RAIRO-Theor. Inf. Appl. **42** (2008) 451–466
DOI: 10.1051/ita:2008009

Available online at:
www.rairo-ita.org

SOME ALGEBRAIC PROPERTIES OF MACHINE POSET OF INFINITE WORDS

Aleksandrs Belovs[1]

Abstract. The complexity of infinite words is considered from the point of view of a transformation with a Mealy machine that is the simplest model of a finite automaton transducer. We are mostly interested in algebraic properties of the underlying partially ordered set. Results considered with the existence of supremum, infimum, antichains, chains and density aspects are investigated.

Mathematics Subject Classification. 03D40, 20F10.

Introduction

A finite automaton can be viewed as a machine model which is as elementary as possible in the sense that the machine has a memory size which is fixed and bounded. The number of possible states of such a machine is itself bounded, whence the notion of a finite-state machine.

A Mealy machine [11,16] is a finite state machine that acts, taking a string on an input alphabet and producing a string of equal length on an output alphabet. This model, namely, Mealy machine, is being investigated intensively since the nineteen fifties (*cf.* [3,6,13,19,20]). It is possible to rise similar questions as in this article for a wider class of finite transducers, but we would like to narrow the investigation only for Mealy machine as a kind of a canonical case.

A word is a sequence of symbols, finite or infinite, taken from a finite alphabet. Words are central objects of automata theory, and in fact in any standard model of computing. Also words themselves is the object of study of the mathematical discipline. Namely, combinatorics on words is dealing with them. During the last twenty years many papers and monographs appeared in this field (*cf.* [1,7–9]). In the contest of infinite words it is worth to mention monograph [12].

Keywords and phrases. Infinite words, Mealy machine, poset, algebraic properties.

[1] Department of Mathematics, University of Latvia, Raiņa bulvāris 19, Rīga, Latvia; stiboh@inbox.lv

Article published by EDP Sciences

In this article we are interested in the complexity of infinite words from the point of view of their transformation with Mealy machines. Nowadays, when streams of information grows enormously, it becomes more homogeneous, but underlying machines pretends to become as simple as possible, sharpened to the specific task, such a question can be indeed interesting.

In any case, it is not a radically new point of view. The subject of finite automata on infinite words was established in the sixties by Büchi [2] and McNaughton [10]. From this core the theory has developed into many directions.

One of the main aspects of a modern mathematics, as it has been widely noticed, most known, probably, in the Bourbaki's "Élements de Mathematique" is not a study of concrete elements, but rather a study of different arising structures. In different areas of mathematics people consider a lot of hierarchies which are typically used to classify some objects according to their complexity.

As an example we can remind a problem of the classification of subsets of the natural number set $\mathbb{N}$ with a relation of a reduction of one set (non-calculable one in general) to another one. Algebraic properties of this structure is already a classics and can be found for example in [14]. A similar question for the case of finite transducers is not studied in such details.

In the world on infinite words similar problems are investigated by the Wagner hierarchy theory [17]. It deals with the Wadge preorder on ω-rational sets, that is an analogous of regular sets for infinite words. The Wadge preoder arose from topology, but is turned out that continuity can be replaced by asynchronous sequential functions. Interested reader may refer to the fifth chapter of [12] for more details. As it has been already said, we study possibly a more natural notion from the computation point of view: the preorder of infinite words themselves using finite automata.

1. Preliminaries

In this section we present most of the notations and terminology that will be needed for understanding the foregoing text. Our terminology is more or less standard (*cf.* [4,7–9]) so that a specialist reader may wish to consult this section only if a need arise.

Let A be a finite non-empty set, we shall call *alphabet*, and A^* be a free monoid generated by A. It is built of finite sequences of elements of A with a concatenation operation. The identity element of A^*, designated as ε, is called the *empty word*.

We shall use a notation

$$\prod_{i=0}^{k} u_i = u_0 u_1 u_2 \cdots u_k$$

for complex concatenations (here u_i are finite words on the same alphabet).

If $w = w_0 w_1 \dots w_{l-1} \in A^*$ (here $w_i \in A$) then l is called *length* of w and is denoted as $|w|$. The length of ε by the definition is equal to zero. We set $w^0 = \varepsilon$ and $w^{i+1} = w^i w$.

The concatenation of sets of words is understood as a set that consists of all element wise concatenations. So, for example the notation A^ℓ stands for the set of all words, of length ℓ, over the alphabet A.

The word w' is a *factor* of w (notation: $w' \leq w$) if $w = uw'v$ for some u and v in A^*. If $u = \varepsilon$ or $v = \varepsilon$ then w' is called respectively *prefix* or *suffix* of w. We denote by $\mathrm{F}(w)$, $\mathrm{Pref}(w)$ and $\mathrm{Suff}(w)$ respectively the sets of all factors, prefixes and suffixes of the finite word w.

If $w = w_0 w_1 \dots w_l$ then the notation $w[i, j]$ (with $0 \leq i \leq j \leq l$) stands for the factor $w_i w_{i+1} \dots w_j$. We shall use notation $w[i]$ instead of $w[i, i]$. An *occurrence* of a factor v in w is such a pair $\langle i, j \rangle$ that $v = w[i, j]$. We use notation $\langle \dots \rangle$ to denote tuples.

An (one-sided) infinite word x on the alphabet A is any map $x : \mathbb{N} \to A$. We write $x = x_0 x_1 x_2 \dots$ The set of infinite words on A is denoted A^ω. All the definitions made before can be applied to this case also, only the concatenation operation xy in $A^* \cup A^\omega$ is defined if x is finite. Hence, prefixes and factors of an infinite word are finite, but suffixes are infinite. The suffix $x_i x_{i+1} \dots$ is denoted by $x[i, \infty[$.

A sequence of finite words $\{w_i\}$ converge to an infinite word y, and we write $y = \lim_{n\to\infty} w_n$ if

$$\forall i \in \mathbb{N} \quad \exists N \in \mathbb{N} \quad \forall m > N : w_m[i] = y[i].$$

As a special case we have

$$\prod_{i=0}^{\infty} u_i = \lim_{k\to\infty} \prod_{i=0}^{k} u_i,$$

and also we set $u^\omega = \lim_{n\to\infty} u^n$ for any non-empty u.

