Codes, unambiguous automata and sofic systems

Marie-Pierre Béal * Dominique Perrin *

Institut Gaspard-Monge, Université de Marne-la-Vallée, 77454 Marne-la-Vallée Cedex 2, France.

Abstract

We study the relationship between codes and unambiguous automata inside a sofic system. We show that a recognizable set is a code in a sofic system if and only if a particular automaton associated to the set and the shift is unambiguous. We discuss an example of a finite complete code in a sofic system in connection with the factorization conjecture.

1 Introduction

This is the third of our papers on codes in sofic shifts. In the first one [1], we have developed the point of view of measures and polynomials in the spirit of the Kraft-McMillan inequality. In the second one [2], we have discussed the notions of complete and maximal codes in sofic shifts and their relationship. This generalization of the theory of (variable length) codes extends previous works of Reutenauer [3], Restivo [4] and Ashley *et al.* [5]. It is also related with recent work by Dalai and Leonardi [6], who study a close problem. They use Markov chains on the source symbols instead of constraints on the channel as we do. Codes with constraints on the source are also studied by Güney Gönenç in [7].

In this paper, we show how the use of unambiguous automata can be adapted to the framework of codes in sofic systems. This gives us a particular a method

* Corresponding author
URLs: `www.univ-mlv.fr/~beal` (Marie-Pierre Béal),
`www.univ-mlv.fr/~perrin` (Dominique Perrin).

for checking whether a regular set is a code in a sofic shift. Another method generalizing the Sardinas-Patterson algorithm is also described.

We use these notions to discuss an interesting example of a complete code in a sofic system, in particular in connection with the factorization conjecture of Schützenberger. We show that its generalization to code in sofic shifts is not true.

We would like to thank Christophe Reutenauer and the anonymous referees for useful comments on an earlier version.

2 Codes in sofic systems

We begin with some definitions from symbolic dynamics. For a general reference, we refer to [8]. A *sofic shift* S is the set of bi-infinite sequences of symbols labelling paths in a finite automaton $\mathcal{A} = (Q, E)$, where Q is the set of states and E the set of edges. We say that $\mathcal{A}$ recognizes S. The set of factors of S, denoted by $\mathrm{Fact}(S)$, is the set of blocks appearing in the elements of S. An *edge shift* is the set of bi-infinite paths in a finite graph. A *shift of finite type* is the set of bi-finite sequences of symbols avoiding a finite set of words. An edge shift is a particular case of a shift of finite type, which is a particular case of a sofic shift. The *full shift* on a finite alphabet A is the set of all bi-infinite sequences of symbols in A.

A sofic shift is *irreducible* if it is recognized by an automaton with a strongly connected graph. There is a unique minimal deterministic automaton recognizing a given irreducible sofic shift. It is called the *Fischer cover* of the shift.

Example 1 *Let S be the irreducible sofic shift whose Fischer cover is represented in Figure 1. This shift is called the* even system*. It is a sofic shift which is not of finite type.*

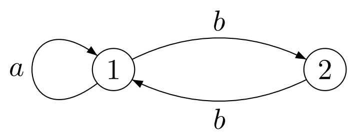

Fig. 1. The even system.

Let S be a sofic shift over the alphabet A. We denote by $\mathbb{Z}[S]$ the algebra of linear combinations with coefficients in $\mathbb{Z}$ of elements of $\mathrm{Fact}(S)$ using the product

$$u \cdot v = \begin{cases} uv & \text{if } uv \in \mathrm{Fact}(S) \\ 0 & \text{otherwise.} \end{cases}$$

For $U \in \mathbb{Z}[S]$ and $w \in \mathrm{Fact}(S)$, we denote by (U, w) the coefficient of w in U. We denote by $\mathbb{N}[S]$ the set of linear combinations with coefficients in $\mathbb{N}$ of elements of $\mathrm{Fact}(S)$. We also use the corresponding large algebra which consists of the infinite linear combinations of elements of $\mathrm{Fact}(S)$. We often identify a subset of $\mathrm{Fact}(S)$ with the sum of all its elements. We have in particular the equation in the large algebra.

$$(1 - A)\,\mathrm{Fact}(S) = \mathrm{Fact}(S)(1 - A) = 1.$$

We may as well write $\mathrm{Fact}(S) = A^*$, provided the star operation is understood to refer to the product defined above.

