Realtime subshifts

Petr Kůrka* and Alejandro Maass†

Abstract

We generalize the class of sofic subshifts, which correspond to regular languages, to subshifts accepted by either nondeterministic or deterministic Turing machines in real time. We show that every substitutive system can be accepted by a deterministic Turing machine in real time.

1 Introduction

Symbolic dynamics, which studies dynamical systems on spaces of symbolic sequences, is intimately connected with formal language theory. Subshifts are in one-to-one correspondence with the class of right central languages and an important class of sofic subshifts (see Weiss [14], Lind & Marcus [9]) corresponds to the family of regular languages. Sofic systems are presented by finite labeled graphs and their properties can be read from the structure of these graphs. There are, however, many interesting subshifts beyond sofic systems (e.g., all minimal ones) and formal language theory offers us the possibility to compare their complexity. If we wish to keep the presentation of subshifts by labeled graphs, we are naturally led to languages accepted by Turing machines in real time. While every subshift can be presented by a countable labeled graph (see Fiebig & Fiebig [5]), in this case the graph has a simple structure directly given by the Turing machine in question.

We consider machines equipped with a finite memory and $n \geq 0$ pushdown stores. Pushdown store is one of the simplest data structures and many results in complexity theory are formulated in terms of them. Since a tape can be simulated by two pushdown stores, a machine with $2n$ pushdown stores has the same computing power as a Turing machine with n tapes. We obtain thus a finer hierarchy, which includes the class of regular languages for $n = 0$ and the class of context-free languages for $n = 1$. Moreover, the resulting formalism is simpler and yields immediately labeled graphs which present the subshifts. In general we consider nondeterministic machines but there is an important class

*Faculty of Mathematics and Physics, Charles University in Prague, Malostranské nám. 25, 118 600 Praha 1, Czechia.

†Departamento de Ingeniería Matemática, Universidad de Chile, Casilla 170/3 correo 3, Santiago, Chile.

of deterministic machines. In both cases, to recognize whether an infinite word belongs to the subshift, the machine reads every time step a letter on the input. If the input word belongs to the subshift, the computation is infinite. If not, the computation ends and the input word is rejected by the time of reading the first error.

We show that the property of being accepted in real time by either nondeterministic or deterministic Turing machine with a given number of pushdown stores is a topological invariant. We show that every substitutive subshift is accepted by a deterministic Turing machine in real time. Among Sturmian subshifts there is an important subclass of subshifts which includes all quadratics, which are accepted by nondeterministic Turing machines in real time (this has been proved in Blanchard & Kůrka [3]). Finally, many subshifts obtained from cellular automata are realtime.

2 Machines, graphs and languages

If A is a finite alphabet, $n \in \mathbb{N} = \{0, 1, 2, \ldots\}$, denote by A^n the set of words over A of length n, $A^+ = \cup_{n>0} A^n$ the set of nonempty finite words over A, $A^* = \cup_{n \in \mathbb{N}} A^n = A^+ \cup \{\lambda\}$ the set of finite words over A, $A^{\mathbb{N}}$ the set of one-way infinite words, and $\overline{A^*} = A^* \cup A^{\mathbb{N}}$. For $u \in \overline{A^*}$, denote by $|u|$ its length ($0 \leq |u| \leq \infty$), and $|u|_a$ the number of occurrences of a letter $a \in A$ in u. The $(i+1)$-st letter of a word $u \in \overline{A^*}$ is denoted by u_i, so $u = u_0 u_1 \cdots$. For $u \in A^*$, $v \in \overline{A^*}$ write $u \sqsubseteq v$ if u is a subword of v, i.e., if there exists $j \geq 0$ such that $u_i = v_{j+i}$ for all $i < |u|$. When $j = 0$ we say that u is a prefix of v and write $u \sqsubseteq_* v$. If $i = |v| - |u|$ we say that u is a suffix of v and write $u \sqsubseteq^* v$. Denote by $u_{|i} = u_0 \cdots u_{i-1}$ the prefix of u of length i. The last letter of a word $u \in A^+$ us denoted by $u_\bullet = u_{|u|-1}$. For $u = u_0 ... u_{n-1}$ denote by $u^{-1} = u_{n-1} ... u_0$ its inverse. The concatenation of the words u and v is written uv; denote the n^{th} concatenation power of u by u^n, and the periodic sequence generated by its infinite repetition by $u^\infty \in A^{\mathbb{N}}$. We frequently use the binary alphabet $\mathbf{2} = \{0, 1\}$.

If A is a finite alphabet, an A-labeled graph is a structure $G = (V, E, s, t, \imath, h)$, where $V = |G|$ is a set of vertices, E is a set of edges, $s, t : E \to V$ are the source and target maps, $\imath \in V$ is an initial vertex and $h : E \to A^*$ is a labeling function (an important special case is the graph labeled by letters, $h : E \to A$). A path in G is a finite or infinite sequence of edges $e = e_0 e_1 \ldots$ such that $s(e_0) = \imath$ and $s(e_i) = t(e_{i-1})$ for $1 < i < |e|$. The label of a path is the concatenation of the labels of its edges $h(e) = h(e_0) h(e_1) \ldots$.

A language $L \subseteq A^*$ is a realtime language, if it is accepted by a nondeterministic Turing machine in real time. We use machines which work with pushdown stores. A pushdown store can store words from an alphabet A. To update a pushdown store, its top (first) letter is replaced by a word. An updating instruction thus can be expressed as a rule $u \to v$ where $u \in A$ is the first letter and

$v \in A^*$ is the word which replaces it. If the content of the pushdown store is uw, where $w \in A^*$, it is updated to vw. We consider generalized pushdown stores, in which $u \in A^*$ can also be a word. This generalization does not increase the computational power. Such a generalized pushdown store can be simulated by a pushdown store whose alphabet is the set of nonempty words with length at most k for suitable k.

We consider a nondeterministic machine over an input alphabet A with finite memory (set of inner states) A_0 and n pushdown stores with alphabets $A_1, \dots, A_n$. The machine works in real time. Every time step it reads a letter on the input, looks at its finite memory and at the top of all its pushdown stores. On the base of this information it chooses nondeterministically some action, in which it updates its finite memory and all its pushdown stores. Such an action is therefore given by a rule

$$(u_0, u_1, ..., u_n) \stackrel{a}{\longrightarrow} (v_0, v_1, ..., v_n)$$

where $a \in A$ is the input letter, $u_0 \in A_0$ is the current inner state and $u_i \in A_i^*$ is a word on the top of i-th pushdown store. The new inner state is then $v_0 \in A_0$ and $v_i \in A_n^*$ is the replacement word for the i-th pushdown store. The dynamics of a machine is given by a finite set of such rules, i.e., by a finite subset M of $A \times (A_0 \times A_1^* \times \cdots \times A_n^*)^2$. Besides we have an initial inner state $q_0 \in A_0$ and a set of accepting inner states $Q_1 \subseteq A_0$. A computation can be viewed as a path in a graph $G_{\mathcal{M}}$, whose vertices $|G_{\mathcal{M}}| = A_0 \times A_1^* \times \cdots \times A_n^*$ correspond to configurations of the machine. A configuration consists of the contents of the finite memory and all the pushdown stores.

