

# Synchronized Shuffle and Regular Languages

M.Latteux Y.Roos  
CNRS URA 369, L.I.F.L.  
Université des Sciences et Technologies de Lille  
U.F.R. I.E.E.A. Informatique  
F59655 Villeneuve d'Ascq cedex  
latteux@lifl.fr yroos@lifl.fr

**Summary.** New representation results for three families of regular languages are stated, using a special kind of shuffle operation, namely the *synchronized shuffle*. First, it is proved that the family of regular star languages is the smallest family containing the language  $(a + bc)^*$  and closed under synchronized shuffle and length preserving morphism. The second representation result states that the family of  $\varepsilon$ -free regular languages is the smallest family containing the language  $(a + bc)^*d$  and closed under synchronized shuffle, union and length preserving morphism. At last, it is proved that Reg is the smallest family containing the two languages  $(a + bb)^*$  and  $a+(ab)^+$ , closed under synchronized shuffle, union and length preserving morphism.

## 1 Introduction

Finite automata are very popular objects in Computer Science and are now used in several other scientific domains. The family of regular languages, called Reg, is a basic class in Chomsky hierarchy. The significance of this family is strengthened by a great number of its nice characterizations. Some of these characterizations state that Reg is equal to the closure of a *small* family under a given set of operations. Besides the famous Kleene's theorem, one can recall the morphic compositional representation of regular languages establishing that Reg is the closure of the family reduced to the single language  $a^*b$  under morphism and inverse morphism. This nice result was proved in 1982 by Culik, Fich and Salomaa [1]. After the existence of such a morphic representation had been proved there followed a sequence of papers improving and generalizing this theorem (see [4], [6] and [9]).

We shall prove here similar characterizations of Reg involving a special kind of shuffle. The shuffle operation appears as a fundamental operation in the theory of concurrency and admits several variations such as literal shuffle, insertion, etc.. Recently, Mateescu, Rozenberg and Salomaa introduced the notion of shuffle on trajectories [7] that provides a way to find again the most of these variations. Synchronized shuffle was introduced in [2] by De Simone under the name of *produit de mixage*(see also [5], where a closely related operation was introduced). This special shuffle can be seen as a shuffle with *rendez-vous* and is very useful in modelling the behaviours of

parallel processes (see [3] , [8]). This powerful operation is quite amazing. For example, contrary to what was stated in [2], it is not associative.

The study of synchronized shuffle in connection with union and length preserving morphism provides us to obtain new representation results for three families of regular languages.

First, we prove that the family of regular star languages is the smallest family containing the language  $(a+bc)^*$  and closed under synchronized shuffle and length preserving morphism. Our second representation result states that the family of  $\varepsilon$ -free regular languages is the smallest family containing the language  $(a+bc)^*d$  and closed under synchronized shuffle, union and length preserving morphism. At last, we prove that  $\text{Reg}$  is the smallest family containing the two languages  $(a+bb)^*$  and  $a+(ab)^+$ , closed under synchronized shuffle, union and length preserving morphism.

## 2 Preliminaries

### 2.1 Notations

Let  $X$  be an alphabet. For any word  $w \in X^*$ , we shall denote by  $|w|$  the length of the word  $w$  and for any letter  $a \in X$ , we shall denote by  $|w|_a$  the number of occurrences of the letter  $a$  that appear in the word  $w$ .

For any language  $L \subseteq X^*$ , we denote by  $\alpha(L)$ , the *alphabet of  $L$*  that is :

$$\alpha(L) = \{x \in X \mid \exists u \in L : |u|_x > 0\}$$

For example we get :  $\alpha(a+ab) = \alpha(a^2+b^3) = \{a, b\}$ ,  $\alpha(\emptyset) = \alpha(\varepsilon) = \emptyset$ .