An infinite word of the form u^ω is called *periodic*. Words of the form uv^ω are called *ultimately periodic*; lengths of the u and v are called respectively *pre-period* and *period* of the ultimately periodic word uv^ω.

We shall use the same notations for prefixes of ultimately periodic words. Of course, each finite word is an ultimately periodic (even periodic) one, the matter is in the sizes of its pre-period and period. We shall get use of the following theorem [5]

Theorem 1.1 (Fine-Wilf)**.** *Let a and b be periodic words of periods m and n respectively. If a and b agree of prefix of length $m + n - \gcd(m, n)$ then $a = b$.*

A function $\mu : A^* \to B^*$ is called *morphism* if $\mu(\varepsilon) = \varepsilon$ and for all $u, v \in A^*$, $\mu(uv) = \mu(u)\mu(v)$. The morphism is uniquely defined by its values on the letters. The morphism μ is called *non-erasing* if $\forall a \in A : \mu(a) \neq \varepsilon$. It is called *uniform* if $\forall a, b \in A : |\mu(a)| = |\mu(b)|$. It is called *literal* if $\forall a \in A : |\mu(a)| = 1$. It is clear that any literal morphism is also an uniform and a non-erasing one. The morphism μ

can be applied also to an infinite word like this

$$\mu(x) = \lim_{n\to\infty} \mu(x[0,n]),$$

if the limit exists. It does if μ is non-erasing.

In a same way it is possible to define a morphism with more than one argument. Let A_i $(i \le k)$, B be alphabets, $x_i \in A_i^\omega$ and f be a function from $A_0 \times A_1 \times \ldots \times A_k$ to B^*. We extend $f : A_0^\omega \times \ldots \times A_k^\omega \to B^\omega$ setting:

$$f(x_0, x_1, \ldots, x_k) = \prod_{i=0}^{\infty} f(x_0[i], x_1[i], \ldots, x_k[i]).$$

Such a morphism is called *literal* if $\forall x_0, x_1, \ldots x_k : |f(x_0, x_1, \ldots, x_k)| = 1$. For example, we can define a *product* of two or more words $x_0 \times x_1 \times \ldots x_k$, taking the identity morphism in the previous definition (with $B = A_0 \times A_1 \times \ldots \times A_k$).

If u and v are finite words of equal length we can define the (perfect) shuffle $u \sqcup v$ as the finite word $u[0]v[0]u[1]v[1]\ldots u[k]v[k]$ consisting of the alternating u and v symbols. If x and y are infinite words we can define $x \sqcup y = \lim_{k\to\infty} u[0,k] \sqcup v[0,k]$.

And vice versa, for each infinite word x on A we can define a *blocking* $B(x)$ as $B(x) = u \times v = \langle x[0], x[1]\rangle\langle x[2], x[3]\rangle \ldots \in (A^2)^\omega$, where $u \sqcup v$ is the only representation of the word x as a shuffle.

Another simple operation on an infinite word is the *shift operation*. If $x \in A^\omega$ then the result of applying the shift operation is $\sigma x = x[1, \infty[$. We shall use also the inverse operation $\sigma^{-1}x = \lambda x$, where $\lambda \in A$ is an arbitrary element and the product on the right side is a concatenation operation in A^ω. We shall not care about the exact value of λ.

A 3-sorted algebra $V = \langle Q, A, B, \circ, * \rangle$ is called *Mealy machine* if Q, A, B are finite non-empty sets and $\circ : Q \times A \to Q$, $* : Q \times A \to B$ are functions. The sets Q, A and B are called respectively *state set*, *input alphabet* and *output alphabet*.

The mappings $\circ$ and $*$ can be extended to $Q \times A^*$ by defining

$$\begin{aligned} q \circ \varepsilon &= q, & q \circ (ua) &= (q \circ u) \circ a, \\ q * \varepsilon &= \varepsilon, & q * (ua) &= (q * u)((q \circ u) * a), \end{aligned}$$

for all $q \in Q, u \in A^*$ and $a \in A$. Henceforth we shall omit parentheses, assuming that $\circ$ and $*$ have equal priorities, that is higher than the priority of concatenation and lower than the one of taking factors. So $q \circ u * x[5,6]y = ((q \circ u) * (x[5,6]))y$. If x is an infinite word and $q \in Q$ we put $q * x = \lim_{n\to\infty} q * x[0,n]$.

A 3-sorted algebra $V_0 = \langle Q, A, B, q_0, \circ, * \rangle$ is called *initial Mealy machine* if $\langle Q, A, B, \circ, * \rangle$ is a Mealy machine with the *initial state* $q_0 \in Q$. We say the machine V_0 *transforms* a word x into a word y (notation: $x \xrightarrow{V_0} y$) if $y = q_0 * x$. We shall write $x \rightharpoonup y$ if there exists such a Mealy machine V that $x \xrightarrow{V} y$.

Whenever possible we would like to describe appearing Mealy machines informally, leaving formal constructions to the reader.

A set L, with a defined binary relation $\geq$ on it, is called a *partially ordered set (poset)* or just an *order*, if it fulfills the following three axioms:

$$\begin{array}{ll} x \geq x & \text{(reflexivity)} \\ x \geq y, y \geq z \Rightarrow x \geq z & \text{(transitivity)} \\ x \geq y, y \geq x \Rightarrow x = y & \text{(anti-symmetry).} \end{array}$$

Whenever $x \geq y$ and $x \neq y$ we write $x > y$. If a relation fulfills only the reflexivity and the transitivity axioms then such an algebraic structure is called a *preorder*.

