The complexity of a particular shift associated to a Turing machine

A. Gajardo

Departamento de Ingeniería Matemática, Universidad de Concepción, Casilla 160-C, Concepción, Chile

Abstract

A subshift is associated to each Turing machine, and its properties are studied. The subshift consists in the set of sequences of symbols that the machine reads, together with the states it takes during each evolution, considering every initial configuration. We focus on machines whose associated subshift is recognized by a deterministic pushdown automaton in real-time. It is proved that all of these machines have restrictions on their movements. Moreover, these machines are characterized as machines that cannot make "zigzag" movements of arbitrary length. We also prove that synchronicity is related to restrictions on the machine movements.

Key words: Turing Machine, Formal Language, Symbolic System

1 Introduction

In previous papers, we introduced the notion of t-shift S_T associated to a Turing machine T. The t-shift consists in the set of all the infinite sequences of pairs (state, read symbol) that the machine reaches during its evolution. The language of finite words $L(S_T)$ of S_T has been also studied, and some relationship has been found between the geometry of the underlying lattice and the complexity of $L(S_T)$ [4]. In this paper, we want to go further and establish that if the complexity of $L(S_T)$ is *low*, then the movements of the machine are *strongly* restricted. We will also study the geometry of the set of cells that the machine can reach.

The notion of *support* is defined for every machine T and word $u \in L(S_T)$. It represents the set of cells that the machine T can reach when it has already

Email address: anahi@ing-mat.udec.cl (A. Gajardo).

[1] This work has been supported by CONICYT FONDECYT #1061036.

done some steps, which are characterized by u. In general, to study the support of a machine T is easier than to study its complete dynamics. Understanding the support helps us to understand the machine behavior. Frequently, it is not the whole underlying lattice but a smaller set.

In order to evaluate the complexity of $L(S_T)$, we look for a suitable automaton that recognizes it. For any machine T, we can define a deterministic automaton in which each node represents the state of the cells that the machine has already visited, together with the machine position and internal state. At each iteration, the automaton passes from one node to another by integrating the information about the last visited cell and the machine actions. This defines in general an infinite automaton. But if the support of the machine is *small*, we do not need to register the state of every visited cell since some of them will never be visited again. Then the automaton may be smaller.

The notion of *synchronized word* also has relevance in this context. In literature, this notion is defined in the context of finite automata and in the theory of transitive subshifts of $A^{\mathbb{Z}}$. Both definitions are not equivalent and they do not correspond to identical concepts. We will take the second context. When a subshift has a synchronized word s, the followers of s are independent from the predecessors of s and they can be combined in an arbitrary way. In some sense, when a synchronized word appears, we can forget all the symbols that were observed before. Synchronized words may be related with the complexity of the language $L(S_T)$.

In this paper, we explore the relationships between these three concepts: language complexity, existence of synchronized words, and support.

2 Basic definitions and properties

Definition 1 *A* Turing Machine *over $\mathbb{Z}$ is a 4-tuple (C, Q, δ) where:*

- *C is a finite set, representing the state of the environment at each lattice point, and called* color set*,*
- *Q is a finite set, representing the internal state of the machine, called* state set*,*
- *$\delta = (\delta_C, \delta_Q, \delta_D)$ is the* transition function*, where $\delta_i : C \times Q \to i$, for each $i = C$, Q or D, and $D = \{-1, 1\}$ are the movement possibilities.*

The elements of $\mathbb{Z}$ are called cells*. A configuration of the system is given by an assignment of symbols to each cell, $c : \mathbb{Z} \to C$, called coloration; a position $g \in \mathbb{Z}$; and a state $q \in Q$, i.e., the phase space is $X = C^{\mathbb{Z}} \times \mathbb{Z} \times Q$.*

The global transition function $T : X \to X$ *is defined by* $T((c, g, q)) = (c', g', q')$, *where*

- $q' = \delta_Q(c(g), q)$,
- $g' = g + \delta_D(c(g), q)$,
- $c'(g) = \delta_S(c(g), q)$ *and* $c'(u) = c(u)$ *for all* $u \neq g$.

Definition 2 *Given a Turing machine* $M = (C, Q, \delta)$ *and its associated dynamical system* (X, T), *let* $\pi : X \to C \times Q$ *be defined by* $\pi(c, g, q) = (c(g), q)$ *and let* $\psi : X \to (C \times Q)^{\mathbb{N}}$ *be defined by* $\psi(x) = (\pi(T^n(x)))_{n\in\mathbb{N}}$. *The* t-shift *of* (X, T) *is* $S_T = \psi(X)$.