We denote by  $\Pi_{X,Y}$  the *projection* over the *sub-alphabet*  $Y$ , i.e. the homomorphism from  $X^*$  to  $Y^*$  defined by:

$$\forall x \in X, \text{ if } x \in Y \text{ then } \Pi_{X,Y}(x) = x, \text{ else } \Pi_{X,Y}(x) = \varepsilon$$

When there is no ambiguity, we shall use notation  $\Pi_Y$  instead of  $\Pi_{X,Y}$ .

$u \sqcup v$  is the *shuffle* of the two words  $u$  and  $v$  that is

$$u \sqcup v = \{u_1v_1u_2v_2\dots u_nv_n \mid u_i \in \Sigma^*, v_i \in \Sigma^*, u = u_1u_2\dots u_n, v = v_1v_2\dots v_n\}$$

This definition is extended over languages by :

$$\forall L_1, L_2 \subseteq X^*, L_1 \sqcup L_2 = \bigcup_{\substack{w_1 \in L_1 \\ w_2 \in L_2}} w_1 \sqcup w_2$$

When there is no ambiguity, for a language reduced to a single word, we shall write this language  $u$  rather than  $\{u\}$ .

## 2.2 Synchronized shuffle

**Definition 1.** [2] Let  $X$  be an alphabet and  $L_1, L_2$  be two languages included in  $X^*$ . The *synchronized shuffle* of  $L_1$  and  $L_2$ , denoted by  $L_1 \sqcap L_2$  is defined by :

$$L_1 \sqcap L_2 = \{w \in (\alpha(L_1) \cup \alpha(L_2))^* \mid \Pi_{\alpha(L_i)}(w) \in L_i, i \in \{1, 2\}\}$$

*Examples :*

1.  $bab \sqcap cac = \{bcabc, bcacb, cbabc, cbacb\}$
2.  $bac \sqcap cab = \emptyset$
3.  $\{a, abb\} \sqcap ab = \emptyset$  and  $a \sqcap ab = ab$
4.  $(a \sqcap \{a, b\}) \sqcap ab = a \sqcap ab = ab$
5.  $a \sqcap (\{a, b\} \sqcap ab) = a \sqcap \emptyset = \emptyset$

Example 3 shows that it generally does not hold that

$$L_1 \sqcap L_2 = \bigcup_{\substack{w_1 \in L_1 \\ w_2 \in L_2}} w_1 \sqcap w_2$$

Examples 4 and 5 show that synchronized shuffle is not an associative operation.

The following properties and proposition easily come from definition :

- the synchronized shuffle is commutative :  $L_1 \sqcap L_2 = L_2 \sqcap L_1$
- $L \sqcap \varepsilon = L$
- $L \sqcap \emptyset = \emptyset$
- if  $\alpha(L_1) = \alpha(L_2)$  then  $L_1 \sqcap L_2 = L_1 \cap L_2$
- if  $\alpha(L_1) \cap \alpha(L_2) = \emptyset$  then  $L_1 \sqcap L_2 = L_1 \sqcup L_2$
- $(L_1^* \sqcap L_2^*) = (L_1 \sqcap L_2)^*$
- $(\varepsilon \in L_1 \sqcap L_2) \iff (\varepsilon \in L_1 \cap L_2)$
- if  $L_1$  and  $L_2$  are  $\varepsilon$ -free languages then for any letter  $a$  :  $(a \in L_1 \sqcap L_2) \iff (a \in L_1 \cap L_2)$

**Proposition 1.** Let  $L_1$  and  $L_2$  be languages with  $X_i = \alpha(L_i)$  for  $i \in \{1, 2\}$ . Then the following equality holds :

$$L_1 \sqcap L_2 = (L_1 \sqcup (X_2 \setminus X_1)^*) \bigcap (L_2 \sqcup (X_1 \setminus X_2)^*)$$

## 2.3 synchronized shuffle and associativity

As we have seen in examples, the synchronized shuffle is not associative. We shall give now a sufficient condition for the synchronized shuffle to be associative over a ( $\sqcap$ -closed) family of language. For every family of language  $\mathcal{L}$ , we denote by  $(\mathcal{L})_{\sqcap}$  the least family of languages containing  $\mathcal{L}$  and closed under  $\sqcap$ . First we can state :

