

LABORATOIRE



INFORMATIQUE, SIGNAUX ET SYSTÈMES  
DE SOPHIA ANTIPOLIS  
UMR 6070

## FAMILIES AND $w$ -AMBIGUITY REMOVAL

*Sandrine JULIA, TRAN Vinh Duc*

*Equipe MC3*

Rapport de recherche  
ISRN I3S/RR-2009-06-FR

Mai 2009

# Families and $\omega$ -ambiguity removal

Sandrine JULIA and TRAN Vinh Duc

Université de Nice - Sophia Antipolis,  
Laboratoire I3S - UNS/CNRS, B.P. 121,  
06903 Sophia Antipolis Cedex, France.  
{julia, vdtran}@i3s.unice.fr

**Abstract.** We consider the following open problem: let  $L$  be a rational language, how to decide whether  $L^\omega$  is generated by an  $\omega$ -code? We investigate languages whose ambiguity and/or  $\omega$ -ambiguity are minimal. To do this, we introduce a notion of *family* over relations fulfilled by words. If  $L$  is finite, the number of families is finite. Each of them produces its proper set of *incompatible prefixes*. In case of a unique family, this leads to two twin languages that are candidates to be  $\omega$ -generator  $\omega$ -code if some exist.

## Introduction

Our research deals with the theory of formal languages and our interest goes to rational languages of infinite words which are recognized by Büchi or Muller automata [12]. The operation  $^\omega$  stands for the infinite iteration and associates to a language its  $\omega$ -power. A language  $L$  is a code if and only if every non-empty word has at most one  $L$ -factorization [3]. By analogy, an  $\omega$ -language  $L$  is an  $\omega$ -code if and only if every  $\omega$ -word has at most one  $L$ -factorization [13]. Those classes of languages are useful as they respectively allow factorizations exempt from ambiguity and  $\omega$ -ambiguity. The open decision problem we tackle is the following: is it decidable to know whether a given rational  $\omega$ -power is generated by a code or by an  $\omega$ -code? This problem is open from years [11] and only solved for prefix codes [10].

In this paper, a framework is given for finitely generated  $\omega$ -languages. Our purpose is to identify, by mean of relations, the cases where ambiguity and/or  $\omega$ -ambiguity is minimal. Intuitively, ambiguity (resp.  $\omega$ -ambiguity) is minimal when only one word (resp. one  $\omega$ -word) is intrinsically responsible of it. We propose the concept of *family*. Roughly speaking, a family gathers the essential information on factorizations of ambiguous words and  $\omega$ -words that look like each other. The set of families is finite and the ambiguity or  $\omega$ -ambiguity is minimal if this set is a singleton. Independently of the others, each family provides its set of *incompatible prefixes*. When the family is unique, we show how to build powerful languages which are able to lead directly to  $\omega$ -generator  $\omega$ -code. Their construction is closely linked to the set of incompatible prefixes. For now, some preliminary results are obtained and our approach seems to be promising.

The paper is divided into three main sections. The first one is dedicated to basic definitions and other preliminaries such as  $\infty$ -relations over a relabelling. Section 2 sufficiently deepens the notion of  $\infty$ -relations in order to introduce the concept of *family*. Ambiguity digraphs are used as illustration. Two twin sets of prefixes originate in each family. The last section deals with the open problem. At first, some conditions for the existence of  $\omega$ -codes are given. Some of them depend on families. After several examples, two crucial sets relative to a single family are presented. A study of their properties and their abilities follows. Clearly, they are candidates to be  $\omega$ -generators  $\omega$ -codes if some exist. Finally, the conclusion recapitulates our approach. Beyond its possible generalization, the remaining questions are discussed.

## 1 Preliminaries

### 1.1 Basic definitions

Let  $\Sigma$  be a finite alphabet. A *word* is a finite concatenation of letters in  $\Sigma$  and an  $\omega$ -word is an infinite one.  $\varepsilon$  denotes the empty word.  $\Sigma^*$  is the set of words over  $\Sigma$  and  $\Sigma^\omega$  is the set of  $\omega$ -words.  $\Sigma^+$  denotes  $\Sigma^* \setminus \{\varepsilon\}$  and  $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ . A *language* is a subset of  $\Sigma^*$  and an  $\omega$ -language is a subset of  $\Sigma^\omega$ .

A word  $u \in \Sigma^+$  is a *prefix* of  $v$  in  $\Sigma^\infty$  if  $v \in u\Sigma^\infty$  and a *suffix* of  $v$  if  $v \in \Sigma^*u$ . The *prefix order* is denoted by  $\leq$  and the strict prefix order by  $<$ .  $u$  is a proper prefix of  $v$  if  $u < v$  and  $u \neq \varepsilon$ .  $Pref(v)$  stands for the set of the prefixes of  $v$  and  $PPref(v)$  for the proper ones. If  $L$  is a language or an  $\omega$ -language,  $Pref(L)$  gathers the prefixes of the elements in  $L$ ,  $PPref(L)$  is the proper prefixes set. Respectively, for a positive integer  $i$ ,  $Pref_i(v)$  and  $Pref_i(L)$  denotes prefixes of length  $i$ . The set of suffixes is called  $Suff(L)$  and  $PSuff(L)$  denotes the proper ones. Let  $L$  be a language,  $L^*$  is defined as  $L^* = \{\varepsilon\} \cup (\bigcup_{n>0} \{a_1 \dots a_n \mid \forall i \ 1 \leq i \leq n, a_i \in L\})$  and the  $\omega$ -power  $L^\omega$  is defined as  $L^\omega = \{a_1 \dots a_n \dots \mid \forall i > 0, a_i \in L \setminus \{\varepsilon\}\}$ . Every language  $G$  is called *generator* of  $L^*$  if  $G^* = L^*$  and  $\omega$ -generator of  $L^\omega$  if  $G^\omega = L^\omega$ .

