Available online at www.sciencedirect.com

SCIENCE DIRECT

Discrete Applied Mathematics 154 (2006) 452–470

DISCRETE APPLIED MATHEMATICS

www.elsevier.com/locate/dam

Linear splicing and syntactic monoid☆

P. Bonizzoni[a], C. De Felice[b,*], G. Mauri[a], R. Zizza[b]

[a] *Dipartimento di Informatica Sistemistica e Comunicazione, Università degli Studi di Milano Bicocca, Via Bicocca degli Arcimboldi 8, 20126 Milano, Italy*

[b] *Dipartimento di Informatica e Applicazioni, Università di Salerno, 84081 Baronissi (SA), Italy*

Received 5 September 2003; received in revised form 28 April 2005; accepted 20 June 2005

Available online 26 September 2005

Abstract

Splicing systems were introduced by Head in 1987 as a formal counterpart of a biological mechanism of DNA recombination under the action of restriction and ligase enzymes. Despite the intensive studies on *linear* splicing systems, some elementary questions about their computational power are still open. In particular, in this paper we face the problem of characterizing the proper subclass of regular languages which are generated by *finite* (*Paun*) *linear splicing systems*. We introduce here the class of *marker languages* L, i.e., regular languages with the form $L = L_1[x]_1 L_2$, where L_1, L_2 are regular languages, $[x]$ is a syntactic congruence class satisfying special conditions and $[x]_1$ is either equal to $[x]$ or equal to $[x] \cup \{1\}$, 1 being the empty word. Using classical properties of formal language theory, we give an algorithm which allows us to decide whether a regular language is a marker language. Furthermore, for each marker language L we exhibit a finite Paun linear splicing system and we prove that this system generates L.

Keywords: Automata; Regular languages; Molecular computing

1. Introduction

Linear splicing systems are generative devices of formal languages, introduced by Head in 1987 to model the recombinant behaviour of DNA molecules under the action of restriction and ligase enzymes (*splicing operation*). Two strands of DNA are cut at specified substrings (sites) by restriction enzymes that recognize a pattern inside the molecule and then the fragments are pasted by ligase enzymes. In particular, Head was concerned with the structure of the languages of those DNA molecules (strings) which could be produced through the splicing operation, performed by a splicing system consisting of a finite set of initial DNA molecules (initial language) and a finite set of enzymes (set of rules). He showed that under some conditions, the generated language is strictly locally testable [12]. As a result of his pioneering work, several variants of the splicing operation were proposed, some of which by Paun and Pixton. Moreover, different variants of splicing systems were introduced together with the more general model of a splicing system which has an infinite set of initial strings and an infinite set of rules (see [19] for a survey).

☆ Partially supported by MIUR Project "*Linguaggi Formali e Automi: Metodi, Modelli e Applicazioni*" (2003), by the contribution of EU Commission under The Fifth Framework Programme (*project MolCoNet* IST-2001-32008) and by 60% Project "*Linguaggi formali e codici: modelli e caratterizzazioni strutturali*" (University of Salerno, 2004).

* Corresponding author.

E-mail addresses: bonizzoni@disco.unimib.it (P. Bonizzoni), defelice@dia.unisa.it (C. De Felice), mauri@disco.unimib.it (G. Mauri), zizza@dia.unisa.it (R. Zizza).

0166-218X/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2005.06.008

The computational power of splicing systems, recently called H systems, i.e., triples $H = (A, I, R)$ where A is a finite alphabet, I is the initial language over A, and R is the set of rules, has been investigated and this computational power also depends on which level in Chomsky hierarchy I, R belong. In particular, these systems can be as powerful as the Turing machines [14,19]. At the lowest level of the hierarchy, the regularity of splicing languages generated by regular initial languages and a finite set of rules is proved in [19,22]. On the other hand, it is known that *finite* linear splicing systems, i.e., H systems with both I, R finite sets, generate languages which belong to a proper subset of the family of regular languages, but the complete structure of these regular languages is still unknown [6,14,16]. This is the problem we deal with in this paper and definitions and results from classical formal language theory are used in order to obtain our results. In particular, we will consider the notion of a *constant* for a regular language L, given by Schützenberger in [23] and the notion of *syntactic congruence* for a regular language L (see Sections 2.2 and 8 for the definitions). These notions seem to play an important role in the description of the structure of the above-mentioned class of regular languages.

To be more precise, we say that $[x]$ is a *marker* if $[x]$ is a syntactic congruence class satisfying some conditions. Specifically, in the transition diagram of the minimal finite state automaton $\mathscr{A}$ recognizing L, we find only a path π with label x. We also suppose that either x is the label of a closed path or $\{x' \in A^* \mid x' \equiv_L x\}$ is a finite set, where $\equiv_L$ denotes the syntactic congruence of L. Under these hypotheses, we set $[x]_1 = [x] \cup \{1\}$ in the former case and $[x]_1 = [x]$ in the latter case, where 1 is the empty word. In this paper we will consider *marker languages* L, i.e., regular languages with the form $L = L_1[x]_1 L_2$, where L_1, L_2 are regular languages and $[x]$ is a marker. We show that we can generate each marker language L through a finite linear Paun splicing system which is defined by means of some particular closed paths (*cycles*) in the minimal finite state automaton recognizing L (Definition 6.3, Theorem 6.1).

A special case is worthy of note. If $[x]$ is a marker, we can show that for each $x' \in [x]$, x' is a constant for L. Now, when $\{x' \in A^* \mid x' \equiv_L x\}$ is a finite set, the marker language $L = L_1[x]_1 L_2 = L_1[x]L_2$ belongs to a family of languages proved to be generated by finite linear Paun splicing systems in [13] by Head. As a matter of fact, L is a finite union of languages named *constant languages* in [3]. Therefore, it is natural to compare these two classes of languages—marker languages and constant languages—with respect to the order of set inclusion. In connection with this problem, we show that constant languages exist which are not marker languages and marker languages exist which are not constant languages (see Propositions 8.2, 8.3).

Concepts, techniques and statements proved in this paper also intervene in a recent result, which characterizes *reflexive* languages (i.e., languages generated by finite *reflexive* Paun linear splicing systems), a class which properly contains the family of languages generated by finite Head splicing systems [3,5]. Once again this characterization, given in [3,5], is obtained via constants (see also [9,10] for other results on reflexive languages).

Decision problems also arise in a natural way. We recall that deciding whether a regular language is a constant language is a question which was first asked in [13]. In [10] the authors have given an answer to this question, along with a solution to the analogous question for reflexive languages. In this framework, we design an algorithm which shows that it is decidable whether a regular language is a marker language (Proposition 5.3).

This paper is organized as follows. Section 2 contains some basics on words and finite automata together with all the necessary preliminary definitions on the splicing operation. The other sections are devoted to marker languages. Precisely, in Section 3 we give an outline of the results. Section 4 contains some definitions and properties of particular closed paths in a graph (cycles) which will be used in the proofs of our results. In Sections 6–7 we prove that marker languages, defined in Section 5, are generated by special finite linear Paun splicing systems which are constructed in Section 6.1. Finally, in Section 8 we discuss the notion of marker and the relationship between constant languages and marker languages. An extended abstract of most of the results contained in this paper has been presented at DLT02 [2].

2. Basics

2.1. Words and finite automata

We denote by A^* the free monoid over a finite alphabet A with the *concatenation* (or *product*) operation, and we set $A^+ = A^* \setminus 1$, where 1 is the empty word. For a word $w \in A^*$, $|w|$ is the length of w and, for a subset L of A^*, we denote by $|L|$ the cardinality of L. A word $x \in A^*$ is a *factor* of $w \in A^*$ if $u_1, u_2 \in A^*$ exist such that $w = u_1 x u_2$; x is a *prefix* (resp. *suffix*) of $w \in A^*$ if $u_1 = 1$ (resp. $u_2 = 1$); x is a *proper prefix* (resp. *proper suffix*) of $w \in A^*$ if $u_2 \neq 1 = u_1$

(resp. $u_1 \neq 1 = u_2$). We denote $Fact(L) = \{w \in A^* \mid A^*wA^* \cap L \neq \emptyset\}$ the set of all the factors of the elements of $L \subseteq A^*$ and we set $Fact(\{w\}) = Fact(w)$. We also recall that the *reverse* w^R of $w \in A^*$ is defined as follows: $1^R = 1$ and, for all $x \in A^*$, $a \in A$, $(xa)^R = ax^R$. Then, for a language $L \subseteq A^*$, $L^R = \{l^R \mid l \in L\}$ is the reverse of L.

We refer the reader to [8,11,15,20] for the notations and results which will be used in the next part of this section. In the following $\mathscr{A} = (Q, A, \delta, q_0, F)$ will be a finite state automaton, where Q is a finite set of states, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of final states. The transition function δ is defined in the classical way. A finite state automaton $\mathscr{A}$ is *deterministic* if, for each $q \in Q, a \in A$, there exists at most one state $p \in Q$ such that $\delta(q, a) = p$. The triple (q, a, p) is called an *edge* of the automaton $\mathscr{A}$. Furthermore, $\mathscr{A}$ is *trim* if each state is accessible and coaccessible, i.e., if for each state $q \in Q$ there exist $x, y \in A^*$ such that $\delta(q_0, x) = q$ and $\delta(q, y) \in F$. As usual, in the transition diagram of a trim deterministic automaton $\mathscr{A}$, each final state will be indicated by a double circle and the initial state will be indicated by an arrow without a label going into it. In this paper, each automaton will be supposed to be deterministic and trim. A *path* π in an automaton $\mathscr{A}$ is a finite sequence $\pi = (q_1, a_1, q_2)(q_2, a_2, q_3) \dots (q_n, a_n, q_{n+1})$ of consecutive edges, i.e., for each $i = 1, \dots, n$ we have $\delta(q_i, a_i) = q_{i+1}$. An abbreviated notation for a path is $\pi = (q_1, a_1a_2 \cdots a_n, q_{n+1})$ and $a_1a_2 \dots a_n$ is called the *label* of π. Moreover, we say that $q_1, \dots, q_{n+1}$ are the *states crossed* by the path $(q_1, a_1 \cdots a_n, q_{n+1})$ and, for each $i \in \{2, \dots, n\}$, q_i is an *internal state crossed* by the same path. For each state q, we consider also the *null path* 1_q with label 1. A path $\pi = (q, x, p)$ is a *closed path* if $\pi \neq 1_q$ and $q = p$.

If not stated, $L \subseteq A^*$ will always be a regular language, at times represented by means of a regular expression. Well known results state that, if L is regular then L^R is regular and, for any set M, $M^{-1}L = \{y \in A^* \mid xy \in L, x \in M\}$, $LM^{-1} = \{y \in A^* \mid yx \in L, x \in M\}$ are also regular [11]. Given a regular language $L \subseteq A^*$, it is also well known that there exists a *minimal* finite state automaton $\mathscr{A}$ recognizing it, i.e., such that $L = L(\mathscr{A})$. This canonical automaton is determined uniquely up to a renaming of the states and (when we make it trim) it has the minimal number of states. It is related to the syntactic monoid of L (the definition of which will be recalled in Section 2.2) and can be obtained by standard construction algorithms. One of these algorithms uses the property reported in Proposition 2.1 [1,15]. Given a deterministic finite state automaton $\mathscr{A} = (Q, A, \delta, q_0, F)$, for all $q \in Q$, we denote $L_q = \{w \in A^* \mid \delta(q, w) \in F\}$.

Proposition 2.1. *Let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be a trim deterministic finite state automaton recognizing the regular language $L = L(\mathscr{A})$. Then, $\mathscr{A}$ is minimal if and only if, for every $q, p \in Q$, we have $L_q \neq L_p$.*

Unless otherwise stated, for a regular language $L \subseteq A^*$, we will always refer to the minimal finite state automaton $\mathscr{A}$ recognizing L. In particular, $\mathscr{A}$ will be deterministic and trim.

2.2. Syntactic monoid

The investigation of a regular language $L \subseteq A^*$ has been thoroughly developed by using the algebraic theory of finite monoids via the so-called *syntactic monoid* associated with L [20]. This is the quotient monoid $\mathscr{M}(L)$ with respect to the *syntactic congruence* $\equiv_L$, defined as follows: two words w, w' are equivalent with respect to the syntactic congruence if they have the same set of contexts, i.e.,

$$w \equiv_L w' \quad \Leftrightarrow \quad [\forall x, y \in A^*, xwy \in L \quad \Leftrightarrow \quad xw'y \in L] \Leftrightarrow C(w) = C(w').$$

Here, we denote $C(w) = \{(x, y) \in A^* \times A^* \mid xwy \in L\}$ the set of *contexts* $C(w)$ of $w \in A^*$ for $L \subseteq A^*$ and $[w] = \{w' \in A^* \mid w' \equiv_L w\}$ the class of w modulo $\equiv_L$. If L is a regular language then the index (i.e., the number of congruence classes) of the syntactic congruence is finite and so $\mathscr{M}(L)$ is a finite monoid. Theorem 2.1, proved in [17], will be widely used in the next part of this paper.

