Mix-Automatic Sequences

Jörg Endrullis[1], Clemens Grabmayer[2], and Dimitri Hendriks[1]

[1] VU University Amsterdam [2] Utrecht University

Abstract. Mix-automatic sequences form a proper extension of the class of automatic sequences, and arise from a generalization of finite state automata where the input alphabet is state-dependent. In this paper we compare the class of mix-automatic sequences with the class of morphic sequences. For every polynomial φ we construct a mix-automatic sequence whose subword complexity exceeds φ. This stands in contrast to automatic and morphic sequences which are known to have at most quadratic subword complexity. We then adapt the notion of k-kernels to obtain a characterization of mix-automatic sequences, and employ this notion to construct morphic sequences that are not mix-automatic.

1 Introduction

Automatic sequences [1] were introduced by Cobham [4] in 1972, and have since been been studied extensively. A sequence $w : \mathbb{N} \to \Delta$ over a finite alphabet Δ is *automatic* if it can be realized by a finite automaton that, for some $k \geq 2$, takes the base-k expansion $(n)_k$ of a number $n \in \mathbb{N}$ as input and outputs the n-th letter of w; in this case w is called *k-automatic*. For multiplicatively independent k and ℓ, k-automaticity and ℓ-automaticity are almost separated notions; e.g., if a sequence is both 2-automatic and 3-automatic, then it is ultimately periodic.

Therefore it is natural to study also nonstandard numeration systems, and the classes of automatic sequences they give rise to. Rigo [10] and Rigo and Maes [11] study 'abstract numeration systems' based on the 'shortlex' order on an infinite regular language, induced by an order on the alphabet. With this concept they precisely capture the class of morphic sequences.

We introduce *dynamic radix* numeration systems which are obtained as a natural generalization from another variation of the standard base-k representation: the *mixed radix* numeration systems [8] in which the base used only depends on the position of a digit. In dynamic radix numeration systems the base used may depend on the input digits read so far. Sequences realized by finite automata that take dynamic radix input we call *mix-automatic*.

We first consider an example of a 2-automatic sequence, the celebrated Thue–Morse sequence, and explain how it is generated by the automaton in Figure 1.

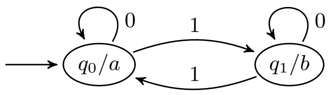


Fig. 1. DFAO generating the Thue–Morse sequence $abbabaabbaababba\cdots$.

The automaton has states $\{q_0, q_1\}$, initial state q_0, input alphabet $\{0, 1\}$ and output alphabet $\{a, b\}$. The output letter assigned to q_0 is a and to q_1 is b (indicated by *state/output* in the states of the automaton). The n-th letter of the sequence is the output of the automaton when reading $(n)_2$, the base-2 expansion of n. For example, for input $(3)_2 = 11$ the automaton ends in state q_0 with output a, and for input $(4)_2 = 100$ in state q_1 with output b.

The automaton of Figure 1 is called a *deterministic finite-state automaton with output (DFAO)*. For $k \geq 2$, a k-DFAO is an automaton over the input alphabet $\mathbb{N}_{<k} = \{0, 1, \dots, k-1\}$. An infinite sequence $w \in \Delta^\omega$ is called k-*automatic* if there exists a k-DFAO such that for every $n \in \mathbb{N}$ the output of the automaton when reading the word $(n)_k \in \mathbb{N}^*_{<k}$ is $w(n)$, with $(n)_k$ the base-k expansion of n.

Mix-Automatic Sequences. The class of automatic sequences is well-known to have good closure properties; for example, it is closed under shifts (prepending letters or removing prefixes), and taking arithmetic subsequences. The class of mix-automatic sequences extends the class of automatic sequences, has all these closure properties, and additionally is closed under k-shuffling, for all $k \geq 2$.

Mix-automatic sequences are defined via *mix-DFAOs*, automata that generalize k-DFAOs by allowing that the alphabet of the symbol to be processed next depends on the current state. Let us consider the example of a mix-DFAO shown in Figure 2. The state q_0 has two outgoing edges, reflecting the input alphabet $\{0, 1\}$, while q_1 has three outgoing edges, reflecting the input alphabet $\{0, 1, 2\}$.

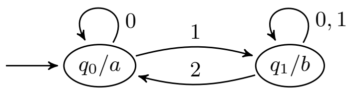


Fig. 2. An example of a mix-DFAO.

Dynamic Radix Numeration Systems. Clearly, the numeration system used for the input of mix-DFAOs cannot be the standard base-k representation. Instead, in the number representation that we let these automata operate on, the base for each digit is determined by the lower-significance digits that have already been read.Thus we let the automata read from the least to the most significant digit (i.e., we let the reading direction be from right to left). We write $(n)_M$ for the number representation of n that serves as input for the automaton M. For M the automaton from Figure 2, the representations of the first eight numbers are

$$(0)_M = \varepsilon \qquad (2)_M = 1_2 0_2 \qquad (4)_M = 1_2 0_2 0_2 \qquad (6)_M = 1_3 1_2 0_2$$
$$(1)_M = 1_2 \qquad (3)_M = 1_3 1_2 \qquad (5)_M = 2_3 1_2 \qquad (7)_M = 1_3 0_3 1_2$$