The *root* of  $L^*$  is the language  $Root(L^*) = (L^* \setminus \{\varepsilon\}) \setminus (L^* \setminus \{\varepsilon\})^2$ .

A *L-factorization* of a word  $u$  in  $L^+$  is a finite sequence of words in  $L \setminus \{\varepsilon\}$   $(u_1, u_2, \dots, u_n)$  such that  $u = u_1 u_2 \dots u_n$ . A *L-factorization* of an  $\omega$ -word  $w$  in  $L^\omega$  is an infinite sequence  $(w_1, w_2, \dots, w_n, \dots)$  such that  $w = w_1 w_2 \dots w_n \dots$ . We will say indifferently *L-factorization* or *factorization over L*. A language  $L$  is a *code* if every word  $u \in \Sigma^*$  has at most one *L-factorization* [3]. A language  $L$  is an  $\omega$ -code if every  $\omega$ -word in  $\Sigma^\omega$  has at most one *L-factorization* [13].

For more convenience, rational languages and  $\omega$ -languages will be denoted in the sequel by their regular expressions or  $\omega$ -regular expressions.

At last, the vocabulary related to graph theory can be found in [1].

### 1.2 Relabelling

We consider a relabelling of words in a given language  $L$  in order to handle more easily distinct factorizations of words and  $\omega$ -words. Unfortunately, a finite alphabet doesn't allow us to relabel the words in an infinite language.

Let  $A$  and  $\Sigma$  be two disjoint alphabets and let  $L$  be a language over  $\Sigma$ . A relabelling of the words in  $L$  is a one-to-one mapping  $\sim$  from  $\Lambda \subseteq A$  over  $L \subseteq \Sigma^+$  extended in the canonical one-to-one morphism from  $(\Lambda^\infty, \cdot)$  over  $(L^\infty, \cdot)$  where  $\cdot$  denotes the concatenation operation.

Hence, a  $L$ -factorization  $(u_1, u_2, \dots, u_n)$  is associated to a word  $\varphi = \varphi_1 \dots \varphi_n$  in  $\Lambda^+$  such that  $\forall i, u_i = \tilde{\varphi}_i$  (idem for a factorization of an  $\omega$ -word).

The following relation takes into account factorizations with different first steps. The underlying words and  $\omega$ -words respectively involved constitute the set  $Amb(L)$  of finite ambiguous words and the set  $Amb^\omega(L)$  of  $\omega$ -ambiguous  $\omega$ -words [8][5].

**Definition 1.** Let  $L$  be a finite language associated to the relabelling  $\Lambda$ . The relation  $\cong$  is a binary relation over  $\Lambda^\infty$  stating that  $\varphi \cong \psi$  if and only if there exists a word or an  $\omega$ -word  $v$  in  $L^\infty$  and two  $L$ -factorizations with different first steps corresponding respectively to  $\varphi$  and  $\psi$  such that:  $v = \tilde{\varphi} = \tilde{\psi}$ .

The items of the above relation are composable and infinitely many ones appear. A refinement of the relation will remove this drawback.

**Definition 2.** Let  $L$  be a finite language associated to the relabelling  $\Lambda$ . The  $\infty$ -relation  $\simeq$  is a binary relation over  $\Lambda^\infty$  defined as follows:

$$(\varphi \simeq \psi) \Leftrightarrow (\varphi \cong \psi \wedge (\widetilde{PPref}(\varphi) \cap \widetilde{PPref}(\psi) = \emptyset))$$

We define by analogy the restricted relations  $\simeq_+$  over  $\Sigma^+$  and  $\omega$ -relations  $\simeq_\omega$  over  $\Sigma^\omega$ . The considered factorizations are with different first steps so that  $Pref(\varphi)$  and  $Pref(\psi)$  are disjoint (except in  $\varepsilon$ ).

## 2 Family

The classical mean to represent relations over words consists in using transducers [2] and can also be found in [4][11]. As usual, we need to remove intrinsic symmetry due to symmetrical  $\infty$ -relations. This requires to define a *directed*  $\infty$ -relation.

**Definition 3.** Let  $L$  be a finite language associated to the relabelling  $\Lambda$ . The directed  $\infty$ -relation  $\dot{\simeq}$  is a binary relation over  $\Lambda^\infty$  stating that

$$(\varphi \dot{\simeq} \psi) \iff ((\varphi \simeq \psi) \wedge (Pref_1(\varphi) < Pref_1(\psi)))$$

So, pairs of different first steps are now directed and the  $\infty$ -relation  $\dot{\simeq}$  can be split into the relations  $\dot{\simeq}_+$  and  $\dot{\simeq}_\omega$ . The relation  $\dot{\simeq}$  induces an ambiguity graph for which we propose the following definition.