Theorem 2.1. *Let L be a regular language and let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal automaton recognizing L. We have $w \equiv_L w'$ if and only if for all $q \in Q$ we have $\delta(q, w) = \delta(q, w')$.*

Let us introduce some notations which will be used in the following sections. We set $Q_w(\mathscr{A}) = \{q \in Q \mid \delta(q, w)$ *is defined*$\}$, simply denoted Q_w when the context makes the meaning evident. Furthermore the observation below defines the notation $Q_{[w]}$ with $w \in A^*$.

Remark 2.1. If $w \equiv_L w'$ then $Q_w = Q_{w'}$. By contradiction, suppose that there exists $q \in Q_w \setminus Q_{w'}$ and let $\delta(q, w) = p$. Since q is accessible and p is coaccessible, then there exist $x, y \in A^*$ such that $\delta(q_0, x) = q$, $\delta(p, y) \in F$. Thus, $(x, y) \in C(w)$, but $(x, y) \notin C(w')$, since $\delta(q_0, x) = q$ and $\delta(q, w')$ is not defined. In conclusion, $Q_w \subseteq Q_{w'}$. Similarly, $Q_{w'} \subseteq Q_w$ and so $Q_w = Q_{w'}$.

Finally, a word $w \in A^*$ can be a label of zero, one path or more in $\mathscr{A}$, but since $\mathscr{A}$ is deterministic, for a given $q \in Q_w$, we have that w is the label of a unique path in $\mathscr{A}$ starting from q.

2.3. Linear splicing

In Section 3 we present the main result of this paper, namely we describe a new class of languages which can be generated by a finite linear splicing system. As a matter of fact, there are three definitions of the linear splicing operation given by Head, Paun and Pixton, respectively [14]. We will only consider the second definition, also reported in [18].

`Paun's definition`. Given a finite alphabet A and two special symbols $|, \$ \notin A$, a *splicing rule* (over A) is denoted $r = u_1 \,|\, u_2 \,\$\, u_3 \,|\, u_4$, $(u_i \in A^*, i = 1, 2, 3, 4)$. For such a rule r we write $(x, y) \vdash_r (w', w'')$ if and only if $x_1, x_2, y_1, y_2 \in A^*$ exist such that $x = x_1u_1u_2x_2$, $y = y_1u_3u_4y_2$, $w' = x_1u_1u_4y_2$ and $w'' = y_1u_3u_2x_2$. A *splicing system* $S_{PA} = (A, I, R)$ consists of a finite alphabet A, an initial language $I \subseteq A^*$ and a set R of rules. Furthermore, given a language $L \subseteq A^*$, we denote by $\sigma'(L) = \{w', w'' \in A^* \,|\, \exists x, y \in L, \exists r \in R : (x, y) \vdash_r (w', w'')\}$. Then, we define $\sigma^0(L) = L$, $\sigma^{i+1}(L) = \sigma^i(L) \cup \sigma'(\sigma^i(L))$, $i \geqslant 0$, and then $\sigma^*(L) = \bigcup_{i \geqslant 0} \sigma^i(L)$.

Definition 2.1. Given a splicing system $S_{PA} = (A, I, R)$, the language $L(S_{PA}) = \sigma^*(I)$ is the *language generated* by the system S_{PA}. A language L is S_{PA} *generated* (or L is a *Paun splicing language*) if a splicing system S_{PA} exists such that $L = L(S_{PA})$.

Given a splicing system $S_{PA} = (A, I, R)$, let us consider a rule $r = u_1 \,|\, u_2 \,\$\, u_3 \,|\, u_4 \in R$. Clearly r may be considered a word, so the reverse $r^R = u_4^R \,|\, u_3^R \,\$\, u_2^R \,|\, u_1^R$ of r may also be defined. Thus, we can define the splicing system $S_{PA}^R = (A, I^R, R^R)$ where $R^R = \{r^R \,|\, r \in R\}$. Proposition 2.2 is more or less folklore.

Proposition 2.2. *Let $L \subseteq A^*$ be a language, let $S_{PA} = (A, I, R)$ be a splicing system. We have $(L(S_{PA}))^R = L(S_{PA}^R)$. In addition, if $L \subseteq L(S_{PA})$ then $L^R \subseteq L(S_{PA}^R)$.*

Proof. Given a splicing system $S_{PA} = (A, I, R)$ and a language $L \subseteq A^*$, we denote by $\sigma'_R(L)$ (resp. $\sigma_R(L)$) the function σ' (resp. σ) when we refer to S_{PA}^R, i.e., $\sigma'_R(L) = \{w', w'' \in A^* \,|\, \exists x, y \in L, \exists r^R \in R^R : (x, y) \vdash_{r^R} (w', w'')\}$. Furthermore, we have $\sigma_R^0(L) = L$, $\sigma_R^{i+1}(L) = \sigma_R^i(L) \cup \sigma'_R(\sigma_R^i(L))$, $i \geqslant 0$.

It is clear that $\sigma_R^0(I^R) = I^R = (\sigma^0(I))^R$. Furthermore, given $x, y \in L$, $r \in R$, $w', w'' \in A^*$, we obviously have $(x, y) \vdash_r (w', w'')$ if and only if $(x^R, y^R) \vdash_{r^R} ((w')^R, (w'')^R)$. This observation shows that $\sigma'_R(L^R) = (\sigma'(L))^R$. Then, looking at Definition 2.1, we have $(L(S_{PA}))^R = L(S_{PA}^R)$.

Suppose that $L \subseteq L(S_{PA})$. Then, for each $w^R \in L^R$, we have either $w \in I$ or there exist $x, y \in L(S_{PA})$, $r \in R$ and $w' \in A^*$ such that $(x, y) \vdash_r (w, w')$ (the other case $(x, y) \vdash_r (w', w)$ is analogous). Consequently, either $w^R \in I^R$ or there is a rule $r^R \in R^R$, $x^R, y^R \in (L(S_{PA}))^R = L(S_{PA}^R)$ and $(w')^R \in A^*$ such that $(x^R, y^R) \vdash_{r^R} (w^R, (w')^R)$, i.e., $w^R \in L(S_{PA}^R)$. □

In the next part of this paper we deal only with finite linear splicing systems, i.e., with both $I, R \in Fin$, where Fin is the class of finite languages. The corresponding class of generated languages will be denoted $S_{PA}(Fin, Fin)$.

3. Outline of the results

As already said, the main result of this paper is the construction of a family of languages generated by finite Paun linear splicing systems and called *marker languages* (Definition 5.4). These languages are defined starting with a regular

language $L = L(\mathscr{A})$ and a particular syntactic congruence class $[x]$ with respect to L. To be more specific, we give the definition of a *marker* for L (Definition 5.3). Intuitively, given $x \in A^*$, $[x]$ is a marker for L if, in the transition diagram of the minimal finite state automaton $\mathscr{A}$ recognizing L, there is only one path π with label x. Furthermore, either $[x]$ is a finite set or x is the label of a closed path in $\mathscr{A}$. Under these hypotheses, we set $[x]_1 = [x]$ in the former case and $[x]_1 = [x] \cup \{1\}$ in the latter case. A marker language L will have the form $L_1[x]_1L_2$, where L_1, L_2 are regular languages and $[x]$ is a marker.

We give an algorithm which shows that we can decide whether a regular language is a marker language. Furthermore, for each marker language L we exhibit a finite linear splicing system (Section 6.1) and we show that this system generates L (Theorem 6.1). Propositions 8.2 and 8.3 prove that the class of marker languages is different from the class of constant languages introduced in [13] and mentioned in Section 1.

4. Cycles

Let L be a regular language and let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L. Intuitively, if a regular infinite language L is generated by a finite linear splicing system, we must exhibit a finite set of rules which are able to produce words with a non-bounded number of occurrences of labels of closed paths as factors. This is the reason for which in [4] the authors give a definition for referring to some of these labels (*cycles*) in $\mathscr{A}$.

The result which follows, also reported in [5], can be easily proved and will be used in Section 6 in the proofs of Proposition 6.2 and Lemma 6.1.

Lemma 4.1. *Let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be a deterministic finite state automaton. For each $w \in A^+$, $q, p \in Q$ such that $\delta(q, w) = p$ there exist a positive integer k, $u_1, \ldots, u_{k+1}, d_1, \ldots, d_k \in A^*$, $p_0, p_1, \ldots, p_k, p_{k+1} \in Q$ such that*

- $p_0 = q$, $p_{k+1} = p$,
- $w = u_1d_1u_2 \cdots u_kd_ku_{k+1}$,
- $\delta(p_i, d_i) = p_i$, $1 \leqslant i \leqslant k$, $\delta(p_{j-1}, u_j) = p_j$, $1 \leqslant j \leqslant k+1$.

Furthermore, $w' = u_1u_2 \cdots u_ku_{k+1}$ satisfies the conditions that follow.

(a) $\delta(q, w') = p$,
(b) *each state crossed by the path (q, w', p) is also crossed by the path (q, w, p),*
(c) *the internal states crossed by the path (q, w', p) are different from one another and with respect to q, p,*
(d) *either $w = w' = u_1u_2$, $k = 1$ or $k \geqslant 1$, $d_1, \ldots, d_k \in A^+$ and, if $k \geqslant 2$, $u_2, \ldots, u_k \in A^+$.*

As we have already said, since $\mathscr{A}$ is deterministic, a path $\pi = (q, c, p)$ is uniquely defined by the pair (q, c).

Definition 4.1 (*q-label*). Let $q \in Q$. A word $c \in A^+$ is a *q-label* in $\mathscr{A}$ (or simply a q-label, if $\mathscr{A}$ is understood) if c is the label of some closed path $\pi = (q, c, q)$ in $\mathscr{A}$. A q-label c is *elementary* if the internal states crossed by the path (q, c, q) are different from q, otherwise it is *non-elementary*.

It is obvious that, for a non-elementary q-label c, there exist (not necessarily different) elementary q-labels $c_1, \ldots, c_n$, $n > 1$, such that $c = c_1 \cdots c_n$. A cycle is a special elementary q-label, its definition, given in [4] and reported below, is mutually recursive with the definition of a *semi-cycle*.

Definition 4.2 (*Simple cycle*). The word $c \in A^+$ is a *simple cycle on q* if c is an elementary q-label and the internal states crossed by the path (q, c, q) are different from one another.

(*Semi-cycle*) The word $c \in A^+$ is a *semi-cycle on q* if there exist (not necessarily different) cycles $c_1, \ldots, c_n$ on q such that $c = c_1 \cdots c_n$ and, if $c_i = c_j$ for $1 \leqslant i < j \leqslant n$, then $c_i = c_k$ for $k \in \{i, i+1, \ldots, j\}$. If $n > 1$ then c is a *proper semi-cycle on q*.

(*Cycle*) The word $c \in A^+$ is a *cycle on q* if c is an elementary q-label such that either c is a simple cycle on q or there exist $u_1, u_2, \ldots, u_k, u_{k+1}, s_1, \ldots, s_k \in A^+$ such that $c = u_1s_1u_2 \cdots u_ks_ku_{k+1}$, with $k \geqslant 1$, $u_1u_2 \cdots u_ku_{k+1}$ being a simple cycle on q, s_j being a semi-cycle on $p_j = \delta(p_{j-1}, u_j)$, $1 \leqslant j \leqslant k$ and $p_0 = q$.

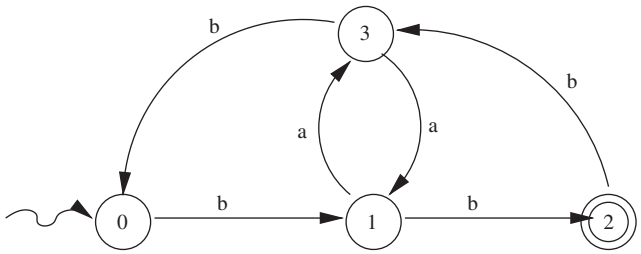

Fig. 1. Automaton $\mathscr{A}_1$.

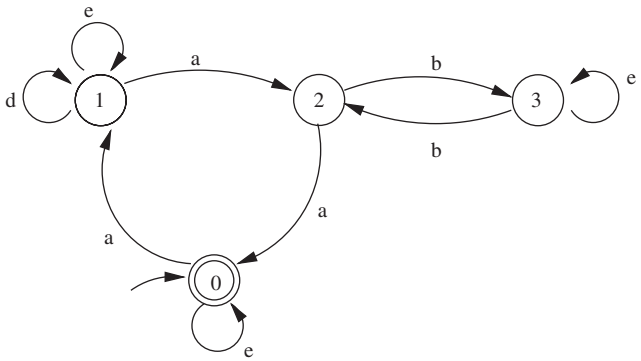

Fig. 2. Automaton $\mathscr{A}_2$.

A cycle c on q is simply named cycle when the context makes the meaning evident.

Example 4.1. The word $c = ba^2b^2a^2b$ is a (non-simple) cycle on state 0 in $\mathscr{A}_1$ (Fig. 1). Indeed, $c = bs_1b^2s_2b$, where bb^2b is a simple cycle on 0, $s_1 = a^2$ is a simple cycle on 1 and $s_2 = a^2$ is a simple cycle on 3. Notice that we can also state that c is a cycle on 0 by observing that $c = basb$, where bab is a simple cycle on 0 and $s = abbaa$ is a semi-cycle on 3. The same word $c = ba^2b^2a^2b$ is a (non-simple) cycle on 2 since $c = bs_1b^2s_2b$, where bb^2b is a simple cycle on 2, $s_1 = a^2$ is a simple cycle on 3 and $s_2 = a^2$ is a simple cycle on 1.

