SOFIC TREE-SHIFTS

NATHALIE AUBRUN AND MARIE-PIERRE BÉAL

Abstract. We introduce the notion of sofic tree-shifts which corresponds to symbolic dynamical systems of infinite ranked trees accepted by finite tree automata. We show that, contrary to shifts of infinite sequences, there is no unique reduced deterministic irreducible tree automaton accepting an irreducible sofic tree-shift, but that there is a unique synchronized one, called the Fischer automaton of the tree-shift. We define the notion of almost of finite type tree-shift which are sofic tree-shifts accepted by a tree automaton which is both deterministic and co-deterministic with a finite delay. It is a meaningful intermediate dynamical class in between irreducible finite type tree-shifts and irreducible sofic tree-shifts. We characterize the Fischer automaton of an almost of finite type tree-shift and we design an algorithm to check whether a sofic tree-shift is almost of finite type. We prove that the the Fischer automaton is a topological conjugacy invariant of the underlying irreducible sofic shift.

1. Introduction

In [1] we introduced the notion of tree-shifts of finite type defined as sets of infinite ranked trees avoiding a finite number of forbidden patterns. Infinite trees have a natural structure of one-sided symbolic dynamical systems equipped with several shift transformations. The ith shift transformation applied to a tree gives the subtree rooted at the child number i of the tree. Tree-shifts are highly interesting to study as they constitute an intermediate class in between one-sided shifts of infinite sequences and multidimensional shifts.

The conjugacy of multidimensional shifts of finite type, also called textile systems or tiling systems (see for instance [17],[20],[13],[10]), is undecidable. The decidability of the conjugacy of finite type shifts of bi-infinite sequences is unknown. However, the conjugacy of finite shifts of one-sided infinite sequences is decidable ([25], see also [18]). In [1], we extended William's result to trees, showing that the conjugacy of irreducible tree-shifts of finite type is decidable.

In this paper, we focus on sofic tree-shifts, which are shifts of infinite trees accepted by finite (bottom-up) tree automata, and thus whose set of patterns is a recognizable set of finite trees. The goal is to extend to trees some results of sofic shifts of infinite sequences, to define a hierarchy a sofic tree-shifts and characterize each level of this hierarchy.

We introduce the notion of irreducible sofic tree-shifts. We show, that, unlike for sofic shifts of sequences, an irreducible sofic tree-shift may be accepted by several reduced deterministic irreducible tree automata. This is due to the lack of a synchronizing block, even in reduced deterministic irreducible automata. We introduce the notion of synchronized tree automaton and the notion of Fischer automaton of an irreducible sofic tree-shift. We prove that the Fischer automaton is the unique reduced synchronized deterministic tree automaton accepting an irreducible sofic tree-shift. We also define the Krieger automaton of a sofic tree-shift and compare it to the Fischer automaton in the case the tree-shift is irreducible.

The existence of the Fischer automaton allows us to introduce the class of almost of finite type tree-shifts, which extends the known notion of almost of finite type shifts of sequences. Almost of finite type tree-shifts are sofic shifts accepted by a tree automaton which is both deterministic and co-deterministic with a finite delay. The almost of finite type concept was introduced by Marcus in [19] for coding purposes. The class of almost of finite type shifts is a meaningful class of shifts between irreducible shifts of finite type and irreducible sofic shifts (see [8], [26], [14], [6], [4]). The class contains strictly the class of irreducible shifts of finite type and is strictly contained in the class of sofic shifts. The class is stable by conjugacy and it is also invariant by a flow equivalence [11]. We characterize the Fischer automaton of an almost of finite type tree-shift and design an algorithm to check whether a sofic tree-shift is almost of finite type. We prove that the the Fischer automaton is a topological conjugacy invariant of the underlying irreducible sofic shift. This extends a similar result from Krieger in [16] (see also [8]) in the framework of sequences.

The paper is organized as follows. In Section 2.1 and Section 2.2, we give basic definitions about tree-shifts and conjugacies between tree-shifts. In Section 3, we define the notion of a tree automaton accepting a tree-shift. We refer to [9], [24], and [21] for more general trees and automata on finite and infinite trees. We define irredudible tree-shifts and irreducible tree automata in Section 3.3, synchronized tree automata in Section 3.4. The notions of Fischer and Krieger automata are introduced in Section 3.5. Almost of finite type tree-shifts are defined and characterized in Section 4. In the algorithmic issue, Section 5, we give a construction of the Fischer automaton of an irreducible sofic tree-shift. We design a polynomial-time algorithm to check whether an irreducible sofic tree-shift given by its Fischer automaton is almost of finite type. A short version of this paper was presented in [2].

2. Definitions

2.1. **Tree-shifts.** We first recall some basic definitions of symbolic dynamics on infinite trees. We consider infinite trees whose nodes have a fixed number of children (ranked trees) and are labeled in a finite alphabet. We restrict to binary trees, but all results extend to the case of trees with d children for a fixed positive integer d.

Let $\Sigma = \{0, 1\}$ and Σ^* be the set of words over Σ. An *infinite tree* t over a finite alphabet A is a complete function from Σ^* to A. A node of an infinite tree is a word of Σ^*. The empty word, that corresponds to the root of the tree, is denoted by ϵ. If x is a node, its children are xi with $i \in \Sigma$. Let t be a tree and let x be a node, we shall denote $t(x)$ by t_x. A sequence of words $(x_k)_{k\geq 0}$ of Σ^* is called a *path* if for all $k, x_{k+1} = x_k i_k$ with $i_k \in \Sigma$.

A *pattern* (or a *finite tree*) is a function $p : L \to A$, where L is a finite prefix-closed subset[1] of Σ^*. The set L is called the *support of the pattern.* A node of a finite tree is a word of L. A node without children is called a *leaf.* A *subtree* of a (finite or infinite) tree t rooted at a node x of t is the tree s defined by $s_y = t_{xy}$ for any $y \in \Sigma^*$ such that xy is a node of t.

If n is nonnegative integer, we denote by $\Sigma^{\leq n}$ the set of words of length at most n on Σ. A *block of height* n, where n is a positive integer, is a pattern whose support is $\Sigma^{\leq n-1}$. The *height* of a block u is denoted by $\text{height}(u)$.

We say that a pattern (resp. block) u of support L is a *pattern of a tree* t (resp. a *block of a tree* t) if there is a word $x \in \Sigma^*$ such that $t_{xy} = u_y$ for any node y of u. We say that u is a pattern (resp. block) of t rooted at the node x. If u is not a pattern (resp. block) of t, one says that t *avoids* p.

Unless otherwise stated, a tree is an infinite tree. When Σ are A are fixed, we denote by $\mathcal{T}$ the set of all infinite trees on A, hence the set A^{Σ^*}.

We define the shift transformations σ_i for $i \in \Sigma$ from $\mathcal{T}$ to itself as follows. If t is a tree, $\sigma_i(t)$ is the tree rooted at the i-th child of t, i.e. $\sigma_i(t)_x = t_{ix}$ for all $x \in \Sigma^*$. The set $\mathcal{T}$ equipped with the shift transformations σ_i is called the *full shift* of infinite trees over A.

We define a *tree-shift* X of $\mathcal{T}$ as the set $\mathsf{X}_{\mathcal{F}}$ of all trees avoiding any element of a set of blocks $\mathcal{F}$. The set $\mathcal{F}$ is called *a set of forbidden blocks* of X. If the set of trees on A is equipped with the usual product topology, where the topology in A is the discrete one, a tree-shift is closed and $\sigma_i(X) \subseteq X$ for any shift transformation σ_i. A *tree-shift of finite type* (SFT) X of $\mathcal{T}$ is a set $\mathsf{X}_{\mathcal{F}}$ of all trees avoiding any block of a *finite* set of blocks $\mathcal{F}$.

We denote by $\mathcal{L}(X)$ the set of patterns of all trees of the tree-shift X, by $\mathcal{B}(X)$ the set of all blocks of X and by $\mathcal{B}_n(X)$ the set of all blocks of height n of X. The set $\mathcal{L}(X)$ is called *the set of allowed patterns* of X. If u is a block of height n with $n \geq 2$ and $i \in \Sigma$, we denote by $\sigma_i(u)$ the block of height $n-1$ such that $\sigma_i(u)_x = b_{ix}$ for $x \in \Sigma^{\leq n-2}$. The block u is written $u = (u_\varepsilon, \sigma_0(u), \sigma_1(u))$.

A set of patterns L is *factorial* if $u \in L$ and v is a subtree of u rooted at a node x of u (i.e. v is a sub-pattern of u) implies $v \in L$. The set L is *extendable* if for any pattern u in L, there is a pattern $v \in L$ such that the support of v contains the support of u, v and v coincide with u on the support of u, and for any node x of the u, $x0$ and $x1$ are nodes of v.

[1]each prefix of L belongs to L.

The language of allowed patterns of a tree-shift is factorial and extendable. Conversely, such a language of finite trees is the set of allowed patterns a tree shift as is shown in the following result which is similar as the one known for shifts of bi-infinite words (see for instance [7]).

Proposition 1. *Let L be a factorial and extendable set of finite trees. The set $\mathcal{X}(L)$ of infinite trees whose patterns belong to L is a tree-shift and $\mathcal{L}(\mathcal{X}(L)) = L$. Conversely, if X is a subshift, then $X = \mathcal{X}(\mathcal{L}(X))$.*

Proof. Let $X = \mathcal{X}(L)$. It is clear that X is a tree-shift and $\mathcal{L}(X) \subseteq L$. We show that $L \subseteq \mathcal{L}(X)$. Let u be a finite tree of L. Since L is extendable and factorial, we can construct an infinite tree t such that u is a subtree of t rooted at the empty node, and such that any block of t belongs to L. Then $u \in \mathcal{L}(X)$.