Definition 1 *A (nondeterministic realtime) machine (with $n \geq 0$ pushdown stores) is a structure $\mathcal{M} = (A, A_0, A_1, \dots, A_n, M, q_0, Q_1)$, where A, A_0, A_1, ..., A_n are finite alphabets, $q_0 \in A_0$ is an initial state, $Q_1 \subseteq A_0$ is a set of accepting states and $M \subset A \times (A_0 \times A_1^* \times ... \times A_n^*)^2$ is a finite set of rules. We associate to $\mathcal{M}$ an A-labeled graph $G_{\mathcal{M}}$ whose vertex set is $|G_{\mathcal{M}}| = A_0 \times A_1^* \times \cdots \times A_n^*$, its initial vertex is $\imath = (q_0, \lambda, \dots, \lambda)$, and whose labeled edges are*

$$(u_0, u_1 w_1, ..., u_n w_n) \stackrel{a}{\longrightarrow} (v_0, v_1 w_1, ..., v_n w_n)$$

where $(u_0, u_1, ..., u_n) \stackrel{a}{\longrightarrow} (v_0, v_1, ..., v_n)$ is any rule of M and $w_i \in A_i^*$ are arbitrary words.

We say that a machine $\mathcal{M}$ is deterministic, if $G_{\mathcal{M}}$ is right-resolving. This means that for every vertex $u \in |G_{\mathcal{M}}|$ and every $a \in A$ there is at most one edge with source u and label a. A machine is deterministic iff following property holds: Whenever

$$(u_0, u_1, ..., u_n) \stackrel{a}{\longrightarrow} (v_0, v_1, ..., v_n), \quad (u_0, u_1', ..., u_n') \stackrel{a}{\longrightarrow} (v_0', v_1', ..., v_n')$$

are different rules of $\mathcal{M}$ such that there exist words $w_i, w_i' \in A_i^*$ with $u_i w_i = u_i' w_i'$, then $v_i w_i = v_i' w_i'$. A sufficient condition for a machine to be deterministic is that their left hand sides together with labels form a code, i.e., whenever in two productions $a = a'$ and $u_0 = u'_0$, then for some i either $u_i \not\sqsubseteq_* u'_i$ or $u'_i \not\sqsubseteq_* u_i$.

Definition 2 *The language $L_{\mathcal{M}}$ accepted by a machine $\mathcal{M}$ consists of all words $u \in A^*$ such that there exists in $G_{\mathcal{M}}$ an accepting path, i.e., a path*

$$(q_0, \lambda, \ldots, \lambda) \xrightarrow{u} (q, u_1, \ldots, u_n)$$

for some $q \in Q_1$ and $u_i \in A_i^*$. Denote by $\mathbb{R}(n)$ the family of languages accepted by deterministic machines with n pushdown stores and $\mathbb{Q}(n)$ the family of languages accepted by nondeterministic machines with n pushdown stores. $\mathbb{R} = \cup_n \mathbb{R}(n)$ is the family of deterministic realtime languages and $\mathbb{Q} = \cup_n \mathbb{Q}(n)$ is the family of nondeterministic realtime languages.

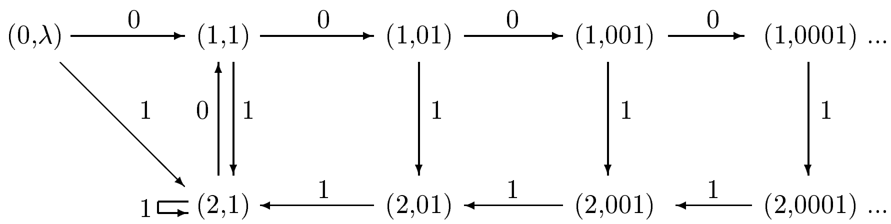
In Wagner and Wechsung [13] the family of deterministic realtime languages is denoted by **REALTIME**. A machine with zero pushdown stores is just a finite automaton, so $\mathbb{R}(0) = \mathbb{Q}(0) = \mathbf{REG}$ is the family of regular languages. A machine with one pushdown store is a pushdown automaton which accepts context-free languages, so $\mathbb{R}(1) \subset \mathbb{Q}(1) = \mathbf{CF}$. A machine with $2n$ pushdown stores is equivalent to a Turing machine with n tapes. By a theorem of Aanderaa [1], the hierarchy of deterministic languages is infinite and $\mathbb{R}(2n)$ is a proper subset of $\mathbb{R}(2n+2)$ (see also Wagner and Wechsung [13], page 137). By a result of Book and Greibach [4], $\mathbb{Q} = \mathbb{Q}(3)$, so the hierarchy of nondeterministic realtime languages is finite and moreover $\mathbb{Q}$ coincides with the class of languages accepted by nondeterministic Turing machines in linear time (see also Wagner and Wechsung [13], page 403). It is easy to see that every nondeterministic realtime language is context-sensitive, $\mathbb{Q} \subseteq \mathbf{CS}$. It is an open question, whether this inclusion is proper.

Example 1 *Consider a language $L \subset \mathbf{2}^*$ consisting of such binary words u, that whenever $0^i 1^j 0 \sqsubseteq u$ then $i \leq j$. Then L is accepted by a deterministic Turing machine with one pushdown store.*

Proof: Put $A_0 = \{0, 1, 2\}$, $A_1 = \mathbf{2}$, the initial state is $q_0 = 0$, $Q_1 = A_0$ and rules are

$$\begin{array}{lcl@{\qquad\qquad}lcl}
(0,\lambda) & \xrightarrow{0} & (1,1) & (0,\lambda) & \xrightarrow{1} & (2,1) \\
(1,\lambda) & \xrightarrow{0} & (1,0) & (2,0) & \xrightarrow{1} & (2,\lambda) \\
(1,\lambda) & \xrightarrow{1} & (2,\lambda) & (2,1) & \xrightarrow{1} & (2,1) \\
 & & & (2,1) & \xrightarrow{0} & (1,1)
\end{array}$$

Then $L = L_{\mathcal{M}}$.

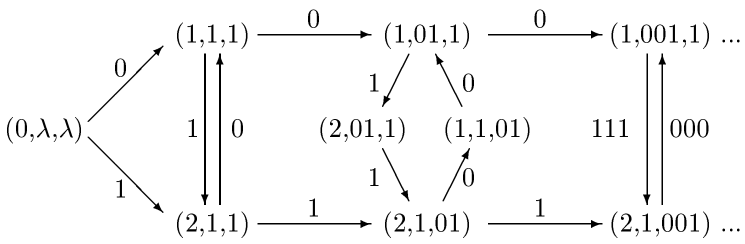


Example 2 *Consider a language $L \subset \mathbf{2}^{\mathbb{N}}$ consisting of such binary words u, that whenever $1^i 0^j 1 \sqsubseteq u$ with $i > 0$, then $i \leq j$, and whenever $0^i 1^j 0 \sqsubseteq u$ with $i > 0$, then $i \leq j$. Then L is accepted by a deterministic Turing machine with two pushdown stores.*

Proof: Consider a Turing machine M with $A_0 = \{0, 1, 2\}$, $A_1 = A_2 = \mathbf{2}$, initial state $q_0 = 0$, $Q_1 = A_0$ and rules

$$\begin{array}{lcl@{\qquad\qquad}lcl}
(0,\lambda,\lambda) & \xrightarrow{0} & (1,1,1) & (0,\lambda,\lambda) & \xrightarrow{1} & (2,1,1) \\
(1,\lambda,0) & \xrightarrow{0} & (1,0,\lambda) & (2,0,\lambda) & \xrightarrow{1} & (2,\lambda,0) \\
(1,\lambda,1) & \xrightarrow{0} & (1,0,1) & (2,1,\lambda) & \xrightarrow{1} & (2,1,0) \\
(1,\lambda,1) & \xrightarrow{1} & (2,\lambda,1) & (2,1,\lambda) & \xrightarrow{0} & (1,1,\lambda)
\end{array}$$

Then $L = L_{\mathcal{M}}$.



