdots w_r$ is a finite word over the alphabet Σ_k, we set $[w]_k := \sum_{i=0}^{r} w_{r-i} k^i$. Given a real number $\xi \in [0,1)$, we let $\langle \xi \rangle_k := 0.a_1 a_2 \cdots$ denote the canonical base-k expansion of ξ, which means that $\xi = \sum_{i=1}^{+\infty} a_i k^{-i}$ with $a_i \in \Sigma_k$ and $a_i \neq k-1$ for infinitely many indices i. The usual notation $\{x\}$, $\lfloor x \rfloor$, and $\lceil x \rceil$, respectively, stands for the fractional part, the floor, and the ceiling of the real number x.

2.1. **Finite automata and automatic sequences.** A sequence $\mathbf{a} := (a_n)_{n \geq 0}$ with values in a finite set is *k-automatic* if it can be generated by a finite automaton used as a transducer. This means that there exists a finite-state machine (a deterministic finite automaton with output) that takes as input the base-k expansion of n and produces as output the symbol a_n. We use the following convention. Inputs are read from left to right, that is, starting from the most significant digit. In this paper, this convention is used for all machines which are used as a transducer (k-automata, k-pushdown automata, k-multistack machines, one-way transducer-like k-machines).

Let us give now a formal definition of a k-automatic sequence. Let $k \geq 2$ be a natural number. A k-automaton is defined as a 6-tuple $\mathcal{A} := (Q, \Sigma_k, \delta, q_0, \Delta, \tau)$, where

- Q is a finite set called the *set of states*,
- Σ_k is called the *set of input symbols*,
- $\delta : Q \times \Sigma_k \to Q$ is called the *transition function*,
- $q_0 \in Q$ is called the *initial state*,
- Δ is a finite set called the *set of output symbols*,
- $\tau : Q \to \Delta$ is called the *output function*.

Given a state q in Q and a finite word $w := w_1 w_2 \cdots w_n$ over the alphabet Σ_k, we define $\delta(q, w)$ recursively by $\delta(q, \epsilon) = q$ and $\delta(q, w) = \delta(\delta(q, w_1 w_2 \cdots w_{n-1}), w_n)$.

Definition 2.1. Let $\mathcal{A} := (Q, \Sigma_k, \delta, q_0, \Delta, \tau)$ be a k-automaton. The output sequence produced by $\mathcal{A}$ is the sequence $(\tau(\delta(q_0, \langle n \rangle_k)))_{n \geq 0}$. Such a sequence is called a *k-automatic sequence.* A sequence or an infinite word is said to be automatic if it is k-automatic for some integer $k \geq 2$.

Example 2.2. By a classical result of Lagrange, it is known that every non-negative integer can be written as the sum of four perfect squares. It is optimal in the sense that some natural numbers cannot be written as the sum of only three squares. More precisely, Legendre proved that

$$\exists\, a, b, c \mid n = a^2 + b^2 + c^2 \iff \nexists\, i, j \mid n = 4^i(8j+7)\,,$$

where n, a, b, c, i, j are non-negative integers. As a consequence, the binary sequence $\mathbf{s} := (s_n)_{n \geq 0}$ defined by $s_n = 1$ if n can be written as the sum of three squares and $s_n = 0$ otherwise is 2-automatic. See Figure 2.1 for a finite automaton generating the sequence $\mathbf{s}$.

Definition 2.3. A real number ξ can be generated by a deterministic k-automaton $\mathcal{A}$ if, for some integer $b \geq 2$, one has $\langle \{\xi\} \rangle_b = 0.a_1 a_2 \cdots$, where $(a_n)_{n \geq 0}$ corresponds to the output sequence produced by $\mathcal{A}$.

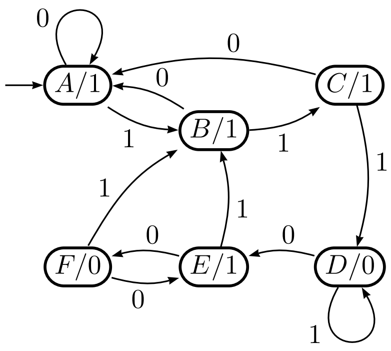

FIGURE 2.1. A 2-automaton generating the sequence $\mathbf{s}$.

Remark 2.4. Note that a_0 is not used in Definition 2.3. We could also have required that $(a_{n+1})_{n\geq 0}$ be produced by $\mathcal{A}$. As the set of k-automatic sequences is invariant under shift [12, Corollary 6.8.5], this would not change the class of real numbers generated by deterministic k-automata.

Theorem AB thus implies that the binary number

$$\xi_0 := 1.111\,111\,011\,111\,110\,111\,111\,101\,111\,011\,011\,111\,110\cdots$$

generated by the 2-automaton of Figure 2.1 is transcendental.

2.2. **Morphic sequences.** Here we recall some basic definitions. Let A be a finite alphabet. A map from A to A^* naturally extends to a map from A^* into itself called an (endo)*morphism*. Given two alphabets A and B, a map from A to B naturally extends to a map from A^* into B^* called a *coding* or *letter-to-letter morphism*. A morphism σ over A is said to be *k-uniform* if $|\sigma(a)| = k$ for every letter a in A, and just *uniform* if it is k-uniform for some k. A useful object associated with a morphism σ is the so-called *incidence matrix* of σ, denoted by M_σ. We first need to choose an ordering of the elements of A, say $A = \{a_1, a_2, \ldots, a_d\}$, and then M_σ is defined by

$$\forall i, j \in \{1, \ldots, d\}, \quad (M_\sigma)_{i,j} := |\sigma(a_j)|_{a_i}\,.$$

The choice of the ordering has no importance. A morphism σ over A is said to be *prolongable* on a if $\sigma(a) = aW$ for some word W and if the length of the word $\sigma^n(a)$ tends to infinity with n. Then the word

$$\sigma^\omega(a) := \lim_{n\to\infty} \sigma^n(a) = aW\sigma(W)\sigma^2(W)\cdots$$

is the unique fixed point of σ that begins with a. An infinite word obtained by iterating a prolongable morphism σ is said to be *pure morphic*. The image of a pure morphic word under a coding is a *morphic word*. Thus, to define a morphic word $\mathbf{a}$, one needs a 5-tuple $\mathcal{T} := (A, \sigma, a, B, \varphi)$ such that $\mathbf{a} = \varphi(\sigma^\omega(a))$, where

- A is a finite set of symbols called the *internal alphabet*,
- a is an element of A called the *starting symbol*,
- σ is a *morphism* of A^* prolongable on a,
- B is a finite set of symbols called the *external alphabet*,
- φ is a letter-to-letter morphism from A to B.

When σ is uniform (resp., k-uniform), the sequence $\mathbf{a}$ is said to be generated by a uniform (resp., k-uniform) morphism.

Definition 2.5. A morphism σ is said to have *exponential growth* if the spectral radius of the matrix M_σ is larger than one. When σ has exponential growth and all letters of A appear in $\sigma^\omega(a)$, the sequence $\mathbf{a} := \varphi(\sigma^\omega(a))$ is said to be generated by a morphism with exponential growth.

Definition 2.6. A real number ξ can be generated by a morphism with exponential growth if, for some integer $b \geq 2$, one has $\langle\{\xi\}\rangle_b = 0.a_1a_2\cdots$, where $\mathbf{a} := (a_n)_{n\geq 0}$ can be generated by a morphism with exponential growth.

Theorem 1.2 thus implies the transcendence of the ternary number

$$\xi_1 := 0.212\,012\,202\,101\,222\,021\,201\,202\,101\,222\,202\,121\,22\cdots$$

whose expansion is the sequence $\mathbf{a} := \varphi_1(\sigma_1^\omega(a))$, where $\sigma_1(a) = acb$, $\sigma_1(b) = abc$, $\sigma_1(c) = c$, $\varphi_1(a) = 0$, $\varphi_1(b) = 1$, and $\varphi_1(c) = 2$. One can check that σ_1 has exponential growth for the spectral radius of M_{σ_1} is 2. In contrast, Theorem 1.2 does not apply to the binary number $\sum 2^{-n^2}$. Though this number can be generated by the morphism σ_2 defined below, the latter has non-exponential growth. Indeed the characteristic sequence of squares can be obtained as $\varphi_2(\sigma_2^\omega(a))$ where $\sigma_2(a) = ab$, $\sigma_2(b) = ccb$, $\sigma_2(c) = c$, $\varphi_2(a) = \varphi_2(b) = 1$, and $\varphi_2(c) = 0$. One can check easily that the spectral radius of M_{σ_2} is equal to 1.

Remark 2.7. Following Cobham [26], there is no loss of generality to assume that the internal morphism σ is a non-erasing morphism, which means that no letter is mapped to the empty word. Indeed, if an infinite word can be generated by an erasing morphism, then there also exists a non-erasing morphism that can generate it. From now on, we will only consider non-erasing morphisms.

It is worth mentioning that the class of sequences generated by uniform morphisms is especially relevant because of the following result of Cobham [27].

Proposition C. *A sequence is k-automatic if and only if it can be generated by some k-uniform morphism.*

Furthermore, the proof of Proposition C is completely constructive and provides a simple way to go from k-uniform morphisms to k-automata and vice versa. This general feature is exemplified below. For a complete treatment see [27] or [12, Chapter 6].

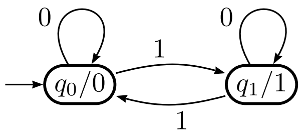
$$\begin{tikzcd} \arrow[r] & q_0/0 \arrow[loop above, "0"] \arrow[r, bend left, "1"] & q_1/1 \arrow[loop above, "0"] \arrow[l, bend left, "1"] \end{tikzcd}$$

FIGURE 2.2. A 2-automaton generating Thue–Morse sequence.

Example 2.8. The Thue–Morse sequence $\mathbf{t} := (t_n)_{n\geq 0}$ is probably the most famous example among automatic sequences. It is defined as follows: $t_n = 0$ if the sum of the binary digits of n is even, and $t_n = 1$ otherwise. It can be generated by the following finite 2-automaton, represented in Figure 2.2: $\mathcal{A} := (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{0, 1\}, \tau)$, where $\delta(q_0, 0) = \delta(q_1, 1) = q_0$, $\delta(q_0, 1) = \delta(q_1, 0) = q_1$, $\tau(q_0) = 0$ and $\tau(q_1) = 1$. The Thue–Morse sequence can as well be generated by a 2-uniform morphism, for one has $\mathbf{t} = \varphi(\sigma^\omega(q_0))$, where $\sigma(q_0) = q_0q_1$, $\sigma(q_1) = q_1q_0$, $\varphi(q_0) = 0$, $\varphi(q_1) = 1$.

2.3. **Pushdown automata.**

A pushdown automaton is a classical device, but most often used in formal language theory as an acceptor, that is, a machine that can accept or reject finite words and thus defines languages, namely context-free languages (see, for instance, [13, 14, 31]). In particular, a classical reference about deterministic context-free languages and deterministic pushdown automata is [31, Chapter 10]. Our point of view here is slightly different, for we use the pushdown automaton as a transducer, that is, a machine that associates a symbol with every finite word over a given input alphabet.

Formally, a k-pushdown automaton is a complete deterministic pushdown automaton with output, or DPAO for short. It is defined as a 7-tuple $\mathcal{M} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$, where

- Q is a finite set called the *set of states*.
- $\Sigma_k := \{0, 1, \ldots, k-1\}$ is called the *set of input symbols*.
- Γ is a finite set called the *set of stack symbols*.
  For convenience, a special symbol $\#$ denotes the empty word of Γ^*, i.e., the empty stack.
  The set $Q \times \Gamma^*$ is called the *set of (internal) configurations*.
- $\delta : E \subset Q \times (\Gamma \cup \{\#\}) \times (\Sigma_k \cup \{\epsilon\}) \to Q \times \Gamma^*$ is called the *transition function*.
- $q_0 \in Q$ is called the *initial state* and $(q_0, \#)$ is called the *initial configuration*.
- Δ is a finite set called the *set of output symbols*.
- $\tau : Q \times (\Gamma \cup \{\#\}) \to \Delta$ is called the *output function*.