For the second assertion, let $L = \mathcal{L}(X)$. It is clear that $X \subseteq \mathcal{X}(\mathcal{L}(X)$. The converse follows by compactness. □

Example 1. In Figure 1 is pictured an infinite tree of a tree-shift X on the alphabet $\{a, b\}$. The forbidden blocks are those containing an even number of a between two b on any path in the tree. This tree-shift is not of finite type.

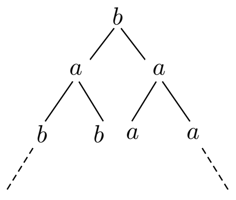


FIGURE 1. An infinite tree of the tree-shift $\mathsf{X}_{\mathcal{F}}$, where $\mathcal{F}$ is the set of patterns containing an even number of a between two b on any path in the tree.

2.2. **Block maps and conjugacies.**

Let A, A' be two finite alphabets, $\mathcal{T}$ (resp. $\mathcal{T}'$) the set of trees on A (resp. A'). Let X be a tree-shift of $\mathcal{T}$ and m be a positive integer. A map $\Phi : X \to \mathcal{T}(A')$ is called an *m-local map* (or an *m-block map*) if there exists a function $\phi : \mathcal{B}_m(X) \to A'$ such that, for all $x \in \Sigma^*$, $\Phi(t)_x = \phi(u)$, where u is the block of t of height m rooted at x. The smallest integer $m-1$ such that Φ is an m-block map, is called the *memory* of the block map. A *block map* is a map which is an m-block map for some positive integer m. The function ϕ is called the block function of Φ.

It is known from the Curtis-Lyndon-Hedlund theorem (see [12]) that block maps are exactly the maps $\Phi : X \to Y$ which are continuous and commute with all tree-shifts transformations, i.e. such that $\sigma_i(\Phi(t)) = \Phi(\sigma_i(t))$ for any $t \in X$ and any $i \in \Sigma$.

The image of X by a block map is also a tree-shift, and is called a *factor* of X. A one-to-one and onto block map from a tree-shift X onto a tree-shift Y has an inverse which is also a block map, as for shifts of sequences. It is called a *conjugacy* from X onto Y. The tree-shifts X and Y are then *conjugate*. We call *sofic* a tree-shift which is a factor of a tree-shift of finite type.

Let X be a tree-shift and m a positive integer. We denote by $X^{(m)}$ the *higher block presentation of* X. It is a tree-shift on the alphabet $\mathcal{B}_m(X)$. For each tree t in $X^{(m)}$, there is tree t' in X such that, for each node x, t_x is the block of height m of t' rooted at x. The shifts X and $X^{(m)}$ are conjugate (see [1]).

3. Sofic tree-shifts

3.1. **Tree automata.** In this section we consider bottom-up tree automata accepting finite or infinite trees. Such an automaton starts its computation from the the leaves, or from the infinite branches, and moves upward. A (finite) *tree automaton* is here a structure $\mathcal{A} = (V, A, \Delta)$ where V is a finite set of states, A is a finite set of input symbols, and Δ is a set of transitions of the form $(p, q), a \to r$, with $p, q, r \in V$, $a \in A$. A transition $(p, q), a \to r$ is called a transition *labeled by* a, *going out of* the pair of states (p, q) and *coming in* the state q. A transition $(p, q), a \to r$ will be pictured by

$$\begin{tikzcd} & a:r \arrow[dl, dash] \arrow[dr, dash] & \\ p & & q \end{tikzcd}$$

Note that no initial nor final state is specified. This means that all states are both initial and final. All tree automata considered here will be finite.

Such an automaton is *deterministic* if for each pair of states (p, q) and for each $a \in A$, there is at most one transition $(p, q), a \to q$. Then the set of transitions defines a partial function δ from $V^2 \times A$ to V. We denote by $\delta(p, q, a)$ the unique state r such that $(p, q), a \to r \in \Delta$ when such a transition exists.

A (bottom-up) *computation* of $\mathcal{A}$ on the infinite tree t is an infinite tree c labeled in V such that, for each node x of t, there is a transition $(c_{x0}, c_{x1}), t_x \to c_x \in \Delta$. A tree t is *accepted* by $\mathcal{A}$ if there exists a computation of $\mathcal{A}$ on t.

Given a tree automaton $\mathcal{A}$, it is always possible to transform it into a deterministic tree automaton which accepts the same set of trees (this process is called a determinization, see for instance [9] for details). In the sequel, we assume that all states of an automaton are *accessible*, i.e. each state ends some computation of the automaton.

A finite tree is *complete binary* if any node has zero or two children. A (bottom-up) *finite computation* of $\mathcal{A}$ on a finite tree t is a finite complete binary tree c on V such that, for each node x of t, there is a transition $(c_{x0}, c_{x1}), t_x \to c_x \in \Delta$.

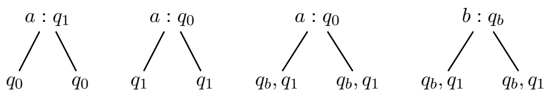
The set of infinite trees accepted by $\mathcal{A}$ is a tree-shift. Indeed, since all states of $\mathcal{A}$ are accessible, the language L of patterns accepted by $\mathcal{A}$ is the factorial and extendable. Then X is the tree shift $\mathcal{X}(L)$.

A tree automaton is called *a transition tree automaton* if all transitions have distinct labels. A *transition tree-shift* is a tree-shift (of finite type) which is accepted by a transition tree automaton.

Let $\mathcal{A} = (V, A, \Delta)$ be a tree automaton. Let E be the set of transitions of $\mathcal{A}$. The set of trees t labeled on the alphabet E such that $t_x = (c_{x0}, c_{x1}), t'_x \to c_x$ for some computation c of $\mathcal{A}$ on a tree t' is called the *transition tree-shift* $X_{\mathcal{A}}$ *of* $\mathcal{A}$.

Let $\mathcal{S}_{\mathcal{A}}$ be the tree-shift accepted by $\mathcal{A}$. There is a one-block map $\lambda_{\mathcal{A}} : X_{\mathcal{A}} \to S_{\mathcal{A}}$ where the block function assigns to each transition $e = (c_{x0}, c_{x1}), a \to c_x$ its label a. The transition tree-shift $X_{\mathcal{A}}$ is a shift of finite type. It is accepted by the automaton $\mathcal{A} = (V, E, \mu)$ whose transitions are $(p, q) \xrightarrow{(p,q,a,r)} r$ for $(p, q, a, r) \in \Delta$. It is defined by the finite set of forbidden blocks (e, f, g) of height 2, where $e = (p, q, a, r)$, $f = (p_1, q_1, a_1, r_1)$, $g = (p_2, q_2, a_2, r_2)$, and either $r_1 \neq p$ or $r_2 \neq q$.

Example 2. We define a tree automaton $\mathcal{A}$ with three state q_b, q_0 and q_1 which accepts the tree-shift X of Example 1. The two states q_0 and q_1 only label nodes with an a, and they control the parity of the number of a encountered from any last b below. The state q_b only labels nodes with a b. The transitions of the tree automaton $\mathcal{A}$ are



The proof of the following proposition is similar to the one for shifts of infinite or bi-infinite sequences (see [18], [15]).

Proposition 2. *A tree-shift is sofic if and only if it is accepted by a tree automaton.*

Proof. Let S be a sofic tree-shift. By definition, it is the image of a tree-shift of finite type X by an m-block map Φ. Without loss of generality, by changing X into its conjugate $X^{(m)}$, we can assume that Φ is a one-block map. Then the block function ϕ of Φ extends to patterns. Since X is of finite type, there is a positive integer n such that $X = \mathsf{X}_{\mathcal{F}}$ where $\mathcal{F}$ is a subset of the set of blocks of height n. The shift X is accepted by the automaton $\mathcal{A} = (V, V, \Delta)$ where V is the set of blocks of height $n - 1$ and $(p, q) \xrightarrow{r} r \in \Delta$ if and only if $(r_\varepsilon, p, q) \notin \mathcal{F}$. The tree-shift S is thus accepted by the automaton $\mathcal{B} = (V, A, \Delta')$ where $(p, q) \xrightarrow{\phi(r)} r \in \Delta'$ if and only if $(p, q) \xrightarrow{r} r \in \Delta$.

Conversely, if S is a set of infinite trees accepted by a tree automaton $\mathcal{B} = (V, A, \Delta')$. Let $\mathcal{A} = (V, \Delta', \Delta)$ where $(p, q) \xrightarrow{(p,q,a,r)} r \in \Delta$ if and only if $(p, q) \xrightarrow{a} r \in \Delta'$. Then $\mathcal{A}$ accepts a shift of finite type X and $S = \Phi(X)$ where Φ is the one-block map defined by $\phi(p, q, a, r) = a$. $\square$

3.2. **Symbolic conjugacy.** We define the notion of symbolic conjugacy for tree automata which extends the notion of symbolic conjugacy of automata on sequences (see [5]). Let $\mathcal{A}$ and $\mathcal{B}$ be two tree automata accepting the sofic tree shifts $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$ and whose transition tree-shifts are $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$. We say that $\mathcal{A}$ and $\mathcal{B}$ are *symbolic conjugate* if there exists conjugacies $\Phi : X_{\mathcal{A}} \to X_{\mathcal{B}}$ and $\Psi : S_{\mathcal{A}} \to S_{\mathcal{B}}$ such that the following diagram commutes.

$$\begin{tikzcd} X_{\mathcal{A}} \arrow[r, "\Phi"] \arrow[d, "\lambda_{\mathcal{A}}"'] & X_{\mathcal{B}} \arrow[d, "\lambda_{\mathcal{B}}"] \\ S_{\mathcal{A}} \arrow[r, "\Psi"] & S_{\mathcal{B}} \end{tikzcd}$$

3.3. **Irreducible tree-shifts.** In this section, we define a notion of irreducibility which is suitable for tree-shifts.

