Journal of Computer Science and Cybernetics, V.36, N.4 (2020), 381–392
DOI 10.15625/1813-9663/36/4/15317

SET DECIPHERABLE LANGUAGES AND GENERATORS

TRAN VINH DUC

School of Information and Communication Technology,
Hanoi University of Science and Technology

Abstract. We investigate the problem to characterize whether the infinite product of a given language L is generated by an ω-code. Up to now, this problem is open even if language L is a finite language.

In this work, we consider a class of languages named ω-set decipherable languages which are very close to the ω-codes. We solve the problem in the restricted case where L is ω-set decipherable and L^* is the greatest generator of L^ω.

Keywords. Codes; ω-codes; Dominoes; Formal languages; Generators; Infinite words.

1. INTRODUCTION

The languages of infinite words (or ω-languages), see [15], are an extension of formal languages where the finite sequences (or finite words) are replaced by infinite sequences (or infinite words). The infinite sequences allow us to model the behavior of systems which are supposed to work indefinitely, for example Web servers and operating systems. The most important class of the ω-languages is the ω-rational languages. This class is exactly finite unions of ω-languages of the form XY^ω, where X, Y are rational languages and the operation $^\omega$ is the infinite product.

In this paper, we study the *ω-powers*, i.e. the ω-languages of the form L^ω where L is a language of finite words. The ω-power L^ω and the language L are closely linked through the infinite product $^\omega$. Thus, two languages X and Y such that $X^\omega = Y^\omega$ are, in some sense, equivalent. Naturally, we define the family $[L^\omega]$ that consists of the languages X such that $X^\omega = L^\omega$. Such language X is called a *generator* of L^ω. The generators give us a way to explore the ω-languages by using the languages of finite words.

In the family $[L^\omega]$, there are some generators that seem to be more interesting. The minimal generators or finite generators are interesting by their sizes. Among minimal generators, the codes or the ω-codes (if exist) are valuable. These generators attract attention from the point of view of easy decoding. Intuitively, a code (respectively an ω-code) is a set of finite words such that any product (respectively any infinite product) of these words can be decoded in unique fashion. Any ω-code is a code.

Codes play important roles in many domains such as data compression and information transmission. The origin of code notations is as follows. Let β be a bijection from an alphabet B into a code C which extends to a morphism from B^* into C^*. The coding procedure associates a word $b_1b_2\dots b_n$ which is the source text with an encoded message $\beta(b_1)\beta(b_2)\cdots\beta(b_n)$ over the channel alphabet by the use of the coding morphism β. The

Corresponding author.
ductv@soict.hust.edu.vn

concepts of code formalize whether the encoded message can be unambiguously decoded. In data compression, the codes (e.g. Huffman codes) are designed for minimizing encoded message length. In information transmission, the codes are designed for detecting and correcting errors.

This paper is motivated by the open question (see [14, p. 229]): Deciding whether a rational ω-power L^ω is equal to C^ω with C an ω-code. The question is natural and classic; and it is open even if language L is a finite language.

Apparently, the above question is similar to the one in free monoids: Deciding whether a rational submonoid M of free monoid is equal to C^* with C a code. In fact, the structure of submonoids and ω-powers is very different.

For any submonoid M of a free monoid, we know [2, p. 60] that M has the greatest generator (with respect to inclusion) which is M itself, and M has the smallest generator $\mathrm{Root}(M) = (M \setminus \varepsilon) \setminus (M \setminus \varepsilon)^2$. Then M is generated by a code if and only if $\mathrm{Root}(M)$ is a code.

For any ω-power L^ω, the situation is more complicated. We know in [13] that L^ω never has the smallest generator, and L^ω does not always have the greatest generator. In case where the greatest generator M exists, for example when L^ω is generated by a finite language, it may happen that $\mathrm{Root}(M)$ is not an ω-code while L^ω is generated by an ω-code.

Our starting point is a result proved in [9] stating that if L is a code such that L^* is the greatest generator of L^ω, then there exists an ω-code C such that $C^\omega = L^\omega$ if and only if L itself is an ω-code. We try to extend this result by defining a class of languages that is as simple as possible after the codes, and we prove the similar result for this class.

Here we consider a superset of codes named set decipherable languages [6, 1]. Intuitively, a set decipherable language is a set of words such that any product of these words can be uniquely deciphered up to both commutativity of elements and the number of occurrences of elements. Based on this idea, we define ω-set decipherable languages where any product of words is replaced by any *infinite product of words*. This class is a superset of ω-codes, and it is very close to the ω-codes.

Our result. In this work, we solve the question in the restricted case where L is an ω-set decipherable language and L^* is the greatest generator of L^ω. We prove that $L^\omega = C^\omega$ for some ω-code C if and only if L is itself an ω-code (see Theorem 27).