where a subscript b (not part of the number representation) in d_b indicates the base employed for d. Let us explain this at the example $(17)_M = 1_2 0_2 2_3 1_2$. Knowing the base for each digit, we can reconstruct the value of the representation as follows: $17 = 1 \cdot 2 \cdot 3 \cdot 2 + 0 \cdot 3 \cdot 2 + 2 \cdot 2 + 1$ where each digit is multiplied with the product of the bases of the lower digits. Given just the representation 1021, the base of each of the digits is determined by the input alphabet of the state of the automaton reading the digit. The states q_0 and q_1 of M have input alphabets $\{0, 1\}$ and $\{0, 1, 2\}$ and thus expect the input in base 2 and 3, respectively. When reading 1021 (right to left) the automaton M visits the states q_0, q_1, q_0, q_0 and q_1. Annotating the input digits with the state of the automaton when reading the digit, we obtain $1_{q_0}0_{q_0}2_{q_1}1_{q_0}$, and taking into account the bases expected by these states, yields $1_2 0_2 2_3 1_2$.

We emphasize that, given a mix-DFAO M, every $n \in \mathbb{N}$ has a unique representation $(n)_M = d_m \cdots d_0$ (without leading zeros). This representation can be computed as follows. Assume that we have determined the value of the digits $d_{i-1} \cdots d_0$ with corresponding bases $b_{i-1} \cdots b_0$. The base b_i of digit d_i is determined by the input alphabet of the state of the automaton after reading $d_{i-1} \cdots d_0$ (right to left), and digit d_i is the remainder of the division of $n - \sum_{0 \le j < i} d_j (b_{j-1} \cdots b_1 \cdot b_0)$ by b_i.

Every mix-DFAO M gives rise to a mix-automatic sequence $w \in \Delta^\omega$ by defining for every $n \in \mathbb{N}$, $w(n)$ as the output of M when reading $(n)_M$.

Zip-Specifications. In [6] it has been shown that k-automatic sequences are precisely the class of sequences definable by *zip-k specifications*, that is, systems of recursion equations $\{X_1 = t_1, \ldots, X_n = t_n\}$ with terms t_i built from the syntax

$$t ::= X_i \mid a : t \mid \mathsf{zip}_k(t, \ldots, t) \qquad\qquad (1 \le i \le n,\ a \in \Delta)$$

Semantically, the term notation $a : t$ indicates the concatenation of a letter with a sequence, and the k-ary symbol zip_k stands for the function of type $(\Sigma^\omega)^k \to \Sigma^\omega$ that zips (or interleaves or shuffles) its k argument sequences, and is defined by

$$\mathsf{zip}_k(w_0, \ldots, w_{k-1})(kn + i) = w_i(n) \qquad\qquad (0 \le i < k)$$

Operationally, zip_k can be defined by the rewrite rule

$$\mathsf{zip}_k(x : t_0, t_1, \ldots, t_{k-1}) \to x : \mathsf{zip}_k(t_1, \ldots, t_{k-1}, t_0) \tag{1}$$

The zip operation on finite words is known in the literature as *perfect shuffle* [2]. An example of a zip-2 specification corresponding to the 2-DFAO from Fig. 1 is

$$\mathsf{M} = a : \mathsf{Q}_1 \qquad \mathsf{Q}_0 = a : \mathsf{zip}_2(\mathsf{Q}_0, \mathsf{Q}_1) \qquad \mathsf{Q}_1 = b : \mathsf{zip}_2(\mathsf{Q}_1, \mathsf{Q}_0) \tag{2}$$

The Thue–Morse sequence is the unique solution for the variable M in this specification, or, from a rewriting perspective, it is the infinite normal form of M in the rewrite system consisting of (1) and (2), orienting the equations from left to right. For further details we refer to [6].

The introduction of mix-automatic sequences was motivated by the characterization of k-automatic sequences as the class of sequences that can defined by zip-k specifications, answering the question: *What class of sequences is obtained when allowing zips of different arities in the same specification?* In [6, 7] the correspondence between such 'zip-mix' specifiable sequences and mix-automatic sequences was established. Moreover, it was shown that mix-automaticity properly extends automaticity: for example, shuffling a 2-automatic and a 3-automatic sequence, both not ultimately periodic, is mix-automatic but not automatic.

Contribution and Overview. We continue the study of mix-automatic sequences started in [6, 7] by exploring the relationship with morphic sequences. In Section 3, we generalize the characterization of k-automatic sequences via finite k-kernels to the setting of mix-automatic sequences. In Sections 4 and 5 we show that neither of the classes (a) mix-automatic sequences and (b) morphic sequences subsumes the other. In particular we show that the subword complexity of mix-automatic sequences can exceed any polynomial, whereas it is known [5] that morphic sequences have at most quadratic subword complexity.

2 Preliminaries

We use standard terminology and notation; for example, see Allouche and Shallit [1]. Let Σ be a finite alphabet. Then we denote by

- Σ^* the set of all *finite words over* Σ, by ε the *empty word*,
- $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$ the set of *finite non-empty words*,
- $\Sigma^\omega = \{w \mid w : \mathbb{N} \to \Sigma\}$ the set of *infinite words over* Σ,
- $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ the set of all *(finite or infinite) words.*

For a word $w \in \Sigma^\infty$ and $n \in \mathbb{N}$, we write $w(n)$ for the n-th letter of w (counting from zero). We write $|x|$ for the *length of* $x \in \Sigma^\infty$, with $|x| = \infty$ if x is infinite. We call a word $v \in \Sigma^*$ a *subword* of $x \in \Sigma^\infty$ if $x = uvy$ for some $u \in \Sigma^*$ and $y \in \Sigma^\infty$, and say that v *occurs at position* $|u|$. The *subword complexity* of a sequence $w \in \Sigma^\omega$ is the function $p_w : \mathbb{N} \to \mathbb{N}$ such that $p_w(n)$ is the number of distinct length-n subwords (factors) of w.