Furthermore, the transition function satisfies the following conditions:

- *Determinism assumption*: If (q, s, ϵ) belongs to E for some $(q, s) \in Q \times (\Gamma \cup \{\#\})$, then no (q, s, a) with $a \in \Sigma_k$ belongs to E. In this case we say that $\mathcal{M}$ has an ϵ-move from internal configuration (q, s).
- *Completeness assumption*: If (q, s, ϵ) does not belong to E for some $(q, s) \in Q \times (\Gamma \cup \{\#\})$, then $\{q\} \times \{s\} \times \Sigma_k \subset E$.
- *Finiteness assumption on ϵ-moves*: There does not exist an infinite sequence of internal configurations $(q_i, S_i)_{i \geq 1}$ such that $\delta(q_i, s_i, \epsilon) = (q_{i+1}, X_i)$ for all $i \geq 1$, where s_i is the first symbol in S_i, or $\#$ if S_i is empty, $S_i = S_i' s_i$, and $S_{i+1} = S_i' X_i$. This assumption is in particular satisfied when all ϵ-moves decrease the stack height (see Remark 2.13).

Remark 2.9. Notice that δ being a function is also a part of the determinism assumption. In a non-deterministic k-pushdown automaton, δ would be defined as a subset of $Q \times (\Gamma \cup \{\#\}) \times (\Sigma_k \cup \{\epsilon\}) \times Q \times \Gamma^*$.

We now want to make sense of the computation $\tau(\delta(q_0, \#, W))$ for any input word W in Σ_k^*. First, the transition function δ of a k-pushdown automaton can naturally be extended to a subset of $Q \times \Gamma^* \times (\Sigma_k \cup \{\epsilon\})$ by setting

$$\forall S \in \Gamma^*,\ \delta(q, Ss, a) = (q', SX)$$

when $\delta(q,s,a) = (q',X)$ with $s \in \Gamma$ (i.e., $s \neq \#$). After reading the symbol a, the pushdown automaton could reach a configuration (q,S) from which ϵ-moves are possible. In such a case, the determinism assumption forces the pushdown automaton to perform all possible ϵ-moves before reading the next input symbol, and the finiteness assumption ensures that this will eventually stop. We stress that this appears to be a classical convention (see the discussion in [14, Section 5.2]) which can be formalized by using, instead of δ, the function $\overline{\delta}$ defined as follows:

$$\begin{cases} \overline{\delta}(q,S,a) = \overline{\delta}(\delta(q,S,a),\epsilon); & \\ \overline{\delta}(q,S,\epsilon) = (q,S) & \text{if } (q,S,\epsilon) \notin E; \\ \overline{\delta}(q,S,\epsilon) = \overline{\delta}(\delta(q,S,\epsilon),\epsilon) & \text{if } (q,S,\epsilon) \in E. \end{cases}$$

Then $\overline{\delta}$ can be extended to a subset of $Q \times \Gamma^* \times \Sigma_k^*$ by setting

$$\overline{\delta}(q,S,wa) = \overline{\delta}\left(\overline{\delta}(q,S,w),a\right) .$$

This means in particular that $\mathcal{M}$ scans its inputs from left to right. We also extend the output function τ to $Q \times \Gamma^*$ by simply setting $\tau(q,Ss) = \tau(q,s)$ for $q \in Q$, $s \in \Gamma$, and $S \in \Gamma^*$.

Definition 2.10. Let $\mathcal{M} := (Q,\Sigma_k,\Gamma,\delta,q_0,\Delta,\tau)$ be a k-pushdown automaton. The sequence $(\tau(\overline{\delta}(q_0,\#,\langle n\rangle_k)))_{n\geq 0}$ is called the output sequence produced by $\mathcal{M}$.

This class of sequences is discussed in [22]. They form a subclass of the context-free sequences (see [22, 38]).

Example 2.11. Usually a deterministic pushdown automaton is represented as a finite graph whose vertices are labelled by the elements of Q and whose edges are labelled by transitions as follows: $\delta(q,s,a) = (q',X)$ is represented by the edge $q \xrightarrow{(a,s|X)} q'$. An example of such internal representation is given by the 2-pushdown automaton $\mathcal{A}$ in Figure 2.3. It outputs the binary sequence

$$\mathbf{a} := 11101110011010000111111011101000000010110\cdots$$

whose nth binary digit is 1 if the difference between the number of occurrences of the digits 0 and 1 in the binary expansion of n is at most 1, and is 0 otherwise.

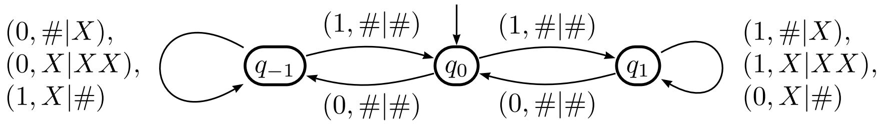

FIGURE 2.3. A 2-pushdown automaton producing the binary expansion of ξ_2.

This automaton works as follows. Being in state q_0 means that the part of the input word that has been already read contains as many 1's as 0's. On the other hand, being in state q_1 means that the part of the input word that has been already read contains more 1's than 0's, while being in state q_{-1} means that it contains more 0's than 1's. Furthermore, in any of these two states, the difference between the number of 0's and 1's (in absolute value) is one more than the number of X's in the stack. Thus, the difference between the number of occurrences of the symbols 1 and 0 in the input word is at most 1 if and only if the reading ends with an empty stack (regardless of the ending state). Choosing the output function accordingly, we see that $\mathcal{A}$ produces the infinite word **a**.

Note that, formally, the pushdown automaton described above is defined as $\mathcal{A} := (\{q_0, q_1, q_{-1}\}, \Sigma_2, \{X\}, \delta, q_0, \{0,1\}, \tau)$, where the transition function δ is given by

$$\begin{array}{lll}
\delta(q_0,\#,0) = (q_{-1},\#), & \delta(q_1,\#,0) = (q_0,\#), & \delta(q_{-1},\#,0) = (q_{-1},X),\\
\delta(q_0,\#,1) = (q_1,\#), & \delta(q_1,\#,1) = (q_1,X), & \delta(q_{-1},\#,1) = (q_0,\#),\\
\delta(q_0,X,0) = (q_0,\#), & \delta(q_1,X,0) = (q_1,\#), & \delta(q_{-1},X,0) = (q_{-1},XX),\\
\delta(q_0,X,1) = (q_0,\#), & \delta(q_1,X,1) = (q_1,XX), & \delta(q_{-1},X,1) = (q_{-1},\#),
\end{array}$$

and where the output function τ is defined by

$$\tau(q_0,\#) = \tau(q_1,\#) = \tau(q_{-1},\#) = 1$$

and

$$\tau(q_0,X) = \tau(q_1,X) = \tau(q_{-1},X) = 0\,.$$

The two transitions from q_0 to q_0 with stack symbol X ensure completeness of the automaton, but they are never used in actual computations, so we omitted them in Figure 2.3.

Definition 2.12. A real number ξ can be generated by a deterministic k-pushdown automaton $\mathcal{M}$ if, for some integer $b \geq 2$, one has $\langle\{\xi\}\rangle_b = 0.a_1a_2\cdots$, where $(a_n)_{n\geq 0}$ corresponds to the output sequence produced by $\mathcal{M}$.

Theorem 1.3 thus implies the transcendence of the binary number

$$\xi_2 := 1.110\,111\,001\,101\,000\,011\,111\,101\,110\,100\,000\,010\,110\cdots$$

whose binary expansion is the infinite word **a** of Example 2.11.

Remark 2.13. *About ϵ-moves.*— Since we only consider deterministic pushdown automata, we can assume without loss of generality that all ϵ-moves decrease the stack height, that is, $\delta(q,s,\epsilon) = (q',X)$ implies $s \neq \#$ and $X = \#$ (see, for instance, [14, Proposition 5.4], or [31, Exercise 10.2(a)]).

Normal forms.— Further assumptions may be made without loss of generality. For instance, we can assume that the transition function has only moves of the form $\delta(q,s,a) = (q',\#)$, known as pop moves, and of the form $\delta(q,s,a) = (q',ss')$, with $s' \in \Gamma$, known as push moves (see [31, Lemma 10.2]).

About input words.— In our model of k-pushdown automaton, we choose to feed our machines only with the canonical base-k expansion of each non-negative integer n. Instead, we could as well imagine to ask that $\tau(\overline{\delta}(q_0,\#,w))$ remain the same for all words $w \in \Sigma_k^*$ such that $[w]_k = n$, that is, $\tau(\overline{\delta}(q_0,\#,w)) = \tau(\overline{\delta}(q_0,\#,0^jw))$ for every natural number j. Such a change would not affect the class of output sequences produced by k-pushdown automata. The discussion is similar to the case of the k-automaton and we refer to [12, Chapter 5] for more details.

Our second remark concerning inputs is more important. In our model, the k-pushdown automaton scans the base-k expansion of a positive integer n starting from the most significant digit. This corresponds to the usual way humans read numbers, that is from left to right. In the case of the k-automaton, this choice is of no consequence because both ways of reading are known to be equivalent. However, this is no longer true for k-pushdown machines, as the class of deterministic context free languages is not closed under reversal (see, for instance, [31, p. 281]).

About uniqueness.— There always exist several different k-pushdown automata producing the same output. In particular, it is possible to choose one with a single state (by encoding the state into the stack). The 2-automaton given in Figure 2.3 is certainly not the smallest one with respect to the number of states, but it makes the process of computation more transparent and it only uses one ordinary stack symbol.

3. A combinatorial transcendence criterion

In this section, we recall the fundamental relation between Diophantine approximation and repetitive patterns occurring in integer base expansions of real numbers.

Let A be an alphabet and let W be a finite word over A. For any positive integer m, we write W^m for the word

$$\underbrace{W \cdots W}_{m \text{ times}}$$

(the concatenation of the word W repeated m times). More generally, for any positive real number x, W^x denotes the word $W^{\lfloor x \rfloor} W'$, where W' is the prefix of W of length $\lceil \{x\} |W| \rceil$. The following natural measure of periodicity for infinite words was introduced in [4] (also see [1,9]).

Definition 3.1. The *Diophantine exponent* of an infinite word $\mathbf{a}$ is defined as the supremum of the real numbers ρ for which there exist arbitrarily long prefixes of $\mathbf{a}$ that can be factorized as UV^α, where U and V are two finite words (U possibly empty) and α is a real number such that

$$\frac{|UV^\alpha|}{|UV|} \geq \rho.$$

The Diophantine exponent of $\mathbf{a}$ is denoted by $\mathrm{dio}(\mathbf{a})$.

Of course, for any infinite word $\mathbf{a}$ one has the following relation:

$$1 \leq \mathrm{dio}(\mathbf{a}) \leq +\infty.$$

Furthermore, $\mathrm{dio}(\mathbf{a}) = +\infty$ for an eventually periodic word $\mathbf{a}$, but the converse is not true. There is some interesting interplay between the Diophantine exponent and Diophantine approximation, which is actually responsible for the name of the exponent. Let ξ be a real number whose base-b expansion is $0.a_1a_2\cdots$. Set $\mathbf{a} := a_1a_2\cdots$. Let us assume that the word $\mathbf{a}$ begins with a prefix of the form UV^α, with $V \neq \epsilon$. Set $q := b^{|U|}(b^{|V|} - 1)$. A simple computation shows that there exists an integer p such that

$$\langle p/q \rangle_b = 0.UVVV\cdots .$$

Since ξ and p/q have the same first $|UV^\alpha|$ digits in their base-b expansion, we obtain that

$$\left| \xi - \frac{p}{q} \right| < \frac{1}{b^{|UV^\alpha|}}$$

and thus

$$\left| \xi - \frac{p}{q} \right| < \frac{1}{q^\rho}, \tag{3.1}$$

where $\rho := |UV^\alpha|/|UV|$.