The set $\psi(X)$ represents all the possible sequences of pairs (symbol, state) that the machine can produce when considering all the possible initial configurations. Given an infinite sequence $y = \binom{\alpha_1\alpha_2\cdots}{q_1q_2\cdots} \in S_T$, we can deduce the machine itinerary. In fact, if we suppose that the initial position is 0, its position at iteration j must be:

$$I(y)_j = \sum_{i=1}^{j-1} \delta_D(\alpha_i, q_i) \quad (\forall 2 \leq j) \quad , \quad I(y)_1 = 0$$

and the set of visited cells is given by

$$V(y) = \{I(y)_j | 1 \leq j\}$$

The initial symbol of the visited cells can be deduced from y and it is given by the following formula

$$c_y(g) = \alpha_i \quad \text{where} \quad i = \min\{j | I(y)_j = g\} \quad (\forall g \in V(y))$$

The partial function c_y is a kind of pre-image of y by ψ in the following sense: if c is an extension of c_y to $\mathbb{Z}^k$, then $\psi(c, 0, q_1) = y$. This means that the sequence $\psi(x)$ contains information about the visited cells and discards the symbols of the other cells. Moreover, it is invariant under translations.

Remark 1 $I(y)$, $V(y)$ *and* c_y *can be defined also if* y *is a finite word. In this context, the following properties hold for every* u, $v \in (C \times Q)^*$:

(1) $I(uv)_j = I(u)_j$, *if* $j \leq |u| + 1$.
(2) $I(uv)_j = I(u)_{|u|+1} + I(v)_{j-|u|}$, *if* $j \geq |u| + 1$.

2.1 Support

Given a language $L \subseteq A^*$ and a monoid morphism $f : (A^*, \cdot) \to (\mathbb{Z}^d, +)$, we define the *support* of L by the set:

$$S = \{f(u) |\ u \in L\}.$$

Given a word w, the *future support* of L on w is:

$$\overrightarrow{S}(w) = \{f(u) |\ wu \in L\},$$

and the *past support* is:

$$\overleftarrow{S}(w) = \{-f(u) |\ uw \in L\},$$

Let us remark that $\overrightarrow{S}(e) = \overleftarrow{S}(e) = S$

In the context of t-shifts, we are interested in the monoid morphism $f(u) = I(u)_{|u|+1}$. In this case, the support of language $L(S_T)$ is called *the support of* T. It represents the set of cells that the machine can reach if it starts at position 0. If $w \in L(S_T)$, then $\overrightarrow{S}(w)$ represents the set of cells that can be reached after producing the word w, and $\overleftarrow{S}(w)$ represents the set of cells that could be visited before producing the word w.

The following property is easily proved for any language L.

Proposition 2 *If w, $v \in A^*$, then*

(1) $\overrightarrow{S}(wv) + f(v) \subseteq \overrightarrow{S}(w)$.
(2) $\overleftarrow{S}(wv) - f(w) \subseteq \overleftarrow{S}(v)$.

Proof.

(1)

$$\begin{aligned}\overrightarrow{S}(wv) + f(v) &= \{f(u) + f(v) |\ wvu \in L\} \\ &= \{f(vu) |\ wvu \in L\} \\ &\subseteq \overrightarrow{S}(w)\end{aligned}$$

(2)

$$\overleftarrow{S}(wv) - f(w) = \{-f(u) - f(w) |\ uwv \in L\}$$

$$= \{-f(uw) |\ uwv \in L\}$$
$$\subseteq \overleftarrow{S}(v)$$

□

Definition 3 *A language L is* factorial *if*

$$(\forall u, v, w \in A^*)\ w \in L \wedge w = uv \Longrightarrow u, v \in L.$$

Proposition 3 *If $L \neq \phi$ is factorial, then*

(1) $0 \in S$.
(2) $\overrightarrow{S}(wv) \subseteq \overrightarrow{S}(v)$.
(3) $\overleftarrow{S}(wv) \subseteq \overleftarrow{S}(w)$.

Proof. L is factorial if $(\forall u \in L)\ v \sqsubseteq u \Rightarrow v \in L$.

(1) Let $w \in L$, $e \sqsubseteq w$, then $e \in L$, since $f(e) = 0$, $0 \in S$.
(2)

$$\begin{aligned}\overrightarrow{S}(wv) &= \{f(u) |\ wvu \in L\}, \text{ but } wvu \in L \Rightarrow vu \in L \\ &\subseteq \{f(u) |\ vu \in L\} \\ &= \overrightarrow{S}(v)\end{aligned}$$

(3) It is analogous.

□

2.2 Support and automata

Let us suppose that a language L is recognized by a given deterministic automaton $M = (A, \Omega, \lambda, o_0, F)$, where Ω, the vertex set, may be infinite. We can observe that the support depends directly on the vertices of the automaton.