Let $\langle L, \geq \rangle$ be some partially ordered set and a be an element of L, a is called the *largest element* iff $\forall x \in L : a \geq x$; a is called the *smallest element* iff $\forall x \in L : x \geq a$; a is called *maximal* iff $\forall x \in L : x \geq a \Rightarrow x = a$; a is called *minimal* iff $\forall x \in L : a \geq x \Rightarrow x = a$

For each two elements x and y of L their *upper bound* is such an element z that $z \geq x$ and $z \geq y$. The *supremum* is the smallest upper bound, i.e. such an element $z = x \vee y$ that not only $z \geq x, y$, but for any other element $t \geq x, y$ holds $t \geq z$. Similarly, z is an *lower bound* if $x, y \geq z$; the largest lower bound is called the *infimum*.

Such a poset that every two elements have a supremum is called an *upper semi-lattice*. If every two elements have infimum, it is called a *lower semi-lattice*. A poset that is both an upper and a lower semi-lattice is called a *lattice*.

2. Machine poset of infinite words

Our main object of investigation is the machine poset of infinite words. In order to avoid some set-theoric problems we shall make some assumptions. Let us take the set

$$\mathfrak{N} = \bigcup_{k=0}^{\infty} \{0, 1, \ldots, k\}^{\omega}.$$

We shall assume that all the states of the Mealy machine, the input and the output alphabets are from the set $\mathbb{N}$. *But if we shall use another input or output alphabet O, we shall assume that there is a bijection $k : O \to \{0, 1, \ldots, |O| - 1\}$ fixed, and this bijection is applied to the input or the output word respectively.*

We would like to explore $\rightharpoonup$ as an algebraic relation on $\mathfrak{N}$.

Proposition 2.1. *The algebraic structure $\langle \mathfrak{N}, \rightharpoonup \rangle$ is a preorder.*

Proof. In other words, the relation $\rightharpoonup$ is reflexive and transitive. Indeed, x can be transformed into itself using a machine with one state that outputs exactly the symbol it gets on the input. If $x \overset{M}{\rightharpoonup} y$ and $y \overset{N}{\rightharpoonup} z$ then $x \rightharpoonup z$ with a machine consisting of the machines M and N. It moves the input symbol to M, its output redirects to the input of N and, finally, the output of N redirects as its own output. □

We can make a canonical transformation to the order, defining $x \sim y$ iff $(x \rightharpoonup y) \vee (y \rightharpoonup x)$, and defining $\mathfrak{K} = \mathfrak{N}/\sim$. The set $\mathfrak{K}$ is build up of equivalence classes

of $\mathfrak{N}$ under the relation $\sim$. We shall say $A \rightharpoonup B$, where $A, B \in \mathfrak{K}$, if for at least one pair (and then for all such pairs) $x \in A$ and $y \in B$ holds $x \rightharpoonup y$ (as it has been defined in the poset $\mathfrak{N}$).

Thus, the algebraic structure $\langle \mathfrak{K}, \rightharpoonup \rangle$ is a poset. This is the poset we shall explore in this article. It consists of the sets of words, but we shall freely switch from the set to an arbitrary its element.

The main technique that can be applied is the cycling of finite automata. As an easy example we shall mention the following theorem due to Yablonsky [18].

Theorem 2.2 (Yablonsky)**.** *The set of ultimately periodic words forms one equivalence class in* $\mathfrak{N}$ *and, henceforth, one element of* $\mathfrak{K}$*. It is the smallest element of* $\mathfrak{K}$*.*

We shall give the proof of this theorem because technics of this proof will be used further in this paper.

Proof. Let us take an ultimately periodic word $x = uv^\omega$ and a machine M with n states and the initial state q_0. Consider its states

$$q_0 \circ u, q_0 \circ uv, q_0 \circ uv^2, \dots, q_0 \circ uv^n. \tag{1}$$

Applying Pigeonhole principle, we can state there are two equal states, say $q_0 \circ uv^k$ and $q_0 \circ uv^l$ $(0 \le k < l \le n)$. Now using induction it is easy to prove $q_0 \circ x[0, |uv^k| + m] = q_0 \circ x[0, |uv^l| + m]$ for any $m > 0$, and hence, $(q_0 * x)[|uv^k| + m] = (q_0 * x)[|uv^l| + m]$. So, $q_0 * x$ is an ultimately periodic word with a period $(l-k)|v|$.

On the other hand, any word (non-periodic as well) can be transformed into an arbitrary ultimately periodic word uv^ω, using a machine that outputs u (denote its state after this operation with q), then ad infinitum outputs v stating itself into the state q after that. □

We say that a machine M *cycles* on an ultimately periodic word uv^n if some two states of (1) are equal. Using the proof of the previous theorem, it is easy to see that the following result holds. We shall widely use it further in the paper.

Lemma 2.3. *If* M *is a Mealy machine with* n *states and* x *is an ultimately periodic word with pre-period* k *and period* l *then the result of transformation of* x *with the machine* M *is an ultimately periodic word with pre-period* $\le k + \ln$ *and period* $\le \ln$*.*

Of course, the same result holds for finite ultimately periodic words as well. In this case we shall be mostly interested in the estimations of pre-period and period. Let us make one more definition:

Definition 2.4. Let uv^k be a finite ultimately periodic word and M such a machine that cycles on it when starts its procession in state q (f.e., such that satisfies the conditions of Lem. 2.3). We say that a symbol $b \ne v[0]$ *breaks the periodicity* if $q * uv^k b \ne q * uv^k v[0]$.

Let us also note some relations between the transformation relation and operations on words.

Proposition 2.5. *Let B_i, C be alphabets. If $x \rightharpoonup y_i$ for $y_i \in B_i$, $i = 0, 1, \ldots, k$ and $f : B_0^\omega \times B_1^\omega \times \cdots \times B_k^\omega \to C$ is an arbitrary literal morphism then $x \rightharpoonup f(y_0, y_1, \ldots, y_k)$.*

Proof. Obvious. □

Transformation relation goes on with the shuffle operation.