We do not claim here that p/q is written in lowest terms. Actually, it may happen that the gcd of p and q is quite large but (3.1) still holds in that case.

By Definition 3.1, it follows that for every $\rho < \mathrm{dio}(\mathbf{a})$, there exist infinitely many rational numbers p/q such that

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{q^\rho}\cdot$$

Note that when $\mathrm{dio}(\mathbf{a}) < 2$, such approximations look quite bad, for the existence of much better ones is ensured by the theory of continued fractions or by the Dirichlet pigeonhole principle. Quite surprisingly, the inequality $\mathrm{dio}(\mathbf{a}) > 1$ is already enough to conclude that ξ is either rational or transcendental. This powerful combinatorial transcendence criterion, proved in [8] and restated in Proposition ABL, emphasizes the relevance of the Diophantine exponent for our purpose.

Proposition ABL. *Let ξ be a real number with $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$. Suppose that $\mathrm{dio}(\mathbf{a}) > 1$ where $\mathbf{a} := a_1a_2\cdots$. Then ξ is either rational or transcendental.*

Proposition ABL is obtained as a consequence of the p-adic subspace theorem. It is the key tool for proving Theorem AB, and it will be the key tool for proving Theorems 1.2 and 1.3 as well.

4. Proof of Theorems 1.2 and 1.3

In this section, we prove our two main results.

4.1. **Proof of Theorem 1.2.**

In order to prove Theorem 1.2, we first need the following definition.

Definition 4.1. Let A be a finite set and let σ be a morphism of A^*. A letter $b \in A$ is said to have *maximal growth* if there exists a real number C such that

$$|\sigma^n(c)| \le C|\sigma^n(b)|$$

for every letter $c \in A$ and every positive integer n.

Lemma 4.2. *Let A be a finite set and $a \in A$. Let σ be a morphism of A^* prolongable on a and such that all letters of A appear in $\sigma^\omega(a)$. Then the letter a has maximal growth. Let θ denote the spectral radius of M_σ. Furthermore, there exist a non-negative integer ℓ and two positive real numbers c_1 and c_2 such that*

$$c_1 n^\ell \theta^n < |\sigma^n(a)| < c_2 n^\ell \theta^n \tag{4.1}$$

for every positive integer n.

Proof. Let c be a letter occurring in $\sigma^\omega(a)$. Then c also occurs in $\sigma^r(a)$, for some positive integer r. Then

$$|\sigma^n(c)| \le |\sigma^{n+r}(a)| = |\sigma^r(\sigma^n(a))| \le \|M_{\sigma^r}\|_\infty |\sigma^n(a)|\,,$$

where $\|\cdot\|_\infty$ stands for the usual infinite norm. This shows that a has maximal growth. Now recall that by a classical result of Salomaa and Soittola (see, for instance, Theorem 4.7.15 in [21]), there exist a non-negative integer ℓ, a real number $\beta \ge 1$, and two positive real numbers c_1 and c_2 such that

$$c_1 n^\ell \beta^n < |\sigma^n(a)| < c_2 n^\ell \beta^n \tag{4.2}$$

for every positive integer n. Since a has maximal growth, a classical theorem on matrices due to Gelfand (see, for instance, [21]) implies that β must be equal to θ, the spectral radius of the incidence matrix of σ. □

Proposition 4.3. *Let* $\mathbf{a}$ *be an infinite sequence that can be generated by a morphism with exponential growth. Then* $\operatorname{dio}(\mathbf{a}) > 1$*.*

Proof. Let $\mathbf{a}$ be an infinite sequence that can be generated by a morphism with exponential growth. Then $\mathbf{a} = \varphi(\sigma^\omega(a))$ for some morphism σ with exponential growth defined over a finite alphabet A, and some coding φ. Furthermore, we recall that the spectral radius θ of the incidence matrix M_σ satisfies $\theta > 1$. Set $\mathbf{u} := \sigma^\omega(a)$. Since by definition σ is prolongable on a and all letters of A appear in $\mathbf{u}$, Lemma 4.2 implies that a has maximal growth and that there exist a non-negative integer ℓ and two positive real numbers c_1 and c_2 such that

$$c_1 n^\ell \theta^n < |\sigma^n(a)| < c_2 n^\ell \theta^n \tag{4.3}$$

for every positive integer n.

Remark 4.4. If $a \in A$ is a letter of maximal growth, and $b \in A$ is not a letter of maximal growth, then for any positive real number ε, one has $|\sigma^n(b)| < \varepsilon|\sigma^n(a)|$ for sufficiently large n.

We now prove that there are infinitely many occurrences of letters with maximal growth in $\mathbf{u}$. Let us argue by contradiction. If there are only finitely many occurrences of letters with maximal growth, then there exists a positive integer n_0 such that $\mathbf{u} = \sigma^{n_0}(a)\mathbf{w}$ where $\mathbf{w}$ is an infinite word that contains no letter with maximal growth. Since $\theta > 1$, there is an integer m_0 such that

$$c_2/\theta^{m_0} < c_1/2. \tag{4.4}$$

Let V_0 denote the unique finite word such that $\sigma^{n_0+m_0}(a) = \sigma^{n_0}(a)V_0$. Then for every positive integer n we get that

$$|\sigma^{n+n_0+m_0}(a)| = |\sigma^{n+n_0}(a)| + |\sigma^n(V_0)| \leq c_2(n+n_0)^\ell \theta^{n+n_0} + |\sigma^n(V_0)|\,.$$

Given $\varepsilon > 0$, we have that $|\sigma^n(V_0)| < \varepsilon n^\ell \theta^n$ for all n sufficiently large, since by construction V_0 contains no letter with maximal growth. Choosing $\varepsilon = c_1/2$, we then infer from (4.4) that

$$\frac{|\sigma^{n+n_0+m_0}(a)|}{(n+n_0+m_0)^\ell \theta^{n+n_0+m_0}} < c_1\,,$$

when n is large enough. This provides a contradiction with (4.3).

Since there are infinitely many occurrences in $\mathbf{u}$ of letters with maximal growth, the pigeonhole principle ensures the existence of such a letter b that occurs at least twice in $\mathbf{u}$. In particular, there exist two possibly empty finite words U and V such that $UbVb$ is a prefix of $\mathbf{u}$. Set $r := |U|$, $s := |bV|$, and for every non-negative integer n, $U_n := \sigma^n(U)$, $V_n := \sigma^n(bV)$. Since by definition $\mathbf{u}$ is fixed by σ, we get that $U_n V_n^{\delta_n}$ is a prefix of $\mathbf{u}$, where $\delta_n := 1 + |\sigma^n(b)|/|\sigma^n(bV)|$. Since b has maximal growth, there exists a positive real number c_3 such that

$$|\sigma^n(c)| \leq c_3|\sigma^n(b)|$$

for every letter c in $\mathbf{u}$. We thus obtain that

$$\frac{|U_n V_n^{\delta_n}|}{|U_n V_n|} = 1 + \frac{|\sigma^n(b)|}{|\sigma^n(UbV)|} \geq 1 + \frac{1}{c_3(r+s)} > 1.$$

This proves that $\operatorname{dio}(\mathbf{u}) > 1$. By definition, one has $\mathbf{a} := \varphi(\mathbf{u})$. It thus follows that $\operatorname{dio}(\mathbf{a}) \geq \operatorname{dio}(\mathbf{u}) > 1$, because applying a coding to an infinite word cannot decrease the Diophantine exponent. This ends the proof. $\square$

Proof of Theorem 1.2. The result follows directly from Propositions ABL and 4.3. □

4.2. **Proof of Theorem 1.3.** In order to prove Theorem 1.3, we first introduce a useful and natural equivalence relation on the set of internal configurations of a pushdown automaton. This equivalence relation is closely related to the classical Myhill-Nerode relation used in formal language theory. Roughly, we think of two configurations as being equivalent if, starting from each configuration, there is no way to distinguish them by feeding the machine with arbitrary inputs.

Let us introduce some notation. An internal configuration of a k-pushdown automaton $\mathcal{M} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$ is a pair $C := (q, S) \in Q \times \Gamma^*$ where q denotes the state of the finite control and S denotes the word on the stack. Given an input word w, $C_{\mathcal{M}}(w)$, or $C(w)$ for short if there is no risk of confusion, denotes the internal configuration reached by the machine $\mathcal{M}$ when it is started from the initial configuration and fed with the input w, that is, $C(w) := \overline{\delta}(q_0, \#, w)$. By the way, $\tau(C(w))$ denotes the corresponding output symbol produced by $\mathcal{M}$. We also use the classical notation $C \overset{w}{\vDash} C'$ to express that starting from the internal configuration C and reading the input word w, the machine enters into the internal configuration C', that is, when $C' = \overline{\delta}(C, w)$. When the input alphabet is Σ_k and n is a natural number, we simply write $C(n)$ instead of $C(\langle n \rangle_k)$.

Definition 4.5. Let $\mathcal{M}$ be a k-pushdown automaton. Given two configurations C_1 and C_2 of $\mathcal{M}$, we say that C_1 and C_2 are equivalent, and we write $C_1 \sim C_2$ if, for every input word $w \in \Sigma_k^*$, one has $\tau(\overline{\delta}(C_1, w)) = \tau(\overline{\delta}(C_2, w))$.

It is obvious that $\sim$ is an equivalence relation. We are now ready to state the following simple but key result.

Proposition 4.6. *Let ξ be a real number generated by a k-pushdown automaton. If there exist two distinct positive integers n and n' such that $C(n) \sim C(n')$, then ξ is either rational or transcendental.*

Proof. Let ξ be a real number whose base-b expansion can be generated by a k-pushdown automaton $\mathcal{M}$. Let $\mathbf{a} := (a_n)_{n \geq 0}$ be the output sequence of $\mathcal{M}$, so that $\langle \{\xi\} \rangle_b = 0.a_1 a_2 \cdots$. Let us assume that there exist two positive integers n and n', $n < n'$, such that $C(n) \sim C(n')$. Set $w_n := \langle n \rangle_k$ and $w'_n := \langle n' \rangle_k$. By definition of the equivalence relation, one has

$$a_{[w_n w]_k} = a_{[w'_n w]_k}$$

for every word $w \in \Sigma_k^*$. Given a positive integer ℓ, we obtain in particular the following equalities:

$$\forall i \in [0, k^\ell - 1], \quad a_{k^\ell n + i} = a_{k^\ell n' + i} \, . \tag{4.5}$$

Set $U_\ell := a_1 a_2 \cdots a_{k^\ell n - 1}$ and $V_\ell := a_{k^\ell n} a_{k^\ell n + 1} \cdots a_{k^\ell n' - 1}$. We thus deduce from (4.5) that the word

$$U_\ell V_\ell^{1 + 1/(n' - n)} = a_1 a_2 \cdots a_{k^\ell n - 1} a_{k^\ell n} a_{k^\ell n + 1} \cdots a_{k^\ell n' - 1} a_{k^\ell n} \cdots a_{k^\ell n + k^\ell - 1}$$

is a prefix of $\mathbf{a}$. Furthermore, one has

$$|U_\ell V_\ell^{1 + 1/(n' - n)}| / |U_\ell V_\ell| = 1 + \frac{1}{n' - 1/k^\ell} \geq 1 + \frac{1}{n'} \cdot$$

Since the exponent $1 + 1/n'$ does not depend on ℓ, this shows that

$$\operatorname{dio}(\mathbf{a}) \geq 1 + 1/n' > 1 .$$

Then Proposition ABL implies that ξ is either rational or transcendental, which ends the proof. □

With this proposition in hand, we observe that Theorem AB becomes obvious.