**Definition 4.** Let  $L$  be a finite language. Its ambiguity digraph  $G = (V, E)$  is defined as follows:

- $V$  is a subset of  $(PSuff(L) \times \{\varepsilon\}) \cup (\{\varepsilon\} \times PSuff(L))$ ;
- $V$  necessarily contains the vertex  $(\varepsilon, \varepsilon)$ , we call it the origin;
- an edge in  $E$  is labelled with  $(e, e')$  in  $(L \cup \{\varepsilon\})^2$  (or  $L^2$  when going out from the origin);
- an edge labelled  $(e, e')$  from the vertex  $(u, u')$  to the vertex  $(v, v')$  may exist if there is a path from the origin to the vertex  $(u, u')$  and also:

$$(v = \varepsilon \wedge v' = (ue)^{-1}(u'e')) \vee (v = (u'e')^{-1}(ue) \wedge v' = \varepsilon)$$

- if an edge  $(e, e')$  goes out from the origin, it must fulfill  $e < e'$ .
- the digraph is cleaned up from vertices (and adjacent edges) neither accessing nor belonging to a cycle accessible from the origin.

Once a language  $L$  is associated to a relabelling language  $\Lambda$ , we get the *relabelled ambiguity digraph*. Hence, an edge labelled  $(e, e')$  in the ambiguity digraph is relabelled by the element  $(\gamma, \gamma') \in (\Lambda \cup \{\varepsilon\})^2$  such that  $\tilde{\gamma} = e$  and  $\tilde{\gamma}' = e'$ .

#### Properties

- (i) Let  $C = ((\gamma_1, \gamma'_1), (\gamma_2, \gamma'_2), \dots, (\gamma_n, \gamma'_n))$  be an elementary cycle starting at the origin. The words  $\gamma = \gamma_1\gamma_2\dots\gamma_n$  and  $\gamma' = \gamma'_1\gamma'_2\dots\gamma'_n$  are elements in  $\Lambda^+$  verifying

$$\gamma \simeq_+ \gamma'$$

- (ii) Let  $P = ((\gamma_1, \gamma'_1), (\gamma_2, \gamma'_2), \dots)$  be an infinite elementary path starting at the origin, the  $\omega$ -words  $\gamma = \gamma_1\gamma_2\dots$  and  $\gamma' = \gamma'_1\gamma'_2\dots$  are elements in  $\Lambda^\omega$  verifying

$$\gamma \simeq_\omega \gamma'$$

- (iii) The language  $L$  is a code iff its ambiguity digraph has no cycle including the origin vertex.
- (iv) The language  $L$  is a code non  $\omega$ -code iff there exists at least a cycle but none of them contains the origin.
- (v) The language  $L$  is an  $\omega$ -code iff its ambiguity digraph is reduced to the origin vertex without any edges.

We illustrate these notions on the next example.

*Example* Let us consider the language  $L = a + ab + ba$  associated to  $\Lambda = 0 + 1 + 2$ . Fig. 1 shows its ambiguity and relabelled ambiguity digraphs. All the relations fulfilled by words and  $\omega$ -words, viewed in the relabelled digraph, are gathered in the following system:

$$\begin{cases} 02 \simeq 10 \\ 02^i 2 \simeq 11^i 0 \text{ for every positive integer } i \\ 02^\omega \simeq_\omega 1^\omega \end{cases}$$

Our example is too simple but, in the general case, we can distinguish different subgraphs logically independent in the ambiguity digraph. Each is linked to a

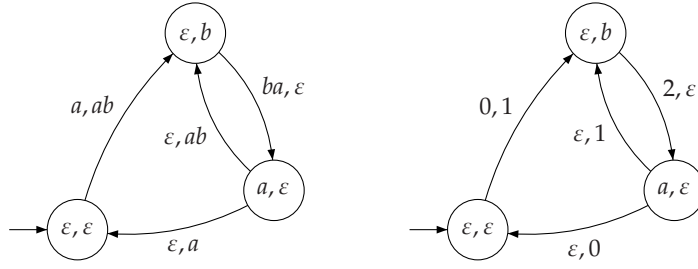


Fig. 1. Ambiguity digraphs for  $L = a + ab + ba$ .

possible directed pair of different first steps beginning two factorizations of the same word or  $\omega$ -word. So, we need to define an equivalence on the items of the directed  $\infty$ -relation  $\simeq$ .

**Definition 5.** Let  $L$  be a finite language associated to the relabelling  $\Lambda$ . The equivalence relation  $\diamond$  is defined on the elements of the relation  $\simeq$  as follows:

$$\begin{aligned} & \left( (\varphi \simeq \psi) \diamond (\varphi' \simeq \psi') \right) \\ & \iff \\ & \left( (\varphi \simeq \psi) \wedge (\varphi' \simeq \psi') \wedge (\text{Pref}_1(\varphi) = \text{Pref}_1(\varphi')) \wedge (\text{Pref}_1(\psi) = \text{Pref}_1(\psi')) \right) \end{aligned}$$

Hence, we call *clan* an equivalence class with respect to  $\diamond$ -relation. The number of classes is clearly equal to the (finite) outer degree  $d$  of the origin vertex on the ambiguity digraph. A subgraph corresponding to items in the same  $\diamond$ -class is called a *cluster*. Although clans are two by two disjoint, on the digraph, their respective clusters can have some parts of the graph in common. In Fig. 1, the whole digraph is identical to the unique cluster due to the edge  $(a, ab)$ , the only one outgoing from the origin.