**Proposition 2.** *Let  $\mathcal{L}$  be a family of languages. If for every languages  $L_1, L_2 \in (\mathcal{L})_{\sqcap}$ , the equality  $\alpha(L_1 \sqcap L_2) = \alpha(L_1) \cup \alpha(L_2)$  holds then  $\sqcap$  is associative over  $(\mathcal{L})_{\sqcap}$ .*

*Proof.* Let  $L_1, L_2, L_3 \in (\mathcal{L})_{\sqcap}$  then, from definition, we have

$$u \in (L_1 \sqcap L_2) \sqcap L_3 \iff (\Pi_{\alpha(L_1 \sqcap L_2)}(u) \in L_1 \sqcap L_2) \wedge (\Pi_{\alpha(L_3)}(u) \in L_3)$$

Since  $\alpha(L_1 \sqcap L_2) = \alpha(L_1) \cup \alpha(L_2)$ , we get :

$$u \in (L_1 \sqcap L_2) \sqcap L_3 \iff (\Pi_{\alpha(L_1 \cup L_2)}(u) \in L_1 \sqcap L_2) \wedge (\Pi_{\alpha(L_3)}(u) \in L_3)$$

Moreover  $\Pi_{\alpha(L_i)}(u) = \Pi_{\alpha(L_i)}(\Pi_{\alpha(L_1) \cup \alpha(L_2)}(u))$  for  $i \in \{1, 2\}$ , hence we have :

$$u \in (L_1 \sqcap L_2) \sqcap L_3$$

$$\Updownarrow$$

$$(\Pi_{\alpha(L_1)}(u) \in L_1) \wedge (\Pi_{\alpha(L_2)}(u) \in L_2) \wedge (\Pi_{\alpha(L_3)}(u) \in L_3)$$

and  $\sqcap$  is associative over  $(\mathcal{L})_{\sqcap}$ .  $\square$

**Definition 2.** Let  $L$  be a language. A language  $R$  is said to be  $L$ -compatible if the two following conditions are satisfied :

1.  $\Pi_{\alpha(R)}(L) \subseteq R$
2.  $\alpha(R) \subseteq \alpha(L)$

A family of languages  $\mathcal{L}$  is said to be  $L$ -compatible if every language  $R$  in  $\mathcal{L}$  is  $L$ -compatible.

**Lemma 1.** *Let  $L$  be a language and  $R_1, R_2$  be two languages which are  $L$ -compatible. Then :*

1.  $\alpha(R_1 \sqcap R_2) = \alpha(R_1) \cup \alpha(R_2)$ .
2.  $R_1 \sqcap R_2$  is  $L$ -compatible.

*Proof.* From 1 of definition 2,  $\Pi_{\alpha(R_1) \cup \alpha(R_2)}(L) \subseteq R_1 \sqcap R_2$  and from 2 of definition 2, we also have

$$\alpha(\Pi_{\alpha(R_1) \cup \alpha(R_2)}(L)) = \alpha(R_1) \cup \alpha(R_2)$$

thus  $\alpha(R_1) \cup \alpha(R_2) \subseteq \alpha(R_1 \sqcap R_2)$ . From definition of synchronized shuffle,  $\alpha(R_1 \sqcap R_2) \subseteq \alpha(R_1) \cup \alpha(R_2)$  then  $\alpha(R_1 \sqcap R_2) = \alpha(R_1) \cup \alpha(R_2) \subseteq \alpha(L)$ . It follows that

$$\Pi_{\alpha(R_1 \sqcap R_2)}(L) = \Pi_{\alpha(R_1) \cup \alpha(R_2)}(L) \subseteq R_1 \sqcap R_2$$

hence  $R_1 \sqcap R_2$  is  $L$ -compatible.  $\square$

Then, by induction and from proposition 2 it directly follows :

**Proposition 3.** *Let  $\mathcal{L}$  be a family of languages. If there exists a language  $L$  such that  $\mathcal{L}$  is  $L$ -compatible, then  $(\mathcal{L})_{\sqcap}$  is  $L$ -compatible and  $\sqcap$  is associative over  $(\mathcal{L})_{\sqcap}$ .*

In the following, we shall omit use of parenthesis in expressions involving synchronized shuffle over such families of languages.