Proof of Theorem AB. Indeed, a finite k-automaton can be seen as a k-pushdown automaton with empty stack alphabet (transitions only depend on the state and do not act on the stack), so a configuration is just given by the state of the finite control, and the empty stack.

Since there are only a finite number of states, a finite automaton has only a finite number of different possible configurations. By the pigeonhole principle, there thus exist two distinct positive integers n and n' such that $C(n) = C(n')$. Then the proof follows from Proposition 4.6. □

We are now ready to prove Theorem 1.3. As mentioned to the authors by the referee, the argument in the following proof is similar to the one given in the proof of Theorem 4.7.4 in [44].

Proof of Theorem 1.3. Let ξ be a real number that can be generated by a k-pushdown automaton, say $\mathcal{M} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$. Given an input word $w \in \Sigma_k^*$, we let q_w denote the state reached by $\mathcal{M}$ when starting from its initial configuration and reading the input w. We also let $S(w) \in \Gamma^*$ denote the corresponding contents of the stack of $\mathcal{M}$ and $H(w)$ denote the corresponding stack height, that is, the length of the word $S(w)$. With this notation, we obtain that starting from the initial configuration $(q_0, \#)$ and reading the input w, $\mathcal{M}$ reaches the internal configuration $(q_w, S(w))$, that is, $(q_0, \#) \overset{w}{\models} (q_w, S(w))$.

For every positive integer m, we consider the set

$$\mathcal{H}_m := \{ w \in \mathcal{R}_k : H(w) \leq m \} .$$

We distinguish two cases.

(i) Let us first assume that there exists a positive integer m such that $\mathcal{H}_m$ is infinite. Note that for all $w \in \mathcal{H}_m$, the configuration $C(w) := (q_w, S(w))$ belongs to the finite set $Q \times \Gamma^{\leq m}$, where $\Gamma^{\leq m}$ denotes the set of words of length at most m defined over Γ. Since $\mathcal{H}_m$ is infinite, the pigeonhole principle ensures the existence of two distinct words w and w' in $\mathcal{H}_m$ such that $C(w) = C(w')$. Setting $n := [w]_k$ and $n' := [w']_k$, we obtain that $n \neq n'$ and $C(n) = C(n')$. In particular, $C(n) \sim C(n')$. Then Proposition 4.6 applies, which concludes the proof in this case.

(ii) We now turn to the case where all the sets $\mathcal{H}_m$ are finite. For every $m \geq 1$, we can thus pick a word v_m in $\mathcal{H}_m$ with maximal length. Note that since $\mathcal{H}_m \subset \mathcal{H}_{m+1}$, we have $|v_m| \leq |v_{m+1}|$. Furthermore, one has

$$\mathcal{R}_k = \bigcup_{m=1}^{\infty} \mathcal{H}_m ,$$

which implies that the set $\{v_m : m \geq 1\}$ is infinite.

As discussed in Remark 2.13, we can assume without loss of generality that all ϵ-moves of $\mathcal{M}$ decrease the stack height. Furthermore, recall that all possible ϵ-moves are performed after reading the last symbol of a given input. This leads to the following alternative. For every internal configuration $(q_w, S(w))$ with $H(w) \neq 0$, each time a new input symbol a is consumed:

- either the stack height is decreased, which means that $H(wa) < H(w)$, and, moreover, $S(wa)$ is a prefix of $S(w)$,
- or only the topmost symbol of the stack has been modified, which means that there exist two words $X, Y \in \Gamma^*$ and a letter $z \in \Gamma$ such that $S(w) = Xz$ while $S(wa) = XY$.

The definition of v_m ensures that

$$\forall w \in \Sigma_k^*, \quad H(v_m) < H(v_m w)\,. \tag{4.6}$$

Furthermore, if m is large enough, we have that $|v_m| > |v_1|$, so that $H(v_m) > 1$. For such m, let us decompose the stack word $S(v_m)$ as

$$S(v_m) = X_m z_m\,,$$

where $z_m \in \Gamma$ is the topmost stack symbol. Inequality (4.6) implies that for all $w \in \Sigma_k^*$, the word X_m is a prefix of the stack word $S(v_m w)$. In other words, the part of the stack corresponding to the word X_m will never be modified or even read during the computation $(q_{v_m}, S(v_m)) \overset{w}{\models} (q_{v_m w}, S(v_m w))$. This means precisely that

$$(q_{v_m}, S(v_m)) \sim (q_{v_m}, z_m)\,.$$

Note that $(q_{v_m}, z_m) \in Q \times \Gamma$, which is a finite set, while we already observed that $\{v_m : m \geq 1\}$ is infinite. The pigeonhole principle thus implies the existence of two distinct integers m and m' such that $v_m \neq v_{m'}$ and $C(v_m) \sim C(v_{m'})$. Setting $n := [v_m]_k$ and $n' := [v_{m'}]_k$, we get that $C(n) \sim C(n')$ and $n \neq n'$. Then Proposition 4.6 applies, which ends the proof. □

5. Some related models of computations: Tag machines and stack machines

In this section, we complete our study by discussing different types of machines. We first consider the *tag machine* that was originally introduced by Cobham [26] and we prove that Theorem 1.2 is equivalent to Cobham's second claim. Then we introduce a general model of computation called a *multistack machine* and we show how Theorem 1.3 allows us to solve the Hartmanis–Stearns problem for this class of machines.

5.1. **Tag machine.**

In [26], Cobham originally investigated a model of computation, called a *tag machine*, whose outputs turn out to be precisely the morphic sequences defined in Section 2.2. Here we describe this model and the associated notion of dilation factor, and prove the equivalence between sequences produced by a tag machine with dilation factor larger than one and sequences generated by a morphism with exponential growth. This shows that Theorem 1.2 is equivalent to Cobham's second claim, as claimed in the introduction.

A tag machine is a two-tape enumerator that can be described as follows. In internal structure, a tag machine $\mathcal{T}$ (see Figure 5.1) has

- a finite state control,
- a working tape on which operate a read-only head $\mathfrak{R}$ and a write-only head $\mathfrak{W}$.

In external structure, $\mathcal{M}$ has

- an output tape on which operates a write-only head $\mathfrak{W}'$ and from which nothing can be erased.

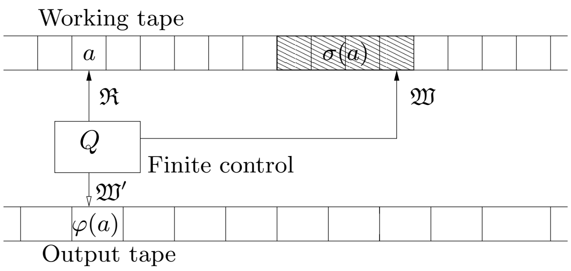

FIGURE 5.1. A tag machine.

Let us briefly describe how a tag machine operates. The finite state control of $\mathcal{T}$ contains the following basic information:

- a finite set of symbols A together with a special starting symbol a_0,
- for every symbol $a \in A$, a (possibly empty) finite word $\sigma(a)$ over A,
- for every symbol $a \in A$, a symbol $\varphi(a)$ in another finite set of symbols B.

When the computation starts, $\mathfrak{R}$ and $\mathfrak{W}$ are both positioned on the leftmost square of the blank working tape and $\mathfrak{W}$ proceeds by writing the word $\sigma(a_0)$, one symbol per square. Then both heads $\mathfrak{R}$ and $\mathfrak{W}$ move one square right, $\mathfrak{R}$ scans the symbol written in the corresponding square, say a, and $\mathfrak{W}$ proceeds by writing the word $\sigma(a)$. Again, both heads move one square to the right and the process keeps on forever unless $\mathfrak{R}$ and $\mathfrak{W}$ reach the same square of the working tape, in which case the machine stops. Meanwhile, each time $\mathfrak{R}$ reads a symbol a on the working tape, $\mathfrak{W}'$ writes the symbol $\varphi(a)$ on the output tape and moves one square right. Each symbol written on the output tape is thus irrevocable and cannot be erased in the process of computation. The output sequence produced by $\mathcal{T}$ is the sequence of symbols written on its output tape.

Using this description [26], Cobham extracts the following usual definition of a tag machine in terms of morphisms which confirms that output of tag machines and morphic sequences are the same.

Definition 5.1. A tag machine is a 5-tuple $\mathcal{T} := (A, \sigma, a, B, \varphi)$ where

- A is a finite set of symbols called the *internal alphabet*,
- a is an element of A called the *starting symbol*,
- σ is a *morphism* of A^* prolongable on a,
- B is a finite set of symbols called the *external alphabet.*,
- φ is a letter-to-letter morphism from A to B.

The output sequence of $\mathcal{T}$ is the morphic sequence $\varphi(\sigma^\omega(a))$. A tag machine is said to be uniform (resp., k-uniform) when the morphism σ has the additional property to be uniform (resp., k-uniform).

In [26], Cobham also introduced the following interesting quantity which measures the rate of production of symbols by a tag machine.

Definition 5.2. The (*minimum*) *dilation* factor of a tag machine $\mathcal{T}$ is defined as

$$\mathfrak{d}(\mathcal{T}) := \liminf_{n\to\infty} \frac{\mathfrak{W}(n)}{n},$$

where $\mathfrak{W}(n)$ denotes the position of the write-only head $\mathfrak{W}$ of $\mathcal{T}$ when the read-only head $\mathfrak{R}$ occupies the nth square of the working tape.

Remark 5.3. It follows from Definition 5.1 that $\mathfrak{W}(n) = |\sigma(u_1u_2\cdots u_n)|$, where $u_1u_2\cdots u_n$ is the prefix of length n of the infinite word $\sigma^\omega(a)$.

It is easy to see that uniform tag machines, or equivalently finite automata used as transducers (see Section 2.2), all have dilation factor at least two. As already mentioned, Cobham claimed that the base-b expansion of an algebraic irrational number cannot be generated by a tag machine with dilation factor larger than one. This result will immediately follow from Theorem 1.2 once we have proved Proposition 5.4 below. It is stated without proof by Cobham in [26].

Proposition 5.4. *Let $\mathcal{T} := (A, \sigma, a, B, \varphi)$ be a tag machine. Then the following statements are equivalent:*

(i) $\mathfrak{d}(\mathcal{T}) > 1$.
(ii) *The spectral radius of M_σ is larger than one.*

Proof. Let us first prove that (i) implies (ii). Since $\mathfrak{d}(\mathcal{T}) > 1$, Remark 5.3 ensures the existence of a positive real number ε such that

$$\frac{|\sigma^{n+1}(a)|}{|\sigma^n(a)|} = \frac{\mathfrak{W}(|\sigma^n(a)|)}{|\sigma^n(a)|} > 1+\varepsilon$$

for every n sufficiently large. This implies that there exists a positive real number c such that

$$|\sigma^n(a)| > c(1+\varepsilon)^n$$

for every positive integer n. By Lemma 4.2, we obtain that θ, the spectral radius of M_σ, must satisfy $\theta \geq 1+\varepsilon > 1$.