Definition 1. A *deterministic finite automaton with output (DFAO)* is a tuple $\langle Q, \Sigma, \delta, q_0, \Delta, \lambda \rangle$ where

- Q is a finite set of states with $q_0 \in Q$ the initial state,
- Σ a finite input alphabet, Δ an output alphabet,
- $\delta : Q \times \Sigma \to Q$ a transition function, and
- $\lambda : Q \to \Delta$ an output function.

We extend the domain of δ to $Q \times \Sigma^*$ by defining, for all $q \in Q$, $\delta(q, \varepsilon) = q$ and

$$\delta(q, xa) = \delta(\delta(q, a), x) \qquad \text{for all } x \in \Sigma^* \text{ and } a \in \Sigma,$$

thus forcing the reading direction from right to left.

For $n, k \in \mathbb{N}$, $k \geq 2$, we let $(n)_k$ denote the canonical *base-k expansion of* n (without leading zeros). More precisely, for $n > 0$ we have

$$(n)_k = d_m d_{m-1} \cdots d_0 \quad \text{where} \quad 0 \leq d_0, \ldots, d_m < k,\ d_m > 0 \text{ and } n = \sum_{i=0}^{m} d_i k^i \,.$$

For $n = 0$ we fix $(n)_k = \varepsilon$. We emphasize that the exclusion of leading zeros in the number representation $(n)_k$ is not crucial. Every DFAO can be transformed into an equivalent DFAO that ignores leading zeros, see [1].

Definition 2. Let $k \geq 2$ and define $\mathbb{N}_{<k} = \{0, \ldots, k-1\}$. A k-DFAO M is a DFAO $\langle Q, \Sigma, \delta, q_0, \Delta, \lambda \rangle$ with the input alphabet $\Sigma = \mathbb{N}_{<k}$.

For $q \in Q$, we define the infinite sequence $\mathrm{seq}(M, q) \in \Delta^\omega$ by $\mathrm{seq}(M, q)(n) = \lambda(\delta(q, (n)_k))$, for every $n \in \mathbb{N}$. We write $\mathrm{seq}(M)$ as shorthand for $\mathrm{seq}(M, q_0)$. The automaton M is said to *generate* the sequence $\mathrm{seq}(M)$.

Now automatic sequences can be defined as follows:

Definition 3. A sequence $w \in \Delta^\omega$ is *k-automatic* if there exists a k-DFAO that generates w. A sequence is called *automatic* if it is k-automatic for some $k \geq 2$.

3 Mix-Automatic Sequences

In this section we introduce mix-automatic sequences. For this purpose, we define finite automata (with output) that have state-dependent input alphabets. As inputs these automata take dynamic radix number representations, which generalize base-k number representations to the effect that the digits are allowed to belong to different bases, and may depend on previously read digits. For specifying the format of the dynamic radix number representation that an automaton can process we use 'base determiners', which are themselves finite automata with (number) output that determine the base of each digit depending on the values of the lower digits. Number representations according to a thus obtained dynamic radix number representation can then serve as inputs for a mix-DFAO. k-DFAOs are special cases of mix-DFAOs. Eventually, we introduce mix-automatic sequences as sequences that are generated by mix-DFAOs.

Deterministic Finite State Automata with State-Dependent Input Alphabet. We introduce finite automata with output for which the input alphabet is dependent on the current state.

Definition 4. A *state-dependent input alphabet DFAO* is a tuple of the form $\langle Q, \Sigma, \delta, q_0, \Delta, \lambda \rangle$ where

- Q is a finite set of states with $q_0 \in Q$ the initial state,
- $\Sigma = \{\Sigma_q\}_{q \in Q}$ is a family of input alphabets,
- $\delta = \{\delta_q : \Sigma_q \to Q\}_{q \in Q}$ is a family of transition functions,
- Δ is an output alphabet, and
- $\lambda : Q \to \Delta$ is an output function.

We interpret δ as a partial function $Q \times \bigcup \Sigma \rightharpoonup Q$, and define $\delta(q, i) = \delta_q(i)$ iff $i \in \Sigma_q$. We extend the domain of δ to $Q \times (\bigcup \Sigma)^*$ by defining for all $q \in Q$, $\delta(q, \varepsilon) = q$, and for all $q \in Q$, $x \in \Sigma^*$, and $a \in \Sigma_q$

$$\delta(q, xa) = \delta(\delta(q, a), x) \qquad \text{if } \delta(\delta(q, a), x) \text{ is defined.}$$

Note that the definition of δ forces the reading direction of input words to be from right to left. An alternative definition of δ is as follows: Let $q \in Q$ and $w = a_{n-1} \cdots a_0$ where $a_i \in \Sigma_{r_i}$ $(0 \leq i < n)$ with $r_i \in Q$ defined (for $0 \leq i \leq n$) by $r_0 = q$ and $r_{i+1} = \delta(r_i, a_i)$; then we set $\delta(q, w) = r_n$.