A set X of elements of $\mathrm{Fact}(S)$ is called an *S-code* if any element of $\mathrm{Fact}(S)$ has at most one decomposition in code words. We also say that X is a code in S. Thus a code in the usual sense is a code in the full shift. A set of words X is *S-complete* (or complete in S) if any element of $\mathrm{Fact}(S)$ occurs within some concatenation of elements of X.

It is known that a maximal S-code is S-complete (see [2]). The converse is not true since for example $X = \{ab\}$ is complete in the shift of finite type avoiding aa and bb. However, it is not maximal since it is included in $\{ab, ba\}$.

If X is an S-code, we have in the large algebra of $\mathrm{Fact}(S)$

$$(1 - X)X^* = X^*(1 - X) = 1,$$

where the star operation is again understood to refer to the product defined above.

A *prefix code* in a sofic shift S is a set X of elements of $\mathrm{Fact}(S)$, such that no proper prefix of a word of X belongs to X. A set of words X is *right complete* in S if any element of $\mathrm{Fact}(S)$ is a prefix of a word in X^*. A prefix code which is maximal is right-complete and conversely (the proof is the same as in the case of the full shift). It is not true however that a prefix code in a sofic shift which is complete is also right-complete as shown by the example of $X = \{ab\}$.

Let X be a maximal prefix code in a sofic shift S, and let P be the set of proper prefixes of words of X. We have in the large algebra of $\mathrm{Fact}(S)$ the equations

$$\begin{aligned} X - 1 &= P(A - 1), \\ \mathrm{Fact}(S) &= X^* P. \end{aligned}$$

For example, if S is the subshift of finite type avoiding aa and bb, and if $X = \{ab, ba\}$, we have in the algebra of $\mathrm{Fact}(S)$

$$X - 1 = (A + 1)(A - 1).$$

Interestingly, for $X = \{ab\}$, we do not have such an identity because it is not maximal. But we have the 3-factors product

$$X - 1 = (a + 1)(a + b - 1)(b + 1).$$

3 Unambiguous automata

There are at least two methods which can be used to check whether a set X is a code, the classical Sardinas-Patterson algorithm and the test of unambiguity for automata. We extend here these methods to codes in sofic systems. The extension of the Sardinas-Patterson algorithm to codes in edge shifts has already been described by C. Reutenauer in [3].

We begin with the extension of the Sardinas-Patterson algorithm. Let S be a sofic system and let X be a subset of $\mathrm{Fact}(S)$. Let $\mathcal{A} = (Q, E, i, T)$ be the minimal deterministic automaton of $\mathrm{Fact}(S)$. We consider the following notion on subsets of $A^* \times Q$. For $U, V \subseteq A^* \times Q$, let

$$U^{-1}V = \{(w, q) \mid (uw, q) \in V \text{ and } p \xrightarrow{w} q \text{ for some } (u, p) \in U\}.$$

We denote

$$Y = \{(x, p) \mid x \in X, p \in Q \text{ and } i \xrightarrow{x} p\}.$$

We then define a sequence $(U_n)_{n\geq 0}$ of subsets of $A^* \times Q$ by the usual formulas.

$$\begin{aligned} U_0 &= Y^{-1}Y - \{(\varepsilon, p), p \in Q\}, \\ U_{n+1} &= Y^{-1}U_n \cup U_n^{-1}Y. \end{aligned}$$

It is easy to verify that there is a finite number of possible sets $(U_n)_{n\geq 0}$ when X is finite (it is also true when X is regular). The proof of the following result is similar to the classical one.

Proposition 1 *The set X is a code if and only if none of the sets U_n contains a pair (ε, p) with $p \in T$.*

Example 2 *We consider the even shift S of Figure 1. Its set of finite factors is recognized by the automaton of Figure 2.*

Let $X = \{a, ab, ba\}$. We have

$$\begin{aligned} Y &= \{(a, 1), (ab, 2), (ba, 1)\}, \\ U_0 &= \{(b, 2)\}, \\ U_1 &= \emptyset. \end{aligned}$$

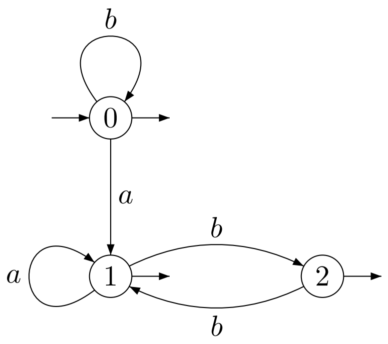

Fig. 2. A deterministic automaton recognizing $\mathrm{Fact}(S)$.