A *finite complete prefix code* of Σ^* is a prefix set [2] P of finite words in Σ^* such that any word of Σ^* which is longer than the words of P has a prefix in P.

A tree-shift X is *irreducible* if for each pair of blocks u, v with $u \in \mathcal{B}_n(X)$, there is a tree t in X and a finite complete prefix code $P \subset \Sigma^{\geq n}$, such that u is a subtree of t rooted at ε, and v is a subtree of t rooted at x for all $x \in P$.

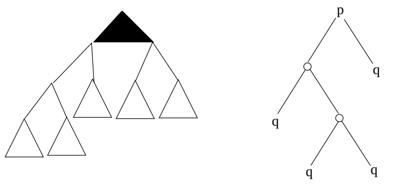


FIGURE 2. (left) An irreducible tree-shift. Let t denotes the tree pictured. If u denotes the black block and v the white one, u is a subtree of t rooted at ε, and v is a subtree of t rooted at each $x \in P$, where P is the complete prefix code $\{00, 010, 011, 1\}$. (right) An hyperpath from q to p in a tree automaton.

A tree automaton is *irreducible* if for each pair of states p, q, there is a finite complete prefix code $P \subset \Sigma^*$ and a finite computation c of the automaton on a pattern u such that $c_\varepsilon = p$ and $c_x = q$ for each $x \in P$. We say in this case that there is an *hyperpath* from q to p labeled by u. For two states p, q of a tree automaton, we say that p is *accessible* from q if there is a hyperpath from q to p.

Proposition 3. *An irreducible automaton accepts an irreducible sofic tree-shift. Conversely, for any irreducible sofic tree-shift, there is an irreducible automaton accepting it.*

[2] i.e. no word is prefix of another one.

Proof. The forward implication is easy. The converse will be proved later by using the notion of Fischer automaton. □

Proposition 4. *Let S and T be two conjugate tree-shifts. Then S is irreducible if and only if T is irreducible.*

Proof. The proof is straightforward. □

3.4. **Synchronizing blocks.** We define below the notion of synchronizing pattern [3] or block of a deterministic tree automaton.

Let $\mathcal{A} = (V, A, \Delta)$ be a deterministic tree automaton accepting a sofic tree-shift X, and u be a pattern (resp. block). We say that u is a *synchronizing pattern (resp. block) of* $\mathcal{A}$ if the set of computations of $\mathcal{A}$ on u is nonempty and all computations of $\mathcal{A}$ on u end in the same state $q \in Q$. We say that the pattern u *focuses* to the state q. A tree t such that all computations of $\mathcal{A}$ on t end in the same state $q \in Q$ is a *synchronizing tree* of $\mathcal{A}$. A deterministic tree automaton which has a synchronizing pattern is called *synchronized.*

Note that if u is a synchronizing pattern, then any pattern v such that u is a subtree of v rooted at ε is also synchronizing.

3.5. **Minimal deterministic tree automata.** Let X be a tree-shift. A *context* ℓ is a pattern where one leaf is replaced by a hole. If u is a pattern, ℓu is the pattern obtained by substituting u in place of the hole of ℓ. If $\ell u \in \mathcal{L}(X)$, we say that ℓ is a *context of* u *in* X. Given a block u, we denote by $\mathrm{cont}_X(u)$ the set of all the contexts of u in X.

Given a tree automaton $\mathcal{A} = (V, A, \Delta)$ accepting a sofic tree-shift X, the *context* of a state $q \in V$ in $\mathcal{A}$ is the set of contexts ℓ, with a hole at the node x, on which there exists a finite computation c of $\mathcal{A}$ with $c_x = q$. We denote this set by $\mathrm{cont}_{\mathcal{A}}(q)$. Note that the context of a pattern u of X is the union of the contexts of the states p such that there is a computation of $\mathcal{A}$ on u ending in p. As a consequence, a sofic tree-shift has only a finite number of distinct contexts.

Let $\mathcal{A} = (V, A, \Delta)$ be an automaton. The automaton $\mathcal{A}$ is said to be *reduced* if $p \neq q$ implies $\mathrm{cont}_{\mathcal{A}}(p) \neq \mathrm{cont}_{\mathcal{A}}(q)$ for all states p, q in V.

Let $\mathcal{A} = (V, A, \Delta)$ and $\mathcal{B} = (V', A, \Delta')$ be two deterministic automata. A *reduction* from $\mathcal{A}$ onto $\mathcal{B}$ is a map h from V onto V' such that, for any letter $a \in A$, one has $(p, q), a \to r \in \Delta$ if and only if $(h(p), h(q)), a \to r \in \Delta'$.

Let $\mathcal{A} = (V, A, \Delta)$ be a deterministic automaton accepting a sofic tree-shift X. We define a deterministic automaton $R(\mathcal{A})$ accepting X called the *reduction of the automaton* $\mathcal{A}$ as follows. The states of $R(\mathcal{A})$ are the classes of the coarsest partition of V such that if p, q belong to the same class, then for each letter a and each state r, $\delta(p, r, a)$ and $\delta(q, r, a)$ (resp. $\delta(r, p, a)$ and $\delta(r, q, a)$) belong to the same class, and $\delta(p, r, a)$ (resp. $\delta(r, p, a)$) is defined if and only if $\delta(q, r, a)$ (resp. $\delta(r, q, a)$) is defined. Let $[p]$ denotes the class

[3]also called a homing pattern or a magic pattern.

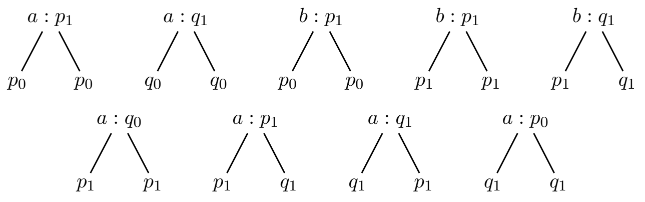
of the state p. The transition $([p], [q]), a \to [\delta(p, q, a)]$ is a transition of $R(\mathcal{A})$ if and only if $\delta(p, q, a)$ is defined. It can be shown that this definition is consistent, i.e. does not depend on the choice of the leader in each class. By definition, the automaton $R(\mathcal{A})$ is reduced and there the map h defined by $h(p) = [p]$ is a reduction from $\mathcal{A}$ to $R(\mathcal{A})$.

The minimization algorithm for deterministic tree automata accepting languages of finite trees, which is for instance described in [9, Section 1.5], can be applied for computing the reductions of tree automata accepting tree-shifts. Remember that all states in tree automata accepting tree-shifts are both initial and final.

In the framework of shifts of bi-infinite words, irreducible sofic shifts have a unique reduced deterministic automaton (i.e. all reduced deterministic automata accepting the shift are equal up to a renaming of the states), see [18]. The situation is quite different for trees since, as is shown below in Example 3, irreducible sofic tree-shifts may have several reduced deterministic tree automata. Indeed, contrary to the situation that we have for shifts of infinite or bi-infinite sequences, an irreducible reduced deterministic tree automaton may not have a synchronizing block.

Proposition 5. *There are reduced irreducible deterministic tree automata which are not synchronized.*

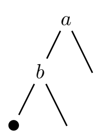
Example 3. Let X be the full tree-shift on the alphabet $A = \{a, b\}$. It is accepted by a trivial one-state automaton with to transitions labeled by a and b. It is also accepted by the deterministic tree automaton $\mathcal{A} = (V, A, \delta)$ described by the following transitions and which is reduced.



This automaton is irreducible. Indeed there is a hyperpath

$$p_0 \rightarrowtail p_1 \rightarrowtail q_0 \rightarrowtail q_1 \rightarrowtail p_0$$

where $p \rightarrowtail q$ denotes a hyperpath from p to q. It is reduced since any two states have distinct contexts. Indeed, the states q_0, q_1 have not the same context as p_0, p_1 since the pattern whose root is b with a left hole is only a context of p_0, p_1. Furthermore the following pattern whose hole is represented with a black dot



is a context of p_0, q_1 only. Hence all states have distinct contexts.

Nevertheless, this automaton is not synchronized. Indeed, let $\mathcal{D}(\mathcal{A})$ be the accessible part from V of the tree automaton $(\mathfrak{P}(V), A, \delta')^4$, where, if $P, Q \in \mathfrak{P}(V)$, $\delta'(P, Q, a) = \{\delta(p, q, a) \mid p \in P, q \in Q\}$ if this set is nonempty, and is not defined otherwise. The automaton $\mathcal{D}(\mathcal{A})$ contains two states $V = \{p_0, q_0, p_1, q_1\}$ and $\{p_1, q_1\}$ and hence no singleton state.

One can overcome this difficulty with the notion of Fischer automaton for irreducible tree-shifts.