3 Subshifts

Define a metric d on the power space $A^{\mathbb{N}}$ by

$$d(u,v) = 2^{-n}, \quad \text{where} \quad n = \min\{i \in \mathbb{N}; u_i \neq v_i\}$$

The space $A^{\mathbb{N}}$ is compact and homeomorphic to the Cantor middle third set. The shift map $\sigma : A^{\mathbb{N}} \to A^{\mathbb{N}}$ is defined by $\sigma(u)_i = u_{i+1}$; thus $\sigma(u_0 u_1 u_2 \cdots) =$

$u_1u_2u_3\cdots$. Then $d(\sigma(u),\sigma(v)) \leq 2d(u,v)$, so σ is continuous. A subshift over A is any nonempty subset $\Sigma \subseteq A^{\mathbb{N}}$ which is topologically closed and σ-invariant, i.e., $\sigma(\Sigma) \subseteq \Sigma$. The restriction of the shift map to Σ is again continuous. We denote by $\sigma^n : \Sigma \to \Sigma$ the n-th iterate, i.e., its composition with itself n times. A point $u \in \Sigma$ is periodic, if $\sigma^n(u) = u$ for some n. A subshift $\Sigma \subseteq A^{\mathbb{N}}$ is minimal, if it does not contain any proper subshifts. A subshift is minimal iff it is the closure if the orbit of any its point, i.e., $\Sigma = \overline{\{\sigma^n(u) : n \geq 0\}}$ for any $u \in \Sigma$.

Given a subshift $\Sigma \subseteq A^{\mathbb{N}}$, define its associated language as

$$\mathcal{L}(\Sigma) = \{u \in A^* \ ; (\exists v \in \Sigma)(u \sqsubseteq v)\}.$$

For $n \in \mathbb{N}$ put also $\mathcal{L}^n(\Sigma) = \mathcal{L}(\Sigma) \cap A^n$. The language $\mathcal{L}(\Sigma)$ is right central, i.e., closed under subwords (if $u \in \mathcal{L}(\Sigma)$ and $v \sqsubseteq u$ then $v \in \mathcal{L}(\Sigma)$) and extendable to the right (if $u \in \mathcal{L}(\Sigma)$, then there exists $a \in A$, such that the concatenation ua is in $\mathcal{L}(\Sigma)$). The converse is true: any right central language is the associated language of a unique subshift (cf. Blanchard and Hansel [2]).

Definition 3 *Let $\mathfrak{L}$ be a family of languages (e.g., $\mathbb{R}(n)$, $\mathbb{R}$, $\mathbb{Q}(n)$, $\mathbb{Q}$). We say that a subshift $\Sigma \subseteq A^{\mathbb{N}}$ is of class $\mathfrak{L}$ if $\mathcal{L}(\Sigma)$ belongs to $\mathfrak{L}$.*

Suppose that $\Sigma \subseteq A^{\mathbb{N}}$ is a subshift, whose language $L(\Sigma)$ is accepted by a deterministic machine $\mathcal{M} = (A, A_0, \ldots, A_n, M, q_0, Q_1)$. Consider a computation of $\mathcal{M}$ on an infinite input word $u \in A^{\mathbb{N}}$. If $u \in \Sigma$, then every its prefix $u_{|n}$ belongs to $\mathcal{L}(\Sigma)$ and there is a unique accepting path with label $u_{|n}$. Thus the computation continues indefinitely for inputs $u \in \Sigma$. If u doest not belong to Σ then there exists least n for which $u_{|n}$ does not belong to $L(\Sigma)$ and this is also the least n for which an accepting path with label $u_{|n}$ does not exist. (Either there is no path with label u or the path with label u ends in a nonaccepting vertex.) Thus if we remove from $\mathcal{M}$ every rule with either u_0 or v_0 from $A_0 \setminus Q_1$, the resulting Turing machine accepts the same language. We see therefore that every deterministic realtime language is accepted by a Turing machine with the full set of accepting states $Q_1 = A_0$.

For nondeterministic machines, the situation is different. If $u \in \Sigma$, then again for every n there exists an accepting path with label $u_{|n}$, but there might be several accepting paths with this label. Moreover an accepting path with label $u_{|(n+1)}$ need not be a prolongation of any accepting path with label $u_{|n}$. To verify whether $u \in A^{\mathbb{N}}$ belongs to Σ therefore means to repeat the computation for every $u_{|n}$ from the scratch. We can prove, however, that even in this case there exists a machine with full set of accepting states $Q_1 = A_0$ which accepts Σ. In this case a word $u \in A^{\mathbb{N}}$ belongs in Σ iff there exists an infinite path with label u. During such an infinite computation, we know that every prefix so far accepted really belongs to $L(\Sigma)$.

Proposition 1 *Let $\Sigma \subseteq A^{\mathbb{N}}$ be a subshift and $\mathcal{L}(\Sigma)$ is accepted by a machine with full set of accepting states ($Q_1 = A_0$). Then $u \in \Sigma$ iff there exists an* *infinite path with label u which starts at the initial vertex. If $u \notin \Sigma$ and if n is the first index for which $u_{|n}$ does not belong to $\mathcal{L}(\Sigma)$, then there is no accepting path with label $u_{|n}$.*

The proof follows from a compactness argument.

To prove that even every nondeterministic realtime subshift can be accepted by a machine with full set of accepting states, we use the concept of realtime computation with delay and a speedup theorem. If we relax the definition of a machine to allow for both labeled and unlabeled rules, we get the set of rules as a finite subset of

$$(A \cup \{\lambda\}) \times (A_0 \times A_1^* \times \ldots \times A_n^*)^2$$

We get a nondeterministic machine with n pushdown stores, which does not work necessarily in real time. Again, the language accepted by such a machine consists of the set of labels of accepting paths. A language is accepted by a machine of this type (with at least 2 pushdown stores) iff it is recursive. We say that a machine of this type works with delay k if there exists $0 \leq i < k$, such that in every path u which starts at the initial vertex, the labeled edges are exactly u_{nk+i}, $n \geq 0$. By a speedup theorem (see Book and Greibach [4]), if a language is accepted by a machine with delay k, then it is a nondeterministic realtime language.

Theorem 1 *If $\Sigma \subseteq A^{\mathbb{N}}$ is a realtime subshift, then it is accepted by a nondeterministic Turing machine with full set of accepting states $Q_1 = A_0$.*

Proof: Suppose that $\mathcal{M}$ is a Turing machine which accepts in realtime the language $\mathcal{L}(\Sigma)$. We construct a machine with delay 8 with the full set of accepting states, which accepts $\mathcal{L}(\Sigma)$. The machine works in stages. At the end of stage n it has guessed a word $u \in \mathcal{L}(\Sigma)$ of length 2^{n+5} and verified that the prefix of the input of length 2^{n+2} equals the prefix of u of the same length. The new machine uses the original machine $\mathcal{M}$ as a subroutine and four additional pushdown stores with alphabets $B = \{a, a' : a \in A\}$, which besides the letters from A contains their primed variants. These primed letters mark the bottom of pushdown stores. Let $n \geq 0$, and let $u_i \in A^*$, $0 \leq i < 16$ be words of length 2^n. Suppose that by time $t = 2^{n+5}$ it has been verified that $u_0 \ldots u_7$ belongs to $\mathcal{L}(\Sigma)$, $u_0u_1u_2u_3$ equals the prefix of the input, and the content of the four new pushdown stores is

$$(u_0 \ldots u_7, \lambda, u_4u_5u_6u_7, \lambda), \quad t = 2^{n+5}$$

and the last letter in the first pushdown store is primed. The n-th stage between times 2^{n+5} and 2^{n+6} is divided into four substages each of length 2^{n+3}. In the first substage it is verified whether the word read on the input coincides with u_4 in pushdown store 3 (a comparison is done every eighth step). If any inequality is found, the process is terminated and the input word is not accepted. Simultaneously with these comparisons, the content of pushdown store 1 is transferred in reverse order to pushdown stores 2 and 4. This transfer is done in a loop, which ends when a primed letter is read on the top of the pushdown store 1. Thus the content of the pushdown stores at the end of the first substage is