### 3 A new characterization of Reg

Here, we shall prove characterization results involving synchronized shuffle. These results concern several families of regular languages. First, let us consider  $\text{Fin}_\varepsilon$ , the family of  $\varepsilon$ -free finite languages.

**Proposition 4.** *The family  $\text{Fin}_\varepsilon$  is the smallest family containing the language  $a + ab$  and closed under union, length preserving morphisms and synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the smallest family containing the language  $a + ab$  and closed under the three above operations. Clearly,  $\mathcal{L} \subseteq \text{Fin}_\varepsilon$ . Conversely, since union is allowed it is sufficient to prove that :

1.  $\emptyset$  is in  $\mathcal{L}$
2. for every alphabet  $X = \{x_1, x_2, \dots, x_n\}, n > 0$ , the language reduced to the single word  $x_1x_2 \dots x_n$  is in  $\mathcal{L}$ .

For 1, we get  $\emptyset = (a + ab) \sqcap (b + ba)$ . For 2, we make an induction on  $n$ . If  $n < 3$ , we get  $a = [(a+ab)+(b+bc)] \sqcap [(a+ac)+(c+cb)]$  and  $ab = (a+ab) \sqcap b$ . If  $n \geq 3$ , it is easily seen that  $x_1x_2 \dots x_n = x_1x_2 \dots x_{n-1} \sqcap x_2 \dots x_n$ .  $\square$

We shall prove a similar result for  $\text{Reg}_*$ , the family of regular star languages. The starting language is  $(a + bc)^*$  and we use only length preserving morphisms and synchronized shuffle. First, we consider monoids generated by special finite languages.

**Definition 3.** A *marked* language  $F$  is a finite language such that :

$$\forall u, v \in F, \forall x \in \alpha(F), (|u|_x > 0 \wedge |v|_x > 0) \implies (u = v \wedge |u|_x = 1)$$

Next, we show that if  $F$  is a marked language, then  $F^*$  can be obtained from languages of the type  $(a + bc)^*$  using synchronized shuffle.

**Definition 4.** Two languages  $L_1$  and  $L_2$  are said equivalent if there is a length preserving morphism which maps bijectively  $L_1$  onto  $L_2$ .

**Lemma 2.** *For every marked language  $F$ ,  $F^*$  can be obtained from languages equivalent to  $(a + bc)^*$  using synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the family of languages which can be obtained by synchronized shuffle from languages equivalent to  $(a + bc)^*$ . The languages  $a^* = (a + bc)^* \sqcap (a + cb)^*$  and  $(ab + cd)^* = (a + cd)^* \sqcap (c + ab)^*$  belong to  $\mathcal{L}$ . Hence,  $(a + b)^* = a^* \sqcap b^*$ ,  $(ab)^* = (ab + cd)^* \sqcap (ab + dc)^*$ ,  $\varepsilon = (ab)^* \sqcap (ba)^*$  are in  $\mathcal{L}$ .

Let us consider now a marked language  $F = u + v$  with  $|v| \geq |u| \geq 0$ . From the above equalities, if  $|v| \leq 2$  then  $F^* \in \mathcal{L}$ . Assume that  $v = x_1x_2 \dots x_k$  with  $k \geq 3$ . The languages  $L_1 = (u + x_1 \dots x_{k-1})^*$ ,  $L_2 = (u + x_2 \dots x_k)^*$  and  $L_3 =$