Let X be a sofic tree-shift. The *context tree automaton* of X is the deterministic automaton $\mathcal{C} = (V, A, \Delta)$, where V is the set of nonempty contexts of blocks of X. Since X is sofic, V is finite. The transitions of $\mathcal{C}$ are $(\mathrm{cont}_X(u), \mathrm{cont}_X(v)), a \to \mathrm{cont}_X(a, u, v)$, where $u, v \in \mathcal{B}(X)$ and $\mathrm{cont}_X(a, u, v)$ is nonempty.

Proposition 6. *The context tree automaton of a tree-shift X accepts X.*

Proof. It is clear that a tree of X is accepted by the context tree automaton of X. Conversely, if there is a computation of context tree automaton of X on a tree t, any block of t belongs to $\mathcal{B}(X)$. Thus t is in X. □

Proposition 7. *The context tree automaton of a sofic tree-shift is synchronized.*

Proof. Let $\mathcal{C}$ be the context tree automaton of a sofic shift X. There is a finite computation c_1 in $\mathcal{C}$ on some block u_1 ending in $\mathrm{cont}_X(u_1)$. Let us assume that u_1 is not a synchronizing block of $\mathcal{C}$. Hence there is another computation c_2 of $\mathcal{C}$ on u_1 ending in $\mathrm{cont}_X(u_2)$ for some block u_2 such that $\mathrm{cont}_X(u_2) \neq \mathrm{cont}_X(u_1)$. We get $\mathrm{cont}_X(u_2) \subsetneq \mathrm{cont}_X(u_1)$ since $\mathrm{cont}_X(u_2) \neq \mathrm{cont}_X(u_1)$. If u_2 is not a synchronizing block of $\mathcal{C}$, there is another computation c_2 of $\mathcal{C}$ on u_2 ending in $\mathrm{cont}_X(u_3)$. We get $\mathrm{cont}_X(u_3) \subsetneq \mathrm{cont}_X(u_2)$. By iterating this process, we either get a synchronizing block for $\mathcal{C}$ or an infinite strictly decreasing sequence of contexts. Hence, since the number of contexts is finite, there is a synchronizing block. □

We say that a tree automaton $\mathcal{A}' = (V, A', \Delta')$ is a *subautomaton* of a tree automaton $\mathcal{A} = (V, A, \Delta)$ if $V' \subseteq V$ and $\Delta' = \Delta \cap (V' \times V' \times A \times V')$. A subautomaton $\mathcal{A}'$ is *minimal* if any transition of $\mathcal{A}$ going out of states of $\mathcal{A}'$ ends in $\mathcal{A}'$. A *minimal irreducible component* of an automaton is a minimal subautomaton which is irreducible.

Proposition 8. *Let X be an irreducible sofic tree-shift and $\mathcal{C}$ its context automaton. Let z be a synchronizing block of $\mathcal{C}$ and $\mathcal{F}$ be the automaton obtained from $\mathcal{F}$ by keeping only the states accessible from $\mathrm{cont}_{\mathcal{C}}(z)$. The automaton $\mathcal{F}$ is the unique minimal irreducible component of $\mathcal{C}$.*

[4] $\mathfrak{P}(V)$ denotes the set of parts of V.

Proof. We assume that z is a block of height n. Let $q = \mathrm{cont}_{\mathcal{C}}(z)$. Let us show $\mathcal{F}$ is a minimal irreducible subautomaton. It is enough to prove that there is a hyperpath from any state of $\mathcal{C}$ to q. Let $p = \mathrm{cont}_{\mathcal{C}}(u)$ be a state of $\mathcal{C}$. Since X is irreducible, there is a pattern w of X and a finite complete prefix code $P \subset \Sigma^{\geq n}$, such that z is a subtree of w rooted at ε, and u is a subtree of w rooted at x for all $x \in P$. Let c be a computation of $\mathcal{C}$ on w. We have $c_\varepsilon = q$, and for all $x \in P$, $c_x = p$. Hence there is an hyperpath from p to q. $\square$

We define the *Fischer automaton*[5] of an irreducible sofic tree-shift X as the unique minimal irreducible component of its context tree automaton.

Note that each state in this automaton is the context of some synchronizing block of the context tree automaton. Furthermore, any computation of $\mathcal{C}$ on a synchronizing block of $\mathcal{C}$ ends in the Fischer automaton.

Proposition 9. *The Fischer automaton of an irreducible tree-shift X accepts X.*

Proof. Since the Fischer automaton $\mathcal{F}$ of X is a subautomaton of the context tree automaton, any tree accepted by $\mathcal{F}$ is in X. Conversely, let t be a tree of X. We show that any block u of t of height n is accepted by $\mathcal{F}$. Let z be a synchronizing block of $\mathcal{C}$. Since X is irreducible, there is a pattern w of X and a finite complete prefix code $P \subset \Sigma^{\geq n}$, such that u is a subtree of w rooted at ε, and z is a subtree of w rooted at x for all $x \in P$. It follows that there is computation of $\mathcal{C}$ on w. Since any computation of $\mathcal{C}$ on a synchronizing block ends in $\mathcal{F}$, and since $\mathcal{F}$ is a minimal subautomaton of $\mathcal{C}$, we obtain a computation of $\mathcal{F}$ on u. Thus u is accepted by $\mathcal{F}$. $\square$

We now prove that the Fischer automaton is the unique reduced deterministic irreducible and synchronized tree automaton accepting an irreducible sofic tree-shift.

Proposition 10. *Two reduced deterministic irreducible and synchronized tree automata accepting the same irreducible sofic tree-shift are equal up to a renaming of the states.*

Proof. Let $\mathcal{M} = (V, A, \Delta)$ and $\mathcal{M}' = (V', A, \Delta')$ be two reduced deterministic irreducible synchronized automata accepting a same sofic tree-shift X. Let $u \in \mathcal{B}_m(X)$ (resp. $v \in \mathcal{B}_m(X)$) be a synchronizing block of $\mathcal{M}$ (resp. $\mathcal{M}'$). We assume that z focuses to the state p in $\mathcal{M}$. Since X is irreducible, there is a pattern z of X and a finite complete prefix code $P \subset \Sigma^{\geq m}$, such that u is a subtree of z rooted at ε, and that v is a subtree of z rooted at x for all $x \in P$. Since $\mathcal{M}$ and $\mathcal{M}'$ are deterministic, the pattern z is a synchronizing block for both $\mathcal{M}$ and $\mathcal{M}'$. Each computation of $\mathcal{M}$ on z focuses to the state p while each computation of $\mathcal{M}'$ on z focuses to some state p' in V'.

[5]also called the Shannon automaton of the tree-shift. It corresponds to the right Fischer automaton of a sofic shift of sequences.

We define a bijection $\varphi : V \to V'$ as follows. Let q be a state of $\mathcal{M}$. Since $\mathcal{M}$ is irreducible, there is a computation c of $\mathcal{M}$ on a pattern w, and a prefix code P such that $c_\varepsilon = q$ and $c_x = p$ for all $x \in P$. Since $\mathcal{M}$ is deterministic and z synchronizing for $\mathcal{M}$, any finite computation of $\mathcal{M}$ on w ends in q. Hence the context of q in $\mathcal{M}$ is the context of w in X. Moreover, any finite computation of $\mathcal{M}'$ on w ends in a same state q' and the context of q' in $\mathcal{M}'$ is the context of w in X.

Since $\mathcal{M}$ (resp. $\mathcal{M}'$) is reduced, two states having the same context in $\mathcal{M}$ (resp. $\mathcal{M}'$) are equal. Hence if $q = cont_X(w)$, $r = cont_X(v)$, and $(q, r), a \to s$ is a transition of Δ, we have $s = \mathrm{cont}_X(a, w, v)$. We define $\varphi(q) = q'$. One can check that φ defines an isomorphism between $\mathcal{M}$ and $\mathcal{M}'$ in the sense that $(q, r), a \to s$ is a transition of Δ if and only if $(\varphi(q), \varphi(r)), a \to \varphi(s)$ is a transition of Δ'. □

Hence two reduced deterministic irreducible and synchronized tree automata accepting a same tree-shift are equal to the Fischer automaton of this tree-shift.

If $t \in \mathcal{T}$ and ℓ a context. Let ℓt denotes the pattern obtained where the hole is replaced by the infinite tree t. If ℓt is an (infinite) pattern of an infinite tree of X, we say that ℓ is a *context of t in X*. Given a tree t in $\mathcal{T}$, we denote by $\mathrm{cont}_X(t)$ the set of all the contexts of t in X. The *Krieger automaton* of a tree-shift X is the automaton whose states are the nonempty sets of the form $\mathrm{cont}_X(t)$, and whose transitions are $(\mathrm{cont}_X(t), \mathrm{cont}_X(t'), a, \mathrm{cont}_X(a, t, t'))$, where $t, t' \in \mathcal{T}$ and $\mathrm{cont}_X(a, t, t')$ is nonempty. Hence the states of the Krieger automaton are contexts of infinite trees while the states of the context automaton are contexts of finite trees.