Once again, a clan is not homogeneous enough. Indeed, it is made of different logical *families* thus we refine the  $\diamond$ -relation.

**Definition 6.** Let  $L$  be a finite language associated to the relabelling  $\Lambda$ . The relation  $\blacklozenge$  is defined as follows:

$$\begin{aligned} & \left( (\varphi \simeq \psi) \blacklozenge (\varphi' \simeq \psi') \right) \\ & \iff \\ & \left( ((\varphi \simeq \psi) \diamond (\varphi' \simeq \psi')) \wedge ((\bar{\varphi} < \bar{\varphi}') \vee (\bar{\varphi}' < \bar{\varphi})) \right) \end{aligned}$$

Now the  $\blacklozenge$ -classes yield to families inside clans. However, it is not immediate to recognize a family at first sight in the digraph. A family is an elementary component of both ambiguities. Inside a family relations, the underlying words are responsible of ambiguity (and induce  $\omega$ -ambiguity) and at most one underlying  $\omega$ -word causes intrinsic  $\omega$ -ambiguity. Moreover, their prefixes are always comparable. The fact that a family is unique coincides with the case where the respective ambiguities, if exist, are *minimal*.

Finally, we need to handle sets of prefixes issued from a same family of  $\infty$ -relations hence a suitable definition is given.

**Definition 7.** Let  $\mathcal{F} = (\varphi_i \simeq \psi_i)_{i \in I}$  be a family of directed  $\infty$ -relations. We define the five respective sets of prefixes:

$$\begin{aligned} LPref_{\mathcal{F}} &= \bigcup_{i \in I} Pref(\varphi_i) \\ RPref_{\mathcal{F}} &= \bigcup_{i \in I} Pref(\psi_i) \\ PLPref_{\mathcal{F}} &= \bigcup_{i \in I} PPref(\varphi_i) \\ PRPref_{\mathcal{F}} &= \bigcup_{i \in I} PPref(\psi_i) \\ Pref_{\mathcal{F}} &= LPref_{\mathcal{F}} \cup RPref_{\mathcal{F}} \end{aligned}$$

These prefix sets are essential to deal with the mentioned open problem. We end this section by giving a first consequence of the previous definitions.

*Property* Let  $\mathcal{F} = (\varphi_i \simeq \psi_i)_{i \in I}$  be a family of directed  $\infty$ -relations, the following assertion is verified:

$$\forall \alpha, \beta \in Pref_{\mathcal{F}} \quad (\tilde{\alpha} \leq \tilde{\beta}) \vee (\tilde{\beta} \leq \tilde{\alpha})$$

### 3 Approach and results

To begin this section, we wonder which are the basic properties of words belonging to  $\omega$ -codes when they are  $\omega$ -generators. A notion of *incompatible prefixes* appears as soon as we give a necessary condition for an  $\omega$ -generator to be an  $\omega$ -code.

Before, we recall a result about the finitely generated  $\omega$ -powers.

**Proposition 1.** [9] Let  $L$  be a language. If  $L$  is finite then the language  $\chi(L^\omega)$  is the greatest  $\omega$ -generator of  $L^\omega$  with:

$$\chi(L^\omega) = \{u \in \Sigma^+, uL^\omega \subseteq L^\omega \text{ and } u^\omega \in L^\omega\}$$

This is the reason why, in the sequel, we choose to consider only finite languages  $L$  which are equal to the root of the greatest  $\omega$ -generator of  $L^\omega$ .

### 3.1 Incompatible prefixes set

Now, we present a necessary condition for a language included in  $\Lambda^+$  to be antecedent of an  $\omega$ -code  $\omega$ -generating  $L^\omega$ .

**Proposition 2.** *Let  $L = \text{Root}(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$ . For every  $\omega$ -code  $G$  such that  $G^\omega = L^\omega$ , a language  $\Gamma \subseteq \Lambda^+$  verifying  $\tilde{\Gamma} = G$  is a prefix code.*

*Proof* Let  $G$  be an  $\omega$ -code such that  $G^\omega = L^\omega$ . Assume that  $\Gamma \subseteq \Lambda^+$  verifying  $\tilde{\Gamma} = G$  is not a prefix code. Without loss of generality,  $\Gamma$  contains a pair  $\{\mu, \nu\}$  such that  $\mu < \nu$ . Thus,  $\sigma = (\mu^{-1}\nu) \in \Lambda^+$ . Consequently, the non-empty word  $\tilde{\sigma} \in (G^{-1}G \cap G^\omega(G^\omega)^{-1})$ . A contradiction happens because  $G$  is an  $\omega$ -code.  $\square$

In a family, we show that the set  $\text{Pref}_{\mathcal{F}}$  gathers *incompatible prefixes* in the sense that if the images of two distinct elements in  $\text{Pref}_{\mathcal{F}}$  are in the same  $\omega$ -generator, necessarily, this  $\omega$ -generator is not  $\omega$ -code.