Proposition 4 *If u, $w \in L$ are the labels of different paths from o_0 to some vertex $\nu \in \Omega$, then $\overrightarrow{S}(u) = \overrightarrow{S}(w)$.*

From this property, given a vertex ν of the automaton that recognizes L, we can define the *support of* ν, $S(\nu)$, as the support of any of the words that correspond to the label of some path from o_0 to ν. In other words, $S(\nu)$ is equal to the support of the language recognized by the automaton $\overline{M}$ which is equal to M but with a starting vertex equal to ν at the place of o_0, and $S(o_0) = S$.

Proposition 5 *If $\nu \in \Omega$ and $\{w^{(i)}\}_{i=1}^{k}$ are the labels of circuits starting*

at ν, then for each i we have that $S(\nu) + f(w^{(i)}) \subseteq S(\nu)$, and $S(\nu)+ < \{f(w^{(i)}\}_{i=1}^{k} > \subseteq S(\nu)$.

Proof. If u is the label of a path from o_0 to ν, then $S(\nu) = \overrightarrow{S}(u) = \overrightarrow{S}(uw^{(i)})$, then

$$\begin{aligned} S(\nu) + f(w^{(i)}) &= \overrightarrow{S}(uw^{(i)}) + f(w^{(i)}) \\ &\subseteq \overrightarrow{S}(u) \\ &= S(\nu). \end{aligned}$$

$$< \{f(w^{(i)})\}_{i=1}^{k} >= \left\{ \sum_{i=1}^{k} a_i f(w^{(i)}) |\ (\forall i)\ a_i \in \mathbb{N} \cup \{0\} \right\}$$

The proof is obtained by induction on $\sum a_i$. □

Now, if we deal with $L(S_T)$, for some machine T, we have additional properties.

Remark 6 *Given a vertex $\nu \in \Omega$, it holds that:*

(1) If ν has an input edge labeled by (α, q), then all the exiting edges of ν have a label of the form $(\beta, \delta_Q(\alpha, q))$, with $\beta \in C$, because the next state of the machine is uniquely determined by α and q and it is $\delta_Q(\alpha, q)$.

(2) Since every word in $L(S_T)$ defines a unique path in G_M, then $\nu \neq o_0$ has only one exiting edge if and only if every path from o_0 to ν corresponds to an itinerary that has already visited its last cell. Otherwise, ν has exactly $|C|$ exiting edges.

(3) The last assertion is not valid when $|C| = 1$. But, in this case, S_T is of finite type, more precisely, it is a finite set composed by eventually periodic sequences. The cycles exist in a finite quantity. The vertices of the automaton that recognizes S_T have degree 1 (except for o_0). See Figure 1.

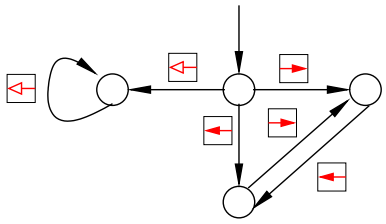

Figure 1. The automaton that recognizes the t-shift of a machine with only one color: white. It has three states. On white arrow, it moves to the left and does not change its state. On red arrows, it moves on the direction of the arrow, but the arrow flips. All the vertices have out-degree one except for the initial one.

This implies that we can distinguish two set of vertices, those with degree 1 corresponding to already visited cells and those with degree $|C|$ corresponding

to unvisited cells.

Proposition 7 *(1) If w is the label of a circuit that passes by a vertex corresponding to an unvisited cell*[2] *, then $f(w) \neq 0$.*
(2) If w is the label of a circuit composed only by vertices of degree 1, then $f(w) = 0$.

The last proposition makes a distinction within terminal connected components of M: those with degree $|C|$ cells and those without them. The first ones correspond to an unbounded propagating movement and the last ones correspond to a bounded periodic movement. Figure 2 shows the automaton of a given machine. We can see the intermediate connected components at the top, and the periodic and final connected components at the bottom.

Labels u and v of different paths that go from o_0 to ν have the same followers. This implies that their future support is the same and that the configurations that they define are identical when restricted to their future support:

$$\overrightarrow{S}(u) = \overrightarrow{S}(v) = H \text{ and } \sigma_{f(u)}(c_u)|_H = \sigma_{f(v)}(c_v)|_H.$$

This suggests a general way for defining a possibly infinite labeled graph that recognizes S_T. The vertices corresponding exactly to the partial configurations and states: $(q, \sigma_{f(w)}(c_w)|_{S(w)})$. If $f(w) \in V(w)$, there is a unique arc whose label is the pair $(q, c_w(f(w)))$. Otherwise, there are $|C|$ arcs with label (q, s). They point to $(\delta(q, s), \sigma_{f(ws)}(c_{ws})|_{S(ws)})$. Such a graph could be interesting when $S(w) \neq \mathbb{Z}^n$, when $V(w) \cap (I(w) + S(w)) \neq V(w)$. For example, in Figure 2, each vertex represents the current state together with the color of the cells that may be visited in the future and whose color is already known.