The following definition generalizes k-DFAOs:

Definition 5. A *mix-DFAO* is a tuple $\langle Q, \beta, \delta, q_0, \Delta, \lambda\rangle$ that represents a state-dependent input alphabet DFAO $\langle Q, \{\mathbb{N}_{<\beta(q)}\}_{q\in Q}, \delta, q_0, \Delta, \lambda\rangle$ with $\beta : Q \to \mathbb{N}_{\geq 2}$.

Obviously, mix-DFAOs require a special number representation as input. The number representation must ensure that the base of each digit matches the input alphabet of the state the automaton is in when reading the digit. This leads to the following generalization of the usual base-k number representations.

Dynamic Radix Numeration Systems and Base Determiners. We now introduce dynamic radix number representations. For defining these representations special mix-DFAOs called 'base determiners' are used to specify the base for each digit depending on the digits that have been read before.

Definition 6. A *base determiner* is a tuple $\langle Q, \beta, \delta, q_0\rangle$ which is a shorthand for the mix-DFAO $\langle Q, \beta, \delta, q_0, \mathbb{N}, \beta\rangle$. The base determiner *underlying* a mix-DFAO $\langle Q, \beta, \delta, q_0, \Delta, \lambda\rangle$ is the base determiner $\langle Q, \beta, \delta, q_0\rangle$.

Let $B = \langle Q, \beta, \delta, q_0\rangle$ be a base determiner. The *base-B representation* of an integer $n \in \mathbb{N}$ is defined by $(n)_B = (n)_{q_0}$ where $(0)_q = \varepsilon$ and for $n > 0$

$$(n)_q = (n')_{\delta(q,d)}\, d\ , \qquad n' = \lfloor n/\beta(q)\rfloor\ , \quad \text{and} \quad d = n - n' \cdot \beta(q)$$

So n' and d are quotient and remainder of division of n by $\beta(q)$, respectively.

Definition 7. Let $B = \langle Q, \beta, \delta, q_0\rangle$ be a base determiner. We define the partial function $[\_]_B : \mathbb{N}^* \rightharpoonup \mathbb{N}$ by $[w]_B = [w, 1]_{q_0}$ where we let $[w, b]_q$ for all $b \in \mathbb{N}$ and $q \in Q$ be defined by

$$[\varepsilon, b]_q = 0 \qquad\qquad [wd, b]_q = [w, b\beta(q)]_{\delta(q,d)} + bd \qquad \text{if } d \in \mathbb{N}_{<\beta(q)}$$

and undefined otherwise.

Note that $[\_]_B$ is the left inverse of $(\_)_B$: for all $b \in \mathbb{N}$ and $q \in Q$ $[(n)_q, b]_q = bn$ follows by induction on $n \in \mathbb{N}$.

We obtain ordinary base-k numbers by defining the base determiner B to consist of a single state q with output k and edges $0, \ldots, k-1$ looping to itself; this is illustrated in Figure 3.

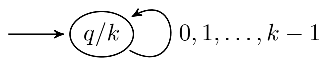
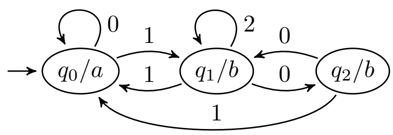


Fig. 3. A base determiner for the standard base-k number representation.

Example 8. Consider the following mix-DFAO M and the dynamic numeration system it defines (where $n > 0$, and $q \in \{q_0, q_1, q_2\}$):



$$\begin{aligned} (2n)_{q_0} &= (n)_{q_0}0 & (0)_q &= \varepsilon \\ (2n+1)_{q_0} &= (n)_{q_1}1 & (3n)_{q_1} &= (n)_{q_2}0 \\ (2n)_{q_2} &= (n)_{q_1}0 & (3n+1)_{q_1} &= (n)_{q_0}1 \\ (2n+1)_{q_2} &= (n)_{q_0}1 & (3n+2)_{q_1} &= (n)_{q_1}2 \end{aligned}$$

Let B be the base determiner underlying M (that is, obtained from M by redefining the output for q_0, q_1 and q_2 as 2, 3 and 2, respectively).

As an example, we compute $(5)_B$, and $(23)_B$ as follows:

$$(5)_B = (5)_{q_0} = (2)_{q_1}1 = (0)_{q_2}21 = 21$$
$$(23)_B = (23)_{q_0} = (11)_{q_1}1 = (3)_{q_1}21 = (1)_{q_2}021 = (0)_{q_0}1021 = 1021 .$$

A k-DFAO is an automaton reading the input in the base-k number format. We generalize this concept to B-DFAOs that expect to read input in the number format defined by the base determiner B.

Definition 9. Let M be a mix-DFAO and B a base determiner. We call M a *B-DFAO* if M is *compatible with* B in the sense that $(n)_B = (n)_{B_M}$ holds for all $n \in \mathbb{N}$, where B_M is the base determiner underlying M.

(Note that M is a B_M-DFAO, i.e., M reads the number format defined by itself.)

A B-DFAO with output alphabet Δ defines a B-automatic sequence $w \in \Delta^\omega$ by defining for all $n \in \mathbb{N}$, $w(n)$ as the output of the DFAO on the input $(n)_B$. Sequences generated by mix-DFAOs we call 'mix-automatic' sequences.

Definition 10. Let B be a base determiner, and $M = \langle Q, \beta, \delta, q_0, \Delta, \lambda \rangle$ a B-DFAO. For states $q \in Q$, we define $\text{seq}(M, q) \in \Delta^\omega$ by:

$$\text{seq}(M, q)(n) = \lambda(\delta(q, (n)_B)) \qquad \text{for all } n \in \mathbb{N}$$

We define $\text{seq}(M) = \text{seq}(M, q_0)$, and say M *generates* the sequence $\text{seq}(M)$.