$$(\lambda, (u_0 \dots u_7)^{-1}, u_5 u_6 u_7, (u_0 \dots u_7)^{-1}), \quad t = 2^{n+5} + 2^{n+3}$$

and the last letters in both stores 2 and 4 is primed. In the second substage, the input is compared with u_5 in the pushdown store 3 and a word $u_8 \dots u_{15}$ is generated nondeterministically and put in reverse order to pushdown store 2. This task runs in a loop in which a letter is read from pushdown store 4 every time step until a primed letter is encountered. The pushdown store 4 is used therefore only to store the length of $u_0 \cdots u_7$. Simultaneously during the substages 1 and 2, the original machine $\mathcal{M}$ is run to verify whether $u_0 \dots u_{15}$ belongs to $\mathcal{L}(\Sigma)$. Every time step the machine is fed with the same symbol which is stored in pushdown store 2. If $u_0 \dots u_{15}$ is not accepted by $\mathcal{M}$, the process is terminated and the input word is not accepted. Otherwise the process continues with content of pushdown stores

$$(\lambda, (u_0 \dots u_{15})^{-1}, u_6 u_7, \lambda), \quad t = 2^{n+5} + 2^{n+4}$$

In the word $(u_0 \dots u_{15})^{-1}$ contained in the pushdown store 2, two letters are now primed: the last one and the last letter of its first half. In the third substage the word $u_8 \cdots u_{15}$ is transferred from the pushdown store 2 to both pushdown stores 1 and 4. This is done in a loop until a primed letter is encountered. Simultaneously, u_6 in the pushdown store 3 is compared with the input. The content of the pushdown stores is then

$$(u_8 \cdots u_{15}, (u_0 \dots u_7)^{-1}, u_7, u_8 \dots u_{15}), \quad t = 2^{n+5} + 3 \cdot 2^{n+3}$$

The last letter in both pushdown stores 1 and 2 is primed. In the fourth substage, $u_0 \dots u_7$ is transferred from pushdown store 2 to pushdown store 1 until a primed letter is encountered. Simultaneously, u_7 is compared with the input. We obtain

$$(u_0 \dots u_{15}, \lambda, \lambda, u_8 \dots u_{15}), \quad t = 2^{n+6}$$

with the last letter in the first pushdown store primed. Thus we know now at this time that $u_0 \dots u_{15}$ belongs to $\mathcal{L}(\Sigma)$ and $u_0 \dots u_7$ has been read on the input. The process now continues with the roles of pushdown stores 3 and 4 interchanged. We leave the technical details of the implementation. In the initial stages up to time 32 the machine has to work slightly differently. It must be initialized so that by time 32 the first pushdown store contains a word of length 8, which belongs to the language, its first half has been read on the input and its second part is stored in pushdown store 3. $\square$

Proposition 2 *If Σ is a realtime subshift, then it is accepted by a Turing machine with full set of accepting states $Q_1 = A_0$ and 7 pushdown stores.*

Proof: By Theorem 3.1 of Book and Greibach [4], every realtime language is accepted by a Turing machine with three pushdown stores. From the construction of Theorem 1, 4 additional pushdown stores are sufficient.

From now on we consider only machines with full set of accepting states. These are structures $\mathcal{M} = (A, A_0, \ldots A_n, M, q_0)$, where M is a finite subset of $A \times (A_0 \times A_1^* \times \cdots \times A_n^*)^2$ and $q_0 \in A_0$.

4 Factors and conjugacies

We say that a subshift $\Sigma_1 \subseteq B^{\mathbb{N}}$ is a factor of a subshift $\Sigma_0 \subseteq A^{\mathbb{N}}$ if there exists a continuous surjective map $F : \Sigma_0 \to \Sigma_1$, which commutes with the shifts, i.e., for every $x \in \Sigma_0$, $F(\sigma(x)) = \sigma(F(x))$. Every factor map F is determined by some local map $f : A^r \to B$, $r \geq 1$ via $F(x)_i = f(x_i, \ldots, x_{i+r-1})$. If F is bijective, we call it a conjugacy and say that Σ_0 and Σ_1 are topologically conjugate. A property which is preserved by topological conjugacy is called topological invariant.

Theorem 2 *If Σ_1 is a factor of Σ_0 and Σ_0 is of class $\mathbb{Q}(n)$, then Σ_1 is of class $\mathbb{Q}(n)$. The property of being a nondeterministic realtime subshift is therefore a topological invariant.*

Proof: Suppose that $\Sigma_0 \subseteq A^{\mathbb{N}}$ is accepted by $\mathcal{M} = (A, A_0, A_1 \ldots, A_n, M, q_0)$. Let $f : A^r \to B$ is a local map of a factor map $F : \Sigma_0 \to \Sigma_1$. We construct a machine $\mathcal{M}' = (B, B_0, A_1, \ldots, A_n, M', q_0')$ which accepts Σ_1. Put $B_0 = (\mathcal{L}^{r-1}(\Sigma_1) \times A_0) \cup \{q_0'\}$, where q_0' is a new symbol and $\mathcal{L}^{r-1}(\Sigma_1)$ is the set of words from $\mathcal{L}(\Sigma_1)$ of length $r-1$. The initial rules of $\mathcal{M}'$ over B are

$$(q_0', \lambda, \ldots, \lambda) \xrightarrow{f(a_0 \ldots a_{r-1})} ((a_1 \ldots a_{r-1}, u_0), u_1, \ldots, u_n)$$

where $(q_0, \lambda, \ldots, \lambda) \stackrel{a_0}{\to} \cdots \stackrel{a_{r-1}}{\to} (u_0, u_1, \ldots, u_n)$ is a path of length r in G_M. The other rules of M' are

$$((a_1 \ldots a_{r-1}, u_0), u_1, \ldots, u_n) \xrightarrow{f(a_1 \ldots a_r)} ((a_2 \ldots a_r, v_0), v_1, \ldots, v_n)$$

where $(u_0, u_1 \ldots u_n) \stackrel{a_r}{\to} (v_0, v_1 \ldots v_n)$ is a rule of M. It follows that every path in $G_{\mathcal{M}}$ with label $a_0 a_1 \ldots$ yields a path in $G_{\mathcal{M}'}$ with label $b_0 b_1 \ldots$, where $b_i = f(a_i \ldots a_{i+r-1})$ and this path belongs to Σ_1. Conversely every point in Σ_1 can be obtained in this way. Thus Σ_1 is of class $\mathbb{Q}(n)$. □

If the machine $\mathcal{M}$ in theorem 2 is deterministic, the machine $\mathcal{M}'$ need not be deterministic, so a factor of a deterministic realtime subshift need not be deterministic realtime. However, suppose that f is a conjugacy, whose inverse conjugacy has local rule $g : B^p \to A$. Then $\mathcal{M}'$ is right-closing with delay $r+p$. This means that if there are two paths with the same label $b_0 \dots b_{n-1}$, then $a_0 \dots a_{n-p} = g(b_0 \dots b_{n-1})$ is fixed and the initial part of the path in $G_{\mathcal{M}}$ of length $n-p-r$ is fixed too. We use this to show that the determinism is a topological invariant.