Example 4.2. As another example, consider the automaton $\mathscr{A}_2$ (Fig. 2). It is easy to see that the elementary 0-labels $adeaa$, $adddeeaa$ are cycles on 0. On the contrary, $c = adedaa$ is not a cycle on 0 since there is a unique simple cycle on 0 having a as a prefix, namely a^3, and ded is not a semi-cycle on 1.

The 0-label $c{=}ad^2e^3a(be^2b)(be^3b)(be^4b)a$ is a cycle on 0. Indeed, we have $c{=}as_1as_2a$, with a^3 being a simple cycle on 0, $s_1 = d^2e^3$ being a semi-cycle on 1, $s_2 = c_3c_4c_5$ being a semi-cycle on 2 since $c_3 = (be^2b)$, $c_4 = (be^3b)$, $c_5 = (be^4b)$ are different cycles on 2. On the contrary, $c = ad^2e^3a(be^2b)(be^3b)(be^2b)a$ is not a cycle on 0 since, once again, there is a unique simple cycle on 0 having a as a prefix, namely a^3, and $(be^2b)(be^3b)(be^2b)$ is not a semi-cycle on 2.

Cycles satisfy a combinatorial property which is described in the lemma below. This lemma, which will be used in the proof of Proposition 4.1, states that a proper semi-cycle on q is not a cycle on q and that a cycle on q cannot be a proper prefix of another cycle on q.

Lemma 4.2. *Let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be a deterministic finite state automaton, let $q \in Q$. A cycle on q is never a proper prefix of another cycle on q.*

Proof. Let c, d be two cycles on q with $c \neq d$ and, by contradiction, suppose that there exists $w \in A^+$ such that $c = dw$ or $d = cw$. Let us suppose that the first case holds (the second case is analogous). Consequently, we have $q = \delta(q, c) = \delta(q, dw) = \delta(\delta(q, d), w) = \delta(q, w)$, i.e., $c = dw$ is not an elementary q-label and this is a contradiction (see Definition 4.2). □

The definition which follows intervenes in the statement of Proposition 4.1. In this proposition, we will state a property of q-labels that are not cycles on q which, in turn, will be used in the proof of the main result of this paper.

Definition 4.2 (*Forbidden q-label*). A q-label c is a *quasi semi-cycle* on q if there exist (not necessarily different) cycles $c_1, \ldots, c_n$ on q such that $c = c_1 \cdots c_n$. If $n > 1$ then c *is a proper quasi semi-cycle* on q. A q-label c is a *forbidden q-label* if $c = c_1 \lambda c_1$, where c_1 is a cycle on q and λ is a q-label which is not a power of c_1 (i.e., $\lambda \neq c_1^t$, $t \geqslant 1$).

Remark 4.1. We notice that if c is a cycle on q then c is not a proper quasi semi-cycle on q. Furthermore, using the same notations as in Definition 4.2, there exists a factorization of c, say $u_1 s_1 \cdots s_k u_{k+1}$, such that no s_j is a forbidden p_j-label. Nevertheless, examples of cycles c on q exist such that we can write $c = hfg$ with f being a forbidden p-label and $\delta(q, h) = p$. For instance, let $\mathscr{A}_1$ be the automaton in Fig. 1. Then $c = ba^2b^2a^4b$ is a cycle on 0 since $c = bs_1b^2s_2b$, where bb^2b is a simple cycle on 0, $s_1 = a^2$ is a cycle on 1 and $s_2 = a^4$ is a semi-cycle on 3. On the other hand, $c = bfab$, where $f = a^2b^2a^3$ is a forbidden 1-label with $c_1 = a^2$, $\lambda = b^2a$. As another example, consider the automaton $\mathscr{A}_2$ and the cycle $c = ad^2e^3a(be^2b)(be^3b)(be^4b)a$ on 0. We have $c = (ad^2e^3abe^2)f(e^4ba)$, where $f = b^2e^3b^2$ is a forbidden 3-label. Proposition 4.1 states that, if a q-label c is not a quasi semi-cycle on q then the above-mentioned factorization hfg of c always exists.

Proposition 4.1. *Let L be a regular language and let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L. Let c be a q-label in $\mathscr{A}$. If c is not a cycle, then one of the following two cases occurs.*

(1) *c is a proper quasi semi-cycle on q.*
(2) *$c = hfg$, where $h, g \in A^+$, f is a forbidden p-label and $\delta(q, h) = p$.*

Proof. Let c be a q-label in $\mathscr{A}$ which is not a cycle. By using induction over $|c|$, we prove that condition (1) or (2) holds.

Let c be the shortest string among all those strings that are p-labels in $\mathscr{A}$ and that are not cycles on p, with $p \in Q$. If c is not an elementary q-label, i.e., $c = c'c''$ with c', c'' being q-labels, then c', c'' are cycles on q (since c', c'' are shorter than c) and c is a proper quasi semi-cycle on q. Otherwise, c is not a simple cycle on q and, for each $u_1, u_2, \ldots, u_k, u_{k+1}, s_1, \ldots, s_k \in A^+$ such that $c = u_1 s_1 u_2 \cdots u_k s_k u_{k+1}$, with $k \geqslant 1$, $u_1 u_2 \cdots u_k u_{k+1}$ being a simple cycle on q, there exists j, $1 \leqslant j \leqslant k$ such that, for $p_j = \delta(p_{j-1}, u_j)$, s_j is a p_j-label which is not a semi-cycle, in particular s_j is not a cycle, a contradiction since $|s_j| < |c|$.

Suppose now that the conclusion holds for words $d \in A^*$, $|d| < |c|$ and let us prove the conclusion for c. The following preliminary observations hold as direct results.

(a) If c', c'' are q-labels such that each of them is a quasi semi-cycle on q or is a product $h'f'g'$, with $h', g' \in A^+$, f' being a forbidden p-label, $\delta(q, h') = p$, then $c = c'c''$ satisfies condition (1) or condition (2).
(b) If $c = hh'fg'g$, where $h, g \in A^+$, $h', g' \in A^*$, f is a forbidden p-label, $\delta(q, hh') = p$, then c satisfies condition (2).

If c is not an elementary q-label, i.e., $c = c'c''$ with c', c'' being q-labels shorter than c, then item (a) along with the induction hypothesis allows us to infer that c satisfies condition (1) or condition (2). Suppose now that c is an elementary q-label. Once again, c is not a simple cycle on q and, for each $u_1, u_2, \ldots, u_k, u_{k+1}, s_1, \ldots, s_k \in A^+$ such that $c = u_1 s_1 u_2 \cdots u_k s_k u_{k+1}$, with $k \geqslant 1$, $u_1 u_2 \cdots u_k u_{k+1}$ being a simple cycle on q, there exists j, $1 \leqslant j \leqslant k$, such that, for $p_j = \delta(p_{j-1}, u_j)$, s_j is a p_j-label which is not a semi-cycle on p_j. In particular, $c = hs_jg$ with $h, g \in A^+$, s_j is not a cycle on p_j and $|s_j| < |c|$. Thus, by induction hypothesis, either $s_j = h'fg'$, where $h', g' \in A^+$, f is a forbidden p-label, $\delta(p_j, h') = p$ or s_j is a proper quasi semi-cycle on p_j. In the latter case, we still have $s_j = h'fg'$ with h', f, g'as above, except that h' or g' can now be the empty word. Indeed, since s_j is a proper quasi semi-cycle on p_j, we have $s_j = c'_1 \cdots c'_n$, with $n > 1$ and c'_k being cycles on p_j. Furthermore, since s_j is not a semi-cycle on p_j, there exists i_1, i_2, $1 \leqslant i_1 < i_2 \leqslant k$, such that $c'_{i_1} = c'_{i_2}$ while $c'_{i_1+1}, \ldots, c'_{i_2-1}$ are different from c'_{i_1}. Then, the above-mentioned factorization exists with $f = c'_{i_1} \lambda c'_{i_2}$, since $\lambda = c'_{i_1+1} \cdots c'_{i_2-1} \in A^+$ is a p_j-label and λ is not a power of c'_{i_1} (in view of Lemma 4.2).

Thus, in view of item (b), c satisfies condition (2). □

In Remark 4.1 we have already observed that Proposition 4.1 does not lead to a characterization of cycles since we have given examples of cycles which satisfy condition (2) contained in the statement of Proposition 4.1.

Example 4.3. Consider $\mathscr{A}_2$ (Fig. 2). We observe that aea^2, $aedabeba$ are elementary 0-labels, while ea^3 is not an elementary 0-label.

Each word $c_{i,j} = d^i e^j$, with $i, j \geqslant 1$, is a proper semi-cycle on 1, hence $c_{i,j}$ is not a cycle on 1. On the other hand, each word $c_{i,j,k} = d^i e^j d^k$, with $i, j, k \geqslant 1$, is a forbidden 1-label. Hence, once again, $c_{i,j,k}$ is not a cycle on 1 since $c_{i,j,k}$ is not an elementary 1-label. Note, that $d^i e^j$ satisfies condition (1) in Proposition 4.1, while $ad^i e^j d^k a^2$ satisfies condition (2) in the same proposition.

The word $c = adeda^2$ is an example of an elementary 0-label that is not a cycle on 0. Indeed, since there is a unique simple cycle on 0 having a as a prefix, namely a^3, there is only one factorization of c which we should consider and that is $c = as_1a^2$, with $s_1 = ded$. On the other hand, s_1 is not a semi-cycle on state 1 and so, c is not a cycle on 0.

5. Markers and languages defined by markers

In this section we will give some preliminary definitions in order to define the class of languages we deal with, i.e., marker languages (Definition 5.4).

Definition 5.1 (*Cyclic class*). A class $[x] \in \mathscr{M}(L)$ is called *cyclic* if there exists $q \in Q$ and $y \in [x]$ such that y is a q-label in $\mathscr{A}$.

For instance, given the finite state automaton $\mathscr{A}_3$ depicted in Fig. 3, $[cb] = ((cb)^*(dh)^*)^* \setminus \{1\}$ is a cyclic class since cb, dh are cycles.

Definition 5.2 (*Singular class*). A class $[x] \in \mathscr{M}(L)$ is called *singular* if $|Q_{[x]}| = 1$.

Proposition 5.1 states a property of a cyclic class which is singular: every word y in $[x]$ is a q-label.

Proposition 5.1. *Let* $[x]$ *be a singular and cyclic class and let* $Q_{[x]} = \{q_x\}$. *For every* $y, z \in [x]$, *we have* $\delta(q_x, y) = \delta(q_x, z) = q_x$ *and* $yz \in [x]$.

Proof. In virtue of Definition 5.1, we have $\delta(q_x, x) = q_x$. Let $y, z \in [x]$. Thus, using Theorem 2.1, we have $\delta(q_x, y) = \delta(q_x, z) = q_x$ and also $\delta(q_x, yz) = \delta(\delta(q_x, y), z) = q_x$. Since Remark 2.1 allows us to state that $Q_x = Q_y = Q_z = \{q_x\}$, we also have $Q_{yz} = \{q_x\}$. Thus, $yz \in [x]$ in virtue of Theorem 2.1. □

Definition 5.3 (*Marker*). Let L be a regular language and let $x \in A^+$ be such that $Q_x \neq \emptyset$. The set $[x]$ is a *marker* for L if $[x] \in \mathscr{M}(L)$ is a singular class that is finite or cyclic. We also set $[x]_1 = [x]$ if $[x]$ is finite, otherwise $[x]_1 = [x] \cup \{1\}$.

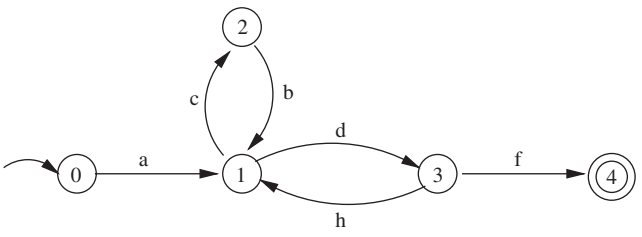

Fig. 3. Automaton $\mathscr{A}_3$ for $a((cb)^*(dh)^*)^*df$.

Example 5.1. For instance, given the finite state automaton $\mathscr{A}_3$ depicted in Fig. 3, $[cb]$ is a marker for $L = a((cb)^*(dh)^*)^*df$ since $[cb] = ((cb)^*(dh)^*)^*\backslash\{1\}$ is a cyclic class and $|Q_{[cb]}| = 1$. In addition, for each $x \in \{a, b, c, d, f, h\}$ we have that $[x]$ is a marker for L since $|Q_{[x]}| = 1$ and $[x] = \{x\}$ is a finite class.

Notice that a marker for L may be a word (when $|[x]| = 1$) or a finite set of equivalent words with respect to $\equiv_L$ (when $[x]$ is a finite class) or a set of equivalent words with respect to $\equiv_L$ which are q_x-labels of closed paths (when $[x]$ is a cyclic class). Furthermore, we have $[x] \neq [1]$ unless we have $|Q| = 1$, where Q is the set of the states in $\mathscr{A}$. In the next part of this paper we will suppose $|Q| > 1$. Proposition 5.2 states that the existence of a marker for L is decidable.