A sequence $w \in \Delta^\omega$ is *B-automatic* if there exists a B-DFAO M such that $w = \text{seq}(M)$. A sequence is called *mix-automatic* if it is B-automatic for some base determiner B.

Example 11. We continue Example 8. The sequence $\text{seq}(M)$ begins with

$$abbab\underline{b}aabbbaaaabbbbbbaaa\underline{a}ababab bbbbbababab aaaaaababbbabbbabbb\cdots$$

with entries 5 and 23 underlined. E.g. $\lambda(\delta(q_0, 1021)) = a$ since starting from q_0 and reading 1021 from right to left brings you back at state q_0 with output a.

Kernels for Mix-Automatic Sequences. Automatic sequences can be characterized in terms of their 'kernels' being finite. For sequences $w \in \Delta^\omega$ and $i, k \in \mathbb{N}$, $k > 0$ we define

$$\pi_{i,k}(w) = w(i)\, w(i+k)\, w(i+2k)\, w(i+3k)\, w(i+4k) \cdots ,$$

the subsequence of w selecting every k-th element starting form the i-th element (counting from 0). The *k-kernel* $\text{Ker}(k, w)$ of a sequence $w \in \Delta^\omega$ is the set of arithmetic subsequences $\text{Ker}(k, w) = \{\pi_{i,k^p}(w) \mid p \in \mathbb{N}, i < k^p\}$. The set $\text{Ker}(k, w)$ can equivalently be defined as the smallest set K such that $w \in K$, and for all $u \in K$ we have $\pi_{i,k}(u) \in K$ for all $0 \le i < k$; see further [6].

Fact ([1, Thm. 6.6.2]). *A sequence is k-automatic iff its k-kernel is finite.*

We now generalize this characterization to mix-automatic sequences.

Definition 12. Let $x : \Delta^\omega \to \mathbb{N}_{\geq 2}$. The *x-kernel* $\mathrm{Ker}(x, w)$ of a sequence $w \in \Delta^\omega$ is defined as the smallest set $K \subseteq \Delta^\omega$ such that $w \in K$, and for all sequences $u \in K$ we have $\pi_{i,x(u)}(u) \in K$ for all $0 \leq i < x(u)$.

The function $x : \Delta^\omega \to \mathbb{N}_{\geq 2}$ determines for every sequence $w \in \Delta^\omega$ the set of derivative functions $\{\pi_{0,x(w)}, \pi_{1,x(w)}, \ldots, \pi_{x(w)-1,x(w)}\}$ to be applied to w. The ordinary k-kernels ($k \in \mathbb{N}$) are obtained by defining $x(w) = k$ for every $w \in \Delta^\omega$.

Theorem 13. *A sequence $w \in \Delta^\omega$ is mix-automatic if and only if there exists a function $x : \Delta^\omega \to \mathbb{N}_{\geq 2}$ such that the x-kernel of w is finite.*

Proof. We show the less obvious direction, from left to right. For this let $M = \langle Q, \beta, \delta, q_0, \Delta, \lambda \rangle$ be a mix-DFAO that generates a sequence w. For every state $q \in Q$ the equality $\mathrm{seq}(M, q) = \mathsf{zip}_{\beta(q)}(\mathrm{seq}(M, \delta(q, 0)), \ldots, \mathrm{seq}(M, \delta(q, \beta(q) - 1)))$ holds, that is, the sequence generated by a state q is the shuffling of the sequences generated by the successor states of q. As a consequence, whenever M contains states $q_1 \neq q_2 \in Q$, $q_2 \neq q_0$ with $\mathrm{seq}(M, q_1) = \mathrm{seq}(M, q_2)$ we can eliminate q_2 after redirecting all its incoming edges to q_1; this changes the number representation, but leaves the sequence generated by the automaton unaltered. Thus we may assume that $\mathrm{seq}(M, q_1) \neq \mathrm{seq}(M, q_2)$ for all $q_1 \neq q_2 \in Q$. Hence we can define the function $x : \Delta^\omega \to \mathbb{N}_{\geq 2}$ as follows: $x(\mathrm{seq}(M, q)) = \beta(q)$ for every $q \in Q$, and $x(u) = 2$ for all other sequences u. Then it follows immediately that $\mathrm{Ker}(x, w) \subseteq \{\mathrm{seq}(M, q) \mid q \in Q\}$, namely, the set of sequences generated by the reachable states, and that $\mathrm{Ker}(x, w)$ is finite. ☐

We refine this characterization with respect to a given number representation.

Definition 14. Let $B = \langle Q, \beta, \delta, q_0 \rangle$ be a base determiner. The *B-kernel* of a sequence $w \in \Delta^\omega$, which is denoted by $\mathrm{Ker}(B, w)$, is the set $\{u \mid (u, q) \in K\}$ where $K \subseteq \Delta^\omega \times Q$ is the smallest set such that $(w, q_0) \in K$, and $(\pi_{i,\beta(q)}(u), \delta(q, i)) \in K$ for all $(u, q) \in K$ and $i \in \mathbb{N}$ with $0 \leq i < \beta(q)$.

Theorem 15. *A sequence $w \in \Delta^\omega$ is B-automatic iff its B-kernel is finite.*

4 The Subword Complexity of Mix-Automatic Sequences

We show for any polynomial φ there exists a mix-automatic sequence with a subword complexity exceeding φ. It immediately follows that there are mix-automatic sequences that are not morphic. This answers a question of [6].