Let us now prove that (ii) implies (i). Let $\theta > 1$ denote the spectral radius of M_σ. We argue by contradiction assuming that $\mathfrak{d}(\mathcal{T}) = 1$. Let $\mathbf{u} := \sigma^\omega(a)$. By Lemma 4.2, there exist a non-negative integer ℓ, and two positive real numbers c_1 and c_2 such that

$$c_1 n^\ell \theta^n < |\sigma^n(a)| < c_2 n^\ell \theta^n \tag{5.1}$$

for every positive integer n. Set $C := \|M_\sigma\|_\infty$. Let ε be a positive real number and let m be a positive integer such that

$$\theta^m > C(1+\varepsilon)c_2/c_1\,.$$

We then infer from (5.1) that

$$|\sigma^{m+n}(a)| > c_1(m+n)^\ell\theta^{m+n} > \theta^m c_1 n^\ell\theta^n > C(1+\varepsilon)c_2 n^\ell \theta^n$$

and thus

$$|\sigma^{m+n}(a)| > C(1+\varepsilon)|\sigma^n(a)|$$

for every positive integer n. Let N be a positive integer and let $u_1u_2\cdots u_N$ denote the prefix of length N of $\mathbf{u}$. Let n be the largest integer such that $\sigma^n(a)$ is a prefix of $u_1u_2\cdots u_N$. It thus follows that

$$|\sigma^m(u_1\cdots u_N)| \geq |\sigma^m(\sigma^n(a))| = |\sigma^{m+n}(a)| > C(1+\varepsilon)|\sigma^n(a)|\,.$$

Since the definition of n ensures that $|\sigma^n(a)| > N/C$, we have

$$|\sigma^m(u_1u_2\cdots u_N)| > (1+\varepsilon)N\,. \tag{5.2}$$

On the other hand, for every $\delta > 0$ there exists a positive integer N such that:

$$\frac{|\sigma(u_1u_2\cdots u_N)|}{N} < 1+\delta\,,$$

since by assumption $\mathfrak{d}(\mathcal{T}) = 1$. Let V be the finite word defined by the relation $\sigma(u_1u_2\cdots u_N) = u_1u_2\cdots u_N V$. Thus $|V| < \delta N$. Now it is easy to see that

$$\sigma^m(u_1u_2\cdots u_N) = u_1u_2\cdots u_N V\sigma(V)\cdots\sigma^{m-1}(V)\,,$$

which implies that

$$|\sigma^m(u_1u_2\cdots u_N)| < N + \delta N + C\delta N + \cdots + C^{m-1}\delta N\,.$$

Choosing $\delta < \varepsilon(C-1)/(C^m-1)$, we get that $|\sigma^m(u_1u_2\cdots u_N)| < (1+\varepsilon)N$, which contradicts (5.2). This ends the proof. $\square$

5.2. **The Hartmanis-Stearns problem for multistack machines.** In this section, we discuss how our main result allows us to revisit the Hartmanis–Stearns problem as follows. Instead of a time constraint, we impose a restriction based on the way the memory may be stored by Turing machines. We consider a classical model of computation called a *multistack machine*. It corresponds to a version of the deterministic Turing machine where the memory is organized as stacks. It is as general as the Turing machine if one allows two or more stacks. Furthermore the one-stack machine turns out to be equivalent to the deterministic pushdown automaton, while a zero-stack machine is just a finite automaton (a machine with a strictly finite memory only stored in the finite state control).

For a formal definition of Turing machines the reader is referred to any of the classical references such as [31, 36, 45]. We will content ourself with the following informal definition of a multistack machine. A multistack machine (see Figure 5.2) is a one-way multitape deterministic Turing machine in which the memory is organized as stacks. Used as a transducer, it can be divided into three parts:

- the input tape, on which there is a read-only head which cannot go to the left (*one-way* machine),
- the internal part, which consists of a finite control and working tapes organized as stacks (the head of each working tape is always located on the rightmost non-blank symbol so that the tape can be thought of as a stack with a head on the topmost symbol),
- the output tape on which there is a write-only head and from which nothing can be erased.

Let us briefly describe how such a machine operates. A move on a multistack machine is based on the current state of the finite control, the input symbol read,

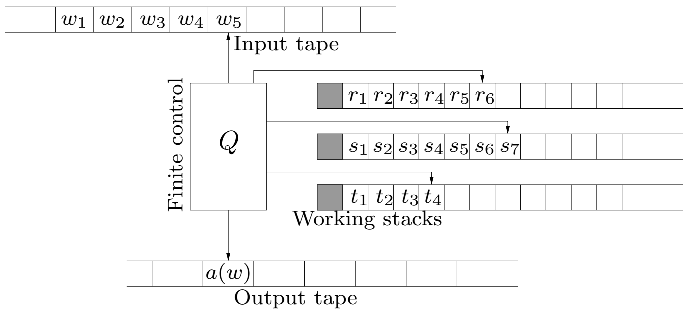
w_1 w_2 w_3 w_4 w_5
Input tape
Finite control
Q
r_1 r_2 r_3 r_4 r_5 r_6
s_1 s_2 s_3 s_4 s_5 s_6 s_7
t_1 t_2 t_3 t_4
Working stacks
$a(w)$
Output tape

FIGURE 5.2. A one-way stack machine.

and the top stack symbol on each of the stacks, and involves the following actions:

- for each stack, replacing the top symbol with a (possibly empty) string of stack symbols;
- moving the head of the input tape to the right;
- changing the finite state control to a new state.

Remark 5.5. Again, to make the machine deterministic, a move is uniquely determined by the knowledge of the input symbol read, the state of the finite control, and the top symbol of each stack. As for pushdown automata, a multistack machine can also possibly perform an ϵ-move: a move for which the head of the input tape does not move. The possibility of such a move depends only on the current state of the finite control and the top stack symbol on each of the stacks. Also, a multistack machine is not allowed to stop its computation in a state from which an ϵ-move is possible.

After reading an input word w, a multistack machine $\mathcal{M}$ produces an output symbol $a(w)$ that belongs to a finite output alphabet. The symbol $a(w)$ depends only on the state of the finite control and the top symbol of each stack. Given an integer $k \geq 2$, a k-multistack machine is a multistack machine that takes as input the base-k expansion of an integer, that is, for which the input alphabet is Σ_k. In that case, the sequence $(a(\langle n \rangle_k))_{n\geq 0}$ is called the *output sequence* produced by $\mathcal{M}$. With these definitions, a deterministic k-pushdown automaton is nothing else than a k-multistack machine with a single stack. One can now define the class of real numbers generated by multistack machines as follows.

Definition 5.6. A real number ξ can be generated by a k-multistack machine $\mathcal{M}$ if, for some integer $b \geq 2$, one has $\langle\{\xi\}\rangle_b = 0.a_1a_2\cdots$, where $(a_n)_{n\geq 0}$ corresponds to the output sequence produced by $\mathcal{M}$. A real number can be generated by a multistack machine if it can be generated by a k-multistack machine for some k.

Theorem 1.3 (resp., Theorem AB) may now be rephrased as follows: *No algebraic irrational can be generated by a one-stack* (*resp., by a zero-stack*) *machine.* Incidentally, this result turns out to provide a complete picture concerning the Hartmanis-Stearns problem for multistack machines. Indeed, since the two-stack machine has the same power as the general Turing machine, any computable number (and in particular any algebraic number) can be generated by a two-stack machine.

5.3. **Beyond pushdown automata.** We stress now that the equivalence relation $\sim$ given in Definition 4.5 and the associated Proposition 4.6 can be naturally extended to more general models of computation. For a multistack machine and more generally for a multitape Turing machine, an internal configuration is determined by the state of the finite control and the complete knowledge of all the working tapes (that is, the word written on each tape). To use the equivalence given in Definition 4.5 and the associated Proposition 4.6 we do not need to concretely describe how the working part of the machine is organized (stacks, tapes, or whatever). All that we need is to work with a machine with a one-way input tape and an output tape on which every symbol written is irrevocable. We refer to these machines as one-way transducer-like machines. When fed with a word, such a machine reads it with a one-way reading head, and outputs a single letter on its output tape.

Let $k \geq 2$ be an integer. A *one-way transducer-like k-machine* is defined as a 6-tuple $\mathcal{M} := (\Sigma_k, \mathcal{C}, \delta, C_0, \Delta, \tau)$ where

- $\Sigma_k := \{0, 1, \ldots, k-1\}$ is called the *alphabet of input symbols*,
- $\mathcal{C}$ is a countable set called the *set of configurations*,
- $\delta : \mathcal{C} \times \Sigma_k \to \mathcal{C}$ is called the *transition function*,
- $C_0 \in \mathcal{C}$ is called the *initial configuration*,
- Δ is a finite set called the *alphabet of output symbols*,
- $\tau : \mathcal{C} \to \Delta$ is called the *output function*.

Note that we assume that the machine is complete, that is, δ is defined on all of $\mathcal{C} \times \Sigma_k$. With this condition, the transition function can be extended to Σ_k^* by setting, for every $w \in \Sigma_k^*$ and $i \in \Sigma_k$, $\delta(C, wi) = \delta(\delta(C, w), i)$.

Definition 5.7. Let $\mathcal{M} := (\Sigma_k, \mathcal{C}, \delta, C_0, \Delta, \tau)$ be a one-way transducer-like k-machine. The *output sequence* produced by $\mathcal{M}$ is the sequence $(\tau(C(n)))_{n \geq 0}$ where $C(n) := \delta(C_0, \langle n \rangle_k)$ is the configuration reached by $\mathcal{M}$ after reading the input word $\langle n \rangle_k$.

Remark 5.8. Without any further restriction, one-way transducer-like k-machines can output any sequence $(a_n)_{n \geq 0}$ with values in a finite alphabet. This comes from the fact that the output function τ can be any function from $\mathcal{C}$ to Δ. In practice, the value of the output functions $\tau(C)$ depends only on some finite amount of information associated with the configuration C, for instance, the current state for a finite automaton, the top symbol of the stack and the current state for a pushdown automaton, the top symbol of each stack and the current state for a multistack machine, etc.

We define now an equivalence relation over the set of configurations of a one-way transducer-like k-machine just as in Definition 4.5.

Definition 5.9. Let $\mathcal{M}$ be a one-way transducer-like k-machine. Given two configurations C_1 and C_2 in $\mathcal{C}$, we say that C_1 and C_2 are equivalent, and we write $C_1 \sim C_2$ if, for every input word $w \in \Sigma_k^*$, $\tau(\delta(C_1, w)) = \tau(\delta(C_2, w))$.

Definition 5.10. A real number ξ can be generated by a one-way transducer-like machine if there exist two integers k and b at least equal to 2, and a one-way transducer-like k-machine $\mathcal{M}$, such that $\langle \{\xi\} \rangle_b = 0.a_1 a_2 \cdots$, where $(a_n)_{n \geq 0}$ corresponds to the output sequence produced by $\mathcal{M}$.

We then have the following proposition, which can be proved in exactly the same way as Proposition 4.6.

Proposition 5.11. *Let ξ be a real number generated by a one-way transducer-like machine. If there exist two distinct positive integers n and n' such that $C(n) \sim C(n')$, then ξ is either rational or transcendental.*

This general result provides a method to prove the transcendence of real numbers which can be generated by machines more powerful than a k-pushdown automaton. For instance, it implies the transcendence of the ternary number

$$\xi_3 := 0.110\,120\,110\,010\,110\,100\,101\,100\,110\,100\,110\,010\,110\,011\,010\,010\,112\cdots$$

whose nth ternary digit is equal to 2 if the binary expansion of n is of the form $1^j0^j1^j$, for some $j \in \mathbb{N}^*$, to 1 if the binary expansion of n has an odd number of occurrences of ones, and to 0 otherwise. This number cannot be generated by a k-pushdown automaton because the set of words of the form $1^j0^j1^j$ for some $j \in \mathbb{N}^*$ is not a context-free language. In particular, ξ_3 is irrational. However, ξ_3 can be generated by a one-way transducer-like 2-machine $\mathcal{M}$. This machine is obtained as the synchronisation product of the linearly bounded one-way two-stack deterministic Turing machine recognizing the context sensitive language $\{1^j0^j1^j, j \in \mathbb{N}\}$ and the Thue-Morse automaton. We give now a precise definition. The machine $\mathcal{M}$ has a finite control given by the set $Q = \{i, q, r, s, t\} \times \{0, 1\}$ and two stacks with a single stack symbol X. For convenience, the set of stack configurations $\{X\}^* \times \{X\}^*$ is identified with $\mathbb{N}^2$ as follows. The stack configuration (X^j, X^k) is simply denoted by (j, k). The set of configurations of $\mathcal{M}$ is then defined by $\mathcal{C} = Q \times \{(j, k) \in \mathbb{N}^2 : j \le k\}$. The initial configuration is $C_0 = (i, 0, 0, 0)$. The transition function $\delta : \mathcal{C} \times \Sigma_2 \to \mathcal{C}$ is defined as follows. We set $\delta((i, 0, 0, 0), 1) = (q, 1, 1, 1)$. For all $(e, j, k) \in \{0, 1\} \times \{(j, k) \in \mathbb{N}^2 : j \le k\}$, we set

$$\begin{aligned}
&\delta((q,e,j,j),1) = (q,1-e,j+1,j+1), && \delta((r,e,0,k),0) = (t,e,0,k),\\
&\delta((s,e,0,0),1) = (t,1-e,0,0), && \delta((s,e,0,k),0) = (t,e,0,k),\\
&\delta((t,e,j,k),1) = (t,1-e,j,k), && \delta((t,e,j,k),0) = (t,e,j,k),
\end{aligned}$$

and, if moreover j and k are positive, we set

$$\begin{aligned}
&\delta((q,e,j,j),0) = (r,e,j-1,j), && \delta((r,e,j,k),0) = (r,e,j-1,k),\\
&\delta((r,e,j,k),1) = (t,1-e,j,k), && \delta((r,e,0,k),1) = (s,1-e,0,k-1),\\
&\delta((s,e,0,k),1) = (s,1-e,0,k-1)\,.
\end{aligned}$$

