

Right and left locally testable languages

Pedro García*, José Ruiz

*Departamento de Sistemas Informáticos y Computación. Universidad Politécnica,
Camina de Vera, s/n 46071 Valencia, Spain*

Received January 1998; revised December 1998

Communicated by D. Perrin

Abstract

The aim of this paper is to give a definition as well as an algebraic characterization of two new varieties of languages that will be referred to as *right* (respectively *left*) *locally testable languages* and denoted as RLT (resp. LLT). Both families strictly contain the class of locally testable languages. Given $k > 0$, the membership of a word x to a RLT (k -RT) language can be decided by means of exploring the prefix and suffix of length $k - 1$ of x and the segments of length k , as well as considering the order of appearance of those segments when we scan the prefixes of x . Membership of x to a k -LT can be decided in a similar way, but we have to change the word “prefixes” for “suffixes” in the above description. In this paper we also show that S is a syntactic semigroup of a k -RT (resp. k -LT) language if and only if the local subsemigroups of S are idempotents and *right* (resp. *left*) *repetition free*. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Recognizable languages; Local testability; Semigroups

1. Introduction

The objective of this paper is to provide a definition as well as an algebraic characterization of two new varieties of languages, the *right* (RLT) and *left* (LLT) *locally testable languages* which strictly include the well-known family of locally testable languages (LT) and which are related to the LT in the same way that right (RPWT) and left (LPWT)¹ piecewise testable languages are related to piecewise testable languages.

Locally testable languages were introduced by McNaughton [9] and algebraically characterized as those languages whose syntactic semigroup is both locally idempotent and locally commutative by Brzozowsky and Simon [2] and by Zalcstein [16]. Given

* Corresponding author.

E-mail address: pgarcia@dsic.upv.es (P. García).

¹ In [5], these languages are denoted as right (left) locally testable. The authors consider that the name given here is more appropriate.

an integer $k > 0$, we say that L is k -testable (k - T) if, given a word $x \in L$, any other word having the same prefix and suffix of length $k - 1$ and containing exactly the same segments of length k than x also belongs to L . The order of appearance of the segments in the words of L is immaterial. The family of k - T languages is a variety of languages and constitutes the boolean closure of the family of k -testable languages in the Strict Sense (this family is not a variety and has been characterized in [7]). A language is *locally testable* if it is k -testable for a value of k .

A natural extension of locally testable languages consists in dropping the condition for prefixes and suffixes in the definition. Those languages are called *strongly locally testable* and have been characterized by Beauquier and Pin in [1], while the strongly locally testable semigroups (concepts that do not correspond each other), have been characterized by Selmi in [12]. Another extension consists in counting the number of occurrences of the segments of length k in the words up to a certain threshold, these languages are called *threshold locally testable* and have been characterized by Straubing [14], Therien and Weiss [15]. Another natural extension are the languages that are described below.

Informally, a language L is called k -right testable (k -RT) (resp. k -left testable (k -LT)) if for a word $x \in L$, any other word y belongs to L if it satisfies the following conditions:

- (a) Begins with the same prefix of length $k - 1$ than x .
 - (b) Ends with the same suffix of length $k - 1$ than x .
 - (c) Contains the same segments of length k than x .
 - (d) The order of appearance of the first occurrences of the segments when we explore the string y left to right (resp. right to left) is the same than when we explore x .
- A language is called *right* (resp. *left*) *locally testable* if it is k -right (resp. left) testable for some value of k .

One should observe that if we change the term “segment” by the term “subword” in the definition of RLT and LLT languages we obtain the definition of right (RPWT) and left (LPWT) piecewise testable languages. These families of languages are related to the family of piecewise testable languages in the same way as RLT and LLT languages are related to LT languages.

The algebraic characterization of these families of languages is solved by proving a theorem that shows that we can relax one of the conditions of a theorem on graphs (Simon) [3] if we consider the order of appearance of the edges in the paths.

In the following, we will omit the proofs related to LLT, which are left–right dual to those of RLT.

2. Preliminaries and notation

We suppose that the reader is familiar with the rudiments of formal languages, semigroup theory and graphs. For further details the reader is referred to Hopcroft and Ullman [4], Pin [10] and Eilenberg [3].

Let Σ be a finite alphabet and Σ^* be the free monoid generated by Σ with concatenation as the internal law and ε as neutral element. A *language* L over Σ is a subset of Σ^* . The length of a word is denoted by $|x|$, while Σ^k represents the set of all words of length k over Σ . Given $x \in \Sigma^*$, if $x = uvw$ with $u, v, w \in \Sigma^*$, then u (resp. w) is called *prefix* (resp. *suffix*) of x , whereas v (also u and w) is called a *segment* of x . $\text{Pr}(L)$ (resp. $\text{Suf}(L)$) denotes the set of prefixes (suffixes) of L . If $X \subseteq \Sigma^*$ is a finite set, $\text{Card}(X)$ denotes the number of elements in X .