Our technique. Our approach is different from [9]. Instead of manipulating directly the language L, we look at the minimal relations satisfied by L. These relations capture the structure of the double L-factorizations of words or infinite words, and they are easy to compute by using a variation of domino graph [6]. Thus, the existence of ω-code generators can be shown by analyzing the system of minimal relations (see Proposition 23). In a broader view, see [17, 16], this approach makes some notations of linear algebra appear on the set of words or infinite words when linear combinations are replaced by relations.

Related work. The generators of rational ω-languages was first studied by Latteux and Timmerman [10]. They gave an algorithm to decide whether a rational ω-power L^ω is finitely generated, that is $L^\omega = F^\omega$ for some finite language F. The structure of the family of generators has been considered in [13]. Generally, the family of generators has no greatest element but has a finite number of maximal generators, and every generator is included in a maximal generator. These maximal generators are rational monoids and can be computed.

The existence of ω-code generators was first investigated by Litovsky [12]. He showed a

method to decide whether a rational ω-power L^ω is generated by a prefix code. Devolder [4] extended this result for the languages with bounded deciphering delay (i.e. languages allowing to perform the decoding of infinite words after some bounded delay, without waiting the whole message). Julia et al. [9] solved the problem for an interesting case: L^ω has the greatest generator that is generated by a code. Notice that the condition of having the greatest generator is only a technical assumption; we can remove it by considering all maximal generators instead of the greatest one. More recently, Tran and Litovsky [19] unraveled the problem when L is a one-relation language (i.e. language satisfied by only one basic relation).

2. PRELIMINARIES

Let A be an alphabet and A^* (resp. A^ω) is the set of all finite (resp. infinite or ω) words. The empty word is denoted by ε and A^+ denotes $A^* \setminus \{\varepsilon\}$. Let $x \in A^*$, we denote by $|x|$ the length of x. The subsets of A^* (resp. A^ω) are called languages (resp. ω-languages). We denote by $A^\infty = A^* \cup A^\omega$ the set of finite or infinite words. The product of a finite word $x = a_0a_1 \cdots a_n$ from A^* with an infinite word $y = b_0b_1 \cdots$ of A^ω is the infinite word $xy = a_0a_1 \cdots a_nb_0b_1 \cdots$.

A word $y \in A^*$ is called a *prefix* of a word $x \in A^\infty$ if $x = yv$ with $v \in A^\infty$. A word y is a *proper prefix* of a word x if y is prefix of x and $x \neq y$. The language $\mathrm{Pref}(x)$ is the set of all prefixes of x. Let $X \subseteq A^\infty$, we define $\mathrm{Pref}(X) = \bigcup_{x\in X} \mathrm{Pref}(x)$.

Let $L \subseteq A^*$ be a language. The language L^* is the set of finite words built with words in L, that is

$$L^* = \{x_1x_2 \cdots x_n \mid n \geq 0 \text{ and } x_1, \ldots, x_n \in L\}.$$

In the same way, the ω-language L^ω is the set of infinite words:

$$L^\omega = \{x_0x_1 \cdots \mid \text{ for all } i \geq 0,\ x_i \in L \setminus \{\varepsilon\}\}.$$

Thus, L^ω is the set of infinite words obtained by concatenating an infinite sequence of nonempty words of L. We denote by $L^+ = LL^* = L^*L$ and by $L^\infty = L^* \cup L^\infty$.

An ω-language of the form L^ω is said to be an *ω-power*. A language of finite words $G \subseteq A^*$ is called a *generator* of L^ω if $G^\omega = L^\omega$.

An L-factorization of a word $x \in A^*$ is a finite sequence $(x_1, \ldots, x_n)$ of words of L such that $x = x_1 \ldots x_n$. An L-factorization of an infinite word $w \in A^\omega$ is an infinite sequence $(x_1, x_2, \ldots)$ of words of L such that $w = x_1x_2 \ldots$.

Definition 1. Let $C \subseteq A^*$ be a language.

- C is a *code* if any word in A^* has at most one C-factorization.
- C is a *ω-code* if any infinite word in A^ω has at most one C-factorization.

Note that the code is called *unique decipherable* code in [6] and ω-code is called *infinitary finite length* code in [18]. Despite their name, ω-codes are finite word languages. It is effectively decidable whether a regular language is a code or an ω-code (see [3, 20]).

Although the definition of codes is on finite words, the codes can be characterized by their way of acting on infinite words. We present a characterization of codes based on the factorizations of infinite periodic words.

Proposition 2 (see [5]). *Let $C \subseteq A^*$ be a language. Then C is a code if and only if for every $u \in C^+$, u^ω has a single C-factorization.*

By this proposition, an ω-code is a code. The following example exhibits a code that is not an ω-code.

Example 3. The code $C = \{a, ab, b^2\}$ is not an ω-code because the infinite word ab^ω has two C-factorizations: $(ab, bb, bb, \dots)$ and $(a, bb, bb, \dots)$.

A variation of codes named set decipherable codes was introduced by Guzmán [6].