Proposition 2.6. *If $x_1 \rightharpoonup y_1$ and $x_2 \rightharpoonup y_2$ then $x_1 \sqcup x_2 \rightharpoonup y_1 \sqcup y_2$. If u is an ultimately periodic word then the mapping $x \mapsto x \sqcup u$ is an isomorphism of $\mathfrak{K}$ on its subset.*

Proof. Indeed, if $x_i \overset{M_i}{\rightharpoonup} y_i$ then $x_1 \sqcup x_2$ is transformable into $y_1 \sqcup y_2$ using a machine that simulates M_1 and M_2 and in the alternating way gives the input symbol on the input of one of them, changing its state, and outputs its output.

The fact that $x \mapsto x \sqcup u$ is a morphism follows from the previous paragraph. Suppose $x \sqcup u \overset{M}{\rightharpoonup} y \sqcup u$. Then we can transform x into y using a machine that contains the machine M and the machine N that produces u (its existence follows from Th. 2.2). It takes a symbol of x, gives it to the input of M, its output redirects to the output tape, gets the output of N, gives it to M ignoring the output, and so on. □

Remark 2.7. Substituting the machine N in the proof of proposition 2.6 with an arbitrary machine N: $x \overset{N}{\rightharpoonup} u$, we can get that the mapping $x \mapsto x \sqcup u$ is an isomorphism of $\{x \in \mathfrak{K} \mid x \rightharpoonup u\}$ onto a subset of $\{x \in \mathfrak{K} \mid x \rightharpoonup u \sqcup u\}$.

As an inversion of the previous proposition, the following result holds:

Proposition 2.8. *If $x \rightharpoonup y$ then the same is for their blockings: $B(x) \rightharpoonup B(y)$.*

Proof. Obvious. □

3. Main properties

Theorem 3.1. *The partially ordered set $\mathfrak{K}$ is an upper semi-lattice.*

Proof. For any $x \in A^\omega$ and any $y \in B^\omega$ as its supremum we can take the word $x \times y \in (A \times B)^\omega$ defined with $(x \times y)[i] = (x[i], y[i])$. That it is an upper bound of x and y and its minimality easy follows now from Proposition 2.5. □

In the proof of the previous result we enlarged the alphabet to $A \times B$ in order to get the supremum. It is possible to proof this enlargement is indeed needed.

Theorem 3.2. *Let us consider three alphabets A, B and C with $|C| < |A| \cdot |B|$. Then there exist two infinite words $x \in A^\omega$ and $y \in B^\omega$ such that for any $z \in C^\omega$ either $z \not\rightharpoonup x$ or $z \not\rightharpoonup y$.*

Proof. We will use the diagonalization method. The set of all Mealy machines S that work from the input alphabet A to the output alphabet C is an enumerable

one. The same is for the set T of machines working from B to C. Hence the set $S \times T$ is also an enumerable one. Let us fix one of its enumerations $r_1, r_2, \ldots, r_k, \ldots$

The words x and y will be build inductively. Let us start with $x_0 = y_0 = \varepsilon$. Now let us assume there are two words $x_n \in A^l$ and $y_n \in B^l$ of equal length l that

$$\forall v \in C^l \quad \forall \langle M_s, M_t \rangle \in \{r_1, \ldots, r_n\} : (M_s * v \neq x_n) \vee (M_t * v \neq y_n).$$

Now we will construct the words x_{n+1} and y_{n+1} with the same properties. Let us denote $r_{n+1} = \langle M_s, M_t \rangle$. There are $|C|^m$ words of length m over the alphabet C. Also there are $(|A| \cdot |B|)^{m-l}$ pairs of words from $A^m \times B^m$ that start with x_n and y_n respectively. For a sufficiently large m: $(|A| \cdot |B|)^{m-l} > |C|^m$. Using the pigeonhole principle we conclude

$$\exists x_{n+1} \in x_n A^{m-l} \quad \exists y_{n+1} \in y_n B^{m-l} \quad \forall z \in C^m : \langle M_s * z, M_t * z \rangle \neq \langle x_{n+1}, y_{n+1} \rangle.$$

The same property for $r_1, r_2, \ldots, r_n$ follows from the inductive assumption and a fact that x_n and y_n are prefixes of x_{n+1} and y_{n+1} respectively.

From the last statement it also follows that there exist $x = \lim_{n\to\infty} x_n$ and $y = \lim_{n\to\infty} y_n$. It is not hard to see they are the words we are actually looking for. □

Theorem 3.3. *The partially ordered set $\mathfrak{K}$ is not a lower semi-lattice.*

Proof. It is sufficient to find two words x and y such that for every z satisfying $x, y \rightharpoonup z$ there exists t with $x, y \rightharpoonup t$ and $z \not\rightharpoonup t$.

Let γ be a fast growing function, f.e. $\gamma(n) = 3^{n+2}$. We shall define:

$$\begin{aligned}
x[i] &= \begin{cases} 1, & \text{if } \exists j\,[i = \gamma(j)]; \\ 0, & \text{otherwise;} \end{cases} \\
\tau(n) &= \begin{cases} \max\{k \in \mathbb{N} \mid n \equiv 0 \pmod{2^k}\}, & \text{if } n \neq 0; \\ 0, & \text{if } n = 0; \end{cases} \\
y[i] &= \begin{cases} 1, & \text{if } \exists j\,[i = \gamma(j) + \tau(j)]; \\ 0, & \text{otherwise;} \end{cases} \\
t_l[i] &= \begin{cases} 1, & \text{if } \exists j\,[i = \gamma((2j+1)2^l) + l]; \\ 0, & \text{otherwise} \end{cases}
\end{aligned}$$

where $l > 0$. Note $(t_l[i] = 1) \Rightarrow (y[i] = 1)$.

It is easy to see $x \rightharpoonup t_l$ and $y \rightharpoonup t_l$. Let us suppose that $x \overset{M_1}{\rightharpoonup} z$ and $y \overset{M_2}{\rightharpoonup} z$, where M_i has m_i states and $m = \max\{m_1, m_2\}$.