Theorem 3 *If Σ_0 is a subshift of class $\mathbb{R}(n)$, and Σ_1 is conjugate to Σ_0, then it is also of class $\mathbb{R}(n)$.*

Proof: Let $\Sigma_0 \subseteq A^{\mathbb{N}}$, $\Sigma_1 \subseteq B^{\mathbb{N}}$, and $f : A^r \to B$, $g : B^p \to A$ are local maps of mutually inverse conjugacies $F : \Sigma_0 \to \Sigma_1$ and $G : \Sigma_1 \to \Sigma_0$. Let $\mathcal{M} = (A, A_0, A_1, ..., A_n, M, q_0)$ be a deterministic machine which accepts Σ_0. We construct a deterministic machine $\mathcal{M}' = (B, B_0, A_1, \dots A_n, M', q_0')$. Here $B_0 = (\mathcal{L}_{r+p-2}(\Sigma_1) \times A_0) \cup \{q_0'\}$, where $\mathcal{L}_{r+p-2}(\Sigma_2)$ is the set of nonempty words of $\mathcal{L}(\Sigma_2)$ of length at most $r+p-2$. The initial rules of M' are

$$(q_0', \lambda, \dots, \lambda) \xrightarrow{b_0} ((b_0, q_0), \lambda, \dots, \lambda), \quad b_0 \in B$$

Further rules are

$$((b_0 \dots b_{i-1}, q_0), \lambda, \dots, \lambda) \xrightarrow{b_i} ((b_0 \dots b_i, q_0), \lambda, \dots, \lambda)$$

for $0 < i < r+p-2$,

$$((b_0 \dots b_{r+p-3}, q_0), \lambda, \dots, \lambda) \xrightarrow{b_{r+p-2}} ((b_1 \dots b_{r+p-2}, u_0), u_1, \dots, u_n)$$

where $(q_0, \lambda, \dots, \lambda) \xrightarrow{g(b_0 \dots b_{p-1})} \cdots \xrightarrow{g(b_{r-1} \dots b_{r+p-2})} (u_0, u_1, \dots u_n)$ is a path in $\mathcal{M}$, and finally

$$((b_1 \dots b_{r+p-2}, u_0), u_1, \dots, u_n) \xrightarrow{b_{r+p-1}} ((b_2 \dots b_{r+p-1}, v_0), v_1, \dots, v_n)$$

where $(u_0, u_1, \dots, u_n) \xrightarrow{g(b_r \dots b_{r+p-1})} (v_0, v_1, \dots v_n)$ is a rule in $\mathcal{M}$. Clearly $\mathcal{M}'$ is deterministic and accepts Σ_1. $\Box$

Example 3 *There exists a realtime subshift which is not deterministic realtime subshift but is a factor of a deterministic realtime subshift.*

Proof: We use the fact that the language

$$L_1 = \{w_1 2 \dots w_j 2 w_i^{-1} : \ 1 \le i \le j, \ \ w_1, \dots, w_j \in \mathbf{2}^*\}$$

is a realtime language but not deterministic realtime (see Wagner and Wechsung [13] page 123). Consider a set

$$X_1 = \{u_0 3 u_1 3 u_2 3 \ldots :\ u_i \in L_1\}$$

and let $\Sigma_1 \subset \{0,1,2,3\}^{\mathbb{N}}$ be the smallest subshift which contains X_1. Then Σ_1 is not deterministic realtime subshift. If it were, we would get an acceptance procedure for the language L_1. On the other hand it is not difficult to construct a nondeterministic Turing machine which accepts Σ_1. If $u_0 3 u_1 3 u_2 3 \ldots \in \Sigma_1$, then u_0 is any suffix of a word from L_1, i.e., a word of the form $w_1 2 \ldots 2 w_j$. Then a nondeterministic Turing machine for L_1 is applied to words $u_1, \ldots$. To obtain Σ_1 as a factor of a deterministic realtime subshift, consider a language

$$L_0 = \{w_1 2 \ldots w_j 4 w_i^{-1} :\ 1 \leq i \leq j,\ \ w_1, \ldots, w_j \in \mathbf{2}^*\}$$

and construct from it in set X_0 and subshift $\Sigma_0 \subset \{0,1,2,3,4\}^{\mathbb{N}}$ in the same way as Σ_1 is constructed from L_1. Then Σ_0 is clearly a deterministic realtime subshift and the map $f(4) = 2$, $f(i) = i$ for $0 \leq i \leq 3$ yields a factor map from Σ_0 to Σ_1. $\Box$

A machine with 0 pushdown stores is just a finite automaton, so subshifts of type $\mathbb{R}(0) = \mathbb{Q}(0$ are just sofic systems. A language is accepted by a nondeterministic machine with one pushdown store iff it is context-free. Context-free subshifts (subshifts with context-free languages) have been considered by Krieger [7]. The languages from both Examples 1 and 2 are right central, the corresponding subshifts are therefore deterministic realtime subshifts.

Proposition 3 *Every context-free subshift has a periodic point.*

Proof: By the Pumping lemma for context-free grammars, there exist words $x, y, z, u, v \in A^*$ such with $|yu| \geq 1$ and $xy^i zu^i v \in L(\Sigma)$ for every $n > 0$, it follows that both y^∞ and u^∞ belong to Σ. $\Box$

5 Substitutive systems

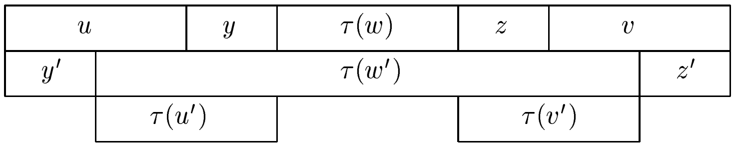
A substitution on alphabet A is a map $\tau : A \to A^+$. We associate to τ a $A \times A$ matrix $M_{ab} = |\tau(a)|_b$. We assume that τ is primitive, i.e., there exists $n > 0$ such that all entries of the power matrix M^n are positive. The language

$$L_\tau = \{u \in A^* :\ \ (\exists k > 0)(\exists a \in A)(u \sqsubseteq \tau^k(a))\}$$

is then right central and the corresponding subshift Σ_τ is minimal (see Queffélec [10]). We use following property proved in Mossé [11, 12].

Proposition 4 *Let $\tau : A \to A^+$ be a primitive substitution.*
1. Any $x \in \Sigma_\tau$ can be written as $x = \sigma^t(\tau(y))$ where $y \in \Sigma_\tau$, $0 \leq t < |\tau(y_0)|$.
2. There exist $N_0, N_1 > 0$ such that every $x \in L_\tau$ of length at least N_1 can be uniquely decomposed as $x = y\tau(w)z$, where $|w| > 0$, $|y|, |z| < N_0$, $|y\tau(w_0)|, |\tau(w_\bullet)z| \geq N_0$ ($w_\bullet$ is the last letter of w).

If x is a subword of x', then the decomposition $x' = y'\tau(w')z'$ is an extension of the decomposition $x = y\tau(w)z$. This means that if $x' = uxv$, then $uy = y'\tau(u')$ for some $u' \sqsubseteq_* w'$ and $zv = \tau(v')z'$ for some $v' \sqsubseteq^* w'$.



By Perron-Frobenius theory of positive matrices, M_τ has a spectral radius $\Theta > 1$. Moreover there exist $0 < M_0 \leq M_1$ such that

$$M_0\Theta^k \leq |\tau^k(a)| \leq M_1\Theta^k$$

for every $a \in A$ and every $k > 0$, and there exist constants $1 < \Theta_0 < \Theta < \Theta_1$ and p_0 such that

$$\Theta_0|u| \leq |\tau(u)| \leq \Theta_1|u|$$

for every word $u \in L_p(\Sigma_\tau)$ for which $p \geq p_0$ (see Queffélec [10]). In the sequel we use letters $N_0, N_1, \Theta, \Theta_0, \Theta_1, M_0, M_1, p_0$ in the meaning introduced above.