Thus X is an S-code although it is not a code in the full shift. We will have more to say about this example later (Section 4).

We now come to the method using automata. It is well known that for a set X of words, one can construct a non-deterministic automaton such that X^* is the stabilizer of one state and that X is a code if and only if this automaton is unambiguous (see [9]). Such an automaton also allows one to check easily whether the code is complete. We extend below this method to codes in sofic systems.

Recall that a nondeterministic automaton is *unambiguous* if whenever there are two paths of the form

$$p \xrightarrow{u} r \xrightarrow{v} q$$
$$p \xrightarrow{u} s \xrightarrow{v} q,$$

we have $r = s$. In other terms there is a unique path with a given origin, end, and label.

Let S be an irreducible sofic system on the alphabet A and let $X \subset \mathrm{Fact}(S)$ be a set of words. We suppose in a first step that X is a finite set. As we shall see later, the construction below also works for a regular set X.

Let $\mathcal{A} = (Q, E)$ be the minimal deterministic automaton recognizing S (see [10] for an exposition of the links between automata and symbolic systems). We build an automaton $\mathcal{B}$ as follows. The set of states is formed by the set Q plus $|x| - 1$ new states for each path in $\mathcal{A}$ of the form $p \xrightarrow{x} q$ for $x \in X$. It is easy to verify that the automaton $\mathcal{B}$ recognizes $X^* \cap \mathrm{Fact}(S)$ with Q as set of initial and terminal states.

When X is regular, we proceed as follows. Let $\mathcal{C} = (P, F, i, t)$ be an unambiguous normalized automaton recognizing X. A classical construction (see [9, p. 185]) allows one to build an automaton $\mathcal{C}^* = (P \cup \omega, \omega, \omega)$, where ω is a new state obtained by merging the states i and t, such that the number of paths from ω to ω labeled by w is the number of factorizations of w in words of X. The automaton $\mathcal{B}$ is now chosen as $\mathcal{A} \times \mathcal{C}^*$. The previous construction corresponds to the choice of the flower automaton of the set X for $\mathcal{C}^*$.

The following statement shows in particular that one can use the automaton $\mathcal{B}$ to verify whether X is an S-code.

Proposition 2 *The set X is an S-code if and only if the automaton $\mathcal{B}$ is unambiguous.*

PROOF. For any two states p, q in Q, let L_{pq} be the set of words labels of paths from p to q in $\mathcal{A}$. Each state of $\mathcal{B}$ is accessible and co-accessible from a state in $Q \times \omega$. Hence $\mathcal{B}$ is unambiguous if and only if, for any pair p, q of states of Q and any word w, there is at most one path in $\mathcal{B}$ labeled by w from (p, ω) to (q, ω). By construction, the number of paths from (p, ω) to (q, ω) labeled by w is the number of factorizations of w in words of X. Thus if X is an S-code, $\mathcal{B}$ is unambiguous. Conversely, if X is not an S-code, there is a word $w \in L_{pq}$ for some states $p, q \in Q$ which has at least two factorizations in words of X, which implies that $\mathcal{B}$ is ambiguous. $\square$

Example 3 *For the set $X = \{a, ab, ba\}$ of Example 2 in the even shift, the automaton $\mathcal{B}$ is represented in Figure 3. It is an unambiguous automaton. A lookahead of one symbol suffices to resolve the nondeterminism in state* 1. *This gives a second proof that X is an S-code.*

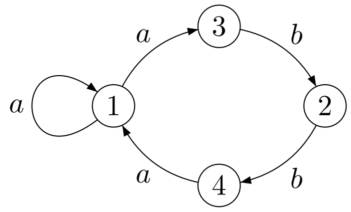

Fig. 3. The unambiguous automaton $\mathcal{B}$.