Since $y \overset{M_2}{\rightharpoonup} z$ then the word $z[\gamma(s2^l - 1) + 1, \gamma(s2^l) + l - 1]$ is ultimately periodic with period and pre-period $\leq m$. But also $x \overset{M_1}{\rightharpoonup} z$, so the word $z[\gamma(s2^l) + 1, \gamma(s2^l + 1) - 1]$ is ultimately periodic with period and pre-period $\leq m$. These two periodic sequences overlap by the segment $z[\gamma(s2^l) + 1, \gamma(s2^l) + l - 1]$, so, for $l \geq 3m + 2$ we can glue them together with the help of Theorem 1.1 and state that the word $z[\gamma(s2^l - 1) + 1, \gamma(s2^l + 1) - 1]$ is ultimately periodic with period and pre-period $\leq m$. So, for sufficiently large odd s it is not possible to output the symbol 1 in the word t_l at the position $\gamma(s2^l) + l$. So, $z \not\rightharpoonup t_l$. □

Let us remind that an *antichain* in a poset $\langle L, \geq \rangle$ is a set of elements S such that for every two different elements x and y from S neither $x \geq y$, nor $y \geq x$. Also let us remind that $\mathfrak{c}$ denotes the cardinality of the set $\mathbb{R}$ of real numbers, it is equal to $2^{\aleph_0}$.

Theorem 3.4. *There exists an antichain with a cardinality* $\mathfrak{c}$ *in* $\mathfrak{K}$.

Proof. We shall construct an injective function $T : \{0,1\}^\omega \to \{0,1\}^\omega$ and prove that its full image forms an antichain.

Let $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be an arbitrary one-to-one mapping. Then T transforms $x \in \{0,1\}^\omega$ to y, where

$$y[n] = \begin{cases} 1, & (n = m^2) \wedge (f(a,b) = m) \wedge (x_a \equiv b \pmod 2) \\ 0, & \text{otherwise.} \end{cases}$$

It is obvious that T is injective.

Let us take two distinct words u and v from $\{0,1\}^\omega$. There exists k such that $u_k \neq v_k$. We can suppose that $u_k = 1$. Then for any l

$$T(u)\left[(f(k,2l)-1)^2+1,\ f(k,2l)^2\right]$$

consists only of zeroes, but

$$T(v)\left[(f(k,2l)-1)^2+1,\ f(k,2l)^2\right]$$

is of a form 0^s1. Since s grows with the growth of l no machine can transform $T(u)$ into $T(v)$. The case of $u_k = 0$ is the same.

The cardinality of the antichain is $\mathfrak{c}$ because the same is the cardinality of $\{0,1\}^\omega$. □

Theorem 3.5. *If* $x \rightharpoonup y$ *and* $y \not\rightharpoonup x$ *then* $x \rightharpoonup y \vee \sigma^{-1}x$, *but still* $y \vee \sigma^{-1}x \not\rightharpoonup x$.

Proof. It is possible to transform x into $\sigma^{-1}x$ using a machine that remembers symbol it has received on the input and outputs it on the next step, then, by Theorem 3.1, $x \rightharpoonup y \vee \sigma^{-1}x$.

Let us assume there exists an initial Mealy machine M that transforms $y \vee \sigma^{-1}x$ into x. In such a case it is possible to transform y into x using the machine M and giving to its input the symbol of the word y together with the symbol it has just produced. □

Corollary 3.6. *For every not ultimately periodic word* x, $x \rightharpoonup \sigma^{-1}x$, *but* $\sigma^{-1}x \not\rightharpoonup x$.

Corollary 3.7. *For any not ultimately periodic word* x *there exists an order preserving injection* $p : \langle \mathbb{Z}, < \rangle \to \langle \mathfrak{K}, \rightharpoonup \rangle$ *such that* $p(0) = x$.

Proof. We can take the mapping $p(a) = \sigma^{-a}x$. □

From the last corollary it also follows that poset $\mathfrak{K}$ has no maximal element and has no atom, *i.e.* such an element that covers (see the next section) the smallest element of $\mathfrak{K}$.

4. Some results on density

It seems interesting to know whether the poset $\mathfrak{K}$ is dense. Let us remind that a poset $\langle L, \geq \rangle$ is called dense iff for every two its elements x, y, with $x > y$, there exists a third element z such that $x > z > y$. We shall call z *intermediate element between x and y*. If this doesn't hold, we shall say that x *covers* y.

We would like to give some equivalent ways how to express the density in $\mathfrak{K}$.

Up to the end of this section let $x \in A^\omega$ and $y \in B^\omega$ be two fixed arbitrary elements from the poset $\mathfrak{K}$ with a property $x \rightharpoonup y$ and $y \not\rightharpoonup x$. We shall be interested to know whether there exists an intermediate element between x and y.

At first, we can assume that y can be got from x using a literal morphism ν. Indeed, if $x \overset{M}{\rightharpoonup} y$ then we can take the universal output t of the machine M proceeding x. That is:

$$t[i+1] = \langle x[i+1], q_0 \circ x[0,i] \rangle,$$

where q_0 is the initial state of M. The word t is obviously equivalent to x, and y can be got from it with a literal morphism $\langle a, q \rangle \mapsto q * a$.

Moreover, if we like, we can assume that the preimage of every element in B except at most one is a one-element set, and it is a two-element set for this exceptional element. We can show this expressing the morphism ν as a composition of morphism with such a property.

Let us make some definitions. A *binary string* u is a word from $\{0,1\}^\omega$. We can identify it with a subset $U \subset \mathbb{N}$, assuming that $i \in U$ iff $u[i] = 1$. So, further we shall freely use a set-theoretical notation, taking in mind this correspondence, for example, we shall talk about non-intersecting binary strings, binary strings that are subsets of other binary strings and so on.

We are interested only in such binary strings u that can be got from x, i.e. $x \rightharpoonup u$. A binary string u shall be called (x,y)-*calculable* or simply *calculable* if $y \rightharpoonup u$. Otherwise (if $y \not\rightharpoonup u$, but still $x \rightharpoonup u$) we shall call it (x,y)-*non-calculable* or just *non-calculable*.