Note that we have just defined δ on $\mathcal{C}' \times \Sigma_2$, where $\mathcal{C}'$ is the set of all configurations that can be reached from the initial configuration, which is of course enough for our purpose. The alphabet of output symbols is $\Delta = \Sigma_3$. The output function $\tau : \mathcal{C} \to \Delta$ is defined by: $\tau(x, 1, j, k) = 1$ if $x \in \{i, q, r, s, t\}$ and $(j, k) \in \{(j, k) \in \mathbb{N}^2 : j \le k\}$, $\tau(s, 0, 0, 0) = 2$, and $\tau(x, 0, j, k) = 0$ if $(x, j, k) \neq (s, 0, 0)$.

The machine $\mathcal{M}$ works as follows. Given a control state (x, e) with $x \in \{i, q, r, s, t\}$ and $e \in \{0, 1\}$, the "e-part" gives the parity of the number of ones already read by the machine, while the "x-part" of the control state and both stacks are devoted to the recognition of the natural numbers whose binary expansion is of the form $1^j0^j1^j$. More precisely, we have that

- in state (q, e) with stacks (j, j), the machine has read a prefix 1^j,
- in state (r, e) with stacks (j, k), the machine has read a prefix 1^k0^{k-j},
- in state (s, e) with stacks $(0, k)$, the machine has read a prefix $1^j0^j1^{j-k}$,
- in state (t, e), the machine has read a word which is not a prefix of the words $1^j0^j1^j$ for any $j \in \mathbb{N}^*$.

From the definition of $\mathcal{M}$, it is easy to check that both configurations $C(10)$ and $C(20)$ are equal to $(t, 0, 0, 0)$. In particular, these two configurations are equivalent, and it follows from Proposition 5.11 that ξ_3 is a transcendental number.

6. Concluding remarks

We end this paper with several comments concerning factor complexity, transcendence measures, and continued fractions, and also provide possible directions for further research.

6.1. **Links with subword complexity.**

Another interesting way to tackle problems concerned with the expansions of classical constants in integer bases is to consider the *subword complexity* (also known as *factor complexity*) of real numbers. Let ξ be a real number, $0 \leq \xi < 1$, and let $b \geq 2$ be a positive integer. Let $\mathbf{a} := (a_n)_{n\geq 1} \in \Sigma_b^{\mathbb{N}}$ denote its base-b expansion. The complexity function of ξ with respect to the base b is the function that associates with each positive integer n the positive integer

$$p(\xi, b, n) := \mathrm{Card}\{(a_j, a_{j+1}, \ldots, a_{j+n-1}),\ j \geq 1\}.$$

When ξ does not belong to $[0, 1)$, we just set $p(\xi, b, n) := p(\{\xi\}, b, n)$.

To obtain lower bounds for the complexity of classical mathematical constants remains a famous challenging problem. In this direction, the main result concerning algebraic numbers was obtained by Bugeaud and the first author [3], who proved that

$$\lim_{n\to\infty} \frac{p(\xi, b, n)}{n} = +\infty \tag{6.1}$$

for all algebraic irrational numbers ξ and all integers $b \geq 2$. This lower bound implies Theorem AB, for it is well known that a real number generated by a finite automaton has subword complexity $O(n)$ [27]. We stress that the situation is really different with pushdown automata and tag machines. Indeed, for every positive integer d, there exist pushdown automata whose output sequence has subword complexity growing at least like n^d [38], while tag machines can output sequences with quadratic complexity (see, for instance, [40]). In particular, Theorems 1.2 and 1.3 do not follow from (6.1). The referee informs the authors that the study of the complexity of sequences related to pushdown automata and context-free languages began with the work of Hamm on quasicontext-free sequences [29]. We now illustrate this difference by providing lower bounds for the complexity of the two numbers ξ_1 and ξ_2 defined in Sections 2.2 and 2.3.

6.1.1. *Estimate for* $p(\xi_1, 3, n)$. It follows from the definition of the number ξ_1 that its ternary expansion is the fixed point of the morphism μ defined by $\mu(0) = 021$, $\mu(1) = 012$, $\mu(2) = 2$. We note that the letter 2 has clearly bounded growth ($|\mu^n(2)| = 1$ for all $n \geq 0$) and that $\mu^\omega(0)$ contains arbitrarily large blocks of consecutive occurrences of the letter 2. Then, a classical result of Pansiot [40] implies that the complexity of the infinite word $\mu^\omega(0)$ is quadratic. In other words, one has

$$c_1 n^2 < p(\xi_1, 3, n) < c_2 n^2$$

for some positive real numbers c_1 and c_2 and for every natural number n.

6.1.2. *Estimate for* $p(\xi_2, 2, n)$. Recall that the binary number ξ_2 is defined as follows: its nth binary digit is 1 if the difference between the number of occurrences of the digits 0 and 1 in the binary expansion of n is at most 1, and is 0 otherwise. We outline a proof of the fact that

$$c_1 n \log^2 n < p(\xi_2, 2, n) < n \log^2 n$$

for some positive real numbers c_1 and c_2, and for every positive integer n. We can first infer from [33] that $p(\xi_2, 2, n) = O(n \log^2 n)$, for this sequence is generated by a pushdown automaton with only one ordinary stack symbol. In order to find a lower bound for $p(\xi_2, 2, n)$, we describe a tag machine-like process (over an infinite alphabet) generating the binary expansion of ξ_2. We first notice that another way to understand the action of the 2-PDA $\mathcal{A}$ in Figure 2.3 that generates the binary expansion of ξ_2 is to unfold it. This representation, given in Figure 6.1, corresponds to the transition graph of $\mathcal{A}$. States in this graph are given by all possible configurations, and transitions between configurations are just labelled by the input digits 0 or 1. In Figure 6.1, the notation qX^n means that, in this configuration, $\mathcal{A}$ is in state q and the content of the stack is $XX \cdots X$ (n times).

$$\begin{tikzcd}
& q_1\# \arrow[r, shift left, "1"] \arrow[d, shift left, "0"] & q_1X \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & q_1X^2 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & q_1X^3 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & q_1X^4 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & q_1X^5 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & \cdots \arrow[l, shift left, "0"] \\
{} \arrow[r] & q_0\# \arrow[u, shift left, "1"] \arrow[d, shift left, "0"] & & & & & & \\
& q_{-1}\# \arrow[u, shift left, "1"] \arrow[r, shift left, "0"] & q_{-1}X \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & q_{-1}X^2 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & q_{-1}X^3 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & q_{-1}X^4 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & q_{-1}X^5 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & \cdots \arrow[l, shift left, "1"]
\end{tikzcd}$$

FIGURE 6.1. The transition graph of $\mathcal{A}$.

In Figure 6.2, states of the transition graph have been renamed as follows. Configurations are replaced with integers, where reading a 1 in state n leads to a move to state $n+1$ and reading a 0 in state n leads to a move to state $n-1$. We easily see that the output state is just the difference between the number of 1's and 0's in the input word. Thus the nth binary digit of ξ_2 is equal to 1 if and only if the reading of the binary expansion of n by this infinite automaton ends in one of the three states labelled by 0, -1 and 1.

$$\begin{tikzcd}
& 1/1 \arrow[r, shift left, "1"] \arrow[d, shift left, "0"] & 2/0 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & 3/0 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & 4/0 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & 5/0 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & 6/0 \arrow[l, shift left, "0"] \arrow[r, shift left, "1"] & \cdots \arrow[l, shift left, "0"] \\
{} \arrow[r] & 0/1 \arrow[u, shift left, "1"] \arrow[d, shift left, "0"] & & & & & & \\
& -1/1 \arrow[u, shift left, "1"] \arrow[r, shift left, "0"] & -2/0 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & -3/0 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & -4/0 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & -5/0 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & -6/0 \arrow[l, shift left, "1"] \arrow[r, shift left, "0"] & \cdots \arrow[l, shift left, "1"]
\end{tikzcd}$$

FIGURE 6.2. Relabelling of the transition graph of $\mathcal{A}$.

The action of 0 and 1 can be summarized by $n \xrightarrow{0} n-1$ and $n \xrightarrow{1} n+1$. This leads to a tag machine-like process over an infinite alphabet $\mathcal{T} := (A, \sigma, s, B, \varphi)$ for generating the expansion of ξ_2. The starting symbol is s, $A := \mathbb{Z} \cup \{s\}$, σ is defined by $\sigma(s) = s1$ and $\sigma(n) = (n-1)(n+1)$, $B := \{0, 1\}$, $\varphi(-1) = \varphi(0) = \varphi(1) = \varphi(s) = 1$, and $\varphi(e) = 0$ if $e \notin \{s, -1, 0, 1\}$. Then we have $\langle \xi_2 \rangle_2 = a_0.a_1a_2\cdots$, where $(a_n)_{n\geq 0} = \varphi(\sigma^\omega(s))$ and

$$\sigma^\omega(s) = s\,1\,0\,2\,(-1)\,1\,1\,3\,(-2)\,0\,0\,2\,0\,2\,2\,4\,(-3)\,(-1)\,(-1)\,1\,(-1)\,1\,1\,3\,(-1)\,1\cdots$$

is the unique fixed point of the morphism σ. The strategy consists now in finding sufficiently many different right special factors, that is, factors w of $\varphi(\sigma^\omega(s))$ for which both factors $w0$ and $w1$ also occur in $\varphi(\sigma^\omega(s))$. We let $r > 2$ and n be two natural numbers with $2^r \leq n < 2^{r+1}$. Arguing as in [32, Lemma 1.13], one can actually show that, for every pair

$$(p, q) \in \mathcal{E} := \{(p, q) \in \mathbb{N}^2 : 2 \leq p \leq q \leq r-1\},$$

both words

$$A := \varphi(\sigma^r(r-2p-2)\sigma^r(r-2p)\sigma^r(r-2q))$$

and

$$B := \varphi(\sigma^r(r-2p-2)\sigma^r(r-2p)\sigma^r(-r-2))$$

occur in $\varphi(\sigma^\omega(s))$ and they have the same factor of length n, say $w(p,q)$, occurring at index $2^{r+1} + 2^q - n - 1$. That is, $w(p,q)$ occurs just after the prefix of length $2^{r+1} + 2^q - n - 2$ in both A and B. In the word A the factor $w(p,q)$ is followed by a 1, while in the word B it is followed by a 0. Thus $w(p,q)$ is a right special factor. It can also be extracted from [32, Lemma 1.13] that the map $(p,q) \mapsto w(p,q)$ is injective on $\mathcal{E}$. This ensures the existence of at least $(r-2)(r-1)/2$ distinct right special factors of length n in $\varphi(\sigma^\omega(s))$. Then it follows that

$$p(\xi_2, 2, n+1) - p(\xi_2, 2, n) \geq \frac{(r-2)(r-1)}{2},$$

from which one easily deduces the lower bound

$$p(\xi_2, 2, n) \geq cn\log^2 n$$

for some positive real number c, and for every positive integer n.