Definition 4 *Let $p > 0$. We define a relation $R_p \subset (A^*)^3$ by*

$$(u, w, v) \in R_p \quad \text{iff} \quad u, v \in \mathcal{L}^p(\Sigma_\tau), uw \sqsubseteq^* \tau(v) = \tau(v_0 ... v_{p-1}),\ uw \not\sqsubseteq^* \tau(v_1 ... v_{p-1}).$$

Define a graph G_p with vertices $|G_p| = \{\imath\} \cup \{\tau^k(u) : \ u \in \mathcal{L}^p(\Sigma_\tau), k > 0\}$, initial vertex $\imath$ and edges

$$\begin{array}{ll} \imath \xrightarrow{u} u, & u \in \mathcal{L}^p(\Sigma) \\ \tau^k(u) \xrightarrow{\tau^k(w)} \tau^{k+1}(v), & k \geq 0, (u, w, v) \in R_p. \end{array}$$

Lemma 1 *Let p be any integer satisfying*

$$p > p_0, \ \ p \geq \frac{2M_1 + M_0\Theta_0}{M_0(\Theta_0 - 1)},$$

Then for every $(u, w, v) \in R_p$ we have $|w| > 1$ and for every $k > 0$ and $a \in A$ we have $|\tau^k(w)| \geq 2|\tau^k(a)|$.

Proof: Let $(u, w, v) \in R_p$. Then $\Theta_0(p-1) \leq |u| + |w| \leq \Theta_1 p$, so $|w| \geq (\Theta_0 - 1)p - \Theta_0 \geq 2M_1/M_0 > 1$. Moreover for every $k > 0$ and every $a \in A$ we get

$$|\tau^k(w)| \geq |w| M_0 \Theta^k \geq 2M_1 \Theta^k \geq 2|\tau^k(a)|.$$

Proposition 5 *If τ is a primitive substitution and p satisfies the conditions of lemma 1, then Σ_τ consists exactly of labels of infinite paths in G_p.*

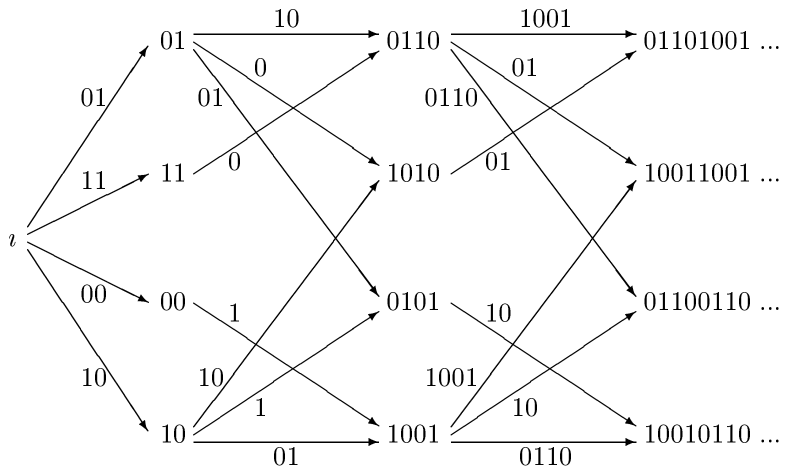
Proof: If

$$\imath \xrightarrow{u^{(0)}} u^{(0)} \xrightarrow{w^{(0)}} \tau(u^{(1)}) \xrightarrow{\tau(w^{(1)})} \tau^2(u^{(2)}) \xrightarrow{\tau(w^{(2)})} \cdots$$

is a label of a path in G_p, then every its initial part is a subword of some $\tau^k(u) \in L_\tau$, and therefore belongs to Σ_τ. Conversely let $x = x^{(0)} \in \Sigma_\tau$. Using recursively proposition 4 we can construct a sequence of points $x^{(i)} \in \Sigma_\tau$ such that, $x^{(i)} = \sigma^{t_i}(\tau(x^{(i+1)}))$ where $0 \leq t_i < |\tau(x_0^{(i+1)})|$. For each $i \in \mathbb{N}$ put $u^{(i)} = x_0^{(i)} \ldots . x_{p-1}^{(i)}$ and $w^{(i)}$ as the unique word in A^* such that $u^{(i)} w^{(i)} \sqsubseteq^* \tau(u^{(i+1)})$. Then $(u^{(i)}, w^{(i)}, u^{(i+1)}) \in R_p$ and for every i, $u^{(0)} w^{(0)} \tau(w^{(1)}) \ldots \tau^i(w^{(i)})$ is a prefix of x. Since every $w^{(i)}$ is nonempty, $x = u_0 w_0 \tau(w_1) \tau^2(w_2) \ldots$ is a path in G_p. $\Box$

Example 4 (Morse) *The Morse substitutive system $\tau : \mathbf{2} \to \mathbf{2}^2$ is given by $\tau(0) = 01$, $\tau(1) = 10$. The Morse subshift is the orbit closure of the Morse sequence $\tau^\infty(0) = 0110100110010110\ldots$.*

For $p = 2$ the graph G_p of the Morse subshift is



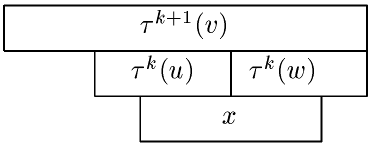
Theorem 4 *Let $\tau : A \to A^+$ be a primitive substitution. Then Σ_τ is a nondeterministic realtime subshift.*

Proof: Suppose that p satisfies the conditions of lemma 1. We describe a nondeterministic Turing machine which accepts Σ_τ following a path in G_p. It works in stages. By the end of stage k it has arrived from the initial state to some vertex $\tau^k(u)$, it keeps in its finite memory the word u and in its pushdown stores all words $\tau^k(a)$, $a \in A$. In stage $k+1$ it chooses for given u nondeterministically some triple $(u, w, v) \in R_p$, verifies whether the next part of the input word is $\tau^k(w)$, stores in its finite memory v and in its pushdown stores it constructs all words $\tau^{k+1}(a)$, $a \in A$. This construction takes $2|\tau^k(a)|$ steps which is less or equal to $|\tau^k(w)|$, so it can be done in real time.

For every $a \in A$ we have 8 pushdown stores denoted by $(a, 0), \ldots, (a, 7)$. The use of pushdown stores alternates in even and odd stages. At the end of an even stage $k = 2n$, $n \in I\!N$, a word u is stored in the finite memory and words $\tau^k(a)$ are stored in pushdown stores $(a, 0)$, $(a, 1)$, $(a, 6)$ (in reading order). In stage $k+1$ an element $(u, w, v) \in R_p$ is chosen nondeterministically and $\tau^k(w)$ is constructed and compared with the input. To construct $\tau^k(w_0)$, the pushdown store $(w_0, 0)$ is read and stored in $(w_0, 4)$ (in reverse order). These two pushdown stores are therefore used as a tape. To construct $\tau^k(w_1)$, $(w_1, 1)$ is read and stored in $(w_1, 5)$ (also in reverse order). Simultaneously with the construction of $\tau^k(w_1)$, the content of $(w_0, 4)$ is stored back to $(w_0, 0)$ at a faster rate, so for the construction of $\tau(w_2)$ the machine can again use pushdown stores $(w_2, 0)$ and $(w_2, 4)$. Simultaneously with the whole construction of $\tau^k(w)$ the words $\tau^{k+1}(a)$ are constructed in the pushdown stores $(a, 2)$, $(a, 3)$ for all $a \in A$. First, during the construction of $\tau^k(w_0)$ the content of $(a, 6)$ is copied to $(a, 7)$, so that $(a, 7)$ contains $\tau^k(a)$ in the reverse order. Then the pushdown store $(a, 7)$ is read letter by letter (until its bottom is reached) and for every letter b, $\tau(b)$ is written to $(a, 2)$, $(a, 3)$ and $(a, 6)$. Thus at the end of stage $k+1$, the pushdown stores $(a, 2)$, $(a, 3)$, $(a, 6)$ contain $\tau^{k+1}(a)$, and they play the role of $(a, 0)$, $(a, 1)$, $(a, 6)$ in the next even stage. The procedure is initialized in stage 0 when firs p letters of the input are read, stored as u into finite memory and $\tau(a)$ are put into pushdown stores $(a, 0)$, $(a, 1)$ and $(a, 6)$. The machine stops and rejects the input word whenever the input symbol does not coincide with the symbol currently constructed inside $\tau^k(w)$. Thus, if the input word belongs to Σ_τ, there exists an infinite computation. If not, the computation must terminate by the occurrence of the first error. $\Box$

In the next theorem we prove that every substitutive system is even deterministic realtime. The idea is to use the algorithm constructed in theorem 4 and follow all possible paths simultaneously. We have only to prove that the number of possible paths is bounded.