Definition 4. A language $C \subseteq A^*$ is a *set decipherable* (SD) if any two C-factorizations of a word of A^* use the same set of elements. More precisely, C is a SD language if, whenever

$$x_1 x_2 \cdots x_n = y_1 y_2 \cdots y_m \quad \text{with} \quad x_i, y_j \in C$$

holds, then necessarily we have

$$\{x_1, x_2, \cdots, x_n\} = \{y_1, y_2, \cdots, y_m\}.$$

Thus, if C is SD, then every word in C^* has unique factorization in words in C up to both commutativity of elements and the number of occurences of elements. A motivating example here, given in [6], is that two rival parties, willing to reveal to each other the presence or absence of certain items without revealing the exact count.

In order to extend the set decipherability for infinite words, we define a new class of languages, called ω-set decipherable languages.

Definition 5. A language $C \subseteq A^*$ is a *ω-set decipherable* (ω-SD) if any two C-factorizations of an infinite word of A^ω use the same set of elements.

By definition, a language C is ω-SD if for all $x \in C$, we have

$$(C \setminus \{x\})^\omega \cap C^* x C^\omega = \emptyset. \tag{1}$$

Note that if C is finite, then the equation (1) can be verified efficiently. Thus we can decide if a given finite language is ω-SD.

Remark 6. As the equality $x_1 \cdots x_n = y_1 \cdots y_m$ implies $(x_1 \dots x_n)^\omega = (y_1 \dots y_m)^\omega$, the ω-SD languages are SD. But the inversion is not true. The following example exhibits a SD language that is neither a code nor an ω-SD language.

Example 7. Consider the SD language $S = \{s_0, s_1, s_2, s_3\}$, where $s_0 = ab$, $s_1 = ba$, $s_2 = a^2baba^2$ and $s_3 = ba^2ba^2b$. The SD language S is *not* a code because the following finite word has two S-factorizations

$$\begin{aligned} s_1 s_0 s_2 s_1 s_3 &= ba \cdot ab \cdot aababaa \cdot ba \cdot baabaab \\ &= baabaab \cdot ab \cdot aababaa \cdot ba \cdot ab \\ &= s_3 s_0 s_2 s_1 s_0. \end{aligned}$$

Moreover, the SD language S is *not* an ω-SD because the following infinite word has two S-factorizations that yield two distinct sets of elements

$$\begin{aligned} (s_3 s_0 s_2)^\omega &= (baabaab \cdot ab \cdot aababaa)^\omega \\ &= ba \cdot ab\,(aababaa \cdot ba \cdot baabaab)^\omega \\ &= s_1 s_0 (s_2 s_1 s_3)^\omega. \end{aligned}$$

Table 1. Examples in brief

Example	ω-code	Code	ω-SD language	SD language
3	No	Yes	No	Yes
7	No	No	No	Yes
8	No	No	Yes	Yes

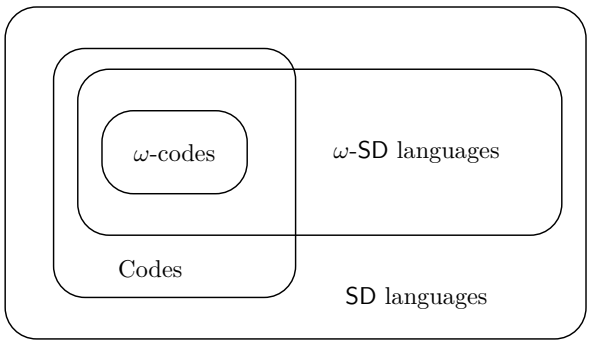

Figure 1. Proper containments of Codes, ω-codes, SD languages and ω-SD languages. The codes and ω-SD languages are incomparable

The following example exhibits a ω-SD language that is not a code.

Example 8. Let $S = \{x_0, x_1, x_2, x_3\}$ where $x_0 = bba, x_1 = bab, x_2 = bbabb$ and $x_3 = abbbabab$. This language is ω-SD (see Example 16). But S is not a code because the following finite word has two factorization on S:

$$\begin{aligned} x_0x_2x_1x_3 &= bba \cdot bbabb \cdot bab \cdot abbbabab \\ &= bbabb \cdot abbbabab \cdot bba \cdot bab \\ &= x_2x_3x_0x_1. \end{aligned}$$

We summarize the examples in Table 2.. The relationship of containment between the codes, ω-code, SD languages and ω-SD languages is shown in Figure 2.

We reiterate the problem which is the motivation for this work.

Problem 9. Given a finite language L, find an algorithm to decide whether $L^\omega = C^\omega$ for some ω-code C.

Example 10. Let language $L = \{a^2, a^3, b\}$. L is not a code, then L is not an ω-code. However, the language $W = \{a^2, a^3b, b\}$ is an ω-code generator of L^ω.

The following theorem is the starting point to this work.