A deterministic finite automaton (DFA) is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is a finite alphabet, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states and δ is a partial function that maps $Q \times \Sigma$ in Q , which can be extended to words by establishing $\delta(q, \varepsilon) = q$ and $\delta(q, xa) = \delta(\delta(q, x), a), \forall q \in Q, \forall x \in \Sigma^*, \forall a \in \Sigma$.

A word x is accepted by an automaton A if $\delta(q_0, x) \in F$. The set of words accepted by A is denoted by $L(A)$. Given an automaton $A, \forall a \in \Sigma$, we can define the function $a^A : Q \rightarrow Q$ as $a^A(q) = \delta(q, a), \forall q \in Q$. For $x \in \Sigma^*$, the function $x^A : Q \rightarrow Q$ is defined inductively: ε^A is the identity on Q and $(xa)^A = x^A a^A, \forall a \in \Sigma$. Clearly, $\forall x, y \in \Sigma^*, (xy)^A = (x)^A (y)^A$. The set $\{a^A : a \in \Sigma\}$ is denoted by M_A . The set of functions $\{x^A : x \in \Sigma^+\}$ is a finite semigroup under the operation of composition of functions, and is denoted as S_A and called *semigroup of A*.

An output automaton is a quintuple $A = (Q, \Sigma, M, \delta, \lambda)$ where Q, Σ and δ are defined in the same way as in a DFA, M is a monoid and the output function λ is a function that maps $Q \times \Sigma$ in M , which can be extended to Σ^* by establishing $\lambda(q, \varepsilon) = 1$, with 1 being the identity of M and $\lambda(q, xa) = \lambda(q, x)\lambda(\delta(q, x), a)$.

Let $L \subseteq \Sigma^*$ and \equiv be an equivalence relation defined in Σ^* . We say that \equiv saturates L , if L is the union of equivalence classes modulo \equiv . An equivalence relation is called a congruence if it is both-sides compatible with the operation of the monoid. The special congruence \sim_L defined as $x \sim_L y \Leftrightarrow (\forall u, v \in \Sigma^*, uxv \in L \Leftrightarrow uyv \in L)$, is called the *syntactic congruence* of L and it is the coarsest congruence that saturates L . Σ^*/\sim_L is called the *syntactic semigroup of L* and is denoted as $S(L)$. The morphism $\varphi : \Sigma^* \rightarrow S(L)$, that maps each word to its equivalence class modulo \sim_L is called the *syntactic morphism of L*. An element $e \in S(L)$ is called *idempotent* if $e^2 = e$. The set of idempotents of $S(L)$ is denoted as $E(S(L))$.

A *labelled directed graph* G , with labels in Σ is given by two sets, a finite set of *vertices* V and a finite set of *edges* $E \subset V \times \Sigma \times V$. The edge (p, a, q) will sometimes be denoted as $p \xrightarrow{a} q$. Two edges (p, a, q) and (r, b, s) are *consecutive* if $q = r$. The set of paths of G is the subset of words in E^+ that does not contain any segments of length two whose edges are non-consecutive. If C is the set of non-consecutive edges in G , the set of paths in G is $P = E^+ - E^* C E^*$. The path $(q_0, a_1, q_1)(q_1, a_2, q_2) \dots (q_{n-1}, a_n, q_n)$ is denoted as $(q_0, a_1 a_2 \dots a_n, q_n)$. The path (p, x, q) , with $x \in \Sigma^+$ is denoted as $p \xrightarrow{x} q$ also. The function $\tau : P \rightarrow 2^E$ is defined by the following conditions:

- $\tau((p, a, q)) = \{(p, a, q)\}$.
- $\tau((p, x, q)(q, y, r)) = \tau((p, x, q)) \cup \tau((q, y, r))$.

For each path (p, x, q) , $\tau((p, x, q))$ gives the set of edges traversed by (p, x, q) without regard to order or multiplicity.

Given $k > 0$, $\forall x \in \Sigma^*$ the prefix and suffix of length $k - 1$ of x is denoted as $i_k(x)$ and $f_k(x)$, respectively, whereas the set of segments of length k of x is denoted as $t_k(x)$. If $\forall x \in \Sigma^*, \forall k > 0$ we define $v_k(x) = (i_k(x), t_k(x), f_k(x))$, the equivalence relation \equiv_k defined $\forall x, y \in \Sigma^*$ as $x \equiv_k y \Leftrightarrow v_k(x) = v_k(y)$ is the congruence that defines the family of k -testable languages. L is k -testable if it is saturated by the congruence \equiv_k . L is locally testable if it is k -testable for some $k > 0$.

3. Right and left locally testable languages

We are going to define the congruences $\equiv_{k,R}$ and $\equiv_{k,L}$ which are a refinement of the congruence \equiv_k in order to define the families of right and left locally testable languages, which is one of the main objectives of the present work.