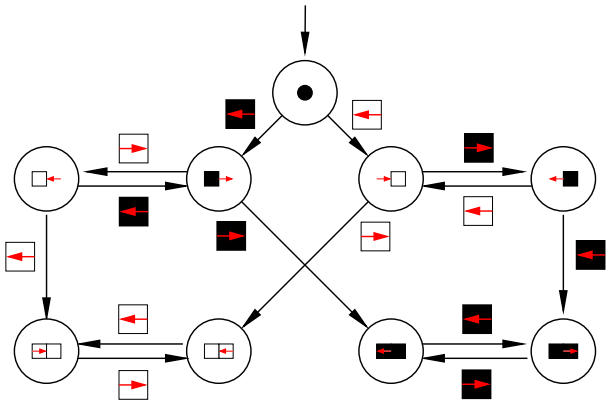

Figure 2. The machine that recognizes this automaton always changes of direction, but it moves backwards over a black cell, and forward over a white cell. The color is always preserved.

[2] Remember that in this case $f(w) = I_{|w|+1}(w)$.

3 Deterministic Pushdown Automata

Now we study S_T when it is recognized in real-time by a pushdown automaton. We take and adapt the definition of real-time recognition of subshifts from [5].

Definition 4 *A deterministic pushdown automaton (DPDA) is a tuple* $M = (A, \Omega, \Gamma, \bot, \lambda, o_0, F)$*, where* A *is the* input alphabet, Ω is the *states set,* Γ *is the stack alphabet,* $\bot \in \Gamma$*,* $\lambda : A \times \Omega \times \Gamma \to \Omega \times (\{e\} \cup \Gamma \cup \Gamma^2)$ *is a partial function called* transition function*,* $o_0 \in \Omega$ *is the* initial state*,* $F \subseteq \Omega$*.*

The transition function λ satisfies: if $\lambda(a, o, \beta) = (o', \mu)$, then μ has at most one $\bot$ and it has one only if $\beta = \bot$ and in that case $\mu_{|\mu|} = \bot$.

A labeled graph, G_M, is associated to M. Its set of vertices is $\Omega \times (\Gamma \setminus \{\bot\})^* \bot$, and the label of an edge $((e, \nu), (f, \mu))$ is 'a' if and only if $(f, \rho) = \lambda(a, e, \nu_0)$ and $\mu = \rho\nu_{[1..|\nu|]}$. The word ν is called the *stack content.*

The language L_M recognized by M consists of all words w in A^* such that there exists a path in G_M with label w, starting on vertex $(o_0, \bot)$.

We say that a subshift Σ is recognized by M in real-time if $L(\Sigma) = L_M$, and we write $\Sigma \in R(1)$.

The language of a subshift is factorial, then we can assume $F = \Omega$ if we allow λ to be a partial function.

It is clear that the language recognized by a DPDA is context-free. Context-free languages satisfy the Pumping Lemma. The following is the strong form of the Pumping Lemma proved by Odgen [6].

Lemma 1 (Odgen, 68) *Given a context-free language* $L \subseteq A^*$*, there exists* $p \in \mathbb{N}$ *such that for all* $w \in L$ *and all set of* p *or more distinguished positions in* w *there exist words* $x, u, y, v, z \in A^*$ *which satisfy:*

(1) $w = xuyvz$*,*
(2) y *contains at least one of the distinguished positions,*
(3) uyv *contains less than* p *of the distinguished positions,*
(4) for all $i \geq 0$*,* $xu^iyv^iz \in L$*.*

This lemma was proved by Odgen using the generating rules of the context-free grammar that generates L. When considering the DPDA that recognizes L, the lemma can also be proved, by regarding the evolution of the states and stack contents. The Odgen's Lemma can be deduced from the following.