Proposition 11. *The Krieger automaton of a sofic tree-shift X accepts X.*

Proof. It is clear that a tree of X is accepted by the Krieger automaton of X. Conversely, if there is a computation the Krieger automaton of X on a tree t, any block of t belongs to $\mathcal{B}(X)$. Thus t is in X. □

Proposition 12. *The Krieger automaton of a sofic tree-shift X is, up to an isomorphism, a subautomaton of the context tree automaton of X. It is reduced and synchronized. If X is irreducible, the Krieger automaton of X has a unique minimal irreducible subautomaton which is the Fischer automaton of X.*

Proof. Let $\mathcal{K} = (V, A, \Delta)$ be the Krieger automaton of a sofic tree-shift X and $\mathcal{C} = (V', A, \Delta')$ its context tree automaton. Let t be an infinite tree and u_n its subtree of height n rooted at ε. We have $\mathrm{cont}_X(u_{i+1}) \subseteq \mathrm{cont}_X(u_i)$ for any $i \geq 0$. Since the number of contexts is finite, there exists a nonnegative integer n such that $\mathrm{cont}_X(u_{n+i}) = \mathrm{cont}_X(u_n)$ for all nonnegative integers i. Let us define $s(t) = \mathrm{cont}_X(u_n)$.

We show that the map $\mathrm{cont}_X(t) \mapsto s(t)$ is well defined and one to one. We have $s(t) = \mathrm{cont}_X(t)$. Indeed, clearly $\mathrm{cont}_X(t) \subseteq s(t)$. Let $c \in s(t) - \mathrm{cont}_X(t)$ with a marked leaf x. There a nonnegative integer n such that

$c(u_i) \in \mathcal{L}(X)$ for all $i \geq n$. Let s be an infinite tree such that c is a subtree of s rooted at ε and t and t is the subtree of s rooted at x. Then $s \in X$ and $c \in \mathrm{cont}_X(t)$. As a consequence, if t, t' are two infinite trees, then $\mathrm{cont}_X(t) = \mathrm{cont}_X(t')$ if and only if $s(t) = s(t')$.

We next show that $\mathcal{K}$ is reduced. If p is a state equal to $\mathrm{cont}_X(t)$ for some tree $t \in \mathcal{T}$, then $\mathrm{cont}_{\mathcal{K}}(p) = \mathrm{cont}_X(t)$. As a consequence, X is reduced.

We now show that $\mathcal{K}$ is synchronized. Let $r_{\mathcal{A}}(t)$ be the number of states of $\mathcal{A}$ ending a computation of an automaton $\mathcal{A}$ on a tree t. By Proposition 7, the context tree automaton is synchronized. Hence, by definition, there is a pattern u such that $r_{\mathcal{C}}(\mathrm{cont}_X(u)) = 1$.

This implies that there for any infinite tree $t \in X$ such that u is a subtree of t rooted at ε, all computations of $\mathcal{C}$ on t end in $\mathrm{cont}_X(u)$. As a consequence, all computations of $\mathcal{K}$ on t end in $\mathrm{cont}_X(u)$. Thus $\mathcal{K}$ is synchronized.

Let us assume that X is irreducible. Let z be a synchronizing block of height n of $\mathcal{K}$ and p ending any finite computation of $\mathcal{K}$ on z. Let W be the set of states accessible from the state p. The automaton $\mathcal{K}' = (W, A, \Delta)$ is a minimal subautomaton of $\mathcal{K}$. Indeed, for any $q \in W$, there is an hyperpath from p to q labeled by u. Further, if v denotes the finite tree obtained by substituting any leaf of u by z, q ends any computation of $\mathcal{K}$ on v. Since X is irreducible, there is a tree t in X and a finite complete prefix code $P \subset \Sigma^{\geq n}$, such that z is a subtree of t rooted at ε, and v is a subtree of t rooted at x for all $x \in P$. Thus there is an hyperpath from q to p in $\mathcal{K}$. Next, if r is a state of a minimal irreducible subautomaton of $\mathcal{K}$ ending a finite computation of $\mathcal{K}$ on some pattern w, then, since X is irreducible, there is a tree t' in X and a finite complete prefix code $Q \subset \Sigma^{\geq n}$, such that z is a subtree of t' rooted at ε, and w is a subtree of t' rooted at x for all $x \in Q$. Thus there is an hyperpath from r to p and $p \in V'$. It follows that K' is the only minimal irreducible subautomaton of $\mathcal{K}$. □

3.6. **Finite type tree-shifts.**

Tree-shifts of finite type are accepted by tree automata having very strong synchronization properties.

Let m be a nonnegative integer. A *deterministic m-local tree automaton* (or an *m-definite tree automaton*) is a deterministic tree automaton $\mathcal{A} = (V, A, \delta)$ such that whenever there are two finite computations c, c' of $\mathcal{A}$ on a same block of height m, then $c_\varepsilon = c'_\varepsilon$ (the computations end in a same state). A tree automaton is *local* (or *definite*) if it is m-local for some nonnegative integer m (and m stands for memory). For instance a transition tree-shift is accepted by a 1-local automaton.

The following proposition was shown in [1].

Proposition 13. *A deterministic local tree automaton accepts a tree-shift of finite type. Conversely, a tree-shift of finite type is accepted by a deterministic local automaton.*

4. Almost of finite type tree-shifts

The following notion of almost of finite type tree-shift extends to trees the notion of almost of finite type shift of bi-infinite sequences. Almost of finite tree-shifts are irreducible and accepted by tree automata which are deterministic and codeterministic with a finite delay. The class contains strictly the class of irreducible tree-shifts of finite type and is strictly contained in the class of irreducible sofic tree-shifts.

Let X, Y be two tree-shifts. A block map $\Phi : X \to Y$ is *left closing* if there are non negative integers m, a (m for memory and a for anticipation) such that, whenever $\Phi(s) = s'$, $\Phi(t) = t'$ with $s'_x = t'_x$ for all $x \in \Sigma^i$ with $0 \leq i \leq (a+m)$, and $s_x = t_x$ for all $x \in \Sigma^i$ with $0 \leq i \leq m-1$, then $s_x = t_x$ for any $x \in \Sigma^m$. Note that he trees have to be pictured horizontally with their root on the right side to see the left closing property.

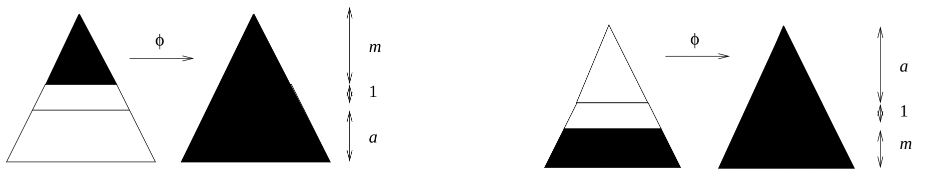


Figure 3. (left) The notion of left closing map. (right) The notion of right closing map.

A tree automaton $\mathcal{A} = (V, A, \Delta)$ is *left closing* if any two computations of $\mathcal{A}$ on a same tree and ending in a same state are equal. Equivalently, there is a nonnegative integer a such any two finite computations c, c' of $\mathcal{A}$ on a same block $u \in \mathcal{B}_a(X)$ such that $c_\varepsilon = c'_\varepsilon$ satisfy $c_x = c'_x$ for all $x \in \{0, 1\}$.

It is straightforward to check that the composition of a left closing map with a conjugacy is still left closing.

A block map $\Phi : X \to Y$ is *right closing* if there are non negative integers m, a such that, whenever $\Phi(s) = s'$, $\Phi(t) = t'$ with $s'_x = t'_x$ for all $x \in \Sigma^i$ with $0 \leq i \leq (a+m)$, and $s_x = t_x$ for all $x \in \Sigma^i$ with $a+1 \leq i \leq (a+m)$, then $s_x = t_x$ for all $x \in \Sigma^a$. The map Φ is *right resolving* if it is a one-block map and if one can choose $m = 1$ and $a = 0$.

We now introduce the notion of resolving block (see [19]). Let Φ be a 1-block map from X to Y. A *resolving block* is a block z of Y for which there is a positive integer m such that if $\phi(u) = \phi(v) = z$ where u and v are two blocks of X of height m and ϕ is the block function of Φ extended two the blocks, then $u_\varepsilon = v_\varepsilon$.

The composition of a one-block map with a resolving block with a conjugacy is still a one-block map with a resolving block.

A sofic tree-shift is *almost of finite type* (AFT) if it is the image of an irreducible tree-shift of finite type via a one-block map which is right resolving, left closing, and has a resolving block.

Proposition 14. *Let S and T be two conjugate irreducible sofic tree-shifts. Then S is AFT if and only if T is AFT.*

Proof. We assume that S is AFT. Let Θ be a one-block map from a tree-shift of finite type X onto a sofic tree-shift S, which is right resolving, left closing, and has a resolving block. Let Φ be a conjugacy from S to T. We can assume that Φ is an m block map with an n-block inverse. Let $X^{(n)}$ (resp. $S^{(n)}$) be the higher block presentation of X (resp. S) of order n, and τ (resp. τ') be the natural conjugacy from X to $X^{(n)}$ (resp. from S to $S^{(n)}$). The tree-shift $X^{(n)}$ is a shift of finite type. The one-block map Θ extends in a natural way to a one-block map Θ from $X^{(n)}$ to $S^{(n)}$. Let $\Phi' = \Phi \circ \tau'^{(-1)}$ and $\Psi = \Phi' \circ \Theta'$. The map $\Psi : X^{(n)} \to T$ is a one-block right resolving map. Since it is also left closing and has a resolving block, the tree-shift T is AFT.

$$\begin{tikzcd}
X \arrow[r, "\tau"] \arrow[d, "\Theta"'] & X^{(n)} \arrow[d, "\Theta'"'] \arrow[dr, "\Psi"] & \\
S \arrow[r, "\tau'"] \arrow[rr, bend right, "\Phi"'] & S^{(n)} \arrow[r, "\Phi'"] & T
\end{tikzcd}$$

□

We say that an automaton is an *almost of finite type automaton* (AFT automaton) if it is irreducible, deterministic, left closing and synchronized.