3.1. The congruences $\equiv_{k,R}$ and $\equiv_{k,L}$

$\forall x, y \in \Sigma^*$ the relation $\equiv_{k,R}$ is defined as follows:

- (1) If $|x| < k$, $x \equiv_{k,R} y \Leftrightarrow x = y$.
- (2) If $|x| \geq k$, $x \equiv_{k,R} y$ if and only if
 - (a) $f_k(x) = f_k(y)$.
 - (b) $\forall u \in \text{Pr}(x) \exists v \in \text{Pr}(y) : t_k(u) = t_k(v)$.
 - (c) $\forall u \in \text{Pr}(y) \exists v \in \text{Pr}(x) : t_k(u) = t_k(v)$.

Note that $i_k(x) = i_k(y)$ follows from the definition. Informally, two words $x, y \in \Sigma^k \Sigma^*$ are $\equiv_{k,R}$ -equivalent if they are \equiv_k -equivalent and the order of appearance of new segments in both words when they are explored left to right is the same.

The relation $\equiv_{k,L}$ is defined in a similar way, by replacing the prefixes with the suffixes of x and y in the above definition.

It is easily seen that,

- $\equiv_{k,R}$ and $\equiv_{k,L}$ are congruences of finite index.
- Both $\equiv_{k,R}$ and $\equiv_{k,L}$, refine the congruence \equiv_k .

Definition 1. We say that a language L is k -RT (resp. k -LT) if it is saturated by the relation $\equiv_{k,R}$ (resp. $\equiv_{k,L}$). L is RLT (resp. LLT) if it is k -RT (resp. k -LT) for any value of $k \geq 1$.

Example 1. The language recognized by the automaton A in Fig. 1 is not 1-testable. It can be seen that $abc \in L(A)$, while $cba \notin L(A)$. One may verify that $L(A)$ is not locally testable as its syntactic semigroup is locally idempotent, but not locally commutative. However, $L(A)$ recognizes any word in which the order of appearance of the segments of length one is $\{a, b, c\}$, $\{b, a, c\}$ or $\{b, c, a\}$ and no other. Hence, according to the definition, $L(A)$ is 1-RT.

$L(B)$ is not locally testable either, as its syntactic semigroup is not locally commutative. However, if we choose the last appearance of the segments of length one

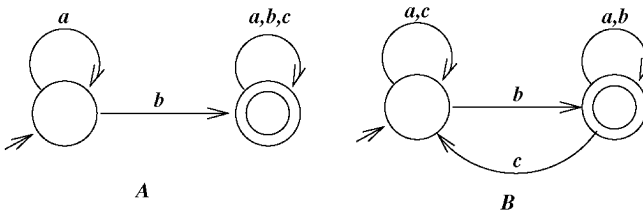


Fig. 1.

in the words accepted by B , the order in which they appear (read left to right) is $\{a, c, b\}$, $\{c, a, b\}$ or $\{c, b, a\}$ and no other, so $L(B)$ is 1-LT.

Proposition 1. Let $\equiv_{k,R}$ and $\equiv_{k,L}$ the congruences over Σ^* defined above. Then

- (1) $\forall x \in \Sigma^{k-1}\Sigma^*, \forall y, z \in \Sigma^* (xy = zx \Rightarrow xy \equiv_{k,R} xy^2)$.
- (2) $\forall x \in \Sigma^{k-1}\Sigma^*, \forall y \in \Sigma^* xyx \equiv_{k,R} xyxyx$.
- (3) $\forall x \in \Sigma^+, \forall y \in \Sigma^* x^n yx^n \equiv_{k,R} x^n yx^n yx^n (0 \leq k - 1 \leq n)$.
- (4) $\forall x \in \Sigma^{k-1}\Sigma^*, \forall y, z \in \Sigma^* xyxzxxyx \equiv_{k,R} xyxzx$.
- (5) $\forall x \in \Sigma^+, \forall y, z \in \Sigma^* x^n yx^n zx^n yx^n \equiv_{k,R} x^n yx^n zx^n (0 \leq k - 1 \leq n)$.

Proof. (1) As $xy^2 = zxy = zx$, it follows that $i_k(xy) = i_k(xy^2) = f_k(xy) = f_k(xy^2) = x$. On the other hand,

- (a) If $u \in \text{Pr}(xy)$ there exists $v \in \text{Pr}(xy^2)$ ($v = u$ in this case) such that $t_k(u) = t_k(v)$.
- (b) If $u \in \text{Pr}(xy^2)$ then,
 - (i) If $u \in \text{Pr}(xy)$ we take $v = u$.
 - (ii) Otherwise, $u = xyu'$ with $u' \in \text{Pr}(y)$. Let $v = xy$. It follows that $t_k(u) = t_k(xy u') = t_k(zx) \cup t_k(xu') = t_k(xy) \cup t_k(xu') = t_k(xy) = t_k(v)$, as $t_k(xu') \subseteq t_k(xy)$.
- (2) Replace in (1) the word y for yx , and z for xy .
- (3) Let m be such that $|x^n| = m - 1$. Then $x^n yx^n \equiv_{m,R} x^n yx^n yx^n$. As $m \geq k$, the congruence $\equiv_{m,R}$ is a refinement of $\equiv_{k,R}$ and the equality holds.
- (4) Both words begin and end with x . On the other hand, $t_k(xy x z x y x) = t_k(x y x z x) \cup t_k(x y x) = t_k(x y x z x)$. Hence the second time that xyx appears as a segment of $xy x z x y x$ it does not contribute with new segments to $t_k(xy x z x y x)$ and both words have the same segments of length k appearing in the same order.
- (5) It follows analogously to (3) and (4). \square