Theorem 11 (see [9])**.** *Let C be a rational code such that C^* is the greatest generator (with respect to inclusion). Then the ω-power C^ω has an ω-code generator if and only if C is an ω-code.*

3. RELATIONS AND DOMINO GRAPH

Given a language $L \subseteq A^*$, we consider a bijection $\tilde{}\colon \Sigma \to L$ between an alphabet Σ and the language L. This mapping is extended to Σ^* as a morphism $\tilde{}\colon \Sigma^* \to L^*$. This morphism is said to be a *coding morphism* for L. This mapping also is extended to $\tilde{}\colon \Sigma^\omega \to L^\omega$ by setting

$$\widetilde{x_0x_1\ldots} = \tilde{x_0}\tilde{x_1}\ldots.$$

Thus each L-factorization of a word in A^∞ is represented by a word in Σ^∞. By abuse of language, the subsets of Σ^∞ are called *languages of factorizations*. For a language $C \subseteq \Sigma^\infty$, we denote $\tilde{C} = \{\tilde{x} \mid x \in C\}$.

Let x and y be two words in Σ^∞ such that $\tilde{x} = \tilde{y}$, we write $x \cong y$ that is called a *relation* in Σ^∞. A relation $x \cong y$ is called *nontrivial* if $x \neq y$. It is obvious that if $x \cong y$ and $x' \cong y'$, then $xx' \cong yy'$. Naturally, we consider the minimal length relations. A nontrivial relation $x \cong y$ is called *minimal* if for all nonempty proper prefix p of x and for all nonempty proper prefix q of y, we have $p \ncong q$. We denote by $R(L)$ and $R_{\min}(L)$ the set of nontrivial relations and minimal relations of L, respectively.

Example 12. Consider the language $L = \{a, ab, bc, c\}$, the alphabet $\Sigma = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}\}$ and the bijection $\mathtt{0} \mapsto a$, $\mathtt{1} \mapsto ab$, $\mathtt{2} \mapsto bc$, $\mathtt{3} \mapsto c$. This language has only one minimal relation $\mathtt{02} \cong \mathtt{13}$. The language L is not SD.

In order to compute the minimal relations satisfied by a finite language, we look at a variation of domino graph that is suggested by Guzmán (see [6]). These graphs have been used for deciding efficiently the multiset decipherability (see [7]) on finite words. We adapt these graphs to finite or infinite words.

Let $G = (V, E)$ be the directed graph with vertex set

$$V = \{\mathtt{open}, \mathtt{close}\} \cup \left\{ \begin{bmatrix} u \\ \varepsilon \end{bmatrix} \mid u \in \mathrm{Pref}(L) \setminus \{\varepsilon\} \right\} \cup \left\{ \begin{bmatrix} \varepsilon \\ u \end{bmatrix} \mid u \in \mathrm{Pref}(L) \setminus \{\varepsilon\} \right\}$$

and with edge set $E = E_1 \cup E_2 \cup E_3 \cup E_4$ where

$$E_1 = \left\{ \left(\mathtt{open}, \begin{bmatrix} \varepsilon \\ u \end{bmatrix} \right) \mid u \in L \right\},$$

$$E_2 = \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \mathtt{close} \right) \mid u \in L \right\},$$

$$E_3 = \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} uv \\ \varepsilon \end{bmatrix} \right) \mid v \in L \right\} \cup \left\{ \left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} \varepsilon \\ uv \end{bmatrix} \right) \mid v \in L \right\},$$

$$E_4 = \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ v \end{bmatrix} \right) \mid uv \in L \right\} \cup \left\{ \left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} v \\ \varepsilon \end{bmatrix} \right) \mid uv \in L \right\}.$$

A finite path in G is *successful* if its starting vertex is `open` and its arrival vertex is `close`. An infinite path in G is *successful* if its starting vertex is `open`.

The *domino graph* associated with L, denote by $G(L)$, is the directed graph (V', E') where V' consists of the vertices $\mathtt{open}, \mathtt{close}$ and the vertices $v \in V$ such that there exists a (finite or infinite) successful path that goes through v; and E' consists of the edges $e \in E$ such that there exists a successful (finite or infinite) path that goes through e.

We denote by $\hat{}: L \to \Sigma$ the bijection defined by $\hat{u} = x$ if $u = \tilde{x}$. The *domino function* associated with L is the mapping d from E to $\left\{ \begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ u \end{bmatrix} \mid u \in L \right\}$ defined on

- E_1 by $d\left(\mathtt{open}, \begin{bmatrix} \varepsilon \\ u \end{bmatrix}\right) = \begin{bmatrix} \widehat{u} \\ \varepsilon \end{bmatrix}$,
- E_2 by $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \mathtt{close}\right) = \begin{bmatrix} \widehat{u} \\ \varepsilon \end{bmatrix}$,
- E_3 by $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} uv \\ \varepsilon \end{bmatrix}\right) = \begin{bmatrix} \varepsilon \\ \widehat{v} \end{bmatrix}$ and $d\left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} \varepsilon \\ uv \end{bmatrix}\right) = \begin{bmatrix} \widehat{v} \\ \varepsilon \end{bmatrix}$,
- E_4 by $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ v \end{bmatrix}\right) = \begin{bmatrix} \widehat{uv} \\ \varepsilon \end{bmatrix}$ and $d\left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} v \\ \varepsilon \end{bmatrix}\right) = \begin{bmatrix} \varepsilon \\ \widehat{uv} \end{bmatrix}$.