Proposition 5.2. *It is decidable whether a regular language L has a marker. There exists an algorithm for constructing the set of markers for L.*

Proof. Given a regular language L, the structure of the syntactic monoid $\mathscr{M}(L)$ of L and, for each $[x] \in \mathscr{M}(L)$, the structure of the regular language $[x]$ can be obtained by standard construction algorithms [21]. Let $\{x_1, \ldots, x_k\}$ be a complete set of representatives of the syntactic congruence classes of L. Obviously, for all $i \in \{1, \ldots, k\}$, we can decide whether $|Q_{x_i}| = 1$ in the minimal automaton $\mathscr{A}$ recognizing L. For this x_i we can also decide whether $[x_i]$ is a cyclic class since, in view of Proposition 5.1, $[x_i]$ is a cyclic class if and only if x_i is a q-label. In addition, we can decide whether $[x_i]$ is a finite class since it is decidable whether a regular set is finite. Therefore, in view of Definition 5.3, it is decidable whether $[x_i]$ is a marker. The argument above allows us to design an algorithm for constructing the set of markers for L. □

Let L be a regular language and let $[x]$ be a marker for L. We now define a regular language $L([x])$ associated with a marker and in Theorem 6.1 we will prove that $L([x])$ is generated by a finite Paun splicing system. We will also introduce notations which will be used in the next part of this paper. In view of Definition 5.3, Theorem 2.1 and Remark 2.1 there exist q_x, $p_x \in Q$ such that $Q_{[x]} = \{q_x\}$ and for each $x' \in [x]$ we have $\delta(q_x, x') = p_x$. In addition, in view of Proposition 5.1, if $[x]$ is a cyclic class then $p_x = q_x$.

Definition 5.4 (*Marker language*). Let L be a regular language and let $[x]$ be a marker for L. Denote $L_1 = \{y \in A^* \mid \delta(q_0, y) = q_x\}$, $L_2 = \{y \in A^* \mid \delta(p_x, y) \in F\} = L_{p_x}$. If $L = L_1[x]_1L_2$, then L is the *marker language* $L([x])$ associated with $[x]$. L is a *marker language* if there exists a marker $[x]$ for L such that L is the marker language associated with $[x]$.

Example 5.2. Consider the finite state automaton $\mathscr{A}_3$ depicted in Fig. 3. The language $L([cb]) = a((cb)^*(dh)^*)^*df$ is the marker language associated with $[cb]$ and we have $L_1 = a((cb)^*(dh)^*)^*$ and $L_2 = ((cb)^*(dh)^*)^*df$. The same language $L([f])=a((cb)^*(dh)^*)^*df$ is also the marker language associated with $[f]$ and we have $L_1=a((cb)^*(dh)^*)^*d$ and $L_2 = \{1\}$.

Remark 5.1. Let $L = L_1[x]_1L_2$ be the marker language associated with a marker $[x]$. Since the minimal automaton recognizing L is trim and $Q_x \neq \emptyset$ (Definition 5.3), there exist $y, z \in A^*$ such that $y[x]z \subseteq L$, i.e., we have $L_1 \neq \emptyset$ and $L_2 \neq \emptyset$. In addition, $L_1 = L([x]_1L_2)^{-1}$, $L_2 = (L_1[x]_1)^{-1}L$ and L_1, L_2 are regular languages (see Section 2.1).

Remark 5.2. Let L be the marker language associated with a marker $[x]$. For $z \in L$, we could have two different factorizations $z = x_1xx_2 = x_1'x'x_2'$ with $x, x' \in [x]$, $x_1, x_1' \in L_1$, $x_2, x_2' \in L_2$, i.e., the product $L_1[x]L_2$ is not necessarily unambiguous.

Proposition 5.3. *It is decidable whether a regular language L is a marker language.*

Proof. Given a regular language L, in Proposition 5.2 we have proved that there exists an algorithm for constructing the set of markers $[x]$ for L. In the proof of this result we have observed that the structure of the regular language $[x]$ can also be described.

Given a marker $[x]$ for L, let us consider the regular sets $F([x]) = A^*[x]A^* \cap L$, $Suf([x]) = (A^*[x])^{-1}F([x])$, $Pref([x]) = F([x])([x]A^*)^{-1}$. Then, L is a marker language if and only if there exists a marker $[x]$ for L such that

$L = F([x])$ if $[x]$ is a finite class, $L = F([x]) \cup Pref([x])Suf([x])$ otherwise (Definition 5.4). Thus, the conclusion holds since given two regular languages X, Y, it is decidable whether $X = Y$ [15]. □

6. Main result

The main result of this section is stated in Theorem 6.1 which will be proved in Sections 6.2 and 7.

6.1. Definition of a splicing system for marker languages

Let L be a regular language, let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L and let $p \in Q$. We will consider the following set $\mathscr{I}_{p,\mathscr{A}}$ associated with $\mathscr{A}$ and p, simply denoted $\mathscr{I}_p$ when $\mathscr{A}$ is understood.

$$\mathscr{I}_{p,\mathscr{A}} = \{y \in A^* \mid p \in Q_y,\ \forall u, v \in A^*, q \in Q \text{ if } y = uc^t v \text{ with } \delta(p, u) = q \text{ and } c \text{ being a } q\text{-label, then } c \text{ is a cycle on } q \text{ in } \mathscr{A} \text{ and } t = 1\}. \quad (1)$$

The result that follows will be used in the next part of this section.

Proposition 6.1. *Let $y \in A^*$. We have $y \in \mathscr{I}_{p,\mathscr{A}}$ if and only if $p \in Q_y$ and each $q \in Q$ is crossed at most twice by the path $(p, y, \delta(p, y))$.*

Proof. Suppose that $y \in \mathscr{I}_{p,\mathscr{A}}$. Then, by definition $p \in Q_y$. By contradiction, suppose that $q \in Q$ exists such that q is crossed at least three times by the path $(p, y, \delta(p, y))$. Then, $z_1, z_2 \in A^*$ and $c_1, c_2 \in A^+$ exist such that $y = z_1c_1c_2z_2$, $\delta(q, c_1) = q$, $\delta(q, c_2) = q$, which contradicts $y \in \mathscr{I}_{p,\mathscr{A}}$ when we take $c_1c_2 = c^t$.

Conversely, let $y \in A^*$, $p \in Q_y$ and suppose that each $q \in Q$ is crossed at most twice by the path $(p, y, \delta(p, y))$. Then, $y \in \mathscr{I}_{p,\mathscr{A}}$ since otherwise, there exist $u, v \in A^*$, $q \in Q$ and a q-label c such that $y = uc^t v$ and either c is not a cycle on q or $t > 1$. In view of Definition 4.2 and Proposition 4.1, q is crossed at least three times by the path $(p, y, \delta(p, y))$: a contradiction. □

Remark 6.1. Each word $y \in \mathscr{I}_{p,\mathscr{A}}$ is named a reduced label on p in [5].

Let $[x]$ be a marker for L and suppose that $L = L([x]) = L_1[x]_1L_2$ is the marker language associated with $[x]$. We now define a splicing system $S_{[x]} = (A, I, R)$ and we will prove that $L([x]) \subseteq L(S_{[x]})$. In the next part of this paper, $\mathscr{A}_R = (Q_R, A, \delta_R, q_{0_R}, F_R)$ will be the minimal finite state automaton recognizing $(L_1)^R$ and we will use the notation $(\mathscr{I}_R)_p$ for the set $\mathscr{I}_{p,\mathscr{A}_R}$, the definition of which is recalled below.

$$(\mathscr{I}_R)_p = \{y \in A^* \mid p \in (Q_R)_y,\ \forall u, v \in A^*, q \in Q_R \text{ if } y = uc^t v \text{ with } \delta_R(p, u) = q \text{ and } c \text{ being a } q\text{-label in } \mathscr{A}_R, \text{ then } c \text{ is a cycle on } q \text{ in } \mathscr{A}_R \text{ and } t = 1\}. \quad (2)$$

Definition 6.1 (*Initial language*). Let $L = L_1[x]_1L_2$ be the marker language associated with $[x]$ and let x_M be a fixed element in $[x]$. The *initial language* I associated with $L[x]$ is the set

$$I = I_1M([x])I_2, \quad (3)$$

where the sets I_1, I_2, $M([x])$, are defined as follows:

$$I_1 = (\mathscr{I}_R)^R_{q_{0_R}} \cap L_1 = \{y \in (\mathscr{I}_R)^R_{q_{0_R}} \mid y^R \in L(\mathscr{A}_R)\},$$
$$I_2 = \mathscr{I}_{p_x} \cap L_2 = \{y \in \mathscr{I}_{p_x} \mid \delta(p_x, y) \in F\},$$
$$M([x]) = \begin{cases} x_M & \text{if } [x] \text{ is cyclic,} \\ [x] & \text{if } [x] \text{ is finite.} \end{cases}$$

Proposition 6.2. *The initial language I is not empty.*

Proof. Let $L = L_1[x]_1L_2$ be the marker language associated with $[x]$ and let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L. By Remark 5.1, we have $L_1 \neq \emptyset$ and $L_2 \neq \emptyset$, so let $l_1, l_2 \in A^*$ be such that $\delta(p_x, l_2) = q_F \in F$, $\delta_R(q_{0_R}, l_1^R) = p_F \in F_R$. In view of Lemma 4.1, there exists $l_2' \in A^*$ (resp. $(l_1')^R \in A^*$) such that $\delta(p_x, l_2') = q_F \in F$ (resp. $\delta_R(q_{0_R}, (l_1')^R) = p_F \in F_R$) and the internal states crossed by the path (p_x, l_2', q_F) (resp. by the path $(q_{0_R}, (l_1')^R, p_F)$) are different from one another and with respect to p_x, q_F (resp. q_{0_R}, p_F). Then, by Proposition 6.1, $l_1' \in I_1$ and $l_2' \in I_2$. Finally, $M([x]) \neq \emptyset$ and for each $x' \in M([x])$, we have $l_1'x'l_2' \in I$. □

In Proposition 6.3 we also prove that the initial language I is finite.

Definition 6.2 (*Set of rules*). Let $[x]$ be a marker and let I be the initial language associated with $L([x])$.

For every $z = z_1x'z_2 \in I$, with $z_1 \in I_1$, $x' \in M([x])$, $z_2 \in I_2$, for every factor c_j of z_2, i.e., $z_2 = z'c_jz''$, which is a cycle on q in $\mathscr{A}$, $q = \delta(p_x, z')$, let

$$r_{c_j} = \quad x'z' \mid 1\,\$\,x'z'c_j \mid 1. \tag{4}$$

For every $z = z_1x'z_2 \in I$, with $z_1 \in I_1$, $x' \in M([x])$, $z_2 \in I_2$, for every factor c_j of $(z_1)^R$, i.e., $(z_1)^R = z''c_jz'$, which is a cycle on q in $\mathscr{A}_R$, $q = \delta_R(q_{0_R}, z'')$, let

$$r'_{c_j} = 1 \mid (c_j)^R(z'')^Rx'\,\$\,1 \mid (z'')^Rx'. \tag{5}$$

In addition, if $[x]$ is a cyclic class, let

$$r_{x_M,x_M} = x_M \mid 1\,\$\,1 \mid x_M. \tag{6}$$

Finally, for each $z = z_1x'z_2 \in I$, let R_z be the set of rules r_{c_j}, r'_{c_j} defined by Eqs. (4), (5). We define $R = \bigcup_{z\in I} R_z$ when $[x]$ is a finite class and $R = (\bigcup_{z\in I} R_z) \cup \{r_{x_M,x_M}\}$ when $[x]$ is a cyclic class.

We also observe that there could be more than one occurrence of c_j as a cycle in z. For instance, we could have $z = z_1xz_2 = z_1xz'c_jz'' = z_1xz_3'c_jz_3''$ with $z' \neq z_3'$. Thus, in this case we have two rules r_{c_j} (at least) and we should denote them r_{z',c_j}, r_{z_3',c_j}. As a matter of fact we will adopt the same notations as in Eqs. (4), (5) since the context will not make them ambiguous.

In Proposition 6.3 we prove that $S_{[x]} = (A, I, R)$ is a finite splicing system by using an argument already given in [4]. In the statements of the propositions proved in this section we will use all the notations introduced in Section 5 and in the first part of this section.

Proposition 6.3. *The languages I and R are finite.*

Proof. We know that $I = I_1M([x])I_2$, where I_2 (resp. I_1^R) is a subset of $\mathscr{I}_{p_x}$ (resp. $(\mathscr{I}_R)_{q_{0_R}}$). Furthermore, $M([x])$ is a finite set. Thus, if we prove that $\mathscr{I}_{p_x}$ (resp. $(\mathscr{I}_R)_{q_{0_R}}$) is a finite set, then I is also a finite set. On the other hand, each $y \in \mathscr{I}_{p_x}$ is the label of a path π in $\mathscr{A}$ such that every $q \in Q$ is crossed at most twice by the path π (see Proposition 6.1). Consequently, $|y| \leqslant 2|Q|$ and $\mathscr{I}_{p_x}$ is a finite set. The same argument maintains for $(\mathscr{I}_R)_{q_{0_R}}$, which is in turn a finite set. Furthermore, for each $z \in I$, R_z is finite since the number of the elements in R_z is at most equal to the number of the factors of z. Finally, if $[x]$ is a finite class then R is finite since R is the union of a finite number ($\leqslant |I|$) of the finite sets R_z. Clearly, R is also finite when $[x]$ is a cyclic class. □

Definition 6.3 (*Paun splicing system for marker languages*). Let I be as in Definition 6.1 and let R be as in Definition 6.2. Then, $S_{[x]} = (A, I, R)$ is the finite Paun splicing system *associated with* $L([x])$.