Remark 1. One may verify that items (1)–(3) of the above proposition follow analogously for the relation $\equiv_{k,L}$. Item (4) has to be replaced by $xyxzxxyx \equiv_{k,L} xzxxyx$, whereas item (5) has to be replaced by $x^n yx^n zx^n yx^n \equiv_{k,L} x^n zx^n yx^n$.

Definition 2. A semigroup S has the property of right (resp. left) repetition elimination if $\forall x, y \in S, xyx = xy$ (resp. $xyx = yx$). The semigroups having that property will be called right (resp. left) repetition free and will be denoted as *rrf* (resp. *lrf*).

Note that if both properties hold simultaneously, the semigroup is commutative.

Proposition 2. For every $k > 0$, $\Sigma^+ / \equiv_{k,R}$ (resp. $\Sigma^+ / \equiv_{k,L}$) is locally idempotent and locally rrf (resp. lrf).

Proof. Let $\pi_{k,R} : \Sigma^* \rightarrow \Sigma^* / \equiv_{k,R}$ be the natural projection associated to $\equiv_{k,R}$. Let e be an idempotent of $\Sigma^+ / \equiv_{k,R}$ and let us consider $s, t \in \Sigma^+ / \equiv_{k,R}$. There exists $x \in \Sigma^k \Sigma^*$ such that $\pi_{k,R}(x) = e$ and $y, z \in \Sigma^*$ such that $\pi_{k,R}(y) = s$ and $\pi_{k,R}(z) = t$.

(a) $ese = \pi_{k,R}(xyx) = \pi_{k,R}(xyxyx) = esese = (ese)(ese)$, so $\Sigma^+ / \equiv_{k,R}$ is locally idempotent.

(b) $(ese)(ete)(ese) = esetese = \pi_{k,R}(xyxzxyx) = \pi_{k,R}(xyxzx) = esete = (ese)(ete)$, so $\Sigma^+ / \equiv_{k,R}$ is locally commutative. \square

Proposition 3. $\Sigma^+ / \equiv_{1,R}$ (resp. $\Sigma^+ / \equiv_{1,L}$) is idempotent and rrf (resp. lrf).

Proof. The proof straightforwardly comes from Proposition 1. More precisely, we can take advantage of the fact that for every $y, z \in \Sigma^*$, $y \equiv_{1,R} y^2$ and $yz y \equiv_{1,R} yz$. \square

Proposition 4. The family of finite locally idempotent and locally rrf (resp. lrf) semigroups is a variety. The same fact occurs in relation with the family of idempotent and rrf (resp. lrf) semigroups.

4. A theorem on graphs

We are going to prove an extension of a theorem on graph congruences by Simon which originally appeared in [2], though the treatment of it as a separate result on graphs is due to Eilenberg [3]. In this theorem, we show that one can relax the condition of commutativity in the paths if we establish the order of appearance of the edges in the paths.

Definition 3. Given $x \in \Sigma^*$, $a \in \Sigma$, we can associate a sequence \mathcal{S} of transitions to each path $p \xrightarrow{x} q$ in the following way:

$$\mathcal{S}(p \xrightarrow{a} q) = \langle (p, a, q) \rangle,$$

$$\mathcal{S}(p \xrightarrow{x} r \xrightarrow{a} q) = \begin{cases} \mathcal{S}(p \xrightarrow{x} r) \wedge \langle (r, a, q) \rangle & \text{if } (r, a, q) \text{ does not appear in } p \xrightarrow{x} r, \\ \mathcal{S}(p \xrightarrow{x} r) & \text{otherwise,} \end{cases}$$

where sequences are denoted by $\langle \dots \rangle$, whereas \wedge denotes concatenation of sequences.

Theorem 1. Let M be a monoid and $A = (Q, \Sigma, M, \delta, \lambda)$ an output DFA where the output function $\lambda : Q \times \Sigma \rightarrow M$ has the property that for any state p and for any cycles $p \xrightarrow{y} p$,

$$\lambda(p, y) = \lambda(p, y^2), \tag{4.1}$$

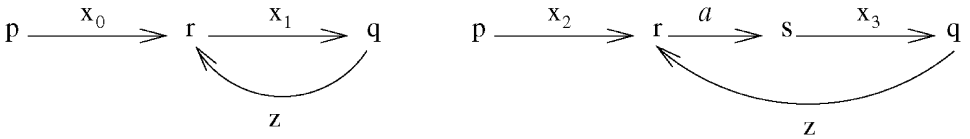


Fig. 2.

$$\lambda(p, yzy) = \lambda(p, yz). \tag{4.2}$$

Then for any paths $p \xrightarrow{x} q$ and $p \xrightarrow{y} q$ such that $\mathcal{S}((p \xrightarrow{x} q)) = \mathcal{S}((p \xrightarrow{y} q))$ we have that $\lambda(p, x) = \lambda(p, y)$.