For $p, n \in \mathbb{N}_{>0}$ with p a prime number, we use $\nu_p(n)$ to denote the *p-adic valuation of n*, that is, the largest integer $k \in \mathbb{N}$ such that p^k divides n. For every prime number p, we define the sequence $\gamma_p \in \{0, 1\}^\omega$ by

$$\gamma_p = (\nu_p(1) \bmod 2)\,(\nu_p(2) \bmod 2)\,(\nu_p(3) \bmod 2)\,\cdots$$

The sequence γ_2 is the well-known *period-doubling sequence* [1, Example 6.4.3]:

$$\gamma_2 = 0100010101000100010001010100010101000101010001000100010101 00\cdots$$

We show that shuffling k sequences from the set $\{\gamma_p \mid p \text{ is prime}\}$ yields a mix-automatic sequence with subword complexity in $\Omega(n^k)$. We first show that each of the sequences has at least linear subword complexity:

Lemma 16. *The subword complexity of γ_p is in $\Omega(n)$ for every prime number p.*

Proof. The Morse–Hedlund theorem [9] asserts that an infinite sequence w is ultimately periodic if and only if for some $n \in \mathbb{N}$ not more than n factors of length n occur in w. Hence, it suffices to show that γ_p is not ultimately periodic. Assume that γ_p would be ultimately periodic. Then there exist $n_0, k > 1$ such that $\nu_p(n) \equiv \nu_p(n+k) \pmod 2$ for every $n \geq n_0$. Let $n = p^{\nu_p(k)+2m+1}$ with $m \in \mathbb{N}$ such that $n \geq n_0$. Then $\nu_p(n) = \nu_p(k) + 2m + 1$ and $\nu_p(n+k) = \nu_p(k)$, and hence $\nu_p(n) \not\equiv \nu_p(n+k) \pmod 2$ contradicting the assumption. ☐

We moreover employ that the sequences have the following regular structure:

Lemma 17. *Let p be a prime number, $k \in \mathbb{N}$ and w the prefix of length $p^k - 1$ of the sequence γ_p. Then w occurs in γ_p at every position $n \cdot p^k$ ($n \in \mathbb{N}$).*

Proof. Let $0 \leq i < p^k - 1$. Then we have $\gamma_p(n \cdot p^k + i) \equiv \nu_p(n \cdot p^k + i + 1) \pmod 2$ and $\nu_p(n \cdot p^k + i + 1) = \nu_p(i+1)$ for every $n \in \mathbb{N}$. ☐

Lemma 18. *Let $k > 0$, $p_1, \ldots, p_k$ be pairwise distinct primes. Then the sequence $\mathsf{zip}_k(\gamma_{p_1}, \gamma_{p_2}, \ldots, \gamma_{p_k})$ is mix-automatic and its subword complexity is in $\Omega(n^k)$.*

Proof. By Lemma 16 the sequences $\gamma_{p_1}, \ldots, \gamma_{p_k}$ have subword complexity in $\Omega(n)$. Consequently, for proving that the subword complexity of $\mathsf{zip}_k(\gamma_{p_1}, \gamma_{p_2}, \ldots, \gamma_{p_k})$ is in $\Omega(n^k)$, it suffices to show the following: for every $n \in \mathbb{N}$ whenever $w_1, \ldots, w_k$ are n-length subwords of the sequences $\gamma_{p_1}, \ldots, \gamma_{p_k}$, respectively, the shuffle $\mathsf{zip}_k(w_1, \ldots, w_k)$ of length kn is a subword of $\mathsf{zip}_k(\gamma_{p_1}, \gamma_{p_2}, \ldots, \gamma_{p_k})$.

To this end, we show $(*)$ there exists a position $q \in \mathbb{N}$ such that for all i ($1 \leq i \leq k$), the word w_i occurs in γ_{p_i} at position q. Let $\ell_1, \ldots, \ell_k$ be such that every w_i ($1 \leq i \leq k$) occurs in the prefix of γ_{p_i} of length $p_i^{\ell_i} - 1$. Let $o_1, \ldots, o_k$ be the positions of the first occurrences of $w_1, \ldots, w_k$ in $\gamma_{p_1}, \ldots, \gamma_{p_k}$, respectively. (All of these positions are in the respective prefixes of γ_{p_i} of length $p_i^{\ell_i} - 1$.) We proceed by induction on $1 \leq i \leq k$ to construct integers $a_i, b_i > 0$ such that

(i) for all $1 \leq j \leq i$, the word w_j occurs at all positions $a_i + m \cdot b_i$ ($m \in \mathbb{N}$), and
(ii) for all $i < j \leq k$, b_i is coprime with p_j, i.e., $\gcd(b_i, p_j) = 1$.

For $i = 1$, we choose $a_1 = o_1$ and $b_1 = p_1^{\ell_1}$. Then, as a consequence of Lemma 17, the word w_1 occurs at every position $a_1 + m \cdot b_1$ ($m \in \mathbb{N}$).

Let $i < k$ and $c_{i+1} = p_{i+1}^{\ell_{i+1}}$. From (ii) it follows that b_i and c_{i+1} are coprime. By Euler's theorem, there exists $1 \leq e_{i+1} \in \mathbb{N}$ such that $b_i^{e_{i+1}} \equiv 1 \pmod{c_{i+1}}$.