Theorem 6.1. *Let L be a regular language. Let* $[x]$ *be a marker for L and suppose that* $L = L([x])$ *is the marker language associated with* $[x]$. *Let* $S_{[x]} = (A, I, R)$ *be the finite Paun splicing system associated with* $L([x])$. *Then, we have* $L([x]) = L(S_{[x]})$.

Example 6.1. Let $L = a((cb)^*(dh)^*)^*df$ be recognized by $\mathscr{A}_3$ (Fig. 3). As observed in Example 5.2, $L = L([f]) = L_1[f]_1L_2 = L_1[f]L_2$ is the marker language associated with $[f]$, where $L_1 = a((cb)^*(dh)^*)^*d$ and $L_2 = \{1\}$. The

minimal finite state automaton $\mathscr{A}_{3,R}$ recognizing $(L_1)^R$ may be easily obtained by reversing all the edges in $\mathscr{A}_3$, by removing state 4 and by taking 3 as initial state and 0 as the unique final state. Therefore, $I_1=\{acbd, adhd, ad\}$ and $S_{[f]}=(A, I, R)$ with $I=\{acbdf, adhdf, adf\}$, $R=\{1 \mid cbdf\,\$\,1 \mid df, 1 \mid dhdf\,\$\,1 \mid df\}$. By Theorem 6.1, $S_{[f]}$ generates L. In Example 5.2, we have observed that $L=L([cb])=L_1[cb]_1L_2$ is also the marker language associated with $[cb]$ and, in this case, we have $L_1=a((cb)^*(dh)^*)^*$ and $L_2=((cb)^*(dh)^*)^*df$. The same automaton $\mathscr{A}_{3,R}$ once again recognizes $(L_1)^R$, when we take 1 as the initial state. We have $I_1=\{acb, adh, a\}$, $I_2=\{cbdf, dhdf, df\}$ and, when we set $M([cb])=cb$, we have $S_{[cb]}=(A, I, R)$ with $I=\{acb, adh, a\}cb\{cbdf, dhdf, df\}$, $R=\{cb \mid 1\,\$\,cbcb \mid 1, cb \mid 1\,\$\,cbdh \mid 1, 1 \mid cbcb\,\$\,1 \mid cb, 1 \mid dhcb\,\$\,1 \mid cb, cb \mid 1\,\$\,1 \mid cb\}$. By Theorem 6.1, $S_{[cb]}$ generates L.

6.2. Generation

In this section we prove that, given a marker language $L([x])=L_1[x]_1L_2$ and the splicing system $S_{[x]}$ associated with $L([x])$ by Definition 6.3, we have $L([x]) \subseteq L(S_{[x]})$. The proof of this result is divided into five steps and the hypothesis that $L([x])$ is a marker language is unnecessary in the first step. Lemma 6.1 will be used in the proof of Proposition 6.4.

Lemma 6.1. *Let L be a regular language, let $\mathscr{A}=(Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L and let $p \in Q$. Let $w_1, w_2 \in A^*$ be such that $w_1 \in \mathscr{I}_{p,\mathscr{A}}=\mathscr{I}_p$, $w_1w_2 \in L_p$ and $\delta(p, w_1)=q_1$, $\delta(q_1, w_2)=q_F$. Suppose that for every $q' \in Q\backslash\{q_1\}$, if q' is crossed by the path (q_1, w_2, q_F), then q' is crossed at most once by the path (p, w_1, q_1). Then, there exists $v' \in A^*$ such that $w_1v' \in \mathscr{I}'_p=\mathscr{I}_p \cap L_p$.*

Proof. If $q_1=q_F$ the conclusion holds with $v'=1$. So, suppose $q_1 \neq q_F$. Let v' be the word satisfying conditions (a)–(d) in Lemma 4.1 applied to w_2 and $\delta(q_1, w_2)=q_F$. In view of condition (a) in Lemma 4.1 we have $w_1v' \in L_p$ and, in view of condition (b) in the same lemma, w_1, v' satisfy the hypothesis in the statement. In addition, by using condition (c) in Lemma 4.1 and since $q_1 \neq q_F$, the states crossed by the path (q_1, v', q_F) are different from one another.

We claim that $w_1v' \in \mathscr{I}'_p$, i.e., $w_1v' \in \mathscr{I}_p$. Indeed, q_1 is crossed at most twice by the path (p, w_1, q_1) (Proposition 6.1) and q_1 is crossed only once by the path (q_1, v', q_F), so q_1 is crossed at most twice by the path (p, w_1v', q_F). Analogously, for each $q' \in Q\backslash\{q_1\}$, either q' is not crossed by the path (q_1, v', q_F) and q' is crossed at most twice by the path (p, w_1, q_1) (Proposition 6.1) or q' is crossed (only once) by the path (q_1, v', q_F) and so, in view of the hypothesis, q' is crossed only once by the path (p, w_1, q_1). We conclude that each $q' \in Q$ is crossed at most twice by the path (p, w_1v', q_F). Since $p \in Q_{w_1v'}$, by using Proposition 6.1, we have $w_1v' \in \mathscr{I}_p$. □

Proposition 6.4. *Let $L([x])$ be a marker language and let $S_{[x]}=(A, I, R)$ be the splicing system associated with $L([x])$. Then we have $L([x]) \subseteq L(S_{[x]})$.*

Proof. The proof is divided into the following five steps.

Step 1. Let M be a finite subset of A^*, with $M \neq \emptyset$. Let L be a regular language, let $\mathscr{A}=(Q, A, \delta, q_0, F)$ be the minimal finite state automaton recognizing L and let $p \in Q$. Let $\mathscr{I}_p=\mathscr{I}_{p,\mathscr{A}}$ and let $\mathscr{I}'_p=\mathscr{I}_p \cap L_p$. Let $S_{PA}=(A, I', R')$ be a splicing system such that for every $x' \in M$, $z_2 \in \mathscr{I}'_p$, for every factor c_j of z_2, i.e., $z_2=z'c_jz''$, which is a cycle in $\mathscr{A}$ on $q=\delta(p, z')$, the rule $r_{c_j}=x'z' \mid 1\,\$\,x'z'c_j \mid 1$ belongs to R'.

For all $Y \subseteq A^*$, if $YM\mathscr{I}'_p \subseteq L(S_{PA})$, then $YML_p \subseteq L(S_{PA})$.

Step 2. $I_1M([x])L_2 \subseteq L(S_{[x]})$.

Step 3. $L_1M([x])L_2 \subseteq L(S_{[x]})$.

Step 4. If $[x]$ is a cyclic class then $L_1M([x])[x]L_2 \subseteq L(S_{[x]})$.

Step 5. $L_1[x]_1L_2 \subseteq L(S_{[x]})$.

As a matter of fact, Steps 1–3 allow us to prove the statement when $[x]$ is a finite class (in this case $M([x])=[x]=[x]_1$). Steps 4–5 are necessary in order to end the proof when $[x]$ is a cyclic class.

Step 1. Let $y \in Y$, $x' \in M$, $l \in L_p$, $z=yx'l \in YML_p$. We prove that $z \in L(S_{PA})$ by induction over $|z|$. If $l \in \mathscr{I}'_p$, then by hypothesis $z \in L(S_{PA})$. Assume that $l \in L_p\backslash\mathscr{I}'_p$ and that, for each $z' \in YML_p\backslash YM\mathscr{I}'_p$ with $|z'|<|z|$, we have $z' \in L(S_{PA})$.

Since $l \in L_p\backslash\mathscr{I}'_p$, then $l \notin \mathscr{I}_p$. Let l_1 be the shortest prefix of l that is a word not in $\mathscr{I}_p$. Then $p \in Q_{l_1}$ and, looking at Eq. (1) and Proposition 4.1, we have that either q, u, c_1, c_2, v exist such that $l_1=uc_1c_2$, $l=uc_1c_2v$, c_1, c_2 (not

necessarily different) cycles on $q=\delta(p,u)$ or $l_1=uhc_1\lambda c_1 g$, where $h, g\in A^+$ and $d=c_1\lambda c_1$ is a forbidden q'-label, with $q'=\delta(q,uh)$. Now, the latter case cannot occur since $l'=uhd$ is a proper prefix of l_1 and $l'\notin \mathscr{I}_p$ (Proposition 6.1), a contradiction with the minimality of $|l_1|$.

In addition, by using the minimality of $|l_1|$, we have that each proper prefix of l_1 is in $\mathscr{I}_p$. In particular, we have $uc_1\in\mathscr{I}_p$, while $uc_1v\in L_p$. Furthermore, uc_1v satisfies one of the following two conditions.

(a) There exists a state q_1 which is crossed by the path $(q,v,\delta(q,v))$, with $q_1\neq q$ and such that q_1 is crossed at least twice by the path (p,uc_1,q).

(b) For each state q_1 crossed by the path $(q,v,\delta(q,v))$, $q_1\neq q$, q_1 is crossed at most once by the path (p,uc_1,q).

Case (a). Suppose that Case (a) occurs. Then, there exist $u_1,u_2,v_1,v_2\in A^*$ such that $uc_1=u_1du_2$, $v=v_1v_2$, with d,u_2v_1 being q_1-labels and $\delta(p,u_1d)=q_1=\delta(p,u_1)=\delta(q,v_1)$, $\delta(q_1,u_2)=q$, $\delta(q_1,v_2)\in F$. So, $u_2\neq 1$ since $q_1\neq q$. Suppose u_1d of minimal length with respect to this condition, that is each state q' crossed by the path $(q,v,\delta(q,v))$, $q'\neq q$, $q'\neq q_1$ is crossed at most once by the path (p,u_1d,q_1) and q_1 is crossed exactly twice by the path (p,u_1d,q_1). Furthermore, observe that since u_1d is a proper prefix of uc_1 then $|u_1d|<|l_1|$ and so $u_1d\in\mathscr{I}_p$, d is a cycle on q_1. In addition, q is crossed at most once by the path (p,u_1d,q_1). Indeed, otherwise, u_1d being a proper prefix of uc_1, q would be crossed more than twice by the path (p,uc_1,q): this is a contradiction since $uc_1\in\mathscr{I}_p$ (Proposition 6.1). Moreover, obviously $l''=u_1dv_2\in L_p$. Thus, $w_1=u_1d$, $w_2=v_2$ satisfy the hypothesis of Lemma 6.1: for every $q'\in Q\backslash\{q_1\}$, if q' is crossed by the path (q_1,w_2,q_F), then q' is crossed at most once by the path (p,w_1,q_1). Consequently, there exists v' such that $u_1dv'\in\mathscr{I}'_p$. So, for each $x'\in M$ we have $r_d=x'u_1\mid 1\,\$\,x'u_1d\mid 1\in R'$. On the other hand, we obviously have $l'=u_1u_2c_2v\in L_p$ and $l''=u_1dv_2\in L_p$. Thus, $z'=yx'l'$, $z''=yx'l''\in YML_p$ with $|z'|<|z|$, $|z''|<|z|$ and, consequently, by the induction hypothesis, $z',z''\in L(S_{PA})$. By applying rule r_d to the pair (z',z'') we generate z, i.e., $z\in L(S_{PA})$.

Case (b). Obviously we have uc_1v, $uc_2v\in L_p$. Then $z'=yx'uc_1v$, $z''=yx'uc_2v\in YML_p$ and by induction hypothesis, $z',z''\in L(S_{PA})$.

Since $uc_1\in\mathscr{I}_p$, $uc_1v\in L_p$ and Case (b) occurs, the words $w_1=uc_1$ and $w_2=v$ satisfy the hypothesis of Lemma 6.1. Consequently, $v'\in A^*$ exists so that $uc_1v'\in\mathscr{I}'_p$. Then, the rule $r_{c_1}=x'u\mid 1\,\$\,x'uc_1\mid 1$ is in R'. By applying r_{c_1} to z'' and z', we generate z, i.e., $z\in L(S_{PA})$.

Step 2. By choosing $Y=I_1$, $M=M([x])$, $p=p_x$ in Step 1, we have $L_p=L_2$ and $\mathscr{I}'_p=I_2$. Furthermore, $S_{[x]}$ satisfies the hypothesis in Step 1. Since $I_1M([x])I_2\subseteq L(S_{[x]})$, by using Step 1, we obtain $I_1M([x])L_2\subseteq L(S_{[x]})$.

Step 3. By using Step 2, we have $I_1M([x])L_2\subseteq L(S_{[x]})$ and so, by using Proposition 2.2, $(L_2)^R(M([x]))^R(I_1)^R\subseteq L(S^R_{[x]})$. By choosing $Y=(L_2)^R$, $M=(M([x]))^R$, $\mathscr{A}=\mathscr{A}_R$, $p=q_{0_R}$ in Step 1, we have $L_p=(L_1)^R$, $\mathscr{I}_p=(\mathscr{I}_R)_{q_{0_R}}$ and $\mathscr{I}'_p=(I_1)^R$. Furthermore, $S^R_{[x]}$ satisfies the hypothesis in Step 1. Then, since $(L_2)^R(M([x]))^R(I_1)^R\subseteq L(S^R_{[x]})$, by using Step 1, we obtain $(L_2)^R(M([x]))^R(L_1)^R\subseteq L(S^R_{[x]})$ and, by using once again Proposition 2.2, we have $L_1M([x])L_2\subseteq L(S_{[x]})$.