**Proposition 3.** *Let  $L = \text{Root}(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. If a subset  $\Gamma$  of  $\Lambda^+$  contains two different elements  $\mu$  and  $\nu$  in  $\text{Pref}_{\mathcal{F}}$  then the language  $G = \tilde{\Gamma}$  is not an  $\omega$ -code.*

*Proof* As  $\mu$  and  $\nu$  share the same family  $\mathcal{F}$ ,  $\tilde{\mu}$  and  $\tilde{\nu}$  are comparable by prefix order. Without loss of generality, we set  $\tilde{\mu} < \tilde{\nu}$ . Then, two possibilities arise. If  $\mu$  and  $\nu$  both belongs to  $\text{RPref}_{\mathcal{F}}$  or to  $\text{LPref}_{\mathcal{F}}$ , we deduce that  $\mu < \nu$ . As a consequence,  $\Gamma$  is not a prefix code and we conclude with Prop. 2. If, without loss of generality,  $\mu \in \text{LPref}_{\mathcal{F}}$  and  $\nu \in \text{RPref}_{\mathcal{F}}$ , there exist  $\varphi, \psi \in \Lambda^\infty$  such that  $\mu < \varphi$ ,  $\nu < \psi$  and a relation  $\varphi \simeq \psi$  holds. Indeed, we extend the path in the graph beyond  $\mu$  and  $\nu$  back to the origin or not. If  $\varphi \simeq_+ \psi$  then the  $\omega$ -word  $(\tilde{\varphi})^\omega$  has two distinct  $G$ -factorizations. If  $\varphi \simeq_\omega \psi$ ,  $\tilde{\varphi}$  has two distinct  $G$ -factorizations. That prevents  $G$  from being an  $\omega$ -code.  $\square$

This result can be extended to pairs of words that identically start and respectively end with two incompatible prefixes.

**Corollary 1.** *Let  $L = \text{Root}(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. If a subset  $\Gamma$  of  $\Lambda^+$  contains two different elements  $\mu$  and  $\nu$  in  $\lambda\text{Pref}_{\mathcal{F}}$  for some  $\lambda \in \Lambda^*$ , then the language  $G = \tilde{\Gamma}$  is not an  $\omega$ -code.*

*Proof* The result is obtained by a generalization of the proof of Prop. 3.  $\square$

The constraints a family puts on the antecedent of an  $\omega$ -code are summarized in the next proposition.

**Proposition 4.** *Let  $L = \text{Root}(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. For every  $\omega$ -code  $G$  such that  $G^\omega = L^\omega$ , a language  $\Pi$  such that  $\tilde{\Pi} = G$  is a prefix code containing at most one element  $\pi$  in  $\text{Pref}_{\mathcal{F}}$ . So,  $\Pi$  verifies:*

$$\Pi \subseteq \left( \{\pi\} \cup (\Lambda^+ \setminus \text{Pref}_{\mathcal{F}}) \right)$$

*Proof*  $\Pi$  is a prefix code due to Prop. 2 and Prop. 3 ensures that  $\Pi$  cannot contain two different elements in  $Pref_{\mathcal{F}}$ . So the other elements in  $\Pi$  are necessarily in  $\Lambda^+ \setminus Pref_{\mathcal{F}}$ .  $\square$

### 3.2 Some examples

Perhaps the following of the paper will be clearer if we present some representative examples now. The construction of the essential sets  $\Pi$  is based on a greedy algorithm: we're searching on line a  $\Pi$ -factorization of an  $\omega$ -word in  $\Lambda^\omega$ , once we have chosen the side of the family in which we take a particular element  $\pi$ . Sometimes, we allow us to rewrite a part or the totality of the  $\omega$ -word to go on factorizing with non forbidden words.

*Continued example* Let us consider the language  $L = a + ab + ba$  associated to  $\Lambda = 0 + 1 + 2$ . Its unique family is:

$$\begin{cases} 02 \simeq 10 \\ 02^i 2 \simeq 11^i 0 \text{ for every positive integer } i \\ 02^\omega \simeq_\omega 1^\omega \end{cases}$$

0 and 1 are clearly incompatible. Here, we decide to keep 0 and so we compute  $\Pi_0$ . We set  $\Pi_0 = 0 + 2 + (1^+(0 + 2) \setminus 11^*0)$  and we get  $\tilde{\Pi}_0 = 0 + 1^*2$ . In  $\Pi_0$ , 0 was chosen, 2 is neutral,  $1^+(0 + 2)$  describes the forbidden right proper prefixes of the family followed by a letter in so far it doesn't extend it into another forbidden prefix, and finally we avoid the finite right hand side of the family.  $\Pi_0$  is clearly a prefix code. Moreover, the infinite language  $G_0$  such that  $G_0 = Root(\tilde{\Pi}_0)$  is an  $\omega$ -code  $\omega$ -generating  $L^\omega$ . Indeed,  $G = a + (ab)^*ba$ . Another possibility is to choose 1. Then, symmetrically  $\Pi_1 = 1 + 2 + (02^*(0 + 1) \setminus 02^+)$  and we get  $\tilde{\Pi}_1 = 1 + 2 + 02^*0 + 02^*1$ . Although  $\Pi_1$  is clearly a prefix code, it doesn't lead to an  $\omega$ -code. Containing a pattern like  $\{00, 01\}$  suffices to prevent  $Root(\tilde{\Pi}_1)$  from being an  $\omega$ -code.