Lemma 2 *Given a DPDA* $M = (A, \Omega, \Gamma, \bot, \lambda, o_0)$*, a path of its graph:* $(o_0, \mu^{(0)}), .., (o_n, \mu^{(n)})$*, and given* $I \subset \{0, .., n\}$ *a set of distinguished posi-*

tions of size larger than $p = 2^{|\Omega|^2|\Gamma|^2+1}$*, we have that there exist four positions* $0 \leq l_1 \leq l_2 < l_3 \leq l_4 \leq n$ *such that*

(1) $(o_{l_1}, \mu_0^{(l_1)}) = (o_{l_2}, \mu_0^{(l_2)})$,
(2) $(o_{l_4}, \mu_0^{(l_4)}) = (o_{l_3}, \mu_0^{(l_3)})$,
(3) for all $i \in \{l_1, ..l_4\}$, $|\mu^{(i)}| \geq |\mu^{(l_1)}|$, *and*
(4) for all $i \in \{l_2, ..l_3\}$, $|\mu^{(i)}| \geq |\mu^{(l_2)}|$.

Moreover, $|I \cap \{l_2, .., l_3 - 1\}| > 0$, $|I \cap$ *and* $|I \cap \{l_1, .., l_4 - 1\}| \leq p$.

Proof. Let us define $h_t = |\mu^{(t)}|$. We recursively define a sequence of pairs of indices $(i_k, j_k)_k$ such that $h_{i_k} = h_{j_k}$ or $j_k = n$, and with h_{i_k} non decreasing in k.

The first pair is $(i_0 = 0, j_0 = n)$. Given (i_k, j_k), the pair (i_{k+1}, j_{k+1}) is choosen such that:

(1) if $|I \cap \{i_k, .., i_k\}| \leq 1$ stop.
(2) else, if there exists $t \in \{i_k + 1, .., j_k - 1\}$ such that $h_t = h_{i_k}$ then
  (a) $i_k \leq i_{k+1} < j_{k+1} \leq j_k$,
  (b) $(i_k, j_k) \neq (i_{k+1}, j_{k+1})$,
  (c) $h_{i_{k+1}} = h_{i_k}$,
  (d) $h_{j_{k+1}} = h_{i_k}$ or $j_{k+1} = n$, and
  (e) $|I \cap \{i_{k+1}, .., j_{k+1}\}|$ be maximal.
(3) else
  (a) $i_{k+1} = i_k + 1$, and
  (b) if $h_{j_k} = h_{i_k}$, then $j_{k+1} = j_k - 1$, else $j_{k+1} = j_k$.

The sequence satisfies that for every k and every $t \in \{i_k, .., j_k\}$, $h_t \geq h_{i_k}$. Let us define K to be the last index. It holds that $0 \leq |I \cap \{i_K, .., j_K\}| \leq 1$. Condition 2(e) assures that $|I \cap \{i_{k+1}, .., j_{k+1}\}| \geq |I \cap \{i_k, .., j_k\}|/2$ for every $k < K - 1$. Thus $3 \geq |I \cap \{i_{K-2}, .., j_{K-2}\}| \geq |I|/2^{K-2}$, so $K \geq \log_2(|I|) \geq |\Omega|^2|\Gamma|^2 + 1$.

Now let us look at the tuples $((o_{i_k}, \nu_0^{(i_k)}), (o_{j_k}, \nu_0^{(j_k)}))$. There are at most $|\Omega|^2|\Gamma|^2$ tuples like these. Then there are $k < k'$ such that $(o_{i_k}, \nu_0^{(i_k)}) = (o_{i_{k'}}, \nu_0^{(i_{k'})})$ and $(o_{j_k}, \nu_0^{(j_k)}) = (o_{j_{k'}}, \nu_0^{(j_{k'})})$. We choose the largest k and k' that satisfy the conditions, and define $l_1 = i_k$, $l_2 = i_{k'}$, $l_3 = j_{k'}$ and $l_4 = j_k$. They satisfy the lemma. □

Definition 5 *We say that a machine makes a zigzag movement of width* p *on configuration* c *if there exist four time steps* $t_1 = 0 < t_2 < t_3 < t_4$ *such that the machine position at time* 0 *is* 0*, at time* t_2 *is* $-p - 1$*, at time* t_3 *is* $p + 1$ *and at time* t_4 *is* $-p$.

It is intuitive that from 0 to t_2 the machine can stack the state of all the visited cells, but from t_2 to t_3 it will need to pop this information. Then,

when it comes back to cell $-p$, it does not know its state and cannot verify the correctness of the state sequence. Therefore, a machine that can make a zigzag movement of arbitrary width cannot have an associated subshift in $R(1)$.

Theorem 1 *$S_T \in R(1)$ if and only if there exists an integer N such that T cannot make any zigzag movement of width greater than N.*

Proof. Let us suppose that $S_T \in R(1)$, and let $M = (A, \Omega, \Gamma, \perp, \lambda, o_0)$ be the DPDA that recognizes it. Let p be the number given by the Lemma 2.