$(u + x_1 x_k)^*$  are  $F^*$ -compatible. From the equality  $(u + v)^* = L_1 \sqcap L_2 \sqcap L_3$ , we get, by induction, that  $(u + v)^* \in \mathcal{L}$ . At last, if  $F = u_1 + u_2 + \dots + u_n$  is a marked language with  $n \geq 3$ , the languages  $R_1 = (u_1 + \dots + u_{n-1})^*$ ,  $R_2 = (u_2 + \dots + u_n)^*$  and  $R_3 = (u_1 + u_n)^*$  are  $F^*$ -compatible. Once again, the equality  $F^* = R_1 \sqcap R_2 \sqcap R_3$  implies by induction that  $F^* \in \mathcal{L}$ .  $\square$

We can now get easily our characterization result for the family  $\text{Reg}_*$ .

**Proposition 5.** *The family of regular star languages is the smallest family containing the language  $(a + bc)^*$  and closed under length preserving morphisms and synchronized shuffle.*

*Proof.* Clearly,  $\text{Reg}_*$  is closed under the above operations. For the reverse inclusion, let  $R^*$  be a regular language. It is known (see [4], [6]) that there exist two finite languages  $F_1$  and  $F_2$  and a length preserving morphism such that  $R^* = g(F_1^* \cap F_2^*)$ . One can assume that  $\alpha(F_1) = \alpha(F_2)$ . Hence,  $R^* = g(F_1^* \sqcap F_2^*)$ . Moreover,  $F_1$  and  $F_2$  are the image by a length preserving morphism of marked languages. Thus, lemma 2 implies the result.  $\square$

We are now able to state our first characterization of  $\text{Reg}$ .

**Proposition 6.** *The family of regular languages  $\text{Reg}$  is the smallest family containing the languages  $(a+bc)^*$  and  $a$ , closed under union, length preserving morphisms and synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the smallest family containing the languages  $(a + bc)^*$  and  $a$ , closed under the three above operations. Clearly  $\mathcal{L}$  is included in  $\text{Reg}$ . For the reverse inclusion, let us consider a language  $R \in \text{Reg}$ . From proposition 5, we know that  $\varepsilon$  is in  $\mathcal{L}$ , so we may suppose that  $\varepsilon \notin R$  since union is allowed. Hence one may assume, without loss of generality, that  $R \subseteq A^* A'$  where  $A$  and  $A'$  are disjoint alphabets. Then  $R = R^* \sqcap A'$ . From proposition 5,  $R^*$  can be obtained from  $(a + bc)^*$  using length preserving morphisms and synchronized shuffle. On the other hand,  $A'$  can be obtained from  $a$  using union and length preserving morphisms.  $\square$

We can observe that it is not possible to enunciate a similar result with a single *generator*. Indeed, since  $\varepsilon \in L_1 \sqcap L_2$  if and only if  $\varepsilon \in L_1 \cap L_2$ , we need to start from a family containing at least two languages  $L_1$  and  $L_2$  with  $\varepsilon \in L_1$  and  $\varepsilon \notin L_2$ . The following result concerns the family of  $\varepsilon$ -free regular languages, that is regular languages which do not contain the empty word. For this family, it is possible to start from a single generator.

**Proposition 7.** *The family of regular  $\varepsilon$ -free languages  $\text{Reg}_\varepsilon$  is the smallest family containing the language  $(a + bc)^* d$  and closed under union, length preserving morphisms and synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the smallest family containing the language  $(a + bc)^*d$  and closed under the three above operations. Clearly  $\mathcal{L}$  is included in  $\text{Reg}_\varepsilon$ . For the reverse inclusion, let us consider a language  $R \in \text{Reg}_\varepsilon$ . Without loss of generality, one may assume that  $R$  is a finite union of languages in the form  $Kd$  with  $K \in \text{Reg}$  and  $d$  a letter not in  $\alpha(K)$ . It remains to prove that such a language  $Kd$  is in  $\mathcal{L}$ . This can be done by induction over the construction of  $K$  with respect to proposition 6.