Proposition 15. *A tree-shift is AFT if and only if it is accepted by an AFT automaton.*

Proof. Let S be an AFT tree-shift on the alphabet A. Let $\Phi : X \to S$ be a one-block map from an irreducible tree-shift of finite type X onto S which is right resolving, left closing and has a resolving block. Without loss of generality, by changing X into a higher block presentation of X, we can assume that Φ is left closing with parameters $m' = 1$.

Let $\mathcal{A} = (V, E, \Delta)$ be an irreducible transition tree automaton accepting X. Let $\mathcal{B} = (V, A, \Delta')$ where $(p, q, \phi(e), r) \in \Delta'$ if and only if $(p, q, e, r) \in \Delta$. Since Φ is a one-block right resolving map, $\mathcal{B}$ is a deterministic automaton.

Since Φ has a resolving block z, the block z is a synchronizing block of $\mathcal{B}$ focusing to some state q. Let us keep in $\mathcal{B}$ only the states accessible from q. Since S is irreducible, $\mathcal{B}$ remains irreducible and still accepts S.

Let us now show that $\mathcal{B}$ is left closing. If $\mathcal{B}$ were not left closing, there would be two distinct computations c, c' of $\mathcal{B}$ on a same tree t ending in a same state p. Let $(p, q, e, r) \in \Delta$ be a transition going out of (p, q) for some states $p, r \in V$ (if there is none, there is a transition of $\mathcal{A}$ going out of (q, p) since $\mathcal{A}$ is irreducible). We get two distinct computations (r, c, d), (r, c', d') of $\mathcal{B}$ ending in r on a same tree $u = (\phi(e), t, t')$ for some tree t'. These two distinct computations of $\mathcal{B}$ also give two distinct computations of $\mathcal{A}$ on two trees s, s' of X with $s_\varepsilon = s'_\varepsilon = e$ and such that $\Phi(s) = \Phi(s')$. Since Φ is left closing with parameters $m' = 1$, we get $s = s'$ and thus $c = c'$.

Conversely, if S is accepted by an AFT automaton $\mathcal{A} = (V, A, \Delta)$, let X be the transition tree-shift accepted by $\mathcal{T} = (V, \Delta, \Delta')$, whose transitions are $(p, q, (p, q, a, r)) \to r$ for $(p, q, a, r) \in \Delta$. Let Φ be the one-block map from X onto S defined by $\phi(p, q, a, r) = a$. This map is right resolving since $\mathcal{A}$ is deterministic. It is left closing since $\mathcal{A}$ is left closing. It has a synchronizing block since $\mathcal{A}$ is synchronized. As a consequence, S is AFT. □

Corollary 1. *An irreducible sofic tree-shift is AFT if and only if its Fischer automaton is AFT.*

Proof. Let S be an irreducible sofic tree-shift. By Proposition 10, any irreducible sofic tree-shift is accepted by an irreducible deterministic synchronized automaton $\mathcal{F}$ equal to the Fischer automaton of S.

Let us assume that S is AFT. By Proposition 15, $\mathcal{F}$ is equal to the minimization of an AFT automaton $\mathcal{A}$. Let us show that $\mathcal{F}$ is left closing. If it is not left closing, then there is a tree $t \in S$ and two distinct computations of $\mathcal{F}$ on t ending in a same state. As a consequence, there are two distinct computations of $\mathcal{A}$ on t ending in two distinct states p, p' which have the same context.

Let z be a synchronizing pattern of $\mathcal{A}$ focusing to the state q. Since $\mathcal{A}$ is irreducible, is an hyperpath from p to q labeled by some pattern w such that z is a subtree of w rooted at ε. Let P be the complete prefix code defining this hyperpath. Thus there is a computation c of $\mathcal{A}$ on a tree s such that, for any node x in P, the tree rooted at x in s is t, $c_x = p$, and the pattern rooted at the root of s is z.

Since p and p' have the same context, there is also a computation c' of $\mathcal{A}$ on s such that, for some node x in P, one has $c'_x = p'$. Since z is synchronizing, $c_\varepsilon = c'_\varepsilon$. This gives two distinct computations of $\mathcal{A}$ on a same tree ending in the same state q, which contradicts the fact that $\mathcal{A}$ is left closing. This proves that the Fischer automaton of an AFT is left closing.

Conversely, if $\mathcal{F}$ is AFT, then X is AFT by Proposition 15. □

Corollary 2. *Let S be an irreducible sofic shift accepted by a deterministic tree automaton. It is decidable whether S is AFT.*

Proof. One can compute the Fischer automaton of the sofic tree-shift and check whether it is AFT as explained in Section 5. □

4.1. **Conjugacy invariants.**

In the case of shifts in dimension one, Krieger proved that the Fischer automaton is a topological conjugacy invariant of the underlying irreducible sofic shift [16] (see also [8]). In this section, we extend Krieger's theorem to sofic tree-shifts.

Lemma 1. *Let $\mathcal{A} = (V, A, \Delta)$ be the Fischer automaton of the irreducible sofic tree-shift $S_\mathcal{A}$ with transition tree shift $X_\mathcal{A}$. Let $\lambda_\mathcal{A} : X_\mathcal{A} \to S_\mathcal{A}$ the one-block map assigning to each transition its label. Let us assume that there is* *a block map Θ such that the following diagram commutes.*

$$\begin{tikzcd}
X_{\mathcal{A}} \arrow[r, "\Theta"] \arrow[d, "\lambda_{\mathcal{A}}"'] & X_{\mathcal{A}} \arrow[d, "\lambda_{\mathcal{A}}"] \\
S_{\mathcal{A}} \arrow[r, "id"] & S_{\mathcal{A}}
\end{tikzcd}$$

Then Θ is the identity map.

Proof. Let u be a synchronizing block of height n of $S_{\mathcal{A}}$. Let z be a block of height n of $X_{\mathcal{A}}$ such that $\lambda_{\mathcal{A}}(z) = u$. Let Z be the subset of $X_{\mathcal{A}}$ containing the trees t of $X_{\mathcal{A}}$ such that

- z is a subtree of t rooted at ε;
- for any node $x \in \Sigma^*$, there is a finite prefix code P depending on x such that, for any $\ell \in \Sigma^n$ and any $y \in P$, z is a subtree of t rooted at the position $x + \ell + y$.

Note that Z is not a subshift of $X_{\mathcal{A}}$. It is clear that $\lambda_{\mathcal{A}}$ is one-to-one on Z. Hence Θ is one-to-one on Z. Let us assume that Θ is an m-block map which is not the identity map on $X_{\mathcal{A}}$. Then there is a block v of height m such that $\theta(v) \neq v_\varepsilon$. Since $X_{\mathcal{A}}$ is irreducible, the block v is a subblock of some tree in Z, which is a contradiction. □

Proposition 16. *Let $\mathcal{A} = (V, A, \Delta)$ (resp. $\mathcal{B} = (V', B, \Delta')$) be the Fischer automaton of the irreducible sofic tree-shift $S_{\mathcal{A}}$ (resp. $S_{\mathcal{B}}$). Let $X_{\mathcal{A}}$ (resp. $X_{\mathcal{B}}$) be the transition tree-shift of $\mathcal{A}$ (resp. $\mathcal{B}$). If $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$ are conjugate, then $\mathcal{A}$ and $\mathcal{B}$ are symbolic conjugate. As a consequence $X_{\mathcal{A}}$ and $X_{\mathcal{B}}$ are conjugate.*

Proof. There is are one-block and onto map $\lambda_{\mathcal{A}} : X_{\mathcal{A}} \to S_{\mathcal{A}}$ (resp. $\lambda_{\mathcal{B}} : X_{\mathcal{B}} \to S_{\mathcal{B}}$). Let Φ be a conjugacy from $S_{\mathcal{A}}$ to $S_{\mathcal{B}}$. We assume that Φ is an m-block map and Φ^{-1} is an n-block map. Since $\mathcal{A}$ is deterministic, the map λ_A is a one-block right resolving map. Let $X_{\mathcal{A}}^{(m+n-1)}$ (resp. $S_{\mathcal{A}}^{(m+n-1)}$) be the higher block presentation of $X_{\mathcal{A}}$ (resp. $\mathcal{S}_{\mathcal{A}}$) of order $(m+n-1)$, and $\tau_{\mathcal{A}}$ (resp. $\mu_{\mathcal{A}}$) be the $m+n-1$-block conjugacy from $X_{\mathcal{A}}$ to $X_{\mathcal{A}}^{(m+n-1)}$ (resp. from $S_{\mathcal{A}}$ to $S_{\mathcal{A}}^{(m+n-1)}$) assigning to each tree t the tree t' whose nodes are blocks of height $m+n-1$ of t. Let $\lambda'_{\mathcal{A}}$ be the natural one-block map such that $\tau_{\mathcal{A}} \circ \lambda'_{\mathcal{A}} = \lambda_{\mathcal{A}} \circ \mu_{\mathcal{A}}$.

Let $\Phi' = \mu_{\mathcal{A}}^{(-1)} \circ \Phi$. Since Φ is an m-block map, the map $\Psi = \Phi' \circ \lambda'_{\mathcal{A}}$ is a one-block map from $X_{\mathcal{A}}^{(m+n-1)}$ to $S_{\mathcal{B}}$. Since $\Phi^{(-1)}$ is an n-block map, Ψ is right resolving.