We will first prove three lemmas. One should observe that in the lemmas we make no use of the order of appearance of the edges in the paths.

Lemma 1. Let $A = (Q, \Sigma, M, \delta, \lambda)$ be an output automaton as in Theorem 1. Given paths $p \xrightarrow{x} q \xrightarrow{y} r$ such that $\tau((q, y, r)) \subset \tau((p, x, q))$, $\exists x_0, x_1 : x = x_0x_1$ and $\lambda(p, x) = \lambda(p, x_0yx_1)$.

Proof. We proceed by induction on $|y|$. If $|y| = 0$ then $x_0 = x$ and $x_1 = \varepsilon$. Assuming that the lemma holds for $|y| \leq k$. Let $y = za$, $a \in \Sigma$, $z \in \Sigma^k$. Then there exists x_0, x_1 such that $x = x_0x_1$ and $\lambda(p, x) = \lambda(p, xzx_1)$ (see Fig. 2).

Let us prove that $x = (x_2a)x_3$ is the required factorization.

$$\begin{aligned} \lambda(p, x) &= \lambda(p, xzx_1) && \text{(by induction hypothesis)} \\ &= \lambda(p, x_0x_1zx_1) && (x = x_0x_1) \\ &= \lambda(p, x_0x_1zx_1zx_1) && (zx_1 \text{ is a cycle about } q \text{ and applying (4.1)}) \\ &= \lambda(p, xzx_1zx_1) && (x = x_0x_1) \\ &= \lambda(p, x_2ax_3zx_1zx_1) && (x = x_2ax_3) \\ &= \lambda(p, x_2ax_3zx_1zax_3zx_1) && (ax_3z \text{ and } x_1z \text{ are cycles about } r \text{ and applying (4.2)}) \\ &= \lambda(p, xzx_1zax_3zx_1) && (x = x_2ax_3) \\ &= \lambda(p, xzx_1zax_3) && (zx_1 \text{ and } zax_3 \text{ are cycles about } q \text{ and applying (4.2)}) \\ &= \lambda(p, xzax_3) && \text{(by induction hypothesis)} \\ &= \lambda(p, xyx_3) && (y = za). \quad \square \end{aligned}$$

Lemma 2. Let $A = (Q, \Sigma, M, \delta, \lambda)$ as in Theorem 1. Given paths $p \xrightarrow{x} q \xrightarrow{y} q$ such that $\tau((q, y, q)) \subset \tau((p, x, q))$, it follows that $\lambda(p, x) = \lambda(p, xy)$.

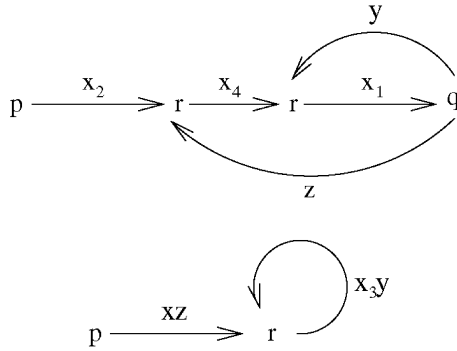


Fig. 3.

Proof. By Lemma 1, we have that, there exists x_0, x_1 such that $x = x_0x_1$ and $\lambda(p, x) = \lambda(p, xyx_1)$ with $p \xrightarrow{x} q \xrightarrow{x_1} q$.

Then $\lambda(p, x) = \lambda(p, xyx_1) = \lambda(p, x_0x_1yx_1) = \lambda(p, x_0x_1y) = \lambda(p, xy)$. \square

Lemma 3. Let $A = (Q, \Sigma, M, \delta, \lambda)$ be an output automaton as in Theorem 1. Given paths $p \xrightarrow{x} q \xrightarrow{y} r$ with $\tau((q, y, r)) \subset \tau((p, x, q))$ and $\tau((q, z, r)) \subset \tau((p, x, q))$ it follows that $\lambda(p, xy) = \lambda(p, xz)$.

Proof. By Lemma 1 we have that $\exists x_0, x_1 : x = x_0x_1$ and $\lambda(p, x) = \lambda(p, xyx_1)$ and also $\exists x_2, x_3 : x = x_2x_3$ with $\lambda(p, x) = \lambda(p, xzx_3)$. Let us suppose that x_2 is a prefix of x_0 (which means no loss of generality), that is, $x_0 = x_2x_4$ with $x_4x_1 = x_3$, which is described in Fig. 3.