Step 4. Let $[x]$ be a cyclic class and let $z=l_1x_Mxl_2\in L_1M([x])[x]L_2$ with $l_1\in L_1$, $l_2\in L_2$ and $x\in[x]$. In view of Proposition 5.1, $\delta(q_x,x)=q_x$ and so, $p_x=q_x$. Furthermore, since $l_2\in L_2$, we have $\delta(p_x,l_2)=\delta(q_x,l_2)\in F$. Consequently, we have $\delta(p_x,xl_2)=\delta(q_x,xl_2)=\delta(\delta(q_x,x),l_2)=\delta(q_x,l_2)\in F$. In conclusion, $xl_2\in L_2$, i.e., $z=l_1x_Mxl_2\in L_1M([x])L_2$ and, by using Step 3, $z\in L(S_{[x]})$.

Step 5. If $[x]$ is a finite class then $[x]_1=[x]=M([x])$, so, by using Step 3, $L_1[x]_1L_2\subseteq L(S_{[x]})$. Thus, suppose that $[x]$ is a cyclic class. Let $z=l_1xl_2\in L_1[x]L_2$ (resp. $z=l_1l_2$) with $l_1\in L_1$, $l_2\in L_2$. Let us prove that $z\in L(S_{[x]})$. By using Step 4 (resp. Step 3) we have $z'=l_1x_Mxl_2\in L(S_{[x]})$ (resp. $z'=l_1x_Ml_2\in L(S_{[x]})$). On the other hand, $r_{x_M,x_M}=x_M\mid 1\,\$\,1\mid x_M\in R$ and by applying the rule r_{x_M,x_M} to the pair (z',z') we generate z, i.e., $z\in L(S_{[x]})$. □

7. Consistency and syntactic monoid

In this section we give results which allow us to end the proof of Theorem 6.1, namely to prove that $L(S_{[x]})\subseteq L([x])$. Once again, we will refer to the minimal finite state automaton $\mathscr{A}$ recognizing a regular language L. Let us also recall the definition of the *left* and *right* contexts of a word $z\in A^*$.

$$C_{\mathscr{L}}(z)=\{u\in A^*\mid\exists q\in Q_z:\delta(q_0,u)=q\},$$

$$C_{\mathscr{R},q}(z)=\{v\in A^*\mid\delta(q,zv)\in F\},\quad C_{\mathscr{R}}(z)=\bigcup_{q\in Q_z}C_{\mathscr{R},q}(z).$$

Remark 7.1. Since the automaton is trim, then it is evident that $Q_z = \{q \mid C_{\mathscr{R},q}(z) \neq \emptyset\}$. Obviously, if $Q_z = Q_{z'}$, then $C_{\mathscr{L}}(z) = C_{\mathscr{L}}(z')$. Furthermore, it is easy to see that $z \equiv_L z'$ if and only if $Q_z = Q_{z'}$, $(C_{\mathscr{L}}(z) = C_{\mathscr{L}}(z'))$, $C_{\mathscr{R},q}(z) = C_{\mathscr{R},q}(z')$, for any $q \in Q_z = Q_{z'}$. Indeed, in view of Remark 2.1, if $z \equiv_L z'$ we have $Q_z = Q_{z'}$ and so, $C_{\mathscr{L}}(z) = C_{\mathscr{L}}(z')$. Moreover, $C_{\mathscr{R},q}(z) = C_{\mathscr{R},q}(z')$, otherwise there exist $(u, v) \in C(z)$ and $(u, v) \notin C(z')$ or vice versa. Indeed, let u be such that $\delta(q_0, u) = q$. If $v \in C_{\mathscr{R},q}(z) \setminus C_{\mathscr{R},q}(z')$, we have $\delta(q_0, uzv) \in F$ and $\delta(q_0, uz'v) \notin F$, since $\delta(q, z'v) \notin F$. Vice versa, let $Q_z = Q_{z'}$, $(C_{\mathscr{L}}(z) = C_{\mathscr{L}}(z'))$, $C_{\mathscr{R},q}(z) = C_{\mathscr{R},q}(z')$, for any $q \in Q_z = Q_{z'}$. If $(u, v) \in C(z)$ then $u \in C_{\mathscr{L}}(z) = C_{\mathscr{L}}(z')$ and $v \in C_{\mathscr{R},q}(z) = C_{\mathscr{R},q}(z')$, for $q = \delta(q_0, u) \in Q_z = Q_{z'}$. Thus $\delta(q_0, uz'v) \in F$, i.e., $(u, v) \in C(z')$. Using a symmetric argument, we can prove that $C(z') \subseteq C(z)$.

In the next part of this paper we suppose that each rule r, in a given splicing system S_{PA}, is *useful*, i.e., there exist $x, y \in L(S_{PA})$ such that $(x, y) \vdash_r (w', w'')$. As already mentioned, in this section we will prove that $L(S_{[x]}) \subseteq L([x])$, i.e., by applying rules in $S_{[x]}$ to words in $L([x])$ we still obtain words in $L([x])$. The definition below naturally arises. It has also been independently introduced with a different name in [9], along with the notion of a useful rule.

Definition 7.1. A language $L \subset A^*$ is *closed with respect to* a rule r if and only if for each $x, y \in L$, if $(x, y) \vdash_r (w', w'')$ then $w', w'' \in L$.

Let $S_{PA} = (A, I, R)$ be a splicing system. Let $r = u_1 \mid u_2 \,\$\, u_3 \mid u_4$ be a rule in R. By definition, if r is applied to $x = x_1u_1u_2x_2$ and $y = y_1u_3u_4y_2$, then r generates $w' = x_1u_1u_4y_2$ and $w'' = y_1u_3u_2x_2$. Now, if $x, y, w', w'' \in L(\mathscr{A})$, the right context x_2 of u_1u_2 is a right context of u_3u_2 and the right context y_2 of u_3u_4 is a right context of u_1u_4. (In other words, splicing allows us to exchange right contexts between strings.) This observation leads to the following result which links the concept of a language closed with respect to a rule with states in the minimal automaton recognizing L.

Proposition 7.1. *Let $S_{PA} = (A, I, R)$ be a finite Paun splicing system, let $L \subseteq A^*$ be a regular language and let $\mathscr{A}$ be the minimal finite state automaton recognizing L. Then $L = L(\mathscr{A})$ is closed with respect to a rule $r = u_1 \mid u_2 \,\$\, u_3 \mid u_4$ if and only if for each pair $(p, q) \in Q_{u_1u_2} \times Q_{u_3u_4}$, we have $(p, q) \in Q_{u_1u_4} \times Q_{u_3u_2}$ and*

(1) $C_{\mathscr{R},p}(u_1u_2) \subseteq C_{\mathscr{R},q}(u_3u_2)$,
(2) $C_{\mathscr{R},q}(u_3u_4) \subseteq C_{\mathscr{R},p}(u_1u_4)$.

Furthermore, if $I \subseteq L(\mathscr{A})$ and $L(\mathscr{A})$ is closed with respect to each rule in R, then $L(S_{PA}) \subseteq L(\mathscr{A})$.

Proof. Since each rule is useful, for each $r = u_1 \mid u_2 \,\$\, u_3 \mid u_4$ we have that $Q_{u_1u_2} \neq \emptyset$ and $Q_{u_3u_4} \neq \emptyset$.

Assume that L is closed with respect to the rule $r = u_1 \mid u_2 \,\$\, u_3 \mid u_4$. Let $(p, q) \in Q_{u_1u_2} \times Q_{u_3u_4}$ and $x_1, y_1 \in A^*$ such that $\delta(q_0, x_1) = p$ and $\delta(q_0, y_1) = q$ ($\mathscr{A}$ is trim). For each $x_2 \in C_{\mathscr{R},p}(u_1u_2)$ and $y_2 \in C_{\mathscr{R},q}(u_3u_4)$ we have $x_1u_1u_2x_2 \in L$ and $y_1u_3u_4y_2 \in L$. Since L is closed with respect to r, we have $x_1u_1u_4y_2 \in L$ and $y_1u_3u_2x_2 \in L$. Consequently, $p \in Q_{u_1u_4}$ and $y_2 \in C_{\mathscr{R},p}(u_1u_4)$, $q \in Q_{u_3u_2}$ and $x_2 \in C_{\mathscr{R},q}(u_3u_2)$, which proves (1) and (2).

Assume now that (1) and (2) hold for the rule $r = u_1 \mid u_2 \,\$\, u_3 \mid u_4$. Let $w', w'' \in A^*$ be such that $x, y \in L$ exist so that $(x, y) \vdash_r (w', w'')$. By definition, $x = x_1u_1u_2x_2$, $y = y_1u_3u_4y_2$, $w' = x_1u_1u_4y_2$ and $w'' = y_1u_3u_2x_2$. Now, for $p = \delta(q_0, x_1)$ and $q = \delta(q_0, y_1)$, we have $(p, q) \in Q_{u_1u_2} \times Q_{u_3u_4}$, $x_2 \in C_{\mathscr{R},p}(u_1u_2) \subseteq C_{\mathscr{R},q}(u_3u_2)$, $y_2 \in C_{\mathscr{R},q}(u_3u_4) \subseteq C_{\mathscr{R},p}(u_1u_4)$. This means that $w'' = y_1u_3u_2x_2 \in L$ and $w' = x_1u_1u_4y_2 \in L$, which proves that L is closed with respect to the rule r.

Now, let $I \subseteq L$ and let L be closed with respect to each rule in R. Let us prove that $L(S_{PA}) = \bigcup_{i \geqslant 0} \sigma^i(I) \subseteq L$, i.e., that for each $w' \in L(S_{PA})$ we have $w' \in L$, by induction on the minimal i such that $w' \in \sigma^i(I)$. Clearly, if $i = 0$, then $w' \in I \subseteq L$. Assume now that $i > 0$. Then there exists a rule r and $x, y \in \sigma^{i-1}(I)$, $w'' \in L(S_{PA})$ such that $(x, y) \vdash_r (w', w'')$ or $(x, y) \vdash_r (w'', w')$. By induction hypothesis, $x, y \in L$ and by using Definition 7.1, $w', w'' \in L$. □

Let $S_{[x]} = (A, I, R)$ be the splicing system associated with $L([x])$. In the next two propositions we prove that $L([x])$ is closed with respect to each rule in R. Since $I \subseteq L([x])$, in view of Proposition 7.1, we obtain $L(S_{[x]}) \subseteq L([x])$.

Proposition 7.2. *Let $L([x]) \subseteq A^*$ be the marker language associated with a marker $[x]$. Let $S_{[x]} = (A, I, R)$ be the splicing system associated with $L([x])$. Then $L([x])$ is closed with respect to each rule as in Eq.* (5).

Proof. Let $r'_{c_j} = 1 \mid (c_j)^R (z'')^R x' \$ 1 \mid (z'')^R x'$ be as in Eq. (5). By definition, there exists $z = z_1 x' z_2 \in I$ and a factor c_j of $(z_1)^R$, i.e., $(z_1)^R = z'' c_j z'$, which is a cycle in $\mathscr{A}_R$ on $q = \delta_R(q, c_j) = \delta_R(q_{0_R}, z'') \in (Q_R)_{c_j}$. Let $x_1, x_2 \in L([x])$ be such that $(x_1, x_2) \vdash_{r'_{c_j}} (w', w'')$. By definition, $k, h, y_2, y'_2 \in A^*$ exist so that

$$x_1 = k(c_j)^R (z'')^R x' y_2, \quad x_2 = h(z'')^R x' y'_2,$$

$$w' = k(z'')^R x' y'_2, \quad w'' = h(c_j)^R (z'')^R x' y_2.$$

Furthermore, since $x' \in M([x])$ and $[x]$ is a marker, we have $\delta(p_x, y_2) = \delta(q_0, k(c_j)^R (z'')^R x' y_2) = \delta(q_0, x_1) \in F$, i.e., $y_2 \in L_2$ and $\delta(p_x, y'_2) = \delta(q_0, h(z'')^R\, x' y'_2) = \delta(q_0, x_2) \in F$, i.e., $y'_2 \in L_2$. In addition, $\delta(q_0, k(c_j)^R (z'')^R) = q_x = \delta(q_0, h(z'')^R)$, i.e., $k(c_j)^R (z'')^R, h(z'')^R \in L_1$.