Let $\mathcal{C} = (\mathcal{B}(X_{\mathcal{A}}^{(m+n-1)}), B, \Theta)$ be the automaton on the alphabet B whose transitions are $(u, v), b \to w$, where $(w_\varepsilon, u, v) \in \mathcal{B}(X_{\mathcal{A}}^{(m+n)})$, w is the sub-block of size $m+n-1$ rooted at ε of (w_ε, u, v), and $\psi(w) = b$. Since Ψ is right resolving, $\mathcal{C}$ is a deterministic tree automaton accepting $S_{\mathcal{B}}$.

We consider the partitioning of the states of $\mathcal{C}$ into classes of states having the same context. We denote by $[u]$ the class of the state u. We define a one-block and onto map Γ from $X_{\mathcal{A}}^{(m+n-1)}$ to $X_{\mathcal{B}}$ by $\gamma((u,v),b \to w) = ([u],[v]),b \to [w]$. We have $\Gamma \circ \lambda_{\mathcal{B}} = \Psi$. Hence the block map $\theta_1 = \tau_{\mathcal{A}} \circ \Gamma$ is onto from $X_{\mathcal{A}}$ to $X_{\mathcal{B}}$. We get the following commutative diagram.

$$\begin{tikzcd}
X_{\mathcal{A}} \arrow[r, "\tau_{\mathcal{A}}"] \arrow[d, "\lambda_{\mathcal{A}}"'] & X_{\mathcal{A}}^{(m+n-1)} \arrow[r, "\Gamma"] \arrow[d, "\lambda'_{\mathcal{A}}"'] \arrow[rd, "\Psi"] & X_{\mathcal{B}} \arrow[d, "\lambda_{\mathcal{B}}"] \\
S_{\mathcal{A}} \arrow[r, "\mu_{\mathcal{A}}"] & S_{\mathcal{A}}^{(m+n-1)} \arrow[r, "\Phi'"] & S_{\mathcal{B}}
\end{tikzcd}$$

By exchanging the roles played by $\mathcal{A}$ and $\mathcal{B}$, we also get a block map $\theta_2 : X_{\mathcal{B}} \to X_{\mathcal{A}}$ such that following diagram commutes.

$$\begin{tikzcd}
X_{\mathcal{A}} \arrow[r, "\theta_1"] \arrow[d, "\lambda_{\mathcal{A}}"'] & X_{\mathcal{B}} \arrow[r, "\theta_2"] \arrow[d, "\lambda_{\mathcal{B}}"'] & X_{\mathcal{A}} \arrow[d, "\lambda_{\mathcal{A}}"] \\
S_{\mathcal{A}} \arrow[r, "\Phi' \circ \mu_{\mathcal{A}}"] & S_{\mathcal{B}} \arrow[r, "(\Phi')^{-1} \circ \mu_{\mathcal{B}}"] & S_{\mathcal{A}}
\end{tikzcd}$$

By Lemma 1, $\theta_2 \circ \theta_1$ is the identity map. As a consequence the maps θ_1 and θ_2 are conjugacies. □

5. The algorithmic issue

In this section, we design algorithms to check whether a tree automaton is synchronized, and whether it is left closing. We also describe an algorithm to compute the Fischer automaton of an irreducible sofic tree-shift given by a tree automaton that accepts it.

5.1. Computation of the Fischer automaton.

Let $\mathcal{A} = (V, A, \Delta)$ be a tree automaton. We define the deterministic tree automaton $\mathcal{D}(\mathcal{A})$, called the *determinized automaton* of $\mathcal{A}$, as the accessible part from the state V of the tree automaton $(\mathfrak{P}(V), A, \delta')$[6], where, for $P, Q \in \mathfrak{P}(V)$, $\delta'(P,Q,a) = \{r \mid (p,q,a,r) \in \Delta, p \in P, q \in Q\}$ if this set is nonempty, and is not defined otherwise.

Proposition 17. *It can be checked in polynomial time whether a tree automaton is irreducible.*

Proof. Let $\mathcal{A} = (V, A, \Delta)$ be a tree automaton. For any state p in V, and any positive integer n, we define the sets

$$P_0 = \{p\},$$
$$P_n = P_{n-1} \cup \{q \in V \mid \in A, r, s \in P_{n-1} \text{ such that } (r,s) \xrightarrow{a} q \in \Delta\}.$$

The sequence of subsets $(P_n)_{n \geq 0}$ is increasing for the inclusion. Thus there is an integer n_0 such that $P_n = P_{n_0}$ for any $n \geq n_0$. By construction, there

[6] $\mathfrak{P}(V)$ denotes the set of parts of V.

is an hyperpath from p to q if and only if $q \in P_{n_0}$. Thus $\mathcal{A}$ is irreducible if and only if $P_{n_0} = V$ for any state p. The set P_{n_0} is computed with at most $|V|$ steps, the time complexity of each step being at most the number of transitions, hence the global polynomial complexity. $\square$

Proposition 18. *It is decidable whether a tree automaton is synchronized.*

Proof. The automaton $\mathcal{A}$ is synchronized if and only if $\mathcal{D}(\mathcal{A})$ contains a singleton state. The time and space complexity for computing $\mathcal{D}(\mathcal{A})$ is exponential in the number of states of $\mathcal{A}$. $\square$

Proposition 19. *Let $\mathcal{A} = (V, A, \delta)$ be a tree automaton accepting an irreducible sofic tree-shift S. The Fischer automaton of S is computable from $\mathcal{A}$.*

Proof. We first compute the determinized automaton $\mathcal{D}(\mathcal{A})$ of $\mathcal{A}$. Let R be a state of $\mathcal{D}(\mathcal{A})$ which is minimal for the inclusion. Let u the label of a hyperpath from V to R. Then u is a synchronizing pattern of $\mathcal{D}(\mathcal{A})$. Indeed, any finite computation of $\mathcal{D}(\mathcal{A})$ on u ends in R by minimality of R. We now keep in $\mathcal{D}(\mathcal{A})$ only the states accessible from the state R and get an irreducible and synchronized automaton accepting S. Its reduction gives the Fischer automaton of S by Proposition 10. It is computed in polynomial time in the size of $\mathcal{D}(A)$ (see for instance [9]). $\square$

We now describe algorithms to check whether an irreducible deterministic automaton is left closing.

5.2. **The pair graph of a tree automaton.**

Given a tree automaton $\mathcal{A} = (V, A, \Delta)$, we define the *square automaton* of $\mathcal{A}$, denoted by $\mathcal{A} \times \mathcal{A} = (V \times V, A, \Delta')$, as the deterministic automaton whose transitions are $(p, p'), (q, q'), a \to (r, r')$ if and only if $(p, q), a \to r$ and $(p', q'), a \to r'$ are transitions of $\mathcal{A}$. A *diagonal state* of $\mathcal{A} \times \mathcal{A}$ is a state (p, p) for some $p \in V$.

Square automata of finite words (see for instance [22, p. 647]) are used to check properties of pairs of paths. We extend this notion, together with a notion a pair graph, to trees, to check properties of pairs of computations. Seidl [23] used branch automata to check the degree of ambiguity of finite tree automata.

Proposition 20. *A tree automaton is not left closing if and only if there is a computation in the square automaton ending in a diagonal state and containing a non diagonal one.*

Proof. By definition of $\mathcal{A} \times \mathcal{A}$, the existence of a computation in $\mathcal{A} \times \mathcal{A}$ ending in a state (p, p) and containing a state (r, s) with $r \neq s$ is equivalent to the existence of two distinct computations of $\mathcal{A}$ on a same tree. $\square$

In order to check the above property, we build *the pair graph* $G_{\mathcal{A}} = (V_G, E_G)$ of $\mathcal{A}$, where $V_G \subseteq (V^2 \times V^2) \cup V^2$ is the set of vertices, $E_G \subseteq V_G \times \{0, 1\} \times A \times V_G$ is the set of edges labeled by 0 or 1 and a letter from A. For more convenience, an edge labeled by 1 is noted by a plain arrow $\longrightarrow$ and is called a plain edge, and an edge labeled by 0 is noted by a dashed arrow

$\dashrightarrow$ and is called a dashed edge. For each pair of transitions $(p,q), a \to r$ and $(p',q'), a \to r'$ of $\mathcal{A}$,

$$((p,p'),(q,q')) \xrightarrow{0,a} ((r,r'),(s,s')),$$
$$((p,p'),(q,q')) \xrightarrow{1,a} ((s,s'),(r,r')),$$
$$((p,p'),(q,q')) \overset{0,a}{\dashrightarrow} (r,r'),$$
$$((p,p'),(q,q')) \overset{1,a}{\dashrightarrow} (r,r'),$$

are edges of $G_{\mathcal{A}}$, for each pair (s,s').

A vertex of $G_{\mathcal{A}}$ is *useful* if it has at least one incoming plain edge and at least one incoming dashed edge. We keep the *essential part* of the pair graph obtained by discarding vertices which are not useful together with their incoming and outgoing edges. A vertex $((p,q),(r,s))$ of G_A is called *non diagonal* if either $p \neq q$ or $r \neq s$.

When $\mathcal{A}$ is a deterministic automaton, we have $|V_G| = O(|V|^4)$ and $|E_G| = O(|V|^6)$. The essential part of the pair graph can be computed in time $O(|V_G| + |E_G|)$ as described in [3].