Then, applying Lemma 2, $\lambda(p, xz) = \lambda(p, xzx_3y) = \lambda(p, xzx_3y) = \lambda(p, xy)$. \square

Proof of the Theorem. Let $p \xrightarrow{x} q$ such that $\mathcal{S}(p \xrightarrow{x} q) = \mathcal{S}(p \xrightarrow{y} q)$ and such that it complies with the rest of the hypothesis of the theorem. Let us see that $\lambda(p, x) = \lambda(p, y)$ by induction in the number of distinct edges appearing in the paths.

If the number of edges is one, we have two possibilities:

- (1) $p = q$, then necessarily $x = a^m, y = a^n$ and as we have $\lambda(p, a) = \lambda(p, a^2)$ and we have $\lambda(p, x) = \lambda(p, y)$.
- (2) $p \neq q$, then $x = a, y = a$ and the theorem also holds.

Suppose that x and y have $k + 1$ different edges. As $\mathcal{S}(p \xrightarrow{x} q) = \mathcal{S}(p \xrightarrow{y} q), \exists r \in V, \exists x_1 \in \text{Pr}(x), y_1 \in \text{Pr}(y)$ and there exist paths $p \xrightarrow{x_1} r$ and $p \xrightarrow{y_1} r$ having k different edges and such that $\mathcal{S}(p \xrightarrow{x_1} r) = \mathcal{S}(p \xrightarrow{y_1} r)$ and the paths $p \xrightarrow{x} q$ and $p \xrightarrow{y} q$ can be expressed as

$$\begin{aligned}
 p &\xrightarrow{x_1} r \xrightarrow{a} r' \xrightarrow{x_2} q, \\
 p &\xrightarrow{y_1} r \xrightarrow{a} r' \xrightarrow{y_2} q.
 \end{aligned}$$

Besides, we have that $\tau(r' \xrightarrow{x_2} q) \subset \tau(p \xrightarrow{x_1 a} r')$ and $\tau(r' \xrightarrow{y_2} q) \subset \tau(p \xrightarrow{y_1 a} r')$. By the induction hypothesis, $\lambda(p, x_1) = \lambda(p, y_1)$ and from Lemma 3 it follows that $\lambda(p, x) = \lambda(p, x_1 a x_2) = \lambda(p, y_1 a y_2) = \lambda(p, y)$. \square

5. Characterization of RLT languages

Definition 4. Let $A = (Q, \Sigma, \delta, q_0)$ be a DFA (in fact an initialized semiautomaton) and let $k \geq 1$. We say that A is k -RT if and only if $\forall x \in \Sigma^{k-1}, \forall y, z \in \Sigma^*$ the following conditions hold:

1. $xy = zx \Rightarrow (xy)^A = (xy^2)^A$,
2. $(xyxzx)A = (xyxzx)^A$.

Definition 5. Let $n = \text{Card}(Q)$. An automaton is RLT if and only if $\forall x \in \Sigma^+, \forall y, z \in \Sigma^*$ the following conditions hold:

1. $(x^n y x^n)^A = (x^n y x^n y x^n)^A$,
2. $(x^n y x^n z x^n y x^n)^A = (x^n y x^n z x^n)^A$.

Theorem 2. Let $A = (Q, \Sigma, \delta, q_0)$ be a DFA with $n = \text{Card}(Q)$ and let S_A the semigroup of transformations of A . A is RLT if and only if S_A is locally idempotent and locally *rrf*.

Proof. (\Rightarrow) Let $e \in E(S_A)$ and let $eS_A e$ be the local subsemigroup associated to e .

- (a) Let $s \in S_A$ and let $x, y \in \Sigma^+$ such that $y^A = s$ and $x^A = e$. It follows that $ese = (xyx)^A = (x^n y x^n)^A = (x^n y x^n y x^n)^A = e^n s e^n s e^n = esese = (ese)(ese)$.
- (b) Let $s, s' \in S_A$ and let $x, y, z \in \Sigma^+$ such that $x^A = e, y^A = s$ and $z^A = s'$. It follows that $(ese)(es'e)(ese) = (xyxzxxyx)^A = (x^n y x^n z x^n y x^n)^A = (x^n y x^n z x^n)^A = eses'e = (ese)(es'e)$.

(\Leftarrow) Let S_A be locally idempotent and locally *rrf*. Since S_A is locally idempotent, $(x^n)^A = (x^{n+1})^A$ for every $x \in \Sigma^+$ [2]. Therefore $(x^n)^A$ is an idempotent.