By contradiction, let us suppose that the machine can make a zigzag movement of width $2p$ on a configuration c, and let $0 < t_2 < t_3 < t_4$ the four time steps such that the machine position at time 0 is p, at time t_2 is $-p-1$, at time t_3 is $p+1$, and at time t_4 is $-p$.

Moreover, let us suppose that c is the configuration that minimizes t_4, i.e., c produces the shorter zigzag of width $2p$. Let m and n be the leftmost and rightmost visited cells, respectively, between time 0 and t_4 ($m < -p < p < n$).

Let $\tilde{t_2}$ be the last time that cell $-p$ is visited before t_4, and let $\tilde{t_3}$ be the first time that cell p is visited between time $\tilde{t_2}$ and t_4. The minimality of t_4 assures that p is never visited before $\tilde{t_3}$ (see Figure 3).

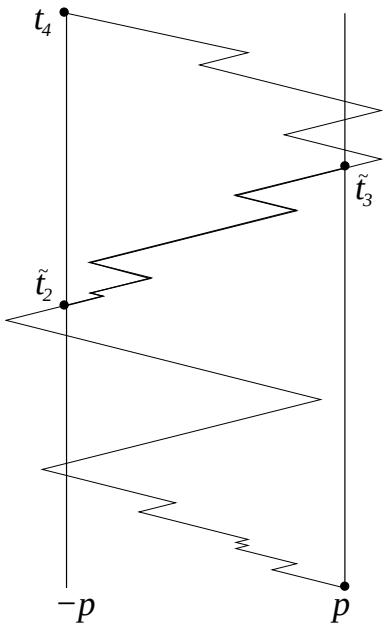

Figure 3. The zigzag movement of width $2p$ and the defined steps $\tilde{t_2}$ and $\tilde{t_3}$. Time goes upward.

Let $w = (\pi(T^i(c, 0, q_0)))_{i=0}^{t_4}$ be the word that T produces on c.

Let $P = (o_0, \mu^{(0)}), ..(o_n, \mu^{(n)})$ be the path in the graph of M, with label w. We apply the Lemma 2 on this sequence with distinguished positions $I = \{\tilde{t_2} + 1, .., \tilde{t_3} - 1\}$, this is possible because $\tilde{t_3} - \tilde{t_2} - 2 \geq 2p - 2 > p$. Let $l_1 \leq l_2 < l_3 \leq l_4$ be the indices given by the lemma.

The main point of our argument is based on the fact that at time $\tilde{t_2}$ the cells between $-p$ and p have already been visited. Then, the vertices $\{(o_{\tilde{t_2}}, \mu^{(\tilde{t_2})}), .., (o_{\tilde{t_3}}, \mu^{(\tilde{t_3})})\}$ have out-degree one. Let us remark that in the case of a DPDA, the out-degree of a vertex depends only on the state o and the first letter of μ.

We distinguish two cases, (a) $\{l_1, .., l_2\} \subseteq I$ and (b) $\{l_1, .., l_2\} \not\subseteq I$.

Case (a): $(o_{l_1}, \mu_0^{(l_1)}) = (o_{l_2}, \mu_0^{(l_2)})$, are both of out-degree one, then $o_{l_1+1} = o_{l_2+1}$. Moreover, from condition 3 and 4 of the lemma, the symbol $\mu_0^{l_1}$ is not popped at l_1 neither at l_2. Then $\mu_0^{l_1+1} = \mu_0^{l_2+1}$, and we can apply induction to conclude that the sequence $((o_t, \mu_0^{(t)}))_{t=l_1}^{t_4}$ is periodic of period smaller than p, then so w, but in such case the position at time t_4 cannot be $-p$, which is a contradiction.

Case (b): Since $\{l_2, .., l_3 - 1\} \cap I \neq \phi$, $\{l_1, .., l_2\} \not\subseteq I$, and $l_4 - l_1 < |I|/2$, then $\{l_3 + 1, .., l_4\} \subseteq I$.

We cannot apply the arguments of case (a) because the stack deepness may decrease.

The conditions of Lemma 2 imply that the concatenation $\tilde{P}$ of paths $(o_t, \mu^{(t)})_{t=0}^{l_1}$, $(o_t, \tilde{\mu}^{(t)})_{t=l_2+1}^{l_3}$ and $(o_t, \mu^{(t)})_{t=l_4+1}^{t_4}$, where $\tilde{\mu}^{(t)} = \mu_{0..h}^{(t)} \mu_{g..h_t}^{(t)}$, $g = |\mu^{(l_1)}|$, $h = |\mu^{(l_2)}|$ and $h_t = |\mu^{(t)}|$, is also a path in the graph of M.