*Example* Let us consider the code non  $\omega$ -code  $L = a^2 + ba + b$  associated to  $\Lambda = 0 + 1 + 2$ . Its unique family is:

$$\{10^\omega \simeq_\omega 20^\omega\}$$

Removing 2 from  $\Lambda$ , we set  $\Pi_1 = 1 + 0 + (20^*(1 + 2) \setminus \emptyset)$  and we get  $\tilde{\Pi}_1 = 0 + 1 + 20^*1 + 20^*2$ . Because of  $\{21, 22\}$  for instance,  $Root(\tilde{\Pi}_1)$  cannot be an  $\omega$ -code. Besides,  $\Pi_2 = 2 + 0 + (10^*(1 + 2) \setminus \emptyset)$ . So,  $\tilde{\Pi}_2 = 0 + 2 + 10^*1 + 10^*2$  and contains for instance  $\{11, 12\}$ . Once again,  $Root(\tilde{\Pi}_2)$  isn't an  $\omega$ -code. Concomitantly, a result from [6] states that, if  $L$  is a code non- $\omega$ -code, there is no  $\omega$ -code among the  $\omega$ -generators of  $L^\omega$ .

*Example* Let us consider the code  $L = a + ab + bc + c$  associated to  $\Lambda = 0 + 1 + 2 + 3$ . Its ambiguity resides only in finite-length words. Its unique family is:

$$\{02 \simeq_+ 13\}$$

Removing 1 from  $\Lambda$ , we set  $\Pi_0 = 0 + 2 + 3 + (1(0 + 1 + 2 + 3) \setminus 13)$  and we get  $\Pi_0 = 0 + 2 + 3 + 10 + 11 + 12$ . Because of  $\{10, 11\}$ ,  $Root(\widetilde{\Pi}_0)$  cannot be an  $\omega$ -code. Besides,  $\Pi_1 = 1 + 2 + 3 + (0(0 + 1 + 2 + 3) \setminus 02)$ . So,  $\Pi_1 = 1 + 2 + 3 + 00 + 01 + 03$ . By reason of  $\{00, 01\}$ ,  $Root(\widetilde{\Pi}_1)$  cannot be an  $\omega$ -code.

We are convinced that if none of these two languages later called  $\Pi_{left}$  and  $\Pi_{right}$  leads to an  $\omega$ -code, there is no hope that one exists.

### 3.3 Two candidates per family

In this section, we describe a way to eventually find an  $\omega$ -code in the case where the family is unique. The idea is to construct a prefix code  $\Pi$  over  $\Lambda$  such that  $Root(\widetilde{\Pi})$  is guaranteed to be an  $\omega$ -generator of  $L^\omega$ . On the one hand, it is always possible, on the other hand, the  $\omega$ -generator  $Root(\widetilde{\Pi})$  may be an  $\omega$ -code.

Let  $L = Root(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. We call  $\pi_{left}$  (resp.  $\pi_{right}$ ) the only prefix of length 1 in  $LPref_{\mathcal{F}}$  (resp. in  $RPref_{\mathcal{F}}$ ). We propose the next two languages:

$$\Pi_{left} = (\Lambda \setminus \{\pi_{right}\}) \cup (\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}})$$

$$\Pi_{right} = (\Lambda \setminus \{\pi_{left}\}) \cup (\overline{PLPref_{\mathcal{F}}} \setminus \Phi_{\mathcal{F}})$$

where:

$$\Phi_{\mathcal{F}} = \{\varphi, (\varphi \simeq_+ \psi) \in \mathcal{F}\}, \Psi_{\mathcal{F}} = \{\psi, (\varphi \simeq_+ \psi) \in \mathcal{F}\}$$

$$\overline{PRPref_{\mathcal{F}}} = \{\mu\sigma, \mu \in PRPref_{\mathcal{F}}, \sigma \in \Lambda, \mu\sigma \notin PRPref_{\mathcal{F}}\}$$

$$\overline{PLPref_{\mathcal{F}}} = \{\mu\sigma, \mu \in PLPref_{\mathcal{F}}, \sigma \in \Lambda, \mu\sigma \notin PLPref_{\mathcal{F}}\}$$

Actually the right hand sides of the unions can be rewritten as follows:

$$\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}} = (PRPref_{\mathcal{F}} \cdot \Lambda) \setminus RPref_{\mathcal{F}}$$

$$\overline{PLPref_{\mathcal{F}}} \setminus \Phi_{\mathcal{F}} = (PLPref_{\mathcal{F}} \cdot \Lambda) \setminus LPref_{\mathcal{F}}$$

With respect to the open problem, the idea is to test, when the family is unique, if the respective languages  $Root(\widetilde{\Pi}_{left})$  and  $Root(\widetilde{\Pi}_{right})$  are  $\omega$ -codes  $\omega$ -generators of  $L^\omega$ . To start with, we give some remarkable properties of these sets.