6.2. **Quantitative aspects.**

Here we discuss some problems related to the quantitative aspects of our results.

6.2.1. *The number theory side: Transcendence measures.*

A real number ξ is transcendental if $|P(\xi)| > 0$ for all non-zero integer polynomials P. A transcendence measure for ξ consists in a lower bound on $|P(\xi)|$, thus refining the transcendence statement. In general, one looks for a non-trivial function f satisfying

$$|P(\xi)| > f(H, d)$$

for all integer polynomials of degree at most d and height at most H. Here, $H(P)$ stands for the naïve height of the polynomial P, that is, the maximum of the absolute values of its coefficients. The degree and the height of an integer polynomial P allow us to take into account the complexity of P. Here we will use the following classification of real numbers defined by Mahler [35] in 1932. For every integer $d \geq 1$ and every real number ξ, we let $w_d(\xi)$ denote the supremum of the exponents w for which

$$0 < |P(\xi)| < H(P)^{-w}$$

has infinitely many solutions in integer polynomials P of degree at most d. Further, we set $w(\xi) := \limsup_{d\to\infty}(w_d(\xi)/d)$ and, according to Mahler [35], we say that ξ is an

$$\begin{gathered} A\text{-number if } w(\xi) = 0; \\ S\text{-number if } 0 < w(\xi) < \infty; \\ T\text{-number if } w(\xi) = \infty \text{ and } w_d(\xi) < \infty \text{ for any integer } d \geq 1; \\ U\text{-number if } w(\xi) = \infty \text{ and } w_d(\xi) = \infty \text{ for some integer } d \geq 1. \end{gathered}$$

An important feature of this classification is that two transcendental real numbers that belong to different classes are algebraically independent. The A-numbers are precisely the algebraic numbers and, in the sense of the Lebesgue measure, almost all numbers are S-numbers.

A Liouville number is a real number ξ such that for any positive real number ρ the inequality

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{q^\rho}$$

has at least one solution $(p, q) \in \mathbb{Z}^2$ with $q > 1$. Thus ξ is a Liouville number if and only if $w_1(\xi) = +\infty$. Liouville numbers are a subclass of U-numbers.

Let ξ be an irrational real number defined through its base-b expansion, say $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$. Let us assume that the base-b expansion of ξ can be generated either by a pushdown automaton or by a tag machine with dilation factor larger than one. As recalled in Proposition ABL (Section 3), the key point for proving that ξ is transcendental is to show that $\text{dio}(\mathbf{a}) > 1$, where $\mathbf{a} := a_1a_2\cdots$. This amounts to finding two sequences of finite words $(U_n)_{n\geq 0}$ and $(V_n)_{n\geq 0}$, a sequence of rational numbers α_n, and a real number $\delta > 0$ such that the word $U_nV_n^{\alpha_n}$ is a prefix of $\mathbf{a}$, the length of the word $U_nV_n^{\alpha_n}$ increases, and

$$\frac{|U_nV_n^{\alpha_n}|}{|U_nV_n|} \geq 1 + \delta\,. \tag{6.2}$$

A look at the proofs of Theorems 1.2 and 1.3 show that one actually has, in both cases, the following extra property. There exists a real number M such that

$$\limsup_{n\to\infty} \frac{|U_{n+1}V_{n+1}|}{|U_nV_n|} < M\,. \tag{6.3}$$

Using an approach introduced in [9] and developed in [7], one can first deduce from Inequalities (6.2) and (6.3) that

$$\text{dio}(\mathbf{a}) - 1 \leq w_1(\xi) \leq c_1 \,\text{dio}(\mathbf{a}) \tag{6.4}$$

for some real number c_1 that depends only on δ and M. In particular, inequalities (6.2) and (6.3) imply that ξ is a Liouville number if and only if $\text{dio}(\mathbf{a})$ is infinite. Then it is proved in [7], following a general approach introduced in [5] and based on a quantitative version of the subspace theorem, that this extra condition leads to transcendence measures. Indeed, taking all parameters into account, one can derive an upper bound of the type

$$w_d(\xi) \leq \max\{w_1(\xi), (2d)^{c_2(\log 3d)(\log\log 3d)}\} \tag{6.5}$$

for all positive integers d and some real number c_2 that depends only on δ and M. The constants c_1 and c_2 can be made effective. In particular, we deduce the following result from inequalities (6.4) and (6.5).

Theorem 6.1. *Let ξ be an irrational real number such that $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$ and let $\mathbf{a} := a_1a_2\cdots$. Let us assume that the base-b expansion of ξ can be generated either by a pushdown automaton or by a tag machine with dilation factor larger than one. Then one of the following holds:*

(i) $\mathrm{dio}(\mathbf{a}) = +\infty$ *and ξ is a Liouville number,*
(ii) $\mathrm{dio}(\mathbf{a}) < +\infty$ *and ξ is an S- or a T-number.*

Of course, in view of Theorem 6.1, it would be interesting to prove whether or not there exist irrational real numbers that can be generated either by a pushdown automaton or by a tag machine, and for which $\mathrm{dio}(\mathbf{a}) = +\infty$. In this direction, it is proved in [9] that $\mathrm{dio}(\mathbf{a})$ is always finite when ξ is generated by a finite automaton. Here we add the following contribution to this problem.

Proposition 6.2. *Let $\mathbf{a} := a_0a_1\cdots$ be an aperiodic pure morphic word generated by a morphism σ defined over a finite alphabet A. Set $M := \max\{|\sigma(i)| : i \in A\}$. Then* $\mathrm{dio}(\mathbf{a}) \leq M + 1$.

Proof. By definition, there exists a letter $a \in \mathcal{A}$ such that σ is prolongable on the letter a and $\sigma^\omega(a) = a_0a_1\cdots$. We argue by contradiction by assuming that $\mathrm{dio}(\mathbf{a}) > M+1$. This assumption ensures that one can find two finite words U and V and a real number $s > 1$ such that

(i) UV^s is a prefix of $\mathbf{a}$, and s is maximal with this property;
(ii) $|UV^s|/|UV| \geq M + 1$;
(iii) V is primitive (i.e., is non-empty and not the integral power of a shorter word).

Note that since $\mathbf{a}$ is fixed by σ then the word $\sigma(UV^s)$ is also a prefix of $\mathbf{a}$. By definition of M, it follows from (ii) that

$$UV^s = \sigma(U)W\,,$$

where $W := \widetilde{V}^\alpha$ for some conjugate $\widetilde{V}$ of V (i.e., $V = AB$ and $\widetilde{V} = BA$ for some A, B) and $\alpha \leq s$. We stress that, since σ is prolongable on a, W and $\sigma(V)$ are non-empty words. On the other hand, $\sigma(V)$ is also a period of W since $UV^s = \sigma(U)W$ is a prefix of $\sigma(UV^s) = \sigma(U)\sigma(V)^{s'}$ for some s'. Thus W has at least two periods: $\widetilde{V}$ and $\sigma(V)$. Furthermore, (ii) implies that

$$|UV^{s-1}| \geq M(|U| + |V|) \geq |\sigma(U)| + |\sigma(V)|$$

and then

$$|W| = |UV^s| - |\sigma(U)| \geq |\sigma(V)| + |V| = |\sigma(V)| + |\widetilde{V}|\,.$$

We can thus apply Fine and Wilf's theorem (see, for instance, [12, Chapter 1]) to the word W and we obtain that there is a word of length $\gcd(|\widetilde{V}|, |\sigma(V)|)$ that is a period of W. But, since V is primitive, the word $\widetilde{V}$ is primitive too, and it follows that $\gcd(|\widetilde{V}|, |\sigma(V)|) = |\widetilde{V}|$. This gives that $\sigma(V) = \widetilde{V}^j$ for some positive integer j. It follows that

$$\sigma(UV^s) = \sigma(U)\widetilde{V}^{js'} = UV^{s-\alpha+js'}$$

is a prefix of $\mathbf{a}$. Now the inequality $|\sigma(UV^s)| > |UV^s|$ gives a contradiction with the maximality of s. This ends the proof. □

6.2.2. *The computer science side.* Theorems AB, 1.2, and 1.3 show that some classes of Turing machines are too limited to produce the base-b expansion of an algebraic irrational real number. Let ξ be an irrational real number that can be generated by a k-pushdown automaton or by a tag machine with dilation factor larger than one. Then the results of Section 6.2 could be rephrased to provide a limitation of the way ξ can be approximated by irrational algebraic numbers. In this section, we suggest to view things from a different angle, changing our target. Indeed, we fix an algebraic irrational real number α and a base b, and ask for how long the base-b expansion of α can be imitated by outputs of a given class of Turing machines.

Let us explain now how to formalize our problem. We can naturally take the number of states as a measure of complexity of a k-automaton. One can also define the size of k-pushdown automata and tag machines as follows. Let us define the size of a k-pushdown automaton $\mathcal{A} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$ to be $|Q| + |\Gamma| + L$, where L is the maximal length of a word that can be added to the stack by the transition function δ of $\mathcal{A}$. Let us also define the size of a tag machine $\mathcal{T} := (A, \sigma, a, \varphi, B)$ to be $|A| + L$, where $L := \max\{|\sigma(i)| : i \in A\}$. Now, let us fix a class $\mathcal{M}$ of Turing machines among k-automata, k-pushdown automata, and tag machines. Let M be a positive integer. We stress that there are only finitely many such machines with size at most M. Then there exists a maximal positive integer $I(\alpha, M)$ for which there exists a machine in $\mathcal{M}$ with size at most M whose output agrees with the base-b expansion of α at least up to the $I(\alpha, M)$th digit. We suggest the following problem.

Problem 6.3. Let α be an algebraic irrational real number and fix a class of Turing machines among k-automata, k-pushdown automata, and tag machines. Given a positive integer M, find an upper bound for $I(\alpha, M)$.

In the case of finite automata, we can give a first result toward this problem. Indeed, the subword complexity of the output $\mathbf{a}$ of a k-automaton with at most M states satisfies $p(\mathbf{a}, n) \leq kM^2 n$ (see, for instance, [12, Theorem 10.3.1]). Let d and H denote the degree and the height of α, respectively. Then the main result of [19] allows us to extract the following upper bound:

$$\begin{aligned} I(\alpha, M) \quad \leq \quad \max \Big\{ & (\max(\log H, e) 100 kM^2)^{8 \log 4kM^2}, \\ & \Big((\log d) 10^{100} (kM^2)^{11/2} \log(kM^2) \Big)^{2.1} \Big\}. \end{aligned}$$

6.3. **Computational complexity of the continued fraction expansion of algebraic numbers.** Replacing integer base expansions with continued fractions leads to similar problems. Rational numbers all have a finite continued fraction expansion, while quadratic real numbers correspond to eventually periodic continued fractions. In contrast, much less is known about the continued fraction expansion of algebraic real numbers of degree at least three such as $\sqrt[3]{2}$. In this direction, an approach based on the subspace theorem was introduced by Bugeaud and the first author [2]. Recently, Bugeaud [20] showed that this approach actually leads to the following analogue of Proposition ABL.

Proposition B. *Let ξ be a real number with $\xi := [a_0, a_1, a_2, \ldots]$ where we assume that $(a_n)_{n\geq 1}$ is a bounded sequence of positive integers. Let us assume that* $\mathrm{dio}(\mathbf{a}) > 1$ *where $\mathbf{a} := a_1 a_2 \cdots$. Then ξ is either quadratic or transcendental.*

In [20], the author deduces from Proposition B that the continued fraction expansion of an algebraic real number of degree at least 3 cannot be generated by a finite automaton. This provides the analogue of Theorem AB in this framework. As a direct consequence of our results and Proposition B, we obtain the following generalization of Bugeaud's result corresponding to the analogue of Theorems 1.2 and 1.3.