- (a) As $(x^n)^A S_A (x^n)^A$ is idempotent it follows that $\forall y \in \Sigma^*, (x^n y x^n)^A (x^n y x^n)^A = (x^n y x^n)^A$ and then $(x^n y x^n y x^n)^A = (x^n y x^n)^A$.
- (b) As $(x^n)^A S_A (x^n)^A$ is right repetition free, $\forall y, z \in \Sigma^*, \forall x \in \Sigma^+$, we have that $(x^n y x^n)^A (x^n z x^n)^A (x^n y x^n)^A = (x^n y x^n)^A (x^n z x^n)^A$, and then $(x^n y x^n z x^n y x^n)^A = (x^n y x^n z x^n)^A$. \square

Proposition 5. Let $A = (Q, \Sigma, \delta, q_0)$ be the canonical acceptor of L . L is k -RT implies that A is k -RT.

Proof. (a) Let $x \in \Sigma^{k-1}, y, z \in \Sigma^*$ such that $xy = zx$. As the congruence $\equiv_{k,R}$ saturates L and $xy \equiv_{k,R} xy^2$ it follows that $xy \sim_L xy^2$, with \sim_L being the syntactic congruence of L , and then $(xy)^A = (xy^2)^A$.

(b) Let $x \in \Sigma^{k-1}, y, z \in \Sigma^*$. By a similar argument we have that $xyxzx \equiv_{k,R} xyxzx$ following that $xyxzx \sim_L xyxzx$ and $(xyxzx)^A = (xyxzx)^A$. \square

Theorem 3. *A recognizable language $L \subset \Sigma^*$ is RLT if and only if $S(L)$ is locally idempotent and locally rrf.*

Proof. (\Rightarrow) If L is RLT there exists some $k \geq 1$ such that L is k -RT. Then, by Proposition 5, the canonical acceptor of L is RLT and (Theorem 2) then $S(L)$ is locally idempotent and locally rrf.

An alternative proof of this fact can be stated as follows: Since L is k -RT, $\Sigma^+ / \equiv_{k,R}$ recognices L and hence $S(L)$ divides $\Sigma^+ / \equiv_{k,R}$. Then by Propositions 2 and 4, $S(L)$ is locally idempotent and locally rrf.

(\Leftarrow) The following proof is inspired in [11] where it was applied to LT languages. Let $M(L)$ be the syntactic monoid of L with $n = \text{Card}(M(L))$ and let $\varphi: \Sigma^* \rightarrow M(L)$ be the syntactic morphism of L . We are going to prove that L is $(n+1)$ -RT.

Let $A = (Q, \Sigma, M(L), \delta, \lambda)$ be the output automaton, where $Q = \Sigma^n$, δ is defined as $\delta(a_1 a_2 \dots a_n, a) = a_2 \dots a_n a$, $\forall a_1 a_2 \dots a_n \in Q$, $\forall a \in \Sigma$ and the output function λ is defined as $\lambda(p, x) = \varphi(px)$, $\forall p \in Q$, $\forall x \in \Sigma^*$.

Let us see that A verifies the hypothesis of Theorem 1, that is, for every pair of loops $p \xrightarrow{y} p$ we have to show that

- (a) $\lambda(p, y) = \lambda(p, y^2)$ and
- (b) $\lambda(p, yzy) = \lambda(p, yz)$.

Let $p = a_1 a_2 \dots a_n$. As $\text{Card}(M(L)) = n$, the elements $1, \varphi(a_1), \varphi(a_1 a_2), \dots, \varphi(a_1 a_2 \dots a_n)$ cannot all be different, so we have a factorization $p = rst$, being $s \neq 1$, such that $rs \sim_L r$ (and therefore $rs^j \sim_L r$ for every positive integer j). As $M(L)$ is finite, $\exists k \geq 1$ such that $x = s^k$ and $\varphi(x) \in E(M(L))$. Let $p \xrightarrow{y} p$ be loops around p . As the words py and pz end with p we can write $sty = y't$ and $stz = z't$ for segments y' and z' such that ry' and rz' end with rs , and then write $ry' = y''rs$. As $rs \sim_L r$ we have that $ry's \sim_L ry'$, $rz's \sim_L rz'$.

In order to prove (a) we will see that $py^2 \sim_L py$:

$$py^2 = rstyy = ry'ty \sim_L ry'sty = ry'y't.$$

We can see that $ry'y't \sim_L ry'y'xt$. Indeed $ry'y't \sim_L y''rsy't \sim_L y''ry't \sim_L y''ry'xt \sim_L y''rsy'xt = ry'y'xt$. Therefore $py^2 \sim_L ry'y'xt \sim_L rxy'xy'xt \sim_L rxy'xt \sim_L ry't = rsty = py$. Then it follows that $\varphi(py^2) = \varphi(py)$ and then $\lambda(p, y^2) = \lambda(p, y)$.

Let us see the proof of (b):

$$\begin{aligned} pyzy &= rstzy = ry'tz \sim_L ry'stz = ry'z'ty \\ &= y''rsz'ty \sim_L y''rz'ty \sim_L y''rz'sty = y''rz'y't \sim_L, \\ y''rsz'y't &\sim_L ry'z'y't \sim_L rxy'xz'xy'xt \sim_L rxy'xz'xt \sim_L, \\ ry'z't &= ry'stz = ry'tz = rstyz = pyz. \end{aligned}$$

The argument used to prove the step $ry'z'y't \sim_L rxy'xz'xy'xt$ above is similar to the one used in (a).