As a consequence we can find some $0 \leq a'_i < c_{i+1}$ and define $a_{i+1} = a_i + a'_i \cdot b_i^{e_{i+1}}$ such that $a_{i+1} \equiv o_{i+1} \pmod{c_{i+1}}$. We let $b_{i+1} = c_{i+1} \cdot b_i^{e_{i+1}}$. Then we have:

$$a_{i+1} = a_i + (a'_i \cdot b_i^{e_{i+1}-1}) \cdot b_i \qquad\qquad b_{i+1} = (c_{i+1} \cdot b_i^{e_{i+1}-1}) \cdot b_i$$

We have $\{a_{i+1} + m \cdot b_{i+1} \mid m \in \mathbb{N}\} \subseteq \{a_i + m \cdot b_i \mid m \in \mathbb{N}\}$, and hence for every $1 \leq j \leq i+1$, the word w_j occurs in γ_{p_j} at all positions $a_{i+1} + m \cdot b_{i+1}$ with $m \in \mathbb{N}$. Moreover $a_{i+1} + m \cdot b_{i+1} \equiv o_{i+1} \pmod{c_{i+1}}$ for every $m \in \mathbb{N}$, and thus by Lemma 17, w_{i+1} occurs in $\gamma_{p_{i+1}}$ at all positions $a_{i+1} + m \cdot b_{i+1}$ with $m \in \mathbb{N}$. We have that $b_{i+1} = (c_{i+1} \cdot b_i^{e_{i+1}}) = p_{i+1}^{\ell_{i+1}} \cdot b_i^{e_{i+1}}$ and thus b_{i+1} and b_j are coprime for every $i+1 < j \leq k$.

Finally, we define $q = a_k$ and by induction hypothesis (i) we have $(*)$. $\square$

Morphic sequences have at most quadratic subword complexity [5]. Hence, by Lemma 18 the mix-automatic sequences $\mathsf{zip}_k(\gamma_{p_1}, \gamma_{p_2}, \ldots, \gamma_{p_k})$ for $k > 2$ are not morphic.

Theorem 19. *The class of mix-automatic sequences is not contained in the class of morphic sequences.*

5 Morphic Sequences that are not Mix-Automatic

In the previous section, we have seen that the class of mix-automatic sequences is not contained in the class of morphic sequences. We now show that the reverse holds as well, that is, there exist morphic sequences that are not mix-automatic. In particular, we consider the characteristic sequence of (positive) squares:

$$\begin{aligned} \mathsf{squares} &= 1001000010000001000000001000000000010000000000001 \cdots \\ &= 10^2 10^4 10^6 10^8 10^{10} 10^{12} 10^{14} 1 \cdots \end{aligned}$$

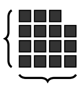
So $\mathsf{squares} \in \{0,1\}^\omega$ is defined by $\mathsf{squares}(n) = 1$ iff $n+1$ is a square number. The sequence is morphic: it can be obtained by iterating the morphism $a \mapsto a001$, $0 \mapsto 0$, $1 \mapsto 001$ on the starting letter a, and applying the coding $a \mapsto 1$, $0 \mapsto 0$ and $1 \mapsto 1$ to the limit word.

We show that $\mathsf{squares}$ is not mix-automatic.

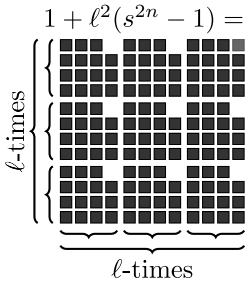
Lemma 20. *Let $\ell, s \in \mathbb{N}$ be such that $\ell, s > 1$. Then there exists a number $n \in \mathbb{N}$ such that $1 + \ell^2(s^n - 1)$ is not a square number.*

Proof. Let $\ell, s \in \mathbb{N}$ be such that $\ell, s > 1$. Let k be large enough to ensure $\ell < 2s^k$. Then $(\ell s^k - 1)^2 = \ell^2 s^{2k} - 2\ell s^k + 1 < 1 + \ell^2(s^{2k} - 1) < (\ell s^k)^2$ follows, which for $n = 2k$ traps $1 + \ell^2(s^n - 1)$ in between consecutive squares.

Alternatively, a geometrical rendering is the following. We view $s^{2n} - 1 = (s^n)^2 - 1$ as a square of $s^n \times s^n$ pieces of which one corner piece has been removed:

s^n-times  $= s^{2n} - 1$

s^n-times



Then $1 + \ell^2(s^{2n} - 1)$ can be visualized as shown on the right. We have $\ell^2 - 1$ cut-out corner pieces. Due to these, $1 + \ell^2(s^{2n} - 1)$ is strictly less than a square $\ell s^n \times \ell s^n$. The next smaller square has size $(\ell s^n - 1) \times (\ell s^n - 1)$ and has precisely $(2\ell s^n - 1)$ less pieces than the larger square. By picking n large enough so that $\ell < 2s^n$, we achieve $\ell^2 - 1 < 2\ell s^n - 1$, and hence there are too many pieces for the next smaller square. ☐