Theorem 6.4. *Let ξ be an algebraic real number of degree at least 3. Then the following holds:*

(i) *The continued fraction expansion of ξ cannot be generated by a one-stack machine, or equivalently, by a deterministic pushdown automaton.*
(ii) *The continued fraction expansion of ξ cannot be generated by a tag machine with dilation factor larger than one.*

Using the approach introduced in [6] and the discussion of Section 6.2, it will also be possible to produce transcendence measures analogous to Theorem 6.1 for real numbers whose continued fraction expansion can be generated by a deterministic pushdown automaton or by a tag machine with dilation factor larger than one.

7. Acknowledgments

We warmly thank the referee for the very careful reading of a previous version of this paper. The current presentation has strongly benefited from the suggestions.

References

[1] Boris Adamczewski, *On the expansion of some exponential periods in an integer base*, Math. Ann. **346** (2010), no. 1, 107–116, DOI 10.1007/s00208-009-0391-z. MR2558889

[2] Boris Adamczewski and Yann Bugeaud, *On the complexity of algebraic numbers. II. Continued fractions*, Acta Math. **195** (2005), 1–20, DOI 10.1007/BF02588048. MR2233683

[3] Boris Adamczewski and Yann Bugeaud, *On the complexity of algebraic numbers. I. Expansions in integer bases*, Ann. of Math. (2) **165** (2007), no. 2, 547–565, DOI 10.4007/annals.2007.165.547. MR2299740

[4] Boris Adamczewski and Yann Bugeaud, *Dynamics for β-shifts and Diophantine approximation*, Ergodic Theory Dynam. Systems **27** (2007), no. 6, 1695–1711, DOI 10.1017/S0143385707000223. MR2371591

[5] Boris Adamczewski and Yann Bugeaud, *Mesures de transcendance et aspects quantitatifs de la méthode de Thue-Siegel-Roth-Schmidt* (French, with English and French summaries), Proc. Lond. Math. Soc. (3) **101** (2010), no. 1, 1–26, DOI 10.1112/plms/pdp054. MR2661240

[6] Boris Adamczewski and Yann Bugeaud, *Transcendence measures for continued fractions involving repetitive or symmetric patterns*, J. Eur. Math. Soc. (JEMS) **12** (2010), no. 4, 883–914, DOI 10.4171/JEMS/218. MR2654083

[7] Boris Adamczewski and Yann Bugeaud, *Nombres réels de complexité sous-linéaire: mesures d'irrationalité et de transcendance* (French, with English summary), J. Reine Angew. Math. **658** (2011), 65–98, DOI 10.1515/CRELLE.2011.061. MR2831513

[8] Boris Adamczewski, Yann Bugeaud, and Florian Luca, *Sur la complexité des nombres algébriques* (French, with English and French summaries), C. R. Math. Acad. Sci. Paris **339** (2004), no. 1, 11–14, DOI 10.1016/j.crma.2004.04.012. MR2075225

[9] Boris Adamczewski and Julien Cassaigne, *Diophantine properties of real numbers generated by finite automata*, Compos. Math. **142** (2006), no. 6, 1351–1372, DOI 10.1112/S0010437X06002247. MR2278750

[10] Boris Adamczewski and Colin Faverjon, *Méthode de Mahler: relations linéaires, transcendance et applications aux nombres automatiques* (French, with English and French summaries), Proc. Lond. Math. Soc. (3) **115** (2017), no. 1, 55–90, DOI 10.1112/plms.12038. MR3669933

[11] J. Albert, *Propriétés combinatoires et arithmétiques de certaines suites automatiques et substitutives*, Thèse de Doctorat de l'Université Paris Sud, 2006.

[12] Jean-Paul Allouche and Jeffrey Shallit, *Automatic sequences*, Cambridge University Press, Cambridge, 2003. Theory, applications, generalizations. MR1997038

[13] J.-M. Autebert, *Langages algébriques* (French), Études et Recherches en Informatique. [Studies and Research in Computer Science], Masson, Paris, 1987. MR888601

[14] Jean-Michel Autebert, Jean Berstel, and Luc Boasson, *Context-free languages and pushdown automata*, Handbook of formal languages, Vol. 1, Springer, Berlin, 1997, pp. 111–174. MR1469995

[15] J. Berstel and L. Boasson, *Modèles de machines*, Encyclopédie de l'informatique et des systèmes d'information, pp. 987–998, Vuibert, 2006.

[16] Daniel Bertrand, *Theta functions and transcendence*, Ramanujan J. **1** (1997), no. 4, 339–350, DOI 10.1023/A:1009749608672. International Symposium on Number Theory (Madras, 1996). MR1608721

[17] É. Borel, *Les probabilités dénombrables et leurs applications arithmétiques*, Rend. Circ. Mat. Palermo **27** (1909), 247–271.

[18] J. M. Borwein and P. B. Borwein, *On the complexity of familiar functions and numbers*, SIAM Rev. **30** (1988), no. 4, 589–601, DOI 10.1137/1030134. MR967961

[19] Yann Bugeaud, *An explicit lower bound for the block complexity of an algebraic number*, Atti Accad. Naz. Lincei Rend. Lincei Mat. Appl. **19** (2008), no. 3, 229–235, DOI 10.4171/RLM/521. MR2439519

[20] Yann Bugeaud, *Automatic continued fractions are transcendental or quadratic* (English, with English and French summaries), Ann. Sci. Éc. Norm. Supér. (4) **46** (2013), no. 6, 1005–1022, DOI 10.24033/asens.2208. MR3134686

[21] Julien Cassaigne and François Nicolas, *Factor complexity*, Combinatorics, automata and number theory, Encyclopedia Math. Appl., vol. 135, Cambridge Univ. Press, Cambridge, 2010, pp. 163–247. MR2759107

[22] Didier Caucal and Marion Le Gonidec, *Context-free sequences*, Theoretical aspects of computing—ICTAC 2014, Lecture Notes in Comput. Sci., vol. 8687, Springer, Cham, 2014, pp. 259–276, DOI 10.1007/978-3-319-10882-7_16. MR3278460

[23] N. Chomsky, *Three models for the description of language*, IRE Trans. Information Theory **18** (1956), 113–124.

[24] A. Cobham, *Functional equations for register machines*, Proccedings of the Hawaii International Conference on System Sciences, Honolulu, pp. 10–13, 1968.

[25] A. Cobham, *A proof of transcendence based on functional equations*, RC–2041, IBM Research Center, Yorktown Heights, New York, 1968.

[26] A. Cobham, *On the Hartmanis-Stearns problem for a class of tag machines*, IEEE Conference Record of 1968 Ninth Annual Symposium on Switching and Automata Theory, pp. 51–60, 1968.

[27] Alan Cobham, *Uniform tag sequences*, Math. Systems Theory **6** (1972), 164–192, DOI 10.1007/BF01706087. MR0457011

[28] Daniel Duverney, Keiji Nishioka, Kumiko Nishioka, and Iekata Shiokawa, *Transcendence of Jacobi's theta series*, Proc. Japan Acad. Ser. A Math. Sci. **72** (1996), no. 9, 202–203. MR1434685

[29] D. Hamm, *Contributions to formal language theory: Fixed points, complexity, and context-free sequences*, M.Sc. Thesis, University of Waterloo, 1998.

[30] J. Hartmanis and R. E. Stearns, *On the computational complexity of algorithms*, Trans. Amer. Math. Soc. **117** (1965), 285–306, DOI 10.2307/1994208. MR170805

[31] John E. Hopcroft and Jeffrey D. Ullman, *Introduction to automata theory, languages, and computation*, Addison-Wesley Publishing Co., Reading, Mass., 1979. Addison-Wesley Series in Computer Science. MR645539

[32] Marion Le Gonidec, *Drunken man infinite words complexity*, Theor. Inform. Appl. **42** (2008), no. 3, 599–613, DOI 10.1051/ita:2008012. MR2434037

[33] Marion Le Gonidec, *On the complexity of a family of k-context-free sequences*, Theoret. Comput. Sci. **414** (2012), 47–54, DOI 10.1016/j.tcs.2011.09.022. MR2896582

[34] J. Liouville, *Sur des classes très étendues de quantités dont la valeur n'est ni algébrique, ni même réductible à des irrationelles algébriques*, C. R. Acad. Sci. Paris **18** (1844), 883–885 et 993–995.

[35] Kurt Mahler, *Zur Approximation der Exponentialfunktion und des Logarithmus. Teil II* (German), J. Reine Angew. Math. **166** (1932), 137–150, DOI 10.1515/crll.1932.166.137. MR1581303

[36] Marvin L. Minsky, *Computation: finite and infinite machines*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1967. Prentice-Hall Series in Automatic Computation. MR0356580

[37] Marston Morse and Gustav A. Hedlund, *Symbolic Dynamics*, Amer. J. Math. **60** (1938), no. 4, 815–866, DOI 10.2307/2371264. MR1507944

[38] Yossi Moshe, *On some questions regarding k-regular and k-context-free sequences*, Theoret. Comput. Sci. **400** (2008), no. 1-3, 62–69, DOI 10.1016/j.tcs.2008.02.018. MR2424342

[39] Yu. V. Nesterenko, *Modular functions and transcendence questions* (Russian, with Russian summary), Mat. Sb. **187** (1996), no. 9, 65–96, DOI 10.1070/SM1996v187n09ABEH000158; English transl., Sb. Math. **187** (1996), no. 9, 1319–1348. MR1422383

[40] Jean-Jacques Pansiot, *Complexité des facteurs des mots infinis engendrés par morphismes itérés* (French, with English summary), Automata, languages and programming (Antwerp, 1984), Lecture Notes in Comput. Sci., vol. 172, Springer, Berlin, 1984, pp. 380–389, DOI 10.1007/3-540-13345-3_34. MR784265

[41] Patrice Philippon, *Groupes de Galois et nombres automatiques* (French, with English and French summaries), J. Lond. Math. Soc. (2) **92** (2015), no. 3, 596–614, DOI 10.1112/jlms/jdv056. MR3431652

[42] N. Pytheas Fogg, *Substitutions in dynamics, arithmetics and combinatorics*, Lecture Notes in Mathematics, vol. 1794, Springer-Verlag, Berlin, 2002. Edited by V. Berthé, S. Ferenczi, C. Mauduit and A. Siegel. MR1970385

[43] Martine Queffélec, *Substitution dynamical systems—spectral analysis*, 2nd ed., Lecture Notes in Mathematics, vol. 1294, Springer-Verlag, Berlin, 2010. MR2590264

[44] J. Shallit, *A second course in formal languages and automata theory*, Cambridge University Press, Cambridge, 2008.

[45] M. Sipser, *Introduction to the theory of computation*, third edition, Wadsworth Publishing Co. Inc., 2012.

[46] Seppo Sippu and Eljas Soisalon-Soininen, *Parsing theory. Vol. I*, Languages and parsing. EATCS Monographs on Theoretical Computer Science, vol. 15, Springer-Verlag, Berlin, 1988. MR960693

[47] A. M. Turing, *On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction*, Proc. London Math. Soc. (2) **43** (1937), no. 7, 544–546, DOI 10.1112/plms/s2-43.6.544. MR1575661

Université Lyon, Université Claude Bernard Lyon 1, CNRS UMR 5208, Institut Camille Jordan, F-69622 Villeurbanne Cedex, France

Email address: Boris.Adamczewski@math.cnrs.fr

CNRS, Aix-Marseille Université, Institut de Mathématiques de Marseille, case 907, 163 avenue de Luminy, 13288 Marseille Cedex 9, France

Email address: Julien.Cassaigne@math.cnrs.fr

Université de la Réunion, Laboratoire d'Informatique et de Mathématiques, Parc technologique universitaire, 2 rue Joseph Wetzell, 97490 Sainte-Clotilde, Réunion France

Email address: marion.le-gonidec@univ-reunion.fr