This new path has a label, say $\tilde{w}$, and corresponds to a machine trajectory over some initial configuration $\tilde{c}$.

The out-degree of these vertices is equal to the out-degree of vertices with equal index in P. Then, a cell is visited for the first time at index t in $\tilde{c}$ if and only if it is also visited for the first time at iteration t in c.

From $\tilde{t_3}$ to t_4 the machine has moved from cell p to cell $-p$, passing over cell $p+1$. Then vertex $\tilde{t_3}$ has out-degree one and vertex $\tilde{t_3}+1$ has out-degree larger than one. Let $i \in \{0, .., l_1 - 1, l_2, .., l_2 - 1, l_4, ..\tilde{t_3} - 1\}$ be the smallest index for which the machine visits p when starting with $\tilde{c}$.

First, we can assert that $i \notin \{l_4, ..\tilde{t_3} - 1\}$, because in c the cell p was visited only at iteration 0, and these vertex are in P.

In $\tilde{P}$, the vertex with index i has out-degree larger than one, then in P too. Before iteration $\tilde{t_3}$ the machine is to the left of p when starting with c. Then,

if a cell x is visited for the first time at iteration i, the position of the machine at iteration $i-1$ is $x+1$. But this is not the case in $\tilde{P}$, where the position of i is maximal. We conclude that $i \in \{0, l_2\}$.

If $i = l_2$, its position is equal to the position of the machine at iteration l_1 in P, which is strictly to the left of p in both P and $\tilde{P}$. The only possibility for i is to be 0.

Let us look at cell $-p$, which is visited with index t_4 in P and $\tilde{P}$. It was already visited in c at time t_4, then it should be already visited in $\tilde{c}$ too, say with index $j \in \{0, .., \tilde{t_3}\}$. This implies that $\tilde{c}$ produces a zigzag movement of width $2p$, but it is shorter than w, which is a contradiction.

In order to prove the converse direction, let us suppose that the width of the zigzag movement that a given machine T can make is bounded, say by N, then after doing a cycle longer than N, it cannot revisit the farthest cells in the cycle, otherwise it will make a zigzag movement of width larger that N. This implies, in particular, that the support of the machine at that time is no longer $\mathbb{Z}$. Up to this iteration, the word can be verified with a DPDA having a stack alphabet of C^N characters. The machine can make another long cycle in the opposite direction, but in such case it will be trapped between both cycles and its movement will eventually become periodic. On the contrary, if it makes no more long cycles, its dynamic will be that of a *sofic* automaton [4], which means that from then on its language can be recognized by finite automaton with C^N vertices. □

Proposition 8 *If $S_T \in R(1)$, then there exists $w \in L(S_T)$ such that $S(w) \neq \mathbb{Z}$.*

Proof: If S_T is sofic, it has already been proved in [2] that its movement from a given configuration is either propagative or periodic. In any case, its support will eventually be either $\mathbb{Z}_-$, $\mathbb{Z}_+$ or a finite set. This means that there is a word with a proper support.

Let N be the widest zigzag movement that T can make. If S_T is not sofic, T can make cycles of arbitrary width (see [4]). In particular, it can make a cycle of width $N+1$. Let w be the word associated to this cycle. If we are supposing that the machine starts at cell 0, and the cycle is to the right, then the cycle visits cell $N+1$. But cell $N+1$ cannot be visited any more, then $S(w) \subseteq \mathbb{Z}_- \cup \{0, .., N\} \neq \mathbb{Z}$. □

The converse of the last proposition is not true. We found several examples of Turing machines with a restricted support but which can make zigzag movements of arbitrary length. For example, the machine with two states $\{\rightarrow, \leftarrow\}$ that follows the arrow if it is over a black cell and rebounds (goes backward and changes of state) over white cells. Its support stops being $\mathbb{Z}$ when it finds

a white cell. But if it starts between two white cells, its movement will be the repetition of zigzag movements with of width equal to the distance between these white cells.

4 Synchronizing words

Definition 6 *Given a language $L \subseteq A^*$, a word $s \subseteq L$ is called* synchronizing *if*

$$(\forall u, v \in A^*)\ us \in L \wedge sv \in L \Rightarrow usv \in L$$

This concept is introduced in the context of transitive and two-sided subshifts [1]. A transitive subshift $\Sigma \subseteq A^{\mathbb{Z}}$ is called *synchronized* if it has a synchronizing word. If a system is synchronized it is also coded, which is an important concept that means that the subshift is recognized by a numerable and strongly connected Fisher automaton.

Our systems S_T are not two-sided shifts ($S_T \subseteq A^{\mathbb{N}}$), and in general not transitive. Anyway, it is not difficult to characterize synchronizing words in terms of the support.