Lemma 21. *The sequence* squares *is morphic but not mix-automatic.*

Proof. The morphic definition of squares is given above. For a contradiction, let us assume that the sequence would be mix-automatic. Then by Theorem 13, there exists $x : \Delta^\omega \to \mathbb{N}_{\geq 2}$ such that the x-kernel K of squares is finite. For every $n \in \mathbb{N}$ we define $w_n \in K$ and $k_n \in \mathbb{N}$ inductively as follows: $w_0 = \mathsf{squares}$ and $w_{n+1} = \pi_{0,k_n}(w_n)$ where $k_n = x(w_n)$. As K is finite, there exist $a, b \in \mathbb{N}$, $a < b$ such that $w_a = w_b$. We define $k = k_0 \cdot k_1 \cdots k_{a-1}$, and $\ell = k_a \cdot k_{a+1} \cdots k_{b-1}$. Then $w_a = \pi_{0,k}(\mathsf{squares})$ and $w_a = w_b = \pi_{0,\ell}(w_a)$, and in particular

$$\pi_{0,k}(\mathsf{squares}) = \pi_{0,\ell}(\pi_{0,\ell}(\pi_{0,k}(\mathsf{squares}))) = \pi_{0,k\ell^2}(\mathsf{squares})\,. \tag{3}$$

Thus (†) for all $n \in \mathbb{N}$, $kn + 1$ is a square if and only if $k\ell^2 n + 1$ is a square.

Let p be a prime that does not divide k, and hence is coprime to k. Then by Euler's theorem there exists $e \in \mathbb{N}$ such that $(p^2)^e \equiv 1 \pmod{k}$. Thus $(*)$ for every $m \in \mathbb{N}$, we have that $(p^{em})^2 = ((p^2)^e)^m \equiv 1 \pmod{k}$ and hence a square number of the form $kn + 1$ for some $n \in \mathbb{N}$.

We define $s = (p^2)^e$. Then an application of Lemma 20 yields that there exists $m \in \mathbb{N}$ such that $1 + \ell^2(s^m - 1)$ is not a square number. We have that $s^m = kn + 1$ for some $n \in \mathbb{N}$ by $(*)$. Thus $1 + \ell^2(s^m - 1) = 1 + \ell^2 kn$ is not a square while $1 + kn$ is. This contradicts (†). ☐

Theorem 22. *The class of morphic sequences is not contained in the class of mix-automatic sequences.*

6 Conclusions and Further Research

Mix-automatic sequences form a natural extension of the class of automatic sequences. While automatic sequences are generated by DFAOs, mix-automatic sequences are generated by DFAOs with state-dependent input alphabets. These automata read number representations $d_n d_{n-1} \cdots d_0$ where the base of a digit d_k depends on the value of the lower-significance digits $d_{k-1} \cdots d_0$.

The results of this paper can be summarized as follows:

(i) A characterization of mix-automatic sequences via a generalization of the concept of k-kernel (by which automatic sequences can be characterized).

(ii) For every polynomial φ there is a mix-automatic sequence whose subword complexity exceeds φ. As a consequence there are mix-automatic sequences that are not morphic, since morphic sequences have quadratic subword complexity at most.
(iii) A morphic sequence that is not mix-automatic, showing that the class of morphic sequences is not contained in the class of mix-automatic sequences.

All of these concepts are very recent, and many interesting questions remain. We highlight three particularly intriguing, and challenging questions:

(1) (J.-P. Allouche) Characterize the intersection of mix-automatic and morphic sequences. (Note that at least all automatic sequences are in.)
(2) Is the following problem decidable: Given two mix-DFAOs, do they generate the same sequence?
(3) Can Cobham's Theorem (below) be generalized to mix-automatic sequences?

Cobham's Theorem ([3]). Let $k, \ell \geq 2$ be multiplicatively independent (i.e., $k^a \neq \ell^b$, for all $a, b > 0$), and let $w \in \Delta^\omega$ be both k- and ℓ-automatic. Then w is ultimately periodic.

In order to generalize this theorem to mix-automatic sequences, one could look for a suitable notion of multiplicative independence for base determiners. Recall that base determiners are themselves finite automata with output.

References

1. J.-P. Allouche and J. Shallit. *Automatic Sequences: Theory, Applications, Generalizations*. Cambridge University Press, New York, 2003.
2. J. Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.
3. A. Cobham. On the Base-Dependence of Sets of Numbers Recognizable by Finite Automata. *Mathematical Systems Theory*, 3(2):186–192, 1969.
4. A. Cobham. Uniform Tag Sequences. *Theory of Computing Systems*, 6:164–192, 1972.
5. A. Ehrenfeucht, K. P. Lee, and G. Rozenberg. Subword Complexity of Various Classes of Deterministic Languages without Interaction. *Theoretical Computer Science*, 1:59–75, 1975.
6. C. Grabmayer, J. Endrullis, D. Hendriks, J. W. Klop, and L. S. Moss. Automatic Sequences and Zip-Specifications. In *Proc. Symp. on Logic in Computer Science (LICS 2012)*, pages 335–344. IEEE Computer Society, 2012.
7. C. Grabmayer, J. Endrullis, D. Hendriks, J. W. Klop, and L. S. Moss. Automatic Sequences and Zip-Specifications. Technical Report 1201.3251, arXiv, 2012.
8. D. E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison-Wesley, 2nd edition, 1981.
9. M. Morse and G. A. Hedlund. Symbolic Dynamics. *American Journal of Mathematics*, 60:815–866, 1938.
10. M. Rigo. Generalization of Automatic Sequences for Numeration Systems on a Regular Language. *Theoretical Computer Science*, 244(1-2):271–281, 2000.
11. M. Rigo and A. Maes. More on Generalized Automatic Sequences. *Journal of Automata, Languages and Combinatorics*, 7(3):351–376, 2002.