If  $K = (a + bc)^*$  then  $Kd$  is in  $\mathcal{L}$ . If  $K = a$ , observe first that  $d = (a + ab)^*d \sqcap (b + ba)^*d$  is in  $\mathcal{L}$ . Moreover  $a^*d$ , which can be obtained from  $(a + bc)^*d$  using a length preserving morphism, is in  $\mathcal{L}$ . Then we get  $ad = a \sqcap a^*d$  belongs to  $\mathcal{L}$ .

Now,

- if  $K = h(K')$  for some  $K' \in \mathcal{L}$  and some length preserving morphism  $h$ , we may suppose that  $d \notin \alpha(K')$  then  $K'd \in \mathcal{L}$  which implies  $Kd \in \mathcal{L}$ .
- if  $K = K_1 + K_2$  with  $K_i \in \mathcal{L}$  for  $i = 1, 2$  then  $Kd = K_1d + K_2d \in \mathcal{L}$ .
- if  $K = K_1 \sqcap K_2$  with  $K_i \in \mathcal{L}$  for  $i = 1, 2$ . We may suppose that  $d \notin \alpha(K_i)$  for  $i = 1, 2$  then  $Kd = K_1d \sqcap K_2d \in \mathcal{L}$ .

□

## 4 Binary generators

The single generator for the family of  $\varepsilon$ -free regular languages  $\text{Reg}_\varepsilon$  used in proposition 7 is built over a four letter alphabet. The following proposition shows that it is possible to start from a language defined over a three letter alphabet :

**Proposition 8.** *The family of regular  $\varepsilon$ -free languages  $\text{Reg}_\varepsilon$  is the smallest family containing the language  $(a + bc)^*b$  and closed under union, length preserving morphisms and synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the smallest family containing the language  $(a + bc)^*b$  and closed under union, length preserving morphisms and synchronized shuffle. From proposition 7, we have to prove that  $(a + bc)^*d$  is in  $\mathcal{L}$ . Observe first that  $b = (a + bc)^*b \sqcap (c + ba)^*b$  is in  $\mathcal{L}$ . Then  $a^*b = (a + bc)^*b \sqcap b$  and  $(a + bc)^*bd = (a + bc)^*b \sqcap b^*d$  are in  $\mathcal{L}$ . It follows that  $(a + bc)^*bc$  is also in  $\mathcal{L}$ . Let us consider now the length preserving morphism  $h$  defined by :  $h(a) = a, h(b) = h(d) = b$  and  $h(c) = c$ . We get

$$(a + bc)^*bc \sqcap h((a + cb)^*c \sqcap d) = a^*(bc)^+ \in \mathcal{L}$$

then  $a^*(bb)^+ \in \mathcal{L}$  and  $a^*b^+ = a^*b + a^*(bb)^+ + h(a^*(bb)^+ \sqcap a^*d)$  is in  $\mathcal{L}$ . Moreover  $a^+b^+ = a^+ + (a^*b^+ \sqcap a^+)$  is also in  $\mathcal{L}$  since, clearly,  $a^+ \in \mathcal{L}$ . Now, with the length preserving morphism  $g$  defined by :  $g(a) = g(e) = a, g(b) = b, g(c) = c$  and  $g(d) = d$ , we obtain

$$(a + bc)^*bca^*d = g((a + bc)^*bc \sqcap c^+e^* \sqcap e^*d \sqcap c^*d) \in \mathcal{L}$$

At last, we get  $(a + bc)^*d = a^*d + (a + bc)^*bca^*d \in \mathcal{L}$ .  $\square$

We have seen that the family of ( $\varepsilon$ -free) regular languages can be obtained from languages whose cardinality alphabet is less or equal to three. A natural question is whether it is possible to start from binary languages which are languages built over two letter alphabets. In a first time, we shall establish a negative answer for the family of star regular languages.

**Definition 5.** A star language  $L$  satisfies the ( $P$ ) property if there exist three distinct letters  $a, b, c$  such that the words  $a$  and  $bc$  are in  $L$ , but the word  $bac$  is not in  $L$ .