Proposition 9 *A word $s \in L(S_T)$ is synchronizing if and only if,*

$$\overleftarrow{S}(s) \cap (\overrightarrow{S}(s) + I(s)) \subseteq V(s).$$

From this property, we obtain the following.

Proposition 10 *If a word $s \in L(S_T)$ is synchronizing, then either $\overleftarrow{S}(s)$ or $\overrightarrow{S}(s)$ is smaller than $\mathbb{Z}$.*

We cannot assure that it is the future support which is restricted. In fact, non-reversible machines can be found such that the past support is finite and the future support is $\mathbb{Z}$. These systems have several synchronizing words. Examples are the machines of class U02 in [3].

Other interesting examples are the systems that admit periodic points. As the following property establishes, periodic points define synchronizing words. But in this case, the synchronizing word only belongs to the periodic trajectory; its existence is more a property of the point than of the system.

Proposition 11 *If $T^p(x) = x$, then $s = (\pi(T^i(x)))_{i=0}^{p-1}$ satisfies $\overrightarrow{S}(s) = V(s) = \overleftarrow{S}(s)$ and it is synchronizing.*

Let us focus now on T reversible, i.e., T one to one. If T is reversible, a two-sided shift can be associated to T, in fact, T^t is well defined for every $t \in \mathbb{Z}$. Thus, we define the shift $\overline{S}_T = \{(\pi(T^t(x)))_{t\in\mathbb{Z}} | x \in C^{\mathbb{Z}} \times \mathbb{Z} \times Q\}$, and all the other definitions work.

In a reversible system, the past support is either infinite or the machine is over a periodic point, in which case the future support is also finite. We can conclude with the following property.

Proposition 12 *If T is reversible and $s \in L(\overline{S}_T)$ is synchronizing, then $\overrightarrow{S}(s)$ and $\overleftarrow{S}(s)$ are proper subsets of $\mathbb{Z}$.*

Again, the converse is not true, the example being a machine that rebounds on a given color, say white. A word containing white color will have a past support equal to the future support, both being different from $\mathbb{Z}$ and not bounded. Such a word is not synchronizing.

Finally, we analyze the case of transitive two-sided subshifts. When $\overline{S}_T$ is transitive, the machine T has no periodic points [4].

Proposition 13 *If T is reversible and it has no periodic points, then for each $w \in L(S_T)$, $\overleftarrow{S}(w)$ and $\overrightarrow{S}(w)$ are not finite.*

Proof. If $\overrightarrow{S}(w)$ is finite, then all the futures of w are periodic, but T has no periodic points; then $\overrightarrow{S}(w)$ cannot be periodic.

If $\overleftarrow{S}(w)$ is finite, then all the pasts (which are infinite) of w are periodic points; then $\overleftarrow{S}(w)$ cannot be finite. □

This property, together with Property 9, imply the following.

Proposition 14 *If T is reversible and has no periodic points, then s synchronizing implies that $\overrightarrow{S}(s) \neq \mathbb{Z}$.*

5 Conclusions

The notion of support of a finite word in the t-shift associated to a Turing machine is introduced. It is the set of cells that the machine can visit after following a given trajectory. Then it expresses the movement restrictions of the machine.

We intended to relate this concept with the complexity of the language of the t-shift, and with the existence of synchronizing words.

We proved that if the language of the t-shift is recognized by a deterministic pushdown automaton (DPDA), then its support is proper. But the converse is not true. We also characterized the class of machines whose t-shift is recognized by a DPDA. They correspond to machines that cannot make zigzag movements of arbitrary length.

Synchronicity of S_T is also related with restrictions in the movement of T, but only when the machine is reversible and has no periodic points. In this case, the existence of a synchronizing word implies that its support is proper. But the converse is open.

References

[1] F Blanchard and G Hansel. Coded systems. *Theor. Comput. Sci.*, 44(1):17–49, 1986.

[2] A. Gajardo. Sofic one head machines. In B. Durand, editor, *Journées Automates Cellulaires*, pages 54–64, 2008.

[3] A. Gajardo and E. Goles. Dynamics of a class of ants on a one-dimensional lattice. *Theor. Comput. Sci.*, 322(2):267–283, 2004.

[4] A. Gajardo and J. Mazoyer. One head machines from a symbolic approach. *Theor. Comput. Sci.*, 370:34–47, 2007.

[5] P. Kůrka and A. Maass. Realtime subshifts. *Theoret. Comput. Sci.*, 237:307–325, 2000.

[6] William F. Ogden. A helpful result for proving inherent ambiguity. *Mathematical Systems Theory*, 2(3):191–194, 1968.