**Proposition 5.** *Let  $L = Root(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. The languages  $\widetilde{\Pi}_{left}$  and  $\widetilde{\Pi}_{right}$  are  $\omega$ -generators of  $L^\omega$ .*

*Proof* Let us prove that  $\widetilde{\Pi}_{left}$  is  $\omega$ -generator of  $L^\omega$ . Clearly,  $(\widetilde{\Pi}_{left})^\omega \subseteq L^\omega$ . Now, consider a given  $\omega$ -word  $w$  in  $L^\omega$  and let  $\lambda = \lambda_1\lambda_2\dots$  be an infinite sequence over  $\Lambda$  such that  $\widetilde{\lambda} = w$ . We intend to show that  $w$  admits at least a  $\widetilde{\Pi}_{left}$ -factorization. Four different cases arise:

- if  $\lambda_1 \in (\Lambda \setminus \{\pi_{right}\})$  then  $\lambda \in \Pi_{left}\Lambda^\omega$  and so  $w \in \widetilde{\Pi}_{left}L^\omega$ ;
- if there exists a prefix of  $\lambda$  say  $\psi$  in  $\Psi_{\mathcal{F}}$  such that  $\lambda = \psi\kappa$  with  $\kappa \in \Lambda^\omega$ . There exists a word  $\varphi \in \Lambda^+$  such that  $\varphi \simeq_+ \psi$ . Since  $\widetilde{\varphi} = \widetilde{\psi}$ , the  $\omega$ -word  $w$  verifies  $w = \widetilde{\varphi}\widetilde{\kappa}$ . Since  $\varphi$  starts with  $\pi_{left} \in \Pi_{left}$ , we get that  $w \in \widetilde{\Pi}_{left}L^\omega$ ;
- if  $\lambda = \psi$  such that the relation  $\varphi \simeq_\omega \psi$  is verified for some  $\varphi \in \Lambda^\omega$ . We obtain that  $w = \widetilde{\varphi}$ . Again, since  $\varphi$  starts with  $\pi_{left}$ , we get that  $w \in \widetilde{\Pi}_{left}L^\omega$ ;
- if there exists a prefix  $\rho \in \overline{PRPref_{\mathcal{F}}}$  such that  $\lambda = \rho\kappa$  for some  $\kappa \in \Lambda^\omega$ .  $\rho$  is chosen of maximal length and we set  $\kappa = \kappa_1\kappa_2\dots$  with  $\forall i > 0, \kappa_i \in \Lambda$ . Because of  $\rho$  maximal length, the word  $\rho\kappa_1$  cannot belong to  $\overline{PRPref_{\mathcal{F}}}$ . Either  $\rho\kappa_1 \in \Psi_{\mathcal{F}}$  and we refer to the second case of this proof, either  $\rho\kappa_1 \notin \Psi_{\mathcal{F}}$  and necessarily  $\rho\kappa_1$  belongs to  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$ . Once again,  $w \in \widetilde{\Pi}_{left}L^\omega$ .

In every case, we've shown that  $w \in \widetilde{\Pi}_{left}L^\omega$ . Hence, by infinite iteration of the inclusion  $L^\omega \subseteq \widetilde{\Pi}_{left}L^\omega$ , we deduce that  $L^\omega \subseteq (\widetilde{\Pi}_{left})^\omega$ . So, the equality holds. The proof is analogous for the language  $\Pi_{right}$ .  $\square$

In addition, it happens that those languages built over  $\Lambda$  are prefix codes.

**Proposition 6.** *Let  $L = \text{Root}(\chi(L^\omega))$  be a finite language associated to the relabelling  $\Lambda$  and  $\mathcal{F}$  be a family. By construction, the languages  $\Pi_{left}$  and  $\Pi_{right}$  are prefix codes.*

*Proof* Let us consider  $\Pi_{left}$ . The length of every element in the set  $\Lambda \setminus \{\pi_{right}\}$  is 1 and none of them is prefix of a word in  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$ . So, it remains to show that no word is prefix of another in  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$ . By definition,  $\overline{PRPref_{\mathcal{F}}} = \{\mu\sigma, \mu \in PRPref_{\mathcal{F}}, \sigma \in \Lambda, \mu\sigma \notin PRPref_{\mathcal{F}}\}$ . This yields that the set  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$  is equal to  $\{\mu\sigma, \mu \in PRPref_{\mathcal{F}}, \sigma \in \Lambda, \mu\sigma \notin RPref_{\mathcal{F}}\}$ . Yet, all the proper prefixes of words in  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$  belong to  $\overline{PRPref_{\mathcal{F}}}$ . As a consequence, whatever the two distinct elements we choose in  $\overline{PRPref_{\mathcal{F}}} \setminus \Psi_{\mathcal{F}}$ , they are not comparable by prefix order. So  $\Pi_{left}$  is a prefix code. The demonstration is analogous for  $\Pi_{right}$ .  $\square$

So, the two languages  $\text{Root}(\widetilde{\Pi}_{right})$  and  $\text{Root}(\widetilde{\Pi}_{left})$  have a lot of interesting properties. Under the assumption of the unique family and from the point of view of the open problem, they appear as the first languages we must test in order to find an  $\omega$ -generator  $\omega$ -code. For now, if both of those languages fail to be an  $\omega$ -code, we cannot deduce anything. Nevertheless, it would be surprising that another (more complicated) language were an  $\omega$ -code and not one of them. We have to pursue the study.