**Lemma 3.** *Let  $S$  be a set of star languages and  $\mathcal{L}$  be the smallest family of languages, containing  $S$  and closed under length preserving morphism and synchronized shuffle. If  $\mathcal{L}$  contains a language which satisfies the ( $P$ ) property then so do  $S$ .*

*Proof.* First, it is clear that  $\mathcal{L}$  contains only star languages. We shall prove this lemma by induction. Let  $L$  be a language in  $\mathcal{L}$  with  $a, bc \in L$  and  $bac \notin L$ .

If there exist some language  $L' \in \mathcal{L}$  and some length preserving morphism  $h$  such that  $L = h(L')$  then it is obvious that  $L'$  satisfies the ( $P$ ) property.

Let us suppose now that  $L = L_1 \sqcap L_2$  such that neither  $L_1$  nor  $L_2$  satisfies ( $P$ ). We shall show that it will lead to the following contradiction  $bac \in L$ . For  $i = 1, 2$ , we denote  $\alpha_i = \alpha(L_i) \cap \{a, b, c\}$ . Observe that  $bac \in L$  if and only if  $\Pi_{\alpha_i}(bac) \in L_i$  for  $i = 1, 2$ . Moreover, since  $L$  is a star language, the words  $abc$  and  $bca$  are in  $L$ . Now, for  $i = 1, 2$  :

- if  $\alpha_i \not\subseteq \{a, b, c\}$ , we get  $\Pi_{\alpha_i}(bac) \in \Pi_{\alpha_i}(abc + bca) \subseteq L_i$
- if  $\alpha_i = \{a, b, c\}$ , then  $a \in L_i$  and  $bc \in L_i$ . Since  $L_i$  does not satisfy ( $P$ ), it follows that  $\Pi_{\alpha_i}(bac) = bac \in L_i$ .

The contradiction  $bac \in L$  implies that  $L_1$  satisfies ( $P$ ) or  $L_2$  satisfies ( $P$ ).  $\square$

Since the language  $(a + bc)^*$  satisfies the ( $P$ ) property, we can state, as a corollary of the above lemma :

**Proposition 9.** *The family of regular star languages can not be obtained as the closure of a set of binary languages under length preserving morphism and synchronized shuffle.*

For the family of ( $\varepsilon$ -free) regular languages, the answer is positive. In order to establish this result, we shall first prove the following lemma :

**Lemma 4.** *Let  $\mathcal{L}$  be a family of languages containing the language  $a^*b^*$  and closed under length preserving morphism and synchronized shuffle, then  $\mathcal{L}$  is closed under product.*

*Proof.* Let  $L_1$  and  $L_2$  be two languages of  $\mathcal{L}$ . We may suppose that  $\alpha_1 = \alpha(L_1)$  and  $\alpha_2 = \alpha(L_2)$  are disjoint. The family  $\{x^*y^* \mid x \in \alpha_1, y \in \alpha_2\}$  is clearly  $L_1L_2$ -compatible, then synchronized shuffle is associative over this family and

$$\bigsqcup_{\substack{x \in \alpha_1 \\ y \in \alpha_2}} x^*y^* = \alpha_1^*\alpha_2^* \in \mathcal{L}$$

Then  $L_1L_2 = (L_1 \sqcap L_2) \sqcap \alpha_1^*\alpha_2^*$  is in  $\mathcal{L}$ .  $\square$

We are now able to enunciate our last proposition which states that Reg, the family of regular languages can be obtained from binary generators.

**Proposition 10.** *The family of regular languages Reg is the smallest family containing the languages  $a + (ab)^+$  and  $(a + bb)^*$ , and closed under union, length preserving morphism and synchronized shuffle.*

*Proof.* Let  $\mathcal{L}$  be the smallest family containing the languages  $a + (ab)^+$  and  $(a + bb)^*$ , closed under the three above operations. Clearly  $\mathcal{L}$  is included in Reg.