The domino function is extended by morphism on the finite or infinite paths by setting, for each finite path p consisting of edges $(e_1, e_2, \ldots, e_m)$,

$$d(p) = d(e_1)d(e_2)\ldots d(e_n) \in \Sigma^* \times \Sigma^*$$

and for each infinite path p consisting of edges $(e_1, e_2, \ldots)$,

$$d(p) = d(e_1)d(e_2)\cdots \in \Sigma^\omega \times \Sigma^\omega.$$

The pair $d(p)$ is called *domino* associated with the path p in G. The first and second components of $d(p)$ are respectively denoted by $d_1(p)$ and $d_2(p)$.

A successful path p of $G(L)$ is trying to find two factorizations of the same word in $L^* \cup L^\omega$ beginning with distinct words. The factorizations obtained so far are $d_1(p)$ and $d_2(p)$. Thus $d_1(p) \cong d_2(p)$ is a nontrivial relation of L. The following proposition (see [6]) shows that these relations are minimal.

Proposition 13. *Let $x, y \in \Sigma^\infty$, $x \cong y$ is a minimal relation of L if and only if there is a finite or infinite successful path p in $G(L)$ such as $d(p) = \left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$ or $d(p) = \left[\begin{smallmatrix} y \\ x \end{smallmatrix}\right]$.*

We denote by $\mathcal{D}(L) = \{d(p) \mid p \text{ is a successful path of } G(L)\}$.

Example 14. Let $L = \{a, ab, ba\}$ and let $\Sigma = \{\mathtt{0}, \mathtt{1}, \mathtt{2}\}$ be the alphabet of relations of L. The domino graph $G(L)$ is shown in the Figure 14. We have

$$\begin{aligned} \mathcal{D}(L) &= \begin{bmatrix} \mathtt{0} \\ \varepsilon \end{bmatrix} \left(\begin{bmatrix} \varepsilon \\ \mathtt{1} \end{bmatrix} \begin{bmatrix} \mathtt{2} \\ \varepsilon \end{bmatrix} \right)^* \begin{bmatrix} \varepsilon \\ \mathtt{1} \end{bmatrix} \begin{bmatrix} \varepsilon \\ \mathtt{0} \end{bmatrix} \begin{bmatrix} \mathtt{2} \\ \varepsilon \end{bmatrix} \cup \begin{bmatrix} \mathtt{0} \\ \varepsilon \end{bmatrix} \left(\begin{bmatrix} \varepsilon \\ \mathtt{1} \end{bmatrix} \begin{bmatrix} \mathtt{2} \\ \varepsilon \end{bmatrix} \right)^\omega \\ &= \begin{bmatrix} \mathtt{0} & \left(\begin{smallmatrix} \mathtt{2} \\ \mathtt{1} \end{smallmatrix} \right)^* & \begin{smallmatrix} \mathtt{2} \\ \mathtt{10} \end{smallmatrix} \end{bmatrix} \cup \begin{bmatrix} \mathtt{0} & \left(\begin{smallmatrix} \mathtt{2} \\ \mathtt{1} \end{smallmatrix} \right)^\omega \end{bmatrix}. \end{aligned}$$

Thus the set of minimal relations of L is exactly the system

$$\begin{cases} \mathtt{0}\mathtt{2}^n \cong \mathtt{1}^n\mathtt{0} & \text{for all} \quad n > 0 \\ \mathtt{0}\mathtt{2}^\omega \cong \mathtt{1}^\omega. \end{cases}$$