Proposition 4.1. *If $x \rightharpoonup y$ and $y \not\rightharpoonup x$ then there always exist two non-intersecting (x,y)-non-calculable binary strings.*

Proof. At first we shall construct one non-calculable binary string. Let us fix an arbitrary injection $\mu : A \to \{0,1\}^k$ for some k, $2^k \geq |A|$. Then we can take binary strings $\pi_i\mu(x)$ for $i = 1, 2, \ldots, k$, where π_i is a projection $\langle a_1, a_2, \ldots, a_i, \ldots, a_k \rangle \mapsto a_i$. If all $\pi_i\mu(x)$ are calculable, then (using proposition 2.5) $y \rightharpoonup x$, so at least one $\pi_i\mu(x)$ is non-calculable. We can take its complement as a second non-calculable binary string. □

Theorem 4.2. *If $x \rightharpoonup y$ and $y \not\rightharpoonup x$ then there exists an intermediate element between x and y if and only if there exist three pairwise non-intersecting (x,y)-non-calculable binary strings.*

Proof. First, let us assume there exists an intermediate element z such that $x \rightharpoonup z$, $z \not\rightharpoonup x$, $z \rightharpoonup y$, $y \not\rightharpoonup z$. Applying proposition 4.1 to the pair (x,z) we get a binary

string u such that $x \rightharpoonup u, \overline{u}$ (the complement binary string) and $z \not\rightharpoonup u, \overline{u}$. In a same way we can find v with a property $z \rightharpoonup v, \overline{v}$ and $y \not\rightharpoonup v, \overline{v}$. Obviously, $v \neq u$ and $v \neq \overline{u}$.

If $v \subset u$ we can take three binary strings $v, u \setminus v$ and $\overline{u}$ satisfying the conclusion. Indeed, if $y \rightharpoonup u \setminus v$, *a fortiori* $z \rightharpoonup u \setminus v$, and then $z \rightharpoonup u = (u \setminus v) \cup v$ that contradicts $z \not\rightharpoonup u$. In a similar way we act if $v \subset \overline{u}$.

If neither holds, then $u \cap v$ and $\overline{u} \cap v$ are nonempty and at least one of them can't be got from y. Assume $y \not\rightharpoonup u \cap v$. If $z \rightharpoonup u \cap v$ we can take $u \cap v$ as a new v. If it is not, we can take $\overline{u \cap v}$ as a new u and $\overline{v}$ as a new v.

Now we shall consider the case where there are three non-calculable pairwise non-intersecting binary strings u_1, u_2, u_3. Let us denote with N_i machines such that $x \overset{N_i}{\rightharpoonup} u_i$. We may assume $y = \nu(x)$ with a literal morphism ν and also $A \cap B = \emptyset$, where $x \in A^\omega$ and $y \in B^\omega$. We shall construct three words $z_i \in (A \cup B)^\omega$, $i = 1, 2, 3$ defined as

$$z_i[j] = \begin{cases} x[j], & j \in u_i \\ y[j], & j \notin u_i. \end{cases}$$

Obviously, $x \rightharpoonup z_i \rightharpoonup y$. Moreover, $y \not\rightharpoonup z_i$, because otherwise $y \rightharpoonup u_i$.

Let us assume $z_i \overset{M_i}{\rightharpoonup} x$ for all i. We will use this fact, but we have to ensure that each of M_i gets the input it has been expected for. We make it with the following machine.

It simulates three machines M_i and three machines N_i. An input symbol b gets redirected to the input of all three M_i. At least two machines will output a correct symbol of x because u_i are pairwise non-intersecting. So, we can take a majority of all these three outputs (denote it a) and output it. After that we give a to all three N_i and check if any machine M_i has been waiting for a symbol of x instead of a symbol of y. If there has been any we reset its state to the one, it had before this step, and give a to its input.

So, at least one z_i is with a property $z_i \not\rightharpoonup x$. □

Definition 4.3. Let $x \in A^\omega$ be a word and z be a binary string. Then the *masking* of x with z is a word t defined with

$$t[i] = \begin{cases} x[i], & \text{if } z[i] = 1, \\ \lambda, & \text{otherwise,} \end{cases}$$

where $\lambda \notin A$.

Theorem 4.4. *If an intermediate element between x and y does not exist then for any set S of pairwise non-intersecting (x, y)-calculable binary strings all maskings of x with elements of S except at most one can be got from y.*

Proof. By contradiction, let us assume that there are two maskings with the binary strings u_1 and u_2 with a desired property. Using as in the proof of Proposition 4.1 an injection $\mu : A \cup \{\lambda\} \to \{0, 1\}^k$ such that $\mu(\lambda) = (0, 0, \ldots, 0)$ we can state that there is a non-calculable binary string $v_{i1} \subset u_i$. As u_i is a calculable binary string,

a binary string $v_{i2} = u_i \setminus v_{i1}$ is also a non-calculable one. So, we have four pairwise non-intersecting non-calculable binary strings, that contradicts Theorem 4.2. □

Corollary 4.5. *If there is no intermediate element between x and y then for each ℓ there exists k such that $y \rightharpoonup t$, where the word t is defined by*

$$t[i] = \begin{cases} x[i], & i \not\equiv k \pmod{\ell}, \\ \lambda, & i \equiv k \pmod{\ell}, \end{cases} \tag{2}$$

satisfies $y \rightharpoonup t$.

In other words, one can get almost the whole word x.

Proof. Obviously, for any integer j satisfying $0 \le j < \ell$ the binary string u_j defined with

$$u_j[i] = \begin{cases} 1, & i \equiv j \pmod{\ell}, \\ \lambda, & \text{otherwise}, \end{cases}$$

is (x, y)-calculable, and any two of them do not intersect. From Theorem 4.4 it follows there exists k such that for all $j \neq k$ all masking of x with u_j can be got from y. It is a straightforward check that t from (2) can be got from the supremum of these words. □

All the previous results seem very natural: they mean that the difficulty of transforming the word y into the word x is one whole entity that cannot be partitioned into several ones.

Let us remind that, as it was shown by a classical result by Sacks [15] that the recursively enumerable degrees are dense. It may seem that there always exists an intermediate element in our case too, but it is not true.