For the reverse inclusion, observe first that

$$(a + bb)^* \sqcap (a + (ab)^+) = a \in \mathcal{L}$$

It remains to prove that  $(a + bc)^* \in \mathcal{L}$ . Let  $h$  be the length preserving morphism defined by  $h(a) = h(b) = a$ , we get  $h((a + bb)^*) = a^* \in \mathcal{L}$  and  $h(a^* \sqcap b) = a^+ \in \mathcal{L}$ . It follows that  $(a + (ab)^+) \sqcap b = ab \in \mathcal{L}$  and  $(a + (ab)^+) \sqcap b^+ = (ab)^+ \in \mathcal{L}$ .

Now, if  $g$  is the length preserving morphism defined from by  $g(a) = g(d) = g(e) = a$ ,  $g(b) = b$  and  $g(c) = c$ , we get

$$g((ac)^+ \sqcap bd) \sqcap g((ca)^+ \sqcap be) = b(ac)^+a \in \mathcal{L}$$

Then  $b(aa)^+a$  in  $\mathcal{L}$ ,  $g(b(aa)^+a \sqcap bd) = b(aa)^+aa \in \mathcal{L}$  and  $g(ba \sqcap ad) = baa \in \mathcal{L}$  so :

$$ba^* = b + ba + baa + b(aa)^+a + b(aa)^+aa \in \mathcal{L}$$

In a same way we can prove  $a^*b \in \mathcal{L}$ . It follows that  $a^*b \sqcap bc^* = a^*bc^*$  is in  $\mathcal{L}$  then  $a^*b^+$  is in  $\mathcal{L}$ . Now, since  $a^*b^+ + a^* = a^*b^* \in \mathcal{L}$  and from lemma 4, we get that  $\mathcal{L}$  is closed under product.

We are now able to obtain  $(a + bc)^*$  :

$$(((a + bb)^* \sqcap (a + cc)^*) \sqcap (bc)^+) + a^* = (a + bcbc)^* \in \mathcal{L}$$

The family  $\mathcal{L}$  is closed under product then we also have  $bc(a + bcbc)^*bc \in \mathcal{L}$ . Then

$$L_1 = (a + bcbc)^* \sqcap bc(d + bcbc)^*bc = a^*bc(d^*bca^*bc)^*d^*bca^* \in \mathcal{L}$$

and, since product is allowed  $L_2 = a^*bcL_1 \in \mathcal{L}$ . We finally get that

$$g(L_1) + g(L_2) + a^*bca^* + a^* = (a + bc)^* \in \mathcal{L}$$

□

## References

1. Culik II K., Fich F.E. and Salomaa A. (1982) A homomorphic characterization of regular languages. *Discrete Applied Mathematics* **4**, 149–152
2. De Simone R. (1984) Langages infinitaires et produit de mixage. *Theoretical Computer Science* **31**, 83–100
3. Duboc C. (1986) Commutation dans les monoïdes libres : Un cadre thorique pour l'étude du parallisme. Thèse de doctorat. Université de Rouen.
4. Karhumaki J., Linna M. (1983) A note on morphic characterization of languages. *Discrete Applied Mathematics* **5**, 243–246
5. Kimura T. (1976) An algebraic system for process structuring and interprocess communication. 8th ACM SIGACTS Symposium on Theory of Computing. 92–100
6. Latteux M., Leguy J. (1983) On the composition of morphisms and inverse morphisms. *Lecture Notes in Computer Science* **154**, 420–432
7. Mateescu A., Rozenberg G. and Salomaa A. (1998) Shuffle on Trajectories : Syntactic Constraints. *Theoretical Computer Science, TCS, Fundamental Study*, **197**, 1-2, 1–56
8. Ryl I. (1998) Langages de synchronisation. Thèse de doctorat. Université de Lille 1.
9. Turakainen P. (1982) A homomorphic characterization of principal semi-AFLs without using intersection with regular sets. *Inform. Sci.* **27**, 141–149