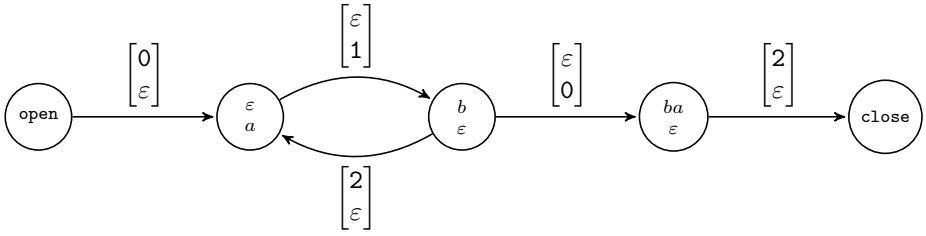

Figure 2. The domino graph of language $L = \{a, ab, ba\}$

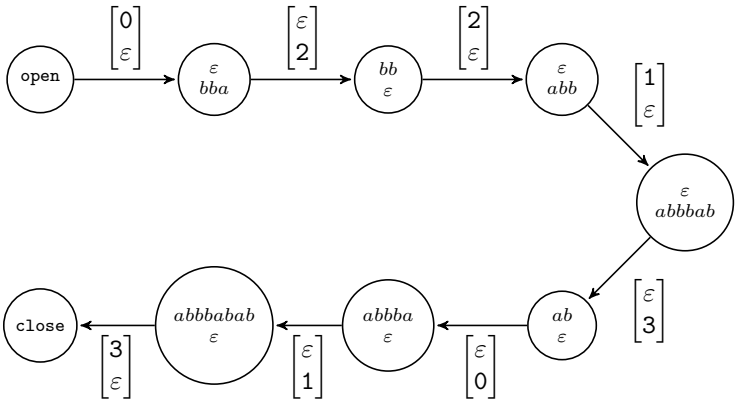

Figure 3. The domino graph of language $S = \{bba, bab, bbabb, abbbabab\}$

The following proposition states that the codes, ω-codes, SD codes, ω-SD codes can be characterized in terms of its domino graph $G(L)$ and the function d_1 and d_2.

Proposition 15. *Let L be a finite language.*

(i) L is a code if and only if no finite successful path exists in $G(L)$.

(ii) L is an ω-code if and only if no finite or infinite successful path exists in $G(L)$.

(iii) L is SD if and only if all finite successful paths p in $G(C)$ are such that $d_1(p)$ and $d_2(p)$ have the same set of letters.

(iv) L is ω-SD if and only if all finite or infinite successful paths p in $G(L)$ are such that $d_1(p)$ and $d_2(p)$ have the same set of letters.

Example 16. Let $S = \{bba, bab, bbabb, abbbabab\}$ and let $\Sigma = \{0, 1, 2, 3\}$ be the alphabet of relations of S. The domino graph $G(S)$ is shown in Figure 16. Thus, the language S has only one minimal relation $0213 \cong 2301$. The language S is ω-SD but non code.

4. ω-SET DECIPHERABLE LANGUAGES AND GENERATORS

In this section, we assume that L is a language such that L^* is the greatest generator (with respect to inclusion) of L^ω and L is in one-to-one mapping with the alphabet Σ.

Note that this assumption is satisfied by some interesting cases, for example the case where L^ω is an ω-power of a finite language (see [11] and [13]).

We denote by $\mathrm{Amb}_\Sigma(L)$ the set of infinite words in Σ^ω such that the images of these infinite words in A^ω have at least two L-factorizations. That is,

$$\begin{aligned}\mathrm{Amb}_\Sigma(L) &= \{x \in \Sigma^\omega \mid \tilde{x} \text{ has at least two } L\text{-factorizations}\}\\ &= \{x \in \Sigma^\omega \mid \exists y \in \Sigma^\omega,\ x \cong y \text{ and } x \neq y\}.\end{aligned}$$

According to Proposition 2, the language L is a code if and only if the set $\mathrm{Amb}_\Sigma(L)$ has no infinite periodic words.

Lemma 17. *Let $C \subseteq \Sigma^+$ such that $\widetilde{C}$ is a generator of L^ω and let $w \in \Sigma^\omega$. If $w \notin Amb_\Sigma(L)$, then $w \in C^\omega$.*

Proof. Since $\widetilde{C}$ is a generator of L^ω, it follows that for each $w \in \Sigma^\omega$ there is an infinite word $w' \in C^\omega$ such that $w \cong w'$. If $w \notin \mathrm{Amb}_\Sigma(L)$ then $w = w'$. So $w \in C^\omega$. ∎

We recall that a language $P \subseteq \Sigma^+$ is a *prefix code* if no word in P is a proper prefix of another word in P.

Lemma 18. *Let $C \subseteq \Sigma^+$ such that the language $\widetilde{C}$ is a code generator of L^ω. Then the language C is a prefix code over Σ.*

Proof. Assume on the contrary that there exist two nonempty words $u, v \in \Sigma^+$ such that $\{u, uv\} \subseteq C$. Since $\widetilde{C}$ is a generator of L^ω, there exists $w \in C^\omega$ such that $(vu)^\omega \cong w$. Then $u(vu)^\omega = (uv)^\omega \cong uw$. As $\tilde{uv} \neq \tilde{u}$, the periodic word $(\tilde{uv})^\omega$ has two factorizations on $\widetilde{C}$: one starts by $\tilde{u}$ and the other by $\tilde{uv}$. According to Proposition 2, $\widetilde{C}$ is not a code. ∎

Definition 19. We say that two words $u, v \in \Sigma^+$ are *incompatible* if there exist $x, y \in \Sigma^\infty$ such that the relation $ux \cong vy$ is minimal.

Remark 20. By the minimality of relation $ux \cong vy$, two incompatible words u and v must have no common prefix.

Let $X \subseteq \Sigma^\infty$. We denote by $\mathrm{P}(X) = \mathrm{Pref}(X) \setminus \{\varepsilon\}$.