In order to prove that $L([x])$ is closed with respect to r'_{c_j}, we must prove that $k(z'')^R x' y'_2, h(c_j)^R\, (z'')^R x' y_2 \in L([x])$. Then, since $y'_2, y_2 \in L_2$ and $x' \in [x]$, we must prove that $k(z'')^R, h(c_j)^R\, (z'')^R \in L_1$, i.e., $z'' k^R, z'' c_j h^R \in L(\mathscr{A}_R)$. By hypothesis, $z'' c_j k^R, z'' h^R \in L(\mathscr{A}_R)$. Thus, since c_j is a cycle in $\mathscr{A}_R$ on q and $\mathscr{A}_R$ is deterministic, we have

$$\begin{aligned}\delta_R(q_{0_R}, z'' k^R) &= \delta_R(\delta_R(q_{0_R}, z''), k^R) \\ &= \delta_R(\delta_R(q_{0_R}, z'' c_j), k^R) = \delta_R(q_{0_R}, z'' c_j k^R) \in F_R\end{aligned}$$

and

$$\delta_R(q_{0_R}, z'' c_j h^R) = \delta_R(q, c_j h^R) = \delta_R(q, h^R) = \delta_R(q_{0_R}, z'' h^R) \in F_R.$$

So $z'' k^R \in (L_1)^R$, $z'' c_j h^R \in (L_1)^R$, i.e., $w' = k(z'')^R x' y'_2 \in L([x])$ and $w'' = h(c_j)^R (z'')^R x' y_2 \in L([x])$. □

Proposition 7.3. *Let $L([x]) \subseteq A^*$ be the marker language associated with a marker $[x]$. Let $S_{[x]} = (A, I, R)$ be the splicing system associated with $L([x])$. Then $L([x])$ is closed with respect to each rule $r \in R$ and so $L(S_{[x]}) \subseteq L([x])$.*

Proof. We refer to the set of the states in the minimal finite state automaton $\mathscr{A}$ recognizing $L([x])$. In Proposition 7.2, we have proved the statement for the rules as in Eq. (5). So, let us consider the rules as in Eqs. (4), (6), i.e., r_{c_j} and, if $[x]$ is a cyclic class, r_{x_M, x_M}.

Let us firstly prove that for each rule $r = u_1 \mid u_2 \$ u_3 \mid u_4$ as in Eqs. (4), (6), we have $Q_{u_1 u_2} = Q_{u_3 u_4} = Q_{u_1 u_4} \subseteq Q_{u_3 u_2}$. Indeed, since $|Q_{x'}| = 1$, for $r_{c_j} = x' z' \mid 1 \$ x' z' c_j \mid 1$ we have $Q_{x'z'} = Q_{x'z'c_j} = \{q_x\}$ and for $r_{x_M, x_M} = x_M \mid 1 \$ 1 \mid x_M$ we have $Q_{x_M} = Q_{x_M x_M} = \{q_x\} \subseteq Q_1 = Q$.

Now, let us prove that $C_{\mathscr{R},q}(u_1 u_2) = C_{\mathscr{R},q}(u_3 u_2)$ and $C_{\mathscr{R},q}(u_3 u_4) = C_{\mathscr{R},q}(u_1 u_4)$ for $q = q_x$, $r = r_{c_j}$ and, if $[x]$ is a cyclic class, for $r = r_{x_M, x_M}$. Indeed, for r_{c_j} we have $C_{\mathscr{R},q_x}(x'z') = C_{\mathscr{R},q_x}(x'z'c_j)$ (c_j is a cycle) and for r_{x_M, x_M} we have $C_{\mathscr{R},q_x}(x_M) = C_{\mathscr{R},q_x}(1) = C_{\mathscr{R},q_x}(x_M x_M)$ (x_M and $x_M x_M$ are q_x-labels, in view of Proposition 5.1).

As a result, since $I \subseteq L([x])$ and $L([x])$ is closed with respect to each rule in R, then $L(S_{[x]}) \subseteq L([x])$ in view of Proposition 7.1. □

Proof of Theorem 6.1. By using Propositions 6.4 and 7.3, Theorem 6.1 can be proved. □

The next result shows a relationship between the splicing operation and classes in a syntactic monoid: if $L(\mathscr{A})$ is closed with respect to $r = u_1 \mid u_2 \$ u_3 \mid u_4$ then $L(\mathscr{A})$ is closed with respect to $[r] = [u_1] \mid [u_2] \$ [u_3] \mid [u_4]$. This statement has been independently proved also in [9].

Proposition 7.4. *Let $S_{PA} = (A, I, R)$ be a finite Paun splicing system, let $L \subseteq A^*$ be a regular language. If L is closed with respect to $r = u_1 \mid u_2 \$ u_3 \mid u_4$ and, for $u'_i \in A^*$, $i \in \{1, 2, 3, 4\}$, we have $u_i \equiv_L u'_i$, then L is closed with respect to $u'_1 \mid u'_2 \$ u'_3 \mid u'_4$.*

Proof. Let us suppose that L is closed with respect to $r = u_1 \mid u_2 \$ u_3 \mid u_4$ and let $u'_i \in A^*$ be such that $u_i \equiv_L u'_i$, $i \in \{1, 2, 3, 4\}$. Then, in virtue of Remark 2.1, we have $Q_{u'_1 u'_2} = Q_{u_1 u_2}$ and $Q_{u_3 u_4} = Q_{u'_3 u'_4}$. Furthermore, by using Proposition 7.1 and Remark 7.1, for each pair $(p, q) \in Q_{u_1 u_2} \times Q_{u_3 u_4} = Q_{u'_1 u'_2} \times Q_{u'_3 u'_4}$ we have $(p, q) \in Q_{u_1 u_4} \times$

$Q_{u_3u_2} = Q_{u'_1u'_4} \times Q_{u'_3u'_2}$, $C_{\mathscr{R},p}(u'_1u'_2) = C_{\mathscr{R},p}(u_1u_2) \subseteq C_{\mathscr{R},q}(u_3u_2) = C_{\mathscr{R},q}(u'_3u'_2)$ and $C_{\mathscr{R},q}(u'_3u'_4) = C_{\mathscr{R},q}(u_3u_4) \subseteq C_{\mathscr{R},p}(u_1u_4) = C_{\mathscr{R},p}(u'_1u'_4)$. Thus, in view of Proposition 7.1, L is closed with respect to $u'_1 \mid u'_2 \,\$\, u'_3 \mid u'_4$. □

8. Marker and constant languages

As mentioned in Section 1, it is natural to investigate the relationship between constant languages and marker languages. In this section we will give results in this direction along with some observations about the definition of a marker. Let us first recall the notion of a constant for a regular language, given by Schützenberger in [23].

Definition 8.1. A word $z \in A^+$ is a *constant* for a regular language L if $A^*zA^* \cap L \neq \emptyset$ and $C(z) = C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$.

Definition 8.2. Let L be a regular language, let $\mathscr{A} = (Q, A, \delta, q_0, F)$ be the minimal automaton recognizing L. A word $w \in A^+$ is singular if $|Q_w| = 1$.

Remark 8.1. Let $w \in A^+$. By Remark 2.1, we already know that $[w]$ is a singular class if and only if w is a singular word.

Remark 8.2. Obviously, if z is a constant for L and $z' \in A^*$ is such that $zz', z'z \in \mathit{Fact}(L)$, then zz' and $z'z$ are also constants [7]. Furthermore, if $z \equiv_L z'$, then z' is also a constant for L ($\mathscr{A}$ is trim) [7]. Finally, if $[x]$ is a singular class (in particular, if $[x]$ is a marker for L), it is clear that x is a constant for L. Indeed, for each $z \in A^*$, we always have $C(z) \subseteq C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$. In addition, if $(z_1, z_2) \in C_{\mathscr{L}}(x) \times C_{\mathscr{R}}(x)$, then $\delta(q_0, z_1) = q_x$, $\delta(p_x, z_2) \in F$ and $\delta(q_0, z_1xz_2) \in F$, where $Q_x = \{q_x\}$ and $p_x = \delta(q_x, x)$.

A characterization of constants, which is more or less folklore, is given in Proposition 8.1.

Proposition 8.1. *Let $L \subseteq A^*$ be a regular language and let $\mathscr{A}$ be the minimal finite state automaton recognizing L. A word $z \in A^+$ is a constant for L if and only if $Q_z \neq \emptyset$ and there exists $q_z \in Q$ such that for all $q \in Q_z$ we have $\delta(q, z) = q_z$.*

Proof. Let $z \in A^+$ and suppose that z is a constant for a regular language L. By definition, $Q_z \neq \emptyset$. By contradiction, suppose that q, p, q_1, q_2 exist with $q, p \in Q_z, q_1, q_2 \in Q, q_1 \neq q_2$ and $\delta(q, z) = q_1$, $\delta(p, z) = q_2$. Let $y \in L_{q_1}$. Then, $y \in C_{\mathscr{R}}(z)$. In addition, since $\mathscr{A}$ is trim, $x' \in A^*$ exists such that $\delta(q_0, x') = p$. Since $(x', y) \in C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$ and z is a constant, we have $\delta(q_0, x'zy) = \delta(q_2, y) \in F$, i.e., $y \in L_{q_2}$. Thus, $L_{q_1} \subseteq L_{q_2}$. A symmetrical argument shows that $L_{q_2} \subseteq L_{q_1}$, i.e., $L_{q_1} = L_{q_2}$, which is in contradiction with Proposition 2.1. Conversely, let $z \in A^+$. Suppose $Q_z \neq \emptyset$ and that $q_z \in Q$ exists such that, for all $q \in Q_z$, $\delta(q, z) = q_z$. Then, $A^*zA^* \cap L \neq \emptyset$, since $\mathscr{A}$ is trim. Furthermore, we always have $C(z) \subseteq C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$. In addition, for each $y \in C_{\mathscr{R}}(z)$, we have that $\delta(q_z, y) \in F$. Thus, for each $(x, y) \in C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$, it holds that $\delta(q_0, xzy) = \delta(q_z, y) \in F$ and $(x, y) \in C(z)$, i.e., $C(z) = C_{\mathscr{L}}(z) \times C_{\mathscr{R}}(z)$. Then, z is a constant for L. □

In Section 1, we mentioned that Head gave a class of regular languages which belong to $S_{PA}(\mathit{Fin}, \mathit{Fin})$ [13]. To be more precise, Head showed that a regular language L is generated by finite linear splicing systems with some special rules (*one-sided context*) if and only if $L = \bigcup_{m \in \mathscr{M}} L(m) \cup X$, where X is a finite subset of A^*, $\mathscr{M} \subseteq A^*$ is a finite set of constants for L and, for each $m \in \mathscr{M}$, there exist $L_{1,m}, L_{2,m} \subseteq A^*$ such that $L(m) = L_{1,m}mL_{2,m}$. We say that $L(m)$ is a *constant* language and L is a *finite union of constant languages*.

Some marker languages are constant languages. For example, if $[x]$ is a marker with $[x]$ being finite, then the marker language $L([x])$ associated with $[x]$ is a finite union of constant languages. Nevertheless, the class of marker languages is not comparable with the class of constant languages with respect to the order of set inclusion, as Propositions 8.2 and 8.3 show.

Proposition 8.2. *Let $A = \{a, b, c\}$, let $L = a^*bc + cbc = (a^* + c)bc \subseteq A^*$. The regular language L is a constant language which is not a marker language.*

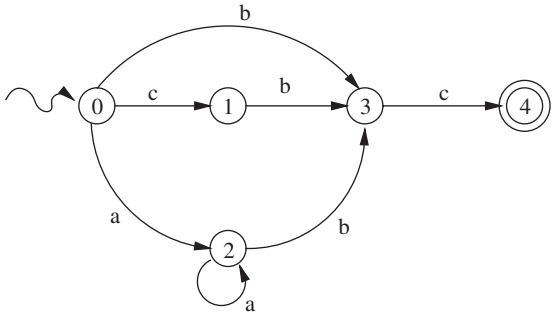

Fig. 4. A constant language which is not a marker language.

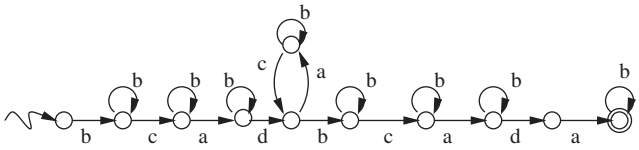

Fig. 5. A marker language which is not a finite union of constant languages.

Proof. It is easy to see that the finite state automaton depicted in Fig. 4 is the minimal automaton recognizing L. Then, in view of Proposition 8.1, b is a constant for L and $L = L_1 b L_2$ is a constant language, where $L_1 = a^* + c$ and $L_2 = \{c\}$. By contradiction, suppose that L is a marker language, i.e., $L = L'_1[x]_1 L'_2$ with $[x]$ being a marker. No word in $\{a, b\}A^* \cap Fact(L)$ is singular and c is not singular. Then, looking at Definition 5.3, either $[x] = [cb]$ or $[x] = [cbc]$. In addition, by using Theorem 2.1, we have $[cb] = \{cb\}$ and $[cbc] = \{cbc\}$. As a result, we have either $L = L'_1 cb L'_2$ or $L = L'_1 cbc L'_2$ and in each case we have a contradiction since $abc \in L$ and $abc \cap A^*\{cb, cbc\}A^* = \emptyset$. □

The next proposition shows that there exist marker languages which are not a finite union of constant languages.