Moreover, the automaton $\mathcal{B}$ can be used to verify whether X is S-complete. Indeed

Proposition 3 *The set X is S-complete if and only if $\mathcal{B}$ recognizes S.*

In practice, if S is irreducible, since $\mathcal{B}$ recognizes a subshift T of S, it is enough to verify that the entropy of T is equal to the entropy of S. This can be done in polynomial time (see [11]).

4 A complete code in the even system

In this section, we develop in some detail an example of a complete code in a sofic system.

We consider again the set of words $X = \{a, ab, ba\}$. It is the simplest example of a set which is not a code. In fact, one has the two factorizations $(ab)(a) = (a)(ba)$. However, one has the following statement

Proposition 4 *The set X is a code in any sofic system such that the block aba is forbidden.*

PROOF. The simplest way to see this is as follows. An ambiguous factorization should begin with $(a)(b\cdots = (ab)\cdots$. Since aba is forbidden, the prefix ab should be followed by a b as $(a)(bb\cdots = (ab)(b\cdots$ which is clearly impossible. $\square$

For example, the set X is a code in the system S_{aba} which is represented on Figure 4, which is the subshift of finite type on the alphabet $A = \{a, b\}$ defined by the unique forbidden block aba.

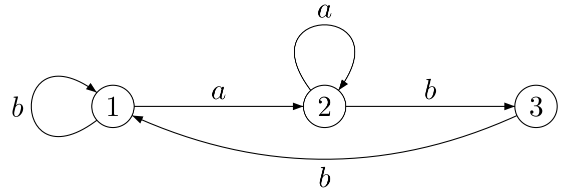

Fig. 4. The system S_{aba}.

We will verify the following statement.

Proposition 5 *The code X cannot be finitely completed in the system S_{aba}.*

PROOF. Let us assume the contrary and let Y be a finite complete code in S_{aba} containing X. Let $n \geq 1$ be such that $b^n \in Y$. Then $(a)(b^n)(ba) = (ab)(b^n)a$, a contradiction. $\square$

This contrasts with the situation for codes (in the full shift) for which the simplest example of a code without any finite completion (*i.e.* the set $\{a^5, b, ab, ba^2\}$) relies on counting modulo some integer (see [12] or [9, p. 64]).

The set X is also a code in the even system represented on Figure 1. Let us now consider the following set containing X.

$$Y = \{a, ab, ba, bab, bbbb\}.$$

We are going to verify that, in contrast with the previous proposition, the following holds.

Proposition 6 *The set Y is a complete code in the even system.*

PROOF. The fact that Y is a code follows from Proposition 1. To see that it is complete, we compute the automaton recognizing Y^* as indicated by the method of Section 3. Up to the merge of some states, we obtain the automaton shown on Figure 5. It recognizes Y^* with 1 and 3 as initial and terminal states.

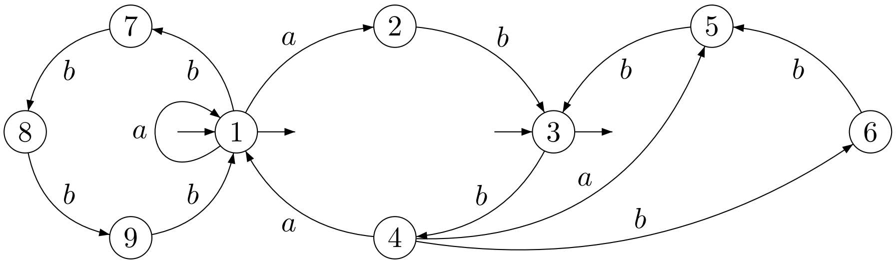

Fig. 5. An automaton recognizing Y^*.

A part of the subset construction applied to this automaton and represented on Figure 6 constitutes an automaton with five states $\{1,2\}$, $\{3,7\}$, $\{4,8\}$, $\{1,5\}$ and $\{6,9\}$ recognizing the even system. This can be seen by minimizing the deterministic automaton with these five states, which gives the Fischer cover of the even system. Thus the code is complete in this system. $\square$

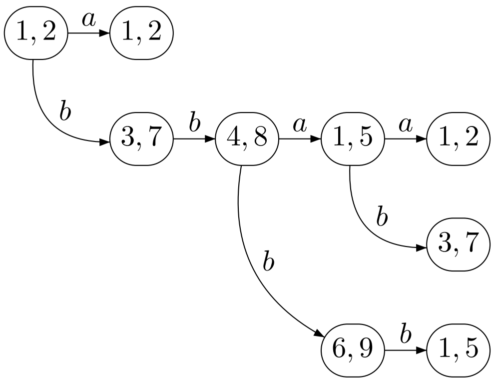

Fig. 6. The state diagram.