Notice that Corollary 1 gives a sufficient condition to check that the languages  $\Pi_{left}$  or  $\Pi_{right}$  have no chance to provide an  $\omega$ -code.

At last, we give two examples with two families. Our method doesn't hold in this case but one can easily imagine its generalization. Indeed, everything we did for a particular family holds, except the chance to reach an  $\omega$ -code without taking into account the other families. Thus, a combination of conditions of each family is needed, otherwise, there is no possibility to find  $\omega$ -codes.

*Examples* The language  $L = a + a^3b + ba^2$ .  $L$  admits two families and finally,  $L^\omega$  is  $\omega$ -generated by an  $\omega$ -code. The language  $L = a^2 + a^3 + b + ba$  has two families as well but no  $\omega$ -generator  $\omega$ -code for its  $\omega$ -power.

## 4 Conclusion

In this work, we introduced the concept of family to understand how ambiguity and  $\omega$ -ambiguity grow. Indeed, the ambiguities, if they exist, are minimal when the family is unique. Afterwards, we showed that every family provides two sets of incompatible prefixes. Mixing prefixes issued from both sets in the same  $\omega$ -generator prevents it from being an  $\omega$ -code. So, the least we have to do to obtain  $\omega$ -codes is to remove all the prefixes from one of each pair of incompatible sets. Already in [7], the method consisted in removing a set of easily identified prefixes. But the structure of their set was limited yet. A lot of questions remain to deepen this study. On the one hand, we are interested in the generalization to several families. On the other hand, a challenge is to establish a link between the existence of  $\omega$ -generators codes and the necessity for acceptable candidates to be  $\omega$ -codes themselves. In addition, some convictions are reinforced by our experimentation. Hence, among the  $\omega$ -generators of a finitely  $\omega$ -generated language, it seems impossible to find at the same time a code non  $\omega$ -code and an  $\omega$ -code, a finite  $\omega$ -code and an infinite one.

## References

1. C. Berge. *The theory of Graphs*. Dover, 2003.
2. J. Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.
3. J. Berstel and D. Perrin. *Theory of codes*. Academic Press, 1985.
4. C. Choffrut and J. Karhumäki. chapter Combinatorics of Words, pages 329–438. Springer, 1997.
5. S. Julia. Ambiguity of infinite words. In *12th Mons Days of Theoretical Computer Science*, Mons, 2008.
6. S. Julia, I. Litovsky, and B. Patrou. On codes,  $\omega$ -codes and  $\omega$ -generators. *Information Processing Letters*, 60(1):1–5, oct. 1996.
7. S. Julia and Tran Vinh Duc. Reduced languages as  $\omega$ -powers. In *Proc. 11<sup>th</sup> Int. Conf. Developments in Language Theory*, Lecture Notes in Computer Science, pages 266–277, Turku, 2007. Springer.
8. J. Karhumäki. On three-element codes. *Theoretical Computer Science*, 40:3–11, 1985.
9. M. Latteux and E. Timmerman. Finitely generated  $\omega$ -languages. *Information Processing Letters*, 23:171–175, 1986.

10. I. Litovsky. Prefix-free languages as  $\omega$ -generators. *Information Processing Letters*, 37:61–65, 1991.
11. M. Lothaire. *Algebraic Combinatorics on Words*. Cambridge, 2002.
12. D. Perrin and J.É. Pin. *Infinite words*. Elsevier, 2004.
13. L. Staiger. On infinitary finite length codes. *Theoretical Informatics and Applications*, 20(4):483–494, 1986.

## Annex

$L = 0 + 1 + 2 \dots$ $= \text{Root}(\chi(L^\omega))$	$\mathcal{F}$	$\Pi_{left}$ $\Pi_{right}$	$\text{Root}(\widetilde{\Pi}_{left})$ $\omega$ -code? $\text{Root}(\widetilde{\Pi}_{right})$ $\omega$ -code?
$a^2 + a^3 + b$	$R\text{Pref}_{\mathcal{F}} = 0(0+1)^*$	$0 + 2 + 12$	yes
	$L\text{Pref}_{\mathcal{F}} = 1(0+1)^*$	$1 + 2 + 02 + 002$	yes
$a + ab + ba$	$02^k \simeq 1^k 0$	$0 + 1^* 2$	yes
	$02^\omega \simeq 1^\omega$	$1 + 2 + 02^*(0+1)$	no
$a + ab + ba^2$	$02^k = (10)^k 0$	$0 + 2 + (10)^*(11 + 12 + 2)$	yes
	$02^\omega \simeq (10)^\omega$	$1 + 2 + 02^*(0+1)$	no
$a^2 + ba + b$	$10^\omega \simeq 20^\omega$	$0 + 1 + 20^*(1+2)$	no (never)
		$0 + 2 + 10^*(1+2)$	
$a + ab + bc + c$	$02 \simeq 13$	$0 + 2 + 3 + 1(0+1+2)$	no (never)
		$1 + 2 + 3 + 0(0+1+3)$	

**Table 1.** Some representative examples