Proposition 8.3. *The regular language* $L = L_1(ab^*c)^* L_2 = L_1(ab^*c)^* L_1 ab^*$, *with* $L_1 = b^+cb^*ab^*d$ *and* $L_2 = b^+cb^*ab^*dab^* = L_1 ab^*$, *is a marker language which is not a finite union of constant languages.*

Proof. It is easy to see that the finite state automaton depicted in Fig. 5 is the minimal automaton recognizing L. Then, let $x \in (ab^*c)^+$. We have $|Q_x| = 1$. Moreover, in view of Theorem 2.1, for each $y \in A^*$ we have $x \equiv_L y$ if and only if $y \in (ab^*c)^+$, i.e., $[x] = (ab^*c)^+$. Since $[x]$ is a cyclic and singular class, then $[x]$ is a marker for L (Definition 5.3). Since $L = L_1(ab^*c)^* L_2$, then L is a marker language (Definition 5.4).

If m is a constant for L, then $m \in Fact(L)$ and we now prove that either there exist $x, x' \in A^*$, $i \in \boldsymbol{N}$ such that $m = xab^i cx'$ or there exist $y, y' \in A^*, j, j' \in \boldsymbol{N}$, $j' > 0$, such that $m = yb^j cb^{j'} cy'$. Indeed, on the one hand if z is a word which has no factor with the form $ab^i c$ or with the form $b^j cb^{j'} c$ and $z \in Fact(L)$, i.e., $z_1 z z_2 = l_1 l_3 l_2$, with $z_1, z_2 \in A^*$, $l_1 \in L_1$, $l_2 \in L_2$, $l_3 \in (ab^*c)^*$, then either a prefix ab^k of l_3 exists such that z is a factor of $l_1 ab^k$ or z is a factor of l_2 and in both the cases $z \in Fact(L_2)$. On the other hand, each word $z \in Fact(L_2)$ cannot be a constant for L. Indeed, by contradiction, suppose that there exists $z \in Fact(L_2)$ such that z is a constant for L. Let $l_2 \in L_2$ be such that $l_2 = z_1 z z_2$, $z_1, z_2 \in A^*$. Thus, there exist $l_1 \in L_1$ and $k \in \boldsymbol{N}$ such that $l_2 = l_1 ab^k = z_1 z z_2$. We also know that $l_1 ab^k c l_1 ab^k = z_1 z z_2 c l_1 ab^k = l_1 ab^k c z_1 z z_2 \in L$. As a result, we have $z_1 \in C_{\mathscr{L}}(z)$, $z_2 \in C_{\mathscr{R}}(z)$, but $(z_1, z_2) \notin C(z)$, since $z_1 z z_2 = l_2 \notin L$. Moreover, each element in $(ab^*c)^+$ is a constant for L, since $(ab^*c)^+$ is a marker and so, if $m = xab^i cx' \in Fact(L)$ then m is a constant for L (Remark 8.2). Also each element z' in b^*cb^+c is a constant for L, since $|Q_{z'}| = 1$ and so, if $m = yb^j cb^{j'} cy' \in Fact(L)$ then m is a constant for L (Remark 8.2). Consequently, each finite set of constants for L is a subset of $Fact(L)$ with the form $\{x_i ab^{k_i} cx'_i \mid k_i \in K\} \cup \{y_i b^{s_i} cb^{j_i} cy'_i \mid s_i \in S, j_i \in J\}$, K, S, J being finite subsets of $\boldsymbol{N}$ with $0 \notin J$.

Let us now show that L is not a finite union of constant languages. By contradiction, let $\mathscr{M}$ be a finite set of constants such that $L = \bigcup_{m\in\mathscr{M}} L(m) \cup X$, X being a finite set. As we have already said, there exist finite subsets K, S, J, of $\mathbf{N}$, with $0 \notin J$, such that $\mathscr{M} = \{x_i ab^{k_i} cx'_i \mid k_i \in K\} \cup \{y_i b^{s_i} cb^{j_i} cy'_i \mid s_i \in S,\ j_i \in J\}$.

With the same notations as in $\mathscr{M}$, let $w_t = zab^{t+k_m} cb^{t+k_m} cz' \in L$ with $z \in L_1$, $b^{t+k_m} cz' \in L_2$, $t > 0$ and $k_m = \max(\{k_i \mid k_i \in K\} \cup \{j_i \mid j_i \in J\} \cup \{s_i \mid s_i \in S\})$. Observe that $z' \in b^*ab^*dab^*$. Let us prove that for each $f \in \mathscr{M}$ we have $w_t \notin A^* f A^*$.

Indeed, we cannot write $zab^{t+k_m} cb^{t+k_m} cz' = y' f x'$, with $f \in \{x_i ab^{k_i} cx'_i \mid k_i \in K\}$ (since $ab^{k_i} c \notin Fact(z)$, $ab^{k_i} c \notin Fact(b^{t+k_m} cz')$, no proper suffix (resp. prefix) of $ab^{t+k_m} c$ can be a proper prefix (resp. suffix) of $ab^{k_i} c$ and $k_m + t \notin K$). Nor can we write $zab^{t+k_m} cb^{t+k_m} cz' = y' y_i b^{s_i} cb^{j_i} cy'_i x'$. Indeed, since $z' \in b^*ab^*dab^*$ then $c \notin alph(z') = Fact(z') \cap A$ and so z' is a suffix of $y'_i x'$. Consequently, $x'' \in A^*$ exists such that $zab^{t+k_m} cb^{t+k_m} c = y' y_i b^{s_i} cb^{j_i} cx''$. In addition, since $z \in L_1$ and za is a prefix of $y' y_i b^{s_i} cb^{j_i} cx''$, then z is a prefix of $y' y_i$, i.e., $y'' \in A^*$ exists such that $ab^{t+k_m} cb^{t+k_m} c = y'' b^{s_i} cb^{j_i} cx''$. Therefore, $b^{s_i} cb^{j_i} c \in Fact(ab^{t+k_m} cb^{t+k_m} c)$, which is a contradiction.

As a result, $w_t \in X$. Since $\{w_t \mid w_t = zab^{t+k_m} cb^{t+k_m} cz', t > 0\}$ is not a finite set, this is a contradiction. □

We end this section with observations concerning the definition of a marker. Precisely, we can imagine different variants of the definition of a marker given in this paper and of the class of languages associated with it. We introduce two of these variants along with results concerning the corresponding families of languages (Propositions 8.4 and 8.5).

Firstly, consider regular languages L with the form $L = L_1 m L_2$, m being a singular word. By Remark 8.2, m is a constant for L and, in view of the above-mentioned Head's result, L is generated by finite Paun splicing systems. Therefore, we can enlarge the class of markers for a regular language by adding singular classes (not necessarily finite or cyclic). Thus, Proposition 8.4 shows that these new marker languages are once again generated by finite Paun splicing systems.

Proposition 8.4. *Let $L = L_1 m L_2$ be a regular language with m being a singular word. Then, $L = L_1[m]L_2$ with $[m]$ being a singular class and L is generated by a finite Paun splicing system.*

Proof. Let L be a regular language. We already know that m is singular if and only if $[m]$ is singular (Remark 8.1). Furthermore, in view of Remark 8.2, for each $m' \in [m]$, m' is a constant for L. In addition, in view of Remark 2.1, for each $m' \in [m]$, we have $Q_{m'} = Q_m = \{q\}$ and in view of Theorem 2.1, we have $\delta(q, m) = p = \delta(q, m')$. Consequently, if $L = L_1 m L_2$ we have $L_1 = \{x \in A^* \mid \delta(q_0, x) = q\} = C_{\mathscr{L}}(m')$, $L_2 = \{y \in A^* \mid \delta(p, y) \in F\} = C_{\mathscr{R}}(m')$. So, looking at Definition 8.1, for the regular languages $L_1 = L(mL_2)^{-1}$, $L_2 = (L_1 m)^{-1} L$ we have $L_1[m]L_2 \subseteq L = L_1 m L_2$, i.e. $L = L_1[m]L_2$. □

As another example, we can define a marker as a set $w[x]$, (resp. $[x]w$) where wx (resp. wx) is a singular word, $[x] \in \mathscr{M}(L)$, and $Q_{wx} \neq \emptyset$ (resp . $Q_{xw} \neq \emptyset$). A further investigation should be done in order to state whether the corresponding languages are generated by a finite Paun linear splicing system.

Proposition 8.5. *Let $L = L_1 w[x] L_2$ be a regular language where wx is a singular word and $[x] \in \mathscr{M}(L)$. Then, we have $L = L_1[wx]L_2$.*

Proof. Since $w[x] \subseteq [w][x] \subseteq [wx]$, we obviously have $L_1 w[x]L_2 \subseteq L_1[wx]L_2$. Now, we can use the same argument as in Proposition 8.4. In view of Remark 2.1, for each $m' \in [wx]$, we have $Q_{m'} = Q_{wx} = \{q\}$. Furthermore, in view of Theorem 2.1, we have $\delta(q, wx) = p = \delta(q, m')$. In addition, by using Remark 8.2, each element in $[wx]$ is a constant for L. Consequently, if $L = L_1 w[x] L_2$, we have $L_1 = \{y_1 \in A^* \mid \delta(q_0, y_1) = q\} = C_{\mathscr{L}}(m')$, $L_2 = \{y_2 \in A^* \mid \delta(p, y_2) \in F\} = C_{\mathscr{R}}(m')$.

So, looking at Definition 8.1, for the regular languages $L_1 = L(w[x]L_2)^{-1}$, $L_2 = (L_1 w[x])^{-1} L$ we have $L_1[wx]L_2 \subseteq L = L_1 w[x] L_2$, i.e. $L = L_1[wx]L_2$. □

Clearly, a dual version of Proposition 8.5 can be stated by considering $[x]w$ instead of $w[x]$, once again with xw being a singular word and $[x] \in \mathscr{M}(L)$.

Acknowledgements

The authors wish to thank J.E. Pin for useful discussions on properties of the syntactic monoid and D. Pixton who allowed us to avail reference [9]. Many thanks to the anonymous referees. The authors are truly indebted to one of them in particular: his/her careful work of reading the submitted version of this paper allowed them to get a more readable paper.

References

[1] J. Berstel, D. Perrin, Theory of Codes, Academic Press, New York, 1985.
[2] P. Bonizzoni, C. De Felice, G. Mauri, R. Zizza, Decision problems on linear and circular splicing, in: M. Ito, M. Toyama (Eds.), Proceedings of DLT 2002, Lecture Notes in Computer Science, vol. 2450, Springer, Berlin, 2003, pp. 78–92.
[3] P. Bonizzoni, C. De Felice, G. Mauri, R. Zizza, Regular languages generated by reflexive finite linear splicing systems, Proceedings of DLT 2003, Lecture Notes in Computer Science, vol. 2710, Springer, Berlin, 2003, pp. 134–145.
[4] P. Bonizzoni, C. De Felice, G. Mauri, R. Zizza, Circular splicing and regularity, Theoret. Inform. Appl. 38 (2004) 189–228.
[5] P. Bonizzoni, C. De Felice, R. Zizza, The structure of reflexive regular splicing languages via Schützenberger constants, Theoret. Comput. Sci. 334 (2005) 71–98.
[6] K. Culik, T. Harju, Splicing semigroups of dominoes and DNA, Discrete Appl. Math. 31 (1991) 261–277.
[7] A. de Luca, A. Restivo, A characterization of strictly locally testable languages and its applications to subsemigroups of a free semigroup, Inform. and Control 44 (1980) 300–319.
[8] S. Eilenberg, Automata, Languages and Machines, vol. A, Academic Press, New York, 1974.
[9] E. Goode Laun, Constants and Splicing Systems, Ph.D. Thesis, Binghamton University, 1999.
[10] E. Goode, D. Pixton, Recognizing splicing languages: syntactic monoids and simultaneous pumping, 2004, submitted for publications (available from http://www.math.binghamton.edu/dennis/Papers/index.html).
[11] M.A. Harrison, Introduction to Formal Language Theory, Addison-Wesley, Reading, MA, 1978.
[12] T. Head, Formal Language Theory and DNA: an analysis of the generative capacity of specific recombinant behaviours, Bull. Math. Biol. 49 (1987) 737–759.
[13] T. Head, Splicing languages generated with one sided context, in: Gh. Paun (Ed.), Computing with Bio-molecules. Theory and Experiments, Springer, Singapore, 1998.
[14] T. Head, Gh. Paun, D. Pixton, Language theory and molecular genetics: generative mechanisms suggested by DNA recombination, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 2, Springer, Berlin, 1996, pp. 295–360.
[15] J.E. Hopcroft, R. Motwani, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 2001.
[16] S.M. Kim, Computational modeling for genetic splicing systems, SIAM J. Comput. 26 (1997) 1284–1309.
[17] R. McNaughton, S. Papert, Counter-Free Automata, MIT Press, Cambridge, MA, 1971.
[18] G. Paun, On the splicing operation, Discrete Appl. Math. 70 (1996) 57–79.
[19] G. Paun, G. Rozenberg, A. Salomaa, DNA computing, New Computing Paradigms, Springer, Berlin, 1998.
[20] D. Perrin, Finite automata, in: J. Van Leeuwen (Ed.), Handbook of Theoretical Computer Science, vol. B, Elsevier, Amsterdam, 1990, pp. 1–57.
[21] J.-E. Pin, Variétés de langages formels, Masson, Paris, 1984 (English translation: Varieties of formal languages, Plenum, New-York, 1986).
[22] D. Pixton, Regularity of splicing languages, Discrete Appl. Math. 69 (1996) 101–124.
[23] M.P. Schützenberger, Sur certaines opérations de fermeture dans les langages rationnels, Sympos. Math. 15 (1975) 245–253.