It follows that $\varphi(pyzzy) = \varphi(pyz)$ and then $\lambda(p, yzy) = \lambda(p, yz)$.

Then, given $p \xrightarrow{y} q$ with $\mathcal{S}(p \xrightarrow{y} q) = \mathcal{S}(p \xrightarrow{z} q)$ it follows (Theorem 1) that $\lambda(p, y) = \lambda(p, z)$, which means $py \sim_L pz$.

Let $u, v \in \Sigma^*$ such that $u \equiv_{n+1, R} v$ and let $p = i_{n+1}(u)$, $q = f_{n+1}(u)$. Let $u = py$, $v = pz$. Then there exist two paths $p \xrightarrow{y} q$ in which all the edges are of the form $r \xrightarrow{a} t$, with $ra \in t_{n+1}(u)$. Besides, $\mathcal{S}(p \xrightarrow{y} q) = \mathcal{S}(p \xrightarrow{z} q)$, then $u \sim_L v$, that is, L is RLTL. \square

Remark 2. In the previous proof a special case occurs whenever there exists j , $1 \leq j \leq n$ such that $\varphi(a_1 \dots a_j) = 1$. Then $s \sim_L 1$ and $S(L)$ is a monoid. Since $S(L)$ is locally idempotent and locally *rrf*, then $S(L)$ is idempotent and *rrf*.

Theorem 4. *A recognizable language $L \subset \Sigma^*$ is 1-RT (resp. 1-LT) if and only if $S(L)$ is idempotent and *rrf* (resp. *lrf*).*

Proof. (\Rightarrow) $S(L)$ divides $\Sigma^+ / \equiv_{1, R}$. Then by Propositions 3 and 4, $S(L)$ is idempotent and *rrf*.

(\Leftarrow) Let $A = (\{1\}, \Sigma, M(L), \delta, \lambda)$ be the output automaton, where δ is defined as $\delta(1, a) = 1$, for every $a \in \Sigma$ and the output function λ is defined as $\lambda(1, x) = \varphi(x)$, for every $x \in \Sigma^*$. So, for every y, z the following facts hold: $1 \xrightarrow{y} 1$, $\lambda(1, y) = \lambda(1, y^2)$ and $\lambda(1, yzy) = \lambda(1, yz)$. The remainder of the proof is analogous to the previous one. \square

Due to all of the above it is easily seen that:

Proposition 6. *The family of RLTL (LLTL) languages is a variety of languages.*

Corollary 1. *The variety of the locally testable languages is the intersection of the varieties of right and left locally testable languages.*

References

[1] Beauquier, J.E. Pin, Languages and scanners, *Theoret. Comput. Sci.* 84 (1991) 3–21.
 [2] J.A. Brzozowski, I. Simon, Characterizations of locally testable events, *Discrete Math.* 4 (1973) 243–271.
 [3] S. Eilenberg, *Automata, Languages and Machines*, vol B, Academic Press, New York, 1976.
 [4] J. Hopcroft, J. Ullman, *Introduction to Automata theory, Languages and Computation*, Addison-Wesley, Reading, MA, 1979.
 [5] R. König, Reduction algorithms for some classes of aperiodic monoids, *R.A.I.R.O. Theor. Inform.* 19 (3) (1985) 233–260.
 [6] M. Lothaire, *Combinatorics on words*, Addison-Wesley, Reading, MA, 1983.
 [7] A. Luca, A. Restivo, A characterization of strictly locally testable languages and its application to subsemigroups of a free semigroup, *Inform. and Control* 44 (1980) 300–319.
 [8] R. McNaughton, Algebraic decision procedures for local testability. *Math. Systems Theory* 8 (1) (1971) 60–76.
 [9] R. McNaughton, *S. Papert Counter-Free Automata*, MIT Press, Cambridge, MA, 1971.
 [10] J.E. Pin, *Variétés de Langages formels*, Masson, Paris, 1984.

- [11] D. Perrin, Finite automata, in: Van Leeuwen (Ed.), *Handbook of Theor. Com. Sci. Vol. B*, Elsevier, Amsterdam, 1990.
- [12] C. Selmi, Over testable languages, *Theoret. Comp. Sci.* 161 (1996) 157–190.
- [13] I. Simon, Piecewise testable events, *Lecture Notes in Comp. Sci.*, vol. 33, 1975, pp. 214–222.
- [14] H. Straubing, *Finite Automata, Formal Logic and Circuit Complexity*, Birkhäuser, Boston, 1994.
- [15] D. Thérien, A. Weiss, Graph congruences and wreath products, *J. Pure Appl. Algebra* 35 (1985) 205–215.
- [16] Y. Zalcstein, Locally testable languages, *J. Comput. Systems Sci.* 6 (1972) 151–167. 1972.