Proposition 4.6. *Consider the following recurrence sequence*

$$x_0 = 1, \quad x_i = x_{i-1}^{i!} 0^{i!},$$

and its limit $x = \lim_{k\to\infty} x_k$. There is no intermediate element between x and $y = \sigma^{-1}x$.

Proof. At first, note that x is not ultimately periodic, so from Corollary 3.6: $x \rightharpoonup y$, but $y \not\rightharpoonup x$.

Let us take an arbitrary Mealy machine M that performs a transformation $x \overset{M}{\rightharpoonup} z$ for some word z. Let q_0 be its initial state and n be the number of states of M. Then for any state q of M and any $k > n$: $q \circ 0^{n!} = q \circ 0^{n!+k!}$ (indeed, M cycles on such a long sequence of zeros. Let ℓ denote the length of this cycle. Because $n! \equiv n! + k!$ (mod ℓ) and $n!$ is larger than the pre-period, the state of M after processing both these words is the same).

Consider occurrences of x_n in x as they naturally appear from the definition of x. Note, that x_n ends with $0^{n!}$ and gaps between two consecutive occurrences of x_n are of the form $0^{k!}$ with $k > n$. Using the claim of the previous paragraph one can see that the states in which machine M starts processing blocks x_n of x are

the same as the corresponding states in word x_n^ω, on which M cycles. Moreover, in $q_0 * x_n^\omega$ the sizes of both pre-period and period are bounded by $n|x_n|$.

Let us take word $v = x_n^{(n+1)!}$ (it is a part of x_{n+1}). From the previous paragraph it follows that machine M ends processing each v in x in one and the same state q_2 and starts processing each v except the first one in one and the same state q_1.

Let us enumerate all the occurrences of v in x: $y[i_k, j_k] = v$, $k = 0, 1, \ldots$ It is obvious there is an infinite number of such occurrences. Again, gaps between them are filled with zeros. Applying Lemma 2.3 one can state the fragments $x[j_k + 1, i_{k+1} - 1]$ are ultimately periodic all with the same pre-period and period, both bounded with n.

Because processing of all v except the first one starts in state q_1, there are only two possibilities. *All* the symbols $x[i_k]$ $(k > 0)$ break the periodicity, or they *all* do not.

In the first case it is possible to get x from the resulting word z. It can be done by a machine that repeat the following cycle:

- Outputs v.
- Goes on outputting zeros until it meets a symbol that breaks the periodicity (the periodic fragment is known, so it is not the problem).

In the second case z can be got from $y = \sigma^{-1}x$ using a machine that acts as follows. At first it outputs $q_0 * v$. Then it repeats the following cycle:

- Goes on outputting ultimately periodic word $q_2 * 0^\omega$ until it meets 1 that is the beginning of v (shifted one position to the right).
- Outputs $q_1 * v$ omitting the first symbol (the machine already produced it on the previous step). □

In the previous proposition each word that can be got from x either can be got from $\sigma^{-1}x$ or is equivalent to x itself. It seems an interesting question whether this result can be improved:

Open Problem 4.7. *Does there exist a binary string $x \in \{0,1\}^\omega$ such that, for each y satisfying $x \rightharpoonup y$ and $y \subset x$, either y is equivalent to x or y is an ultimately periodic word?*

So, $\mathfrak{K}$ is not dense. But there are some ways to get something like it. Let us remind the mapping $y \mapsto y \sqcup \lambda^\omega$ is an isomorphism of $\mathfrak{K}$ on a subset of itself. Using such a map it is possible to insert intermediate elements.

Theorem 4.8. *If $x \rightharpoonup y$ and $y \not\rightharpoonup x$ then $z = y \sqcup x$ is an intermediate element between $x' = x \sqcup \lambda^\omega$ and $y' = y \sqcup \lambda^\omega$. Moreover, if there is no intermediate element between x and y then z is the only intermediate element between x' and y'.*

Proof. Everything in the first statement of the Theorem except $y' \not\rightharpoonup z$ follows from Proposition 2.6 and Theorem 3.5. Let us prove the remaining claim.

If we assume that $y' \overset{M}{\rightharpoonup} z$ then we can get x' from y' with a following machine. If it gets a symbol of y' that is not λ, we know that the next symbol of y' is λ. We give these two symbols to the input of M and we get a symbol of y followed by a

symbol of x. Then we output the symbol of x. If machine gets λ on the input, it gives it to the output, but gives nothing to M.

Now we shall suppose x covers y and $z = u \sqcup v$ is an intermediate element between $x \sqcup \lambda^\omega$ and $y \sqcup \lambda^\omega$. Using proposition 2.8, we have $x \sim x \times \lambda^\omega \rightharpoonup u \times v \rightharpoonup y \times \lambda^\omega \sim y$. If $u \times v \sim y$ then $y \sqcup \lambda^\omega \rightharpoonup z$. It is a contradiction, so $u \times v \sim x$, hence, $u \sqcup v \rightharpoonup \lambda^\omega \sqcup x$. Now it is almost obvious that $z \rightharpoonup y \sqcup x$.

By blocking $\sigma^{-1}x' = \lambda^\omega \sqcup x$, $\sigma^{-1}z = \sigma^{-1}v \sqcup u$ and $\sigma^{-1}y' = \lambda^\omega \sqcup y$ we have $x \rightharpoonup \sigma^{-1}v \times u \rightharpoonup y$. From $\sigma^{-1}v \sqcup u \not\rightharpoonup \lambda^\omega \sqcup x$ it follows that $u \times \sigma^{-1}v \not\rightharpoonup x$, so $u \times \sigma^{-1}v \sim y$, hence $y \rightharpoonup u$. Finally, $y \sqcup x \rightharpoonup u \sqcup v = z$.

So, we we have $z \sim y \sqcup x$. □

Corollary 4.9. *The mapping $x \mapsto x \sqcup \lambda^\omega$ is not a bijection* onto $\mathfrak{K}$.