It is easy to verify that a vertex $((p,p'),(q,q'))$ is a vertex of the (essential part of the) pair graph if and only if there are two computations of $\mathcal{A}$ on a tree s one ending in p, the other one in p', and there are two computations of $\mathcal{A}$ on a tree t one ending in q, the other one in q'.

Note also that there is an edge $((p,p'),(q,q')) \xrightarrow{0} ((r,r'),(s,s'))$ in the pair graph if and only if there is a letter a and transitions $(p,q) \xrightarrow{a} r$ and $(p',q') \xrightarrow{a} r'$ in $\mathcal{A}$ (or transitions $(p,q) \xrightarrow{a} s$ and $(p',q') \xrightarrow{a} s'$ in $\mathcal{A}$). There is an edge $((p,p'),(q,q')) \overset{0}{\dashrightarrow} (r,r')$ in the pair graph if and only if there is a letter a and transitions $(p,q) \xrightarrow{a} r$ and $(p',q') \xrightarrow{a} r'$ in $\mathcal{A}$.

Proposition 21. *A tree automaton is not left closing if and only its there is a path in its pair graph starting at non diagonal vertex and ending in the vertex (p,p).*

Proof. Let $\mathcal{A}$ be a tree automaton and $G_{\mathcal{A}}$ its pair graph. Suppose that there is a path in $G_{\mathcal{A}}$ from vertex $((p,p'),(q,q'))$ with $p \neq p'$ or $q \neq q'$ to a vertex (r,r). Without loss of generality, we can assume that this path is an edge. Then there are two computations c, c' of $\mathcal{A}$ on a same tree t such that c ends in p and c' ends in p', and there are two computations d, d' of $\mathcal{A}$ on a same tree s such that d ends in q and d' ends in q'. There are also transitions $(p,q) \xrightarrow{a} r$ and $(p',q') \xrightarrow{a} r'$. This implies that there are two distinct computations of $\mathcal{A}$ on a same tree ending in r. Thus $\mathcal{A}$ is not left closing.

Conversely, if $\mathcal{A}$ is not left closing, there are two distinct computations c, c' of $\mathcal{A}$ on a same tree t ending in a same state r. Let x be a node of the tree t such that $c_x \neq c'_x$. Then there is path in $G_{\mathcal{A}}$ labeled by (x,w) going from some vertex $((c_x,c'_x),(s,s'))$ or some vertex $((s,s'),(c_x,c'_x))$ (depending on the last letter of x) to (r, r), where w is is the label of the path of the tree t from the root to the node x. Hence the conclusion. □

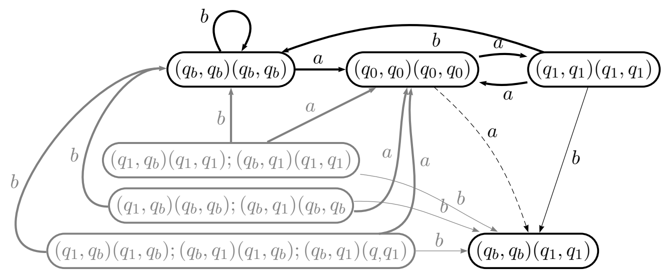

$(q_b, q_b)(q_b, q_b)$
$(q_0, q_0)(q_0, q_0)$
$(q_1, q_1)(q_1, q_1)$
$(q_1, q_b)(q_1, q_1); (q_b, q_1)(q_1, q_1)$
$(q_1, q_b)(q_b, q_b); (q_b, q_1)(q_b, q_b)$
$(q_1, q_b)(q_1, q_b); (q_b, q_1)(q_1, q_b); (q_b, q_1)(q, q_1)$
$(q_b, q_b)(q_1, q_1)$
a
b

Figure 4. The pair graph for the tree automaton of Example 1. A thick edge represents a plain edge and a dashed edge with the same label. The non useful edges and vertices are drawn in gray. Each vertex (p, q) is identified with the vertex $(p, q)(p, q)$. For the test, the pair $((p, q), (r, s))$ may not be represented if $((r, s), (p, q))$ does. The tree-shift S accepted by $\mathcal{A}$ satisfies the property of Proposition 21 since there is no path from a diagonal state to a non diagonal one. Then $\mathcal{A}$ is a left closing automaton and as a consequence the tree-shift S is AFT.

If $\mathcal{A}$ is a deterministic tree automaton, the number of vertices of $G_{\mathcal{A}}$ is at most $O(|V|^4)$ and its number of edges of $G_{\mathcal{A}}$ is at most $O(|V|^6)$. The property of Proposition 21 can be checked in a linear time in the size of $G_{\mathcal{A}}$. As a consequence, it can be checked in polynomial time whether the Fischer automaton of an irreducible sofic tree-shift is AFT. Note that Seidl's check of the finite degree of ambiguity of tree automata in [23] has a similar complexity (the cube of the size of the transitions of the tree automaton). The pair graph for the tree automaton $\mathcal{A}$ of Example 1 is given in Figure 4.

Conclusion

In this article, we have shown that tree-shifts differ from one-sided shifts of infinite sequences at least concerning the following property: there may be more than one reduced deterministic irreducible tree automata accepting the same irreducible sofic tree-shift. The reason is that such automata do not always have a synchronizing block. For irreducible sofic tree-shifts, the Fischer automaton remedy for this lack and allows us to define the class of almost of finite type tree-shifts.

References

[1] N. Aubrun and M.-P. Béal. Decidability of conjugacy of tree-shifts of finite type. In *ICALP '09: Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, pages 132–143, Berlin, Heidelberg, 2009. Springer-Verlag.

[2] N. Aubrun and M.-P. Béal. Sofic and almost of finite type tree-shifts. In *CSR*, volume 6072 of *Lecture Notes in Computer Science*, pages 12–24. Springer, 2010.

[3] N. Aubrun and M.-P. Béal. Tree-shifts of finite type. Extended version of ICALP '09, 2011.

[4] T. Bates, S. Eilers, and D. Pask. Reducibility of covers of AFT shifts. *Israel Journal of Mathematics*, 2010. to appear.

[5] M.-P. Béal, J. Berstel, S. Eilers, and D. Perrin. Symbolic dynamics. *CoRR*, abs/1006.1265, 2010.

[6] M.-P. Béal, F. Fiorenzi, and D. Perrin. A hierarchy of shift equivalent sofic shifts. *Theor. Comput. Sci.*, 345(2-3):190–205, 2005.

[7] M.-P. Béal and D. Perrin. Symbolic dynamics and finite automata. In *Handbook of formal languages, Vol. 2*, pages 463–505. Springer, Berlin, 1997.

[8] M. Boyle, B. Kitchens, and B. Marcus. A note on minimal covers for sofic systems. *Proc. Amer. Math. Soc.*, 95(3):403–411, 1985.

[9] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. http://www.grappa.univ-lille3.fr/tata, 2007. release October, 12th 2007.

[10] E. Coven, A. Johnson, N. Jonoska, and K. Madden. The symbolic dynamics of multidimensional tiling systems. *Ergodic Theory and Dynamical Systems*, 23(02):447–460, 2003.

[11] M. Fujiwara and M. Osikawa. Sofic systems and flow equivalence. *Math. Rep. Kyushu Univ.*, 16(1):17–27, 1987.

[12] G. Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Theory of Computing Systems*, 3(4):320–375, 1969.

[13] A. S. A. Johnson and K. M. Madden. The decomposition theorem for two-dimensional shifts of finite type. *Proc. Amer. Math. Soc.*, 127(5):1533–1543, 1999.

[14] N. Jonoska and B. Marcus. Minimal presentations for irreducible sofic shifts. *IEEE Trans. Inform. Theory*, 40(6):1818–1825, 1994.

[15] B. P. Kitchens. *Symbolic dynamics*. Universitext. Springer-Verlag, Berlin, 1998. One-sided, two-sided and countable state Markov shifts.

[16] W. Krieger. On sofic systems. I. *Israel J. Math.*, 48(4):305–330, 1984.

[17] D. Lind and K. Schmidt. Symbolic and algebraic dynamical systems. In *Handbook of dynamical systems, Vol. 1A*, pages 765–812. North-Holland, Amsterdam, 2002.

[18] D. A. Lind and B. H. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1995.

[19] B. H. Marcus. Sofic systems and encoding data. *IEEE Transactions on Information Theory*, 31(3):366–377, 1985.

[20] M. Nasu. *Textile Systems for Endomorphisms and Automorphisms of the Shift*. American Mathematical Society, 1995.

[21] D. Perrin and J. Pin. *Infinite words*. Elsevier Boston, 2004.

[22] J. Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

[23] H. Seidl. On the finite degree of ambiguity of finite tree automata. In *Fundamentals of computation theory (Szeged, 1989)*, volume 380 of *Lecture Notes in Comput. Sci.*, pages 395–404. Springer, New York, 1989.

[24] W. Thomas. Automata on infinite objects. In *Handbook of theoretical computer science, Vol. B*, pages 133–191. Elsevier, Amsterdam, 1990.

[25] R. F. Williams. Classification of subshifts of finite type. In *Recent advances in topological dynamics (Proc. Conf. Topological Dynamics, Yale Univ., New Haven, Conn.*, pages 281–285. Lecture Notes in Math., Vol. 318. Springer, Berlin, 1973.

[26] S. Williams. Covers of non-almost-finite type sofic systems. *Proc. Amer. Math. Soc.*, 104(1):245–252, 1988.

Université Paris-Est, Laboratoire d'informatique Gaspard-Monge, CNRS