We now consider the polynomial of the code Y. This is by definition the determinant of the matrix $I - M(Y)$ (with entries in $\mathbb{Z}[A]$), where $M(Y)$ is the matrix associated with the action of the words of the code on the minimal automaton of the shift. This action is represented on Figure 7.

The matrix $M(Y)$ is

$$M(Y) = \begin{bmatrix} a + b^4 & ab \\ ba & bab + b^4 \end{bmatrix}.$$

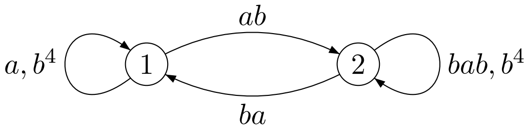

Fig. 7. The action of Y on the even system.

Thus the polynomial of the code is

$$\begin{aligned} p(Y) &= 1 - a - 2b^4 - a^2b^2 - ab^2 + (a + b^4)(ab^2 + b^4) \\ &= 1 - a - ab^2 - 2b^4 + ab^4 + ab^6 + b^8. \end{aligned}$$

In accordance with the main result of [2], the polynomial $p(Y)$ is divisible by $p(A) = 1 - a - b^2$. Indeed, we have

$$p(Y) = (1 + b^2)(1 - a - b^2)(1 - b^4).$$

It is interesting to remark that this factorization can be lifted to a non-commutative one. Indeed, one has *in non-commuting variables*

$$\begin{bmatrix} 1 - a - b^4 & -ab \\ -ba & 1 - bab - b^4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ b & 1 + b^2 \end{bmatrix} \begin{bmatrix} 1 - a & -b \\ -b & 1 \end{bmatrix} \begin{bmatrix} 1 & b \\ b^3 & 1 \end{bmatrix}$$

Thus, we have obtained the existence of matrices P, Q with elements in the subsets of A^* such that

$$I - M(Y) = P(I - M(A))Q. \tag{1}$$

We will have more to say on this equation in the next section.

5 A complete code in an edge shift

We finally consider what happens if one replaces the even system by the subshift of finite type S represented on Figure 8 and consisting in giving distinct names to the edges of the automaton of the graph of Figure 1. This is actually the edge shift of the graph of Figure 1.

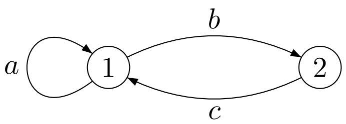

Fig. 8. A subshift of finite type.

We replace Y by the set

$$Z = \{a, ab, ca, cab, bcbc, cbcb\}$$

obtained by renaming the paths labeled by the words of Y in the graph of the automaton recognizing the even system. The set Z is again a complete code in S. The matrix $M(Z)$ is

$$M(Z) = \begin{bmatrix} a + bcbc & ab \\ ca & cab + cbcb \end{bmatrix}$$

and we have the factorization

$$\begin{bmatrix} 1 - a - bcbc & -ab \\ -ca & 1 - cab - cbcb \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ c & 1 + cb \end{bmatrix} \begin{bmatrix} 1 - a & -b \\ -c & 1 \end{bmatrix} \begin{bmatrix} 1 & b \\ cbc & 1 \end{bmatrix}$$

This factorization has the same form $I - M(Z) = P(I - M(A))Q$ as the factorization (1) but this time, the matrices P, Q are the matrices of the action of sets U, V with $U = \{1, c, cb\}$, $V = \{1, b, cbc\}$. We can even write simply

$$1 - Z = U(1 - A)V, \tag{2}$$

provided the expressions on both sides are computed in the algebra $\mathbb{Z}[S]$.

A finite subset Z of $\mathrm{Fact}(S)$ such that there exists two polynomials $U, V \in \mathbb{Z}[S]$ (resp. two sets $U, V \subseteq \mathrm{Fact}(S)$) satisfying Equality (2) is called $\mathbb{Z}$-*factorizing* (resp. $\mathbb{N}$-*factorizing*).