Let A be an alphabet with $\lambda \notin A$. A word of a form $\prod_{i=0}^{\infty}(a_i\lambda^{k_i})$, with $a_i \in A$ and $\lim_{k\to\infty} k_i = \infty$, is called a *rear word in a standard form.* A word equivalent to such a word is a *rear word.* A *complexity* of a rear word x is the smallest cardinality of A that x is equivalent to a rear word in a standard form from $(A \cup \{\lambda\})^\omega$.

As a contrast to proposition 4.6 we may have the following theorem.

Theorem 4.10. *If x is a rear word of complexity 1 then $\{y \in \mathfrak{K} \mid (x \rightharpoonup y) \wedge (y \not\rightharpoonup x)\}$ is without a maximal element.*

Proof. Let $x \in \{\lambda, 1\}^\omega$ be a rear word in a standard form and $x \overset{M}{\rightharpoonup} y$, where M is a machine with n states. Taking suffix, we may assume that x does not have a sequence of λ's shorter than $3n$.

It is not hard to describe the structure of y. If x has a symbol 1 at position i then y has a periodic fragment starting from $i+n$ with a period $\leq n$. It continues up to the next occurrence of 1 in x. We shall say that there is a *hold* if this periodic fragment continues further (in other words, if symbol 1 in x does not break the periodicity).

Lemma 4.11. *With the previous assumptions, $y \not\rightharpoonup x$ if and only if there is an infinite number of holds.*

Proof. If there are infinitely many holds then $y \not\rightharpoonup x$ due to Lemma 2.3.

If there is a finite number of holds, taking suffices again, we can suppose that there are no hold at all. Moreover, we may suppose x starts with a sequence of zeros longer than n. Then it is possible to transform y into x using a Mealy machine with a following algorithm.

It simulates machine M and on each step sends symbol λ to its input. If the output of M is equal to the current symbol of y then the machine outputs λ. If they are not equal then the periodicity is broken and x has 1 in the current position. So, the machine outputs 1 and recalculates the state of M, *i.e.* sets the state of M to $q * 1$, where q is the state of M before this step and $*$ is the output function of M. □

So, if $y \not\leftarrow x$ then there is an infinite number of holds. Let us enumerate all holds and construct word z as follows:

$$z[i] = \begin{cases} 1, & \text{if } x[i]=1 \text{ and the hold has an odd number or there is no hold at all} \\ 2, & \text{if } x[i-1]=1 \text{ and the hold has an even number} \\ \lambda, & \text{otherwise.} \end{cases}$$

The word z can be got from the word x by a Mealy machine that simulates machine M, counts the parity of the number of holds, and depending on it produces the output. The word y can be got from the word z by a machine that also simulates behaviour of the machine M, remembers its last and previous states, and getting a symbol 2, moves one position back and recalculates its state. Note, that the output on the previous step is no affected by such a recalculation. It is not possible to transform z into x and y into z due to the infinite number of holds. □

Note that the word y in Theorem 4.6 is not a rear one, although it contains long fragments consisting only of zeros, the lengths of them do not tend to the infinity.

Acknowledgements. The author is thankful to the anonymous referees for their thoughout reports.

References

[1] J. Berstel and J. Karhumäki, *Combinatorics on Words – A Tutorial.* TUCS Technical Report (No 530, June) (2003).
[2] J.R. Büchi, *On a Decision Method in Restricted Second Order Arithmetic*, in *Proc. Internat. Congr. on Logic, Methodology and Philosophy of Science*, edited by E. Nagel *et al.*, Stanford Univ. Press, Stanford, CA (1960) 1–11.
[3] J. Dassow, *Completeness Problems in the Structural Theory of Automata.* Mathematical Research (Band 7), Akademie–Verlag, Berlin (1981).
[4] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order.* Cambridge University Press (2002).
[5] N.J. Fine and H.S. Wilf, Uniqueness theorem for periodic functions. *Proc. Amer. Math. Soc.* **16** (1965) 109–114.
[6] J. Hartmanis and R.E. Stearns, *Algebraic Structure Theory of Sequential Machines.* Prentice–Hall, Inc., Englewood Cliffs, New Jersey (1966).
[7] M. Lothaire, *Combinatorics on Words.* Encyclopedia of Mathematics and its Applications, Vol. 17, Addison–Wesley, Reading, Massachusetts (1983).
[8] M. Lothaire, *Algebraic Combinatorics on Words.* Encyclopedia of Mathematics and its Applications, Vol. 90, Cambridge University Press, Cambridge (2002).
[9] A. Luca and S. Varricchio, *Finiteness and Regularity in Semigroups and Formal Languages.* Springer-Verlag, Berlin, Heidelberg (1999).
[10] R. McNaughton, Testing and generating infinite sequences by a finite automaton. *Inform. Control* **9** (1966) 521–530.
[11] G.H. Mealy, A method for synthesizing sequential circuits. *Bell System Tech. J.* **34** (1955) 1045–1079.
[12] D. Perrin and J.-E. Pin, Infinite words: automata, semigroups, logic and games. *Pure Appl. Math.* **141** (2004).

[13] B.I. Plotkin, I.Ja. Greenglaz and A.A. Gvaramija, *Algebraic Structures in Automata and Databases Theory.* World Scientific, Singapore, New Jersey, London, Hong Kong (1992).
[14] H. Rogers, *Theory of recursive functions and effective computability.* McGraw-Hill Book Company (1967).
[15] G.E. Sacks, The recursively enumerable degrees are dense. *Ann. Math.* **80** (1964) 300–312.
[16] S. Seshu, Mathematical models for sequential machines. *IRE Mat. Convent, Rec.* **7** (1959) 4–16.
[17] K. Wagner, On ω-regular sets. *Informatics and Control* **43** (1979) 123–177.
[18] V.B. Kudryavcev, S.V. Aleshin and A.S. Podkolzin, An introduction to the theory of automata. *Moskva Nauka* (1985) (Russian).
[19] A.A. Kurmit, Sequential decomposition of finite automata. *Riga Zinatne* (1982) (Russian).
[20] B.A. Trahtenbrot and Ya.M. Barzdin, Finite automata, behaviour and synthesis. *Moskva Nauka* (1970) (Russian).