Lemma 21. *Let $C \subseteq \Sigma^+$ such that $\widetilde{C}$ is an ω-code generator of L^ω. Let u and v be two incompatible words. Then for all $m \in \Sigma^*$, the set $m\mathrm{P}(\{u, v\}) \cap C$ is the empty set or a singleton.*

Proof. According to Lemma 18, the language C is a prefix code. For each $m \in \Sigma^*$, each set of $m\mathrm{P}(u) \cap C$ and $m\mathrm{P}(v) \cap C$ is either the empty set or a singleton. Therefore, it is sufficient to show that $m\mathrm{P}(u) = \emptyset$ or $m\mathrm{P}(v) = \emptyset$.

Assume on the contrary that there exist $p \in \mathrm{P}(u)$ and $q \in \mathrm{P}(v)$ such that $\{mp, mq\} \subseteq C$. As two words u and v are incompatible, there exist $x, y \in \Sigma^\omega$ such that $px \cong qy$ is a minimal relation. Then we have $(mp)x \cong (mq)y$. There are two cases:

- If $\tilde{mp} \neq \tilde{mq}$, then $\widetilde{C}$ is not an ω-code generator of L^ω;
- If $\tilde{mp} = \tilde{mq}$, that is $mp \cong mq$, then $p \cong q$ which contradicts the minimality of the relation $px \cong qy$.

In both cases, we obtain a contradiction. ∎

This is a corollary of Lemma 21.

Corollary 22. *Let $C \subseteq \Sigma^+$ such that $\widetilde{C}$ is an ω-code generator of L^ω. Let u and v be two incompatible words. Then for all $m \in \Sigma^*$, there exists $z \in \{u, v\}$ such that $m\mathrm{P}(z) \cap C = \emptyset$.*

Proof. According to Lemma 21, we have $m\mathrm{P}(u) \cap C = \emptyset$ or $m\mathrm{P}(v) \cap C = \emptyset$. ∎

Proposition 23. *Let u and v be two incompatible words. If*

$$Amb \cap \{u, v\}^\omega = \emptyset,$$

then L^ω has no ω-code generator.

Proof. Assume that there exists $C \subseteq \Sigma^+$ such that $\widetilde{C}$ is an ω-code generator of L^ω. We build, by induction, an infinite sequence $(z_i)_{i\geq 0}$ of words such that for all $i \geq 0$,

$$\begin{cases} z_i \in \{u, v\} \\ \mathrm{P}(z_0 \dots z_i) \cap C = \emptyset. \end{cases} \tag{2}$$

Indeed, according to Corollary 22, there exists $z_0 \in \{u, v\}$ such that

$$\mathrm{P}(z_0) \cap C = \emptyset.$$

Assume that we have the sequence $(z_0, \dots, z_{n-1})$ which verifies the condition (2). According to Corollary 22, there exists $z_n \in \{u, v\}$ such that

$$z_0 \dots z_{n-1}\mathrm{P}(z_n) \cap C = \emptyset$$

and by induction hypothesis we have

$$\mathrm{P}(z_0 \dots z_{n-1}) \cap C = \emptyset.$$

Then $z_0, \dots, z_n$ verifies the condition (2).

Now consider the infinite word

$$w = z_0 z_1 \dots z_n \dots \notin \mathrm{Amb}_\Sigma(L),$$

according to Lemma 17, we have $w \in C^\omega$. However, by the construction, we have

$$\mathrm{P}(w) \cap C = \emptyset.$$

This is a contradiction. ∎

The following examples point out how to apply Proposition 23 to prove that some ω-powers have no ω-code generator.

Example 24. Let $L = \{a, ab, bc, c\}$ and $\Sigma = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}\}$. The language L has only one minimal relation $\mathtt{02} \cong \mathtt{13}$. Then $\mathrm{Amb}_\Sigma(L) = \Sigma^*\{\mathtt{02}, \mathtt{13}\}\Sigma^\omega$. We have

$$\mathrm{Amb}_\Sigma(L) \cap \{\mathtt{0}, \mathtt{1}\}^\omega = \emptyset.$$

As two words $\mathtt{0}, \mathtt{1}$ are incompatibles, according to Proposition 23, L^ω has no ω-code generator.

Example 25. Let $L = \{a, ab, b^2\}$ be a suffix code and $\Sigma = \{0, 1, 2\}$. The language L has only one minimal relation $02^\omega \cong 12^\omega$. Thus $\mathrm{Amb}_\Sigma(L) = \Sigma^*\{02^\omega, 12^\omega\}$. We have therefore $\mathrm{Amb}_\Sigma(L) \cap \{0, 1\}^\omega = \emptyset$. According to Proposition 23, the ω-language L^ω has no ω-code generator.

Now we apply the above results for ω-SD codes. Before going to the main theorem, we need the following lemma.