An $\mathbb{N}$-factorizing set is an S-complete code. Indeed, Equation (2) is equivalent to $\mathrm{Fact}(S) = V(1 - Z)^{-1}U$ in the large algebra of $\mathrm{Fact}(S)$. The last equality implies that $(1 - Z)^{-1}$ has coefficients equal to 0 or 1, and thus that it is an S-code. It also implies that it is S-complete. It is conjectured that any finite complete code in the full shift is $\mathbb{N}$-factorizing. This is called the *factorization conjecture* (see [9]).

C. Reutenauer has proved that any finite complete code in the full shift is $\mathbb{Z}$-factorizing (see [13]). He has conjectured in [3] that any finite code which is complete and minimal for this property in an edge shift is $\mathbb{Z}$-factorizing.

The following statement shows that the extension to sofic shifts of Reutenauer's conjecture is not true. Indeed, the set Y defined below is a complete code in the even system by Proposition 6. It is also minimal for this property as one may verify.

Proposition 7 *The code $Y = \{a, ab, ba, bab, bbbb\}$ is not $\mathbb{Z}$-factorizing in the even shift.*

PROOF. Let S be the even shift. Suppose that U, V are two polynomials in $\mathbb{Z}[S]$ such that $1 - Y = U(1 - A)V$. Since any word in Y has at most one occurrence of a, the monomials of U and V belong to b^*. The equation $1 - Y = U(1 - A)V$ is equivalent to the equation $A^* = VY^*U$. We have $(U, 1) = (V, 1) = 1$. Next, we have either $(U, b) = 1$ and $(V, b) = 0$, or $(V, b) = 1$ and $(U, b) = 0$. Suppose for instance that $(U, b) = 1$ and $(V, b) = 0$. Then

$$(VY^*U, ba) = (V, b)(Y^*, a)(U, 1) + (V, 1)(Y^*, ba)(U, 1) = 2,$$

which is a contradiction. $\square$

References

[1] M.-P. Béal, D. Perrin, Codes and sofic constraints, Theoret. Comput. Sci. 340 (2) (2005) 381–393.

[2] M.-P. Béal, D. Perrin, Complete codes in sofic systemsTo appear, STACS'06.

[3] Ch. Reutenauer, Ensembles libres de chemins dans un graphe, Bull. Soc. Math. France 114 (2) (1986) 135–152.

[4] A. Restivo, Codes and local constraints, Theoret. Comput. Sci. 72 (1) (1990) 55–64.

[5] J. Ashley, B. Marcus, D. Perrin, S. Tuncel, Surjective extensions of sliding-block codes, SIAM J. Discrete Math. 6 (4) (1993) 582–611.

[6] M. Dalai, R. Leonardi, Non prefix-free codes for constrained sequences, in: International Symposium on Information Theory, 2005. ISIT 2005, IEEE, 2005, pp. 1534 – 1538.

[7] G. Gönenç, Unique decipherability of codes with constraints with application to syllabification of Turkish words, in: COLING 1973: Computational And Mathematical Linguistics: Proceedings of the International Conference on Computational Linguistics, Vol. 1, 1973, pp. 183–193.

[8] D. A. Lind, B. H. Marcus, An Introduction to Symbolic Dynamics and Coding, Cambridge, 1995.

[9] J. Berstel, D. Perrin, Theory of Codes, Academic Press, 1985, http://igm.univ-mlv.fr/~berstel/LivreCodes/Codes.html.
URL `http://igm.univ-mlv.fr/ berstel/LivreCodes/Codes.html`

[10] M.-P. Béal, D. Perrin, Symbolic dynamics and finite automata, in: Handbook of formal languages, Vol. 2, Springer, Berlin, 1997, pp. 463–505.

[11] M.-P. Béal, M. Crochemore, F. Mignosi, A. Restivo, M. Sciortino, Computing forbidden words of regular languages, Fund. Inform. 56 (1-2) (2003) 121–135.

[12] A. Restivo, On codes having no finite completions, Discrete Math. 17 (3) (1977) 309–316.

[13] J. Berstel, Ch. Reutenauer, Rational Series and their Languages, Springer-Verlag, 1988.