Definition 5 *Let $x \in L_\tau$, $p > 0$. An admissible p-configuration for x is any quintuple (k, l, u, w, v), such that $k \geq 0$, $(u, w, v) \in R_p$, $l < |\tau(u_0)|$, $l + |x| > |\tau^k(u)|$, and $x_i = \tau^k(uw)_{l+i}$ for $0 \leq i < |x|$.*



Lemma 2 *There exists $p > 0$ and $M > 0$ such that for every $x \in L_\tau$ there exist at most M p-admissible configurations for x.*

Proof: Suppose that p satisfies the conditions of lemma 1 and an additional inequality

$$p > 1 + \left(N_1 + \frac{2N_0}{\Theta - 1}\right)\frac{M_1}{M_0}$$

Let (k, l, u, w, v) be a p-admissible configuration for x. Then

$$M_0(p-1)\Theta^k \leq |\tau^k(u_1 \dots u_{p-1})| < |x| \leq |\tau^{k+1}(v_0 \dots v_{p-1})| \leq pM_1\Theta^{k+1}$$

so

$$\frac{|x|}{pM_1\Theta} \leq \Theta^k < \frac{|x|}{M_0(p-1)}$$

and we get at most $M_2 = \log_\Theta \frac{pM_1\Theta}{M_0(p-1)}$ possibilities for k. Consider now admissible p-configurations with fixed k. Put $x^{(0)} = x$ and for $i > 0$ construct repeatedly by proposition 4 the unique decompositions $x^{(i-1)} = y^{(i)}\tau(x^{(i)})z^{(i)}$ as long as $|x^{(i-1)}| \geq M_1$. Then

$$\tau^{i-1}(x^{(i-1)}) = \tau^{i-1}(y^{(i)})\tau^i(x^{(i)})\tau^{i-1}(z^{(i)})$$

and since $|\tau^{i-1}(y^{(i)})|, |\tau^{i-1}(z^{(i)})| < N_0M_1\Theta^{i-1}$, we get

$$|\tau^i(x^{(i)})| > |\tau^{i-1}(x^{(i-1)})| - 2N_0M_1\Theta^{i-1}$$

Adding these inequalities for i from 1 to j we get

$$|\tau^j(x^{(j)})| > |x^{(0)}| - 2N_0M_1(1 + \cdots + \Theta^{j-1}) > M_0(p-1)\Theta^k - 2N_0M_1\frac{\Theta^j}{\Theta - 1}$$

For $j \leq k$ we get

$$|x^{(j)}| > \frac{|\tau^j(x^{(j)})|}{M_1\Theta^j} > (p-1)\frac{M_0}{M_1} - \frac{2N_0}{\Theta - 1} > N_1$$

Thus we get a unique decomposition

$$x = y^{(1)}\tau(y^{(2)}) \dots \tau^{k-1}(y^{(k)})\tau^k(x^{(k)})\tau^{k-1}(z^{(k)}) \dots \tau(z^{(2)})z^{(1)}$$

Since x is a subword of $\tau^k(uw)$, the decomposition of $\tau^k(uw)$ must be an extension of the decomposition of x (see the remark after proposition 4). Thus the number of possibilities for l is at most the length of uw, which is at most $\Theta_1 p$. Thus if M_3 is the size of the (finite) relation R_p, then the number of admissible p-configurations is at most $M = M_2 M_3 \Theta_1 p$. □

Theorem 5 *Let $\tau : A \to A^+$ be a primitive substitution. Then Σ_τ is a deterministic realtime subshift.*

Proof: Let p satisfy the conditions from lemma 1 and 2, let M be the upper bound for the number of admissible p-configurations from lemma 2. Consider first a machine $\mathcal{M}'$ which differs from the machine constructed in the proof of theorem 4 only in having for every letter $a \in A$ $8M$ pushdown stores denoted (a, i, j), where $0 \leq i < M$ and $0 \leq j < 8$. To retrieve $\tau^k(a)$ in an even stage, the machine uses stores $(a, 0, 0)$ and $(a, 0, 1)$ but constructs the words $\tau^{k+1}(a)$ in all stores $(a, i, 2)$ and $(a, i, 3)$. The stores $(a, i, 0)$ and $(a, i, 1)$ are therefore not used for $i > 0$. In an odd stage the machine uses stores $(a, 0, 2)$ and $(a, 0, 3)$ but constructs the words $\tau^{k+1}(a)$ in all stores $(a, i, 0)$ and $(a, i, 1)$, so the stores $(a, i, 2)$ and $(a, i, 3)$ are therefore not used for $i > 0$. We consider now M machines $\mathcal{M}'$ which work in paralel and follow different paths for a given input. When, during the comparison with the input, an inequality is found, the concerned machine is deactivated. When at the end of a particular stage the i-th machine finds for given u several possible $(u, w, v) \in R_p$, it takes one of these posibilities for itself and activates other dormant machines for the other possibilities. In the next stage the j-th newly activated machine looks for words $\tau^k(a)$ in the stores $(a, j, 0)$ and $(a, j, 1)$ (in an even stage) of the i-th machine. A deactivated machine is immediately ready for such a task even when its stores are not empty. A special symbol can be put in these stores to signify the new bottom. By lemma 2 the number of activated machines never exceeds M. The whole process runs as long as at least one machine is active. □.

6 Sturmian subshifts

The irrational rotations, which are an important class of dynamical systems, can be obtained as factors of Sturmian subshifts. Like substitutive systems, they are minimal. A Sturmian sequence is determined by an infinite sequence of positive integers $a = (a_i)_{i \geq 0}$. The irrational number $a = [0; a_0, a_1, \ldots]$, whose continued fraction expansion is just the sequence a is the angle of the corresponding rotation of the circle. Define a sequence of binary words

$$W_{-1} = 1, \;\; W_0 = 0, \;\; W_1 = 0^{a_0 - 1}1, \;\; W_2 = W_1^{a_1} W_0, \ldots W_{n+1} = W_n^{a_n} W_{n-1}$$

Then $W_i \sqsubseteq_* W_{i+1}$. The Sturmian sequence corresponding to a is the limit $W = \lim_{n \to \infty} W_n$, and the Sturmian subshift Σ_a is the closure of its σ-orbit. An integer sequence $a = (a_i)_{i \geq 0}$ is called polynomial, if there exists a polynomial $p(x)$ with integer coefficients such that $a_n = p(n)$ for every n. An integer sequence a is called polynomial mixture, if there exists an integer m such that for every $0 \leq j < m$ the sequence $b_i = a_{im+j}$ is polynomial. The Sturmian subshifts with polynomial mixtures include all quadratic irrationals (solutions of quadratic equations with integer coefficients) as well as number e. Following theorem has been proved in Blanchard and Kůrka [3].