Lemma 26 (see [14, p. 207])**.** *If two words satisfy a nontrivial relation on finite or infinite words, then they are powers of the same word.*

Theorem 27. *Let L be an ω-SD language such that L^* is the greatest generator (with respect to inclusion) of L^ω. Then L^ω has an ω-code generator if and only if L is an ω-code.*

Proof. If L is an ω-code, then L itself is an ω-code generator of L^ω. Otherwise, there exists a minimal relation $0x \cong 1y$ where 0 and 1 are two distinct letters of alphabet Σ. Then 0 and 1 are incompatible. Let us suppose that L is an ω-SD such that L^* is the greatest generator of L^ω and L^ω has an ω-code generator. By Proposition 23, there exists an infinite word

$$w \in \mathrm{Amb}_\Sigma(L) \cap \{0, 1\}^\omega.$$

That is, there exists a nontrivial relation $w \cong v$. Since L is ω-SD and $w \in \{0, 1\}^\omega$, we also have $v \in \{0, 1\}^\omega$. According to Lemma 26, two words $\tilde{0}$ and $\tilde{1}$ are powers of the same word. Thus we have $0^\omega \cong 1^\omega$, which contradicts the assumption that L is ω-SD. ∎

5. CONCLUSIONS

In this paper, we have faced the open problem to characterize whether the infinite product of a given language L is generated by an ω-code. We define a new class of languages named ω-set decipherable languages, and we solve the problem in the restricted case where L is ω-set decipherable and L^* is the greatest generator of L^ω. Our approach here is to study the minimal relations satisfied by L. These relations capture the structure of the double factorizations of words or infinite words, and they are easy to compute by using the domino graphs.

In the future, it would be interesting to consider the problem in the case where L is set decipherable.

ACKNOWLEDGMENT

This research is funded by the Hanoi University of Science and Technology (HUST) under project number T2018-PC-207.

REFERENCES

[1] P. C. Bell, D. Reidenbach, and J. O. Shallit, "Unique decipherability in formal languages," *Theoretical Computer Science*, vol. 804, pp. 149 – 160, 2020.

[2] J. Berstel, D. Perrin, and C. Reutenauer, *Codes and Automata*, ser. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2009, vol. 129.

[3] T. Dang Quyet, H. Nguyen Dinh, and H. Phan Trung, "Algorithms based on finite automata for testing of ω-codes," in *Future Information Technology, Application, and Service*. Springer Netherlands, 2012, pp. 271–279.

[4] J. Devolder, "Generators with bounded deciphering delay for rational ω-languages," *Journal of Automata, Languages and Combinatorics*, vol. 4, no. 3, pp. 183–204, 1999.

[5] J. Devolder, M. Latteux, I. Litovsky, and L. Staiger, "Codes and infinite words," *Acta Cybernetica*, vol. 11, no. 4, pp. 241–256, 1994.

[6] F. Guzmán, "Decidability of codes," *Journal of Pure and Applied Algebra*, pp. 13–35, 1999.

[7] T. Head and A. Weber, "Deciding multiset decipherability," *in IEEE Trans. Inf. Theory*, vol. 41, no. 1, pp. 291–297, 1995.

[8] S. Julia, "On ω-generators and codes," in 23^d *ICALP (Int. Coll. on Automata, Languages and Programming)*, ser. Lecture Notes in Computer Sciences, vol. 1099. Berlin: Springer, 1996, pp. 393–402.

[9] S. Julia, I. Litovsky, and B. Patrou, "On codes, ω-codes and ω-generators," *Information Processing Letters*, vol. 60, no. 1, pp. 1–5, oct 1996.

[10] M. Latteux and E. Timmerman, "Finitely generated ω-langages," *Information Processing Letters*, vol. 23, pp. 171–175, 1986.

[11] I. Litovsky, "Générateurs des langages rationnels de mots infinis," Ph.D. dissertation, Université de Lille, 1988.

[12] ——, "Prefix-free languages as ω-generators," *Information Processing Letters*, vol. 37, pp. 61–65, 1991.

[13] I. Litovsky and E. Timmerman, "On generators of rational ω-power languages," *Theoretical Computer Science*, vol. 53, pp. 187–200, 1987.

[14] M. Lothaire, *Algebraic Combinatorics on Words*, ser. Encyclopedia of Mathematics and its Applications. Cambridge: Cambridge University Press, 2002, vol. 90.

[15] D. Perrin and J. E. Pin, *Infinite Words*. Academic Press, 2002.

[16] A. Saarela, "Independent systems of word equations: From Ehrenfeucht to eighteen," in *Combinatorics on Words*, Springer International Publishing, 2019, pp. 60–67.

[17] ——, "Word equations with kth powers of variables," *Journal of Combinatorial Theory, Series A*, vol. 165, pp. 15 – 31, 2019.

[18] L. Staiger, "On infinitary finite length codes," *Theoretical Informatics and Applications*, vol. 20, no. 4, pp. 483–494, 1986.

[19] V. D. Tran and I. Litovsky, "One-relation languages and ω-code generators," in *Proc. of the* 13^{th} *Mons Theoretical Computer Science Days*, 2010.

[20] R. Zaccagnino, R. Zizza, and C. Zottoli, "Testing DNA code words properties of regular languages," *Theoretical Computer Science*, vol. 608, pp. 84 – 97, 2015.

Received on July 28, 2020
Revised on September 10, 2020