Theorem 6 *If $a = (a_n)_{n \geq 0}$ is a polynomial mixture, then Σ_a is a nondeterministic realtime subshift.*

7 Cellular automata

Many realtime subshifts occur naturally in the study of cellular automata. If A is a finite alphabet, denote by $A^{\mathbb{Z}} = \{u : \mathbb{Z} \to A\}$ the set of two-way infinite sequences of letters from A ($\mathbb{Z}$ is the set of integers). Define a metric on $A^{\mathbb{Z}}$ by $d(x, y) = 2^{-n}$, where $n = \min\{i : \; x_i \neq y_i \; \text{ or } \; x_{-i} \neq y_{-i}\}$. Then $A^{\mathbb{Z}}$ becomes a compact metric space homeomorphic to the Cantor set. A cellular automaton is a continuous map $F : A^{\mathbb{Z}} \to A^{\mathbb{Z}}$, which commutes with the shift, i.e., $F(\sigma(x)) = \sigma(F(x))$, where $\sigma(x)_i = x_{i+1}$. Every cellular automaton is determined by some local map $f : A^{2r+1} \to A$ by formula $F(x)_i = f(x_{i-r}, \ldots, x_{i+r})$. If we write the sequence of iterations of a cellular automaton into consecutive lines, we obtain subshifts in columns. For $m \geq 0$ consider an alphabet $B = A^{2m+1}$ and consider the subshift which occurs in $2m + 1$ central columns $-m, \ldots, m$. It is defined by

$$\Sigma_m = \{u \in (A^{2m+1})^{\mathbb{N}} : \; (\exists x \in A^{\mathbb{Z}})(\forall n \geq O)(f^n(x)_{[-m,m]} = u_n)\}$$

Given a class of languages $\mathcal{L}$, we say that a cellular automaton is of class $\mathcal{L}$, if for all m, $L(\Sigma_n)$ belongs to $\mathcal{L}$. Cellular automata with regular languages have been studied in Kůrka) [8]. An example of a cellular automaton with nonregular languages has been given by Gilman. We give first an example of a context-free cellular automaton.

Example 5 *Take $A = \{0, 1, 2\}$, with transition function given by*

2	*XY2*	*1*	*XY1*	*XYZW1*	*0*
2	*2*	*1*	*1*	*1*	*0*

where $X, Y, Z, W \in \{0, 1, 2, 3\}$, and the first applicable rule is used. Then the cellular automaton is context-free.

A sample development of this cellular automaton is

...0000000000120...
...0000000011220...
...0000001112220...
...0000111122220...
...0011112222220...

It is easy to see that the one-column subshift is

$$\Sigma_1 = \{0^n 1^m 2^\infty : \ 0 \le n \le m\} \cup \{0^n 1^\infty : \ 0 \le n\} \cup \{0^\infty\}$$

This is a context-free subshift which is a slight modification of Example 1. Put $Q = \{0, 1, 2, 3\}$, $A_1 = \mathbf{2}$, the initial state is $q_0 = 0$ and productions

$$\begin{array}{ccccccccc} (0,\lambda) & \xrightarrow{0} & (1,1) & (0,\lambda) & \xrightarrow{1} & (2,1) & (0,\lambda) & \xrightarrow{2} & (3,1) \\ (1,\lambda) & \xrightarrow{0} & (1,0) & (1,\lambda) & \xrightarrow{1} & (2,\lambda) & (2,0) & \xrightarrow{1} & (2,\lambda) \\ (2,1) & \xrightarrow{1} & (2,1) & (2,1) & \xrightarrow{2} & (3,1) & (3,1) & \xrightarrow{2} & (3,1) \end{array}$$

It is easy to see that the subshifts obtained in any central $2m+1$ columns remain context-free.

A slightly more complicated is cellular automaton considered by Gilman

Example 6 *Take* $A = \mathbf{2}$ *and* $f(x)_i = x_{i+1}x_{i+2}$.

The Σ_1 consists of binary words $u \in \mathbf{2}^{\mathbb{N}}$ such that if $u_{k-1} = 0$ and $u_k = 1$, then $u_i = 1$ for $k < i < 2k$. this is a realtime subshift which is accepted with Turing machine with states $Q = \{0, 1, 2, 3, 4\}$, two binary pushdown stores, initial state $q_0 = 0$ and rules

$$\begin{array}{cccccc} (0,\lambda,\lambda) & \xrightarrow{1} & (1,01,1) & (0,\lambda,\lambda) & \xrightarrow{0} & (2,01,1) \\ (1,\lambda,0) & \xrightarrow{1} & (1,00,1) & (3,0,\lambda) & \xrightarrow{1} & (3,\lambda,00) \\ (1,\lambda,1) & \xrightarrow{1} & (1,0,1) & (3,1,\lambda) & \xrightarrow{1} & (3,1,0) \\ (1,\lambda,1) & \xrightarrow{0} & (2,0,1) & (3,1,\lambda) & \xrightarrow{0} & (4,1,0) \\ (2,\lambda,\lambda) & \xrightarrow{0} & (2,0,\lambda) & (4,\lambda,\lambda) & \xrightarrow{0} & (4,\lambda,0) \\ (2,0,\lambda) & \xrightarrow{1} & (3,\lambda,00) & (4,\lambda,0) & \xrightarrow{1} & (1,00,\lambda) \end{array}$$

If $(0,\lambda,\lambda) \xrightarrow{u} (k, 0^n 1, 0^m 1)$ is a path in the graph, then $|u| = n + m$ so both pushdown stores store integers whose sum is the length of the word accepted. One of the stores holds moreover the number of ones which have yet to appear on the input.

We conjecture that every cellular automaton yields in this way only realtime subshifts. A counterexample to this conjecture would resolve a long standing open problem whether the family of nondeterministic realtime languages is a proper subset of the family of context-sensitive languages.

References

[1] S.O.Aanderaa: On k-tape versus $(k-1)$-tape real time computation, SIAM-AMS 7, pp. 75-96, 1974.

[2] F.Blanchard, G.Hansel: Languages and subshifts, in: *Automata on Infinite Words*, M.Nivat and D.Perrin eds., Lecture Notes in Computer Science 192, 138–146, Springer-Verlag, Berlin, 1985.

[3] F.Blanchard, P.Kůrka: Language complexity of rotations and Sturmian sequences, to appear in *Theoretical Computer Science.*

[4] R.V.Book, S.A.Greibach: Quasi-realtime languages, *Mathematical Systems Theory* 4, 97-111, 1970.

[5] D.Fiebig, U.R.Fiebig: Covers for Coded Systems. *Symbolic Dynamics and its Applications* (P.Walters, ed.) Contemporary Mathematics 135, pp.139-180, AMS 1992

[6] J.E.Hopcroft, J.D.Ullman: *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[7] W.Krieger: On a class of algebraic subshifts. Universität Heidelberg, Mathematisches Institut, Heft Nr 62, Mai 1993.

[8] P.Kůrka: Languages, equicontinuity and attractors in cellular automata. *Ergod. Th. & Dynam. Sys.* 17, 417-433, 1997.

[9] D.Lind, B.Marcus: *An introduction to Symbolic Dynamics and Coding*, Cambridge University Press. Cambridge 1995.

[10] M.Queffélec: Substitution Dynamical Systems Spectral Analysis, Lecture Notes in Mathematics 1294, Springer-Verlag, Berlin 1987.

[11] B. Mossé: Puissances de mots et reconnaissabilité des points fixes d'une substitution, *Theoretical Computer Sciences* 99, 327–334, 1992.

[12] B. Mossé: Reconnaissabilité des substitutions et complexité des suites automatiques, *Bull. Soc. Math. France* 124, 101-118, 1996.

[13] K.Wagner, G.Wechsung: *Computational Complexity.* VEB Deutscher Verlag der Wissenschaften, Berlin 1986.

[14] B.Weiss: Subshifts of finite type and sofic systems, *Monats. Math.* 77, 462-474, 1973.