Jean-Eric Pin ⋆

LIAFA, Université Paris VII and CNRS, Case 7014, 2 Place Jussieu, 75251 Paris Cedex 05, France

Jean-Eric.Pin@liafa.jussieu.fr

The expressive power of existential first-order sentences of Büchi's sequential calculus

Abstract

The aim of this paper is to study the first-order theory of the successor, interpreted on finite words. More specifically, we are interested in the hierarchy based on quantifier alternations (or Σ_n-hierarchy). It was known (Thomas, 1982) that this hierarchy collapses at level 2, but the expressive power of the lower levels was not characterized effectively. We give a semigroup theoretic description of the expressive power of $\mathcal{B}\Sigma_1$, the boolean combinations of existential formulas. We also give an $O(n^7)$-time algorithm to decide whether the language accepted by a deterministic n-state automaton is expressible by a first order sentence (respectively a $\mathcal{B}\Sigma_1$-sentence).

Key words: logic, first-order, successor, automata, semigroups

1 Introduction

A short version of this paper (without detailed proofs) was presented in [18].

The connections between formal languages and mathematical logic were first studied by Büchi [5]. But although Büchi was primarily interested in infinite words, only finite words will be considered in this paper.

Büchi's sequential calculus is a logical formalism to specify some combinatorial properties of a finite word, for instance "the factor *bba* occurs three times in the word, but the factor *bbb* does not occur". Thus, each logical sentence of this calculus defines a language, namely the set of all words that satisfy the property expressed by the formula. More precisely, to each word $u \in A^+$ is

⋆ Work supported by INTAS project 1224.

associated a structure

$$\mathcal{M}_u = (\{1, 2, \ldots, |u|\}, \mathbf{S}, (\mathbf{a})_{a\in A})$$

where $\mathbf{S}$ denotes the successor relation on $\{1, 2, \ldots, |u|\}$ and $\mathbf{a}$ is the set of all i such that the i-th letter of u is an a. For instance, if $A = \{a, b\}$ and $u = abaab$, then $\mathbf{a} = \{1, 3, 4\}$ and $\mathbf{b} = \{2, 5\}$. The logical language appropriate to such models has $\mathbf{S}$ and the $\mathbf{a}$'s as non logical symbols, and formulas are built in the standard way by using these non-logical symbols, variables, boolean connectives, equality between elements (positions) and quantifiers. Note that the symbol $<$ is not used in this logic.

Given a sentence φ, we denote by $L(\varphi)$ the set of all words which satisfy φ, when words are considered as models. Two formulas are said to be (elementary) *equivalent* if they define the same languages.

It is a well-known result of Büchi that monadic second order sentences exactly define the recognizable (or regular) languages. That is, for each monadic second order sentence φ, $L(\varphi)$ is a recognizable language and, for every recognizable language L, there exists a monadic second order sentence φ such that $L(\varphi) = L$. Actually, monadic second order logic constitutes a border line in the study of the sequential calculus. Beyond that border, one enters the hard world of complexity classes [8].

The effective characterization of the languages defined by first-order formulas follows from two results by Thomas [27] and Thérien and Weiss [26] recalled below. The details of the landscape can be refined by considering the Σ_n-hierarchy of first order logic. In fact, it was already shown in [27] that every first-order formula is equivalent to a Σ_2-formula, that is, a formula of the form

$$\exists x_1 \ \cdots \ \exists x_n \ \forall y_1 \ \cdots \ \forall y_m \ \psi(x_1, \cdots, x_n, y_1, \cdots, y_m)$$

where ψ is quantifier-free.

The aim of this paper is to give an effective characterization of the languages recognized by boolean combinations of Σ_1-formulas, that is, boolean combinations of formulas of the form

$$\exists x_1 \ \cdots \ \exists x_n \ \psi(x_1, \cdots, x_n)$$

where ψ is quantifier-free. This result leads to an algorithm to decide whether a given first-order formula (or even a monadic second order formula) is equivalent with a $\mathcal{B}\Sigma_1$-formula.

It is fair to say that the $\mathcal{B}\Sigma_1$-problem for formulas interpreted on infinite words has been solved by Wilke [30], but the given characterization makes use of the usual topology on infinite words and therefore could not be directly adapted to finite words.

Our characterization is expressed by an algebraic condition on the syntactic semigroup of the language, which can also be interpreted directly on its minimal deterministic automaton. This leads to an $O(n^7)$-time algorithm to decide whether the language accepted by a deterministic n-state automaton is expressible by a first-order sentence (respectively a $\mathcal{B}\Sigma_1$-sentence).

Originally, we also intended to give an effective characterization of the Σ_1-expressible languages. However this result requires auxiliary techniques that could not fit into this paper. Therefore, the proof is delayed to a future paper.

2 A combinatorial description

The expressive power of first-order formulas was first studied by Thomas [27]. Before stating this result, let us introduce some convenient definitions.

Recall that *boolean operations* on languages comprise finite union, finite intersection and complement. Given a word x and a positive integer k, it is not very difficult to express in first-order logic a property like "a factor x occurs at least k times". Let us denote by $F(x,k)$ the language defined by this property.

A language L of A^+ is *strongly threshold locally testable* (STLT for short) if it is a boolean combination of sets of the form $F(x,k)$ where $x \in A^+$ and $k > 0$. It is *threshold locally testable* (TLT) if it is a boolean combination of sets of the form uA^*, A^*v or $F(x,k)$ where $u, v, x \in A^+$ and $k > 0$. Note that uA^* (resp. A^*v) is the set of words having u as a prefix (resp. v as a suffix), a property that can also be expressed in Σ_2. Thomas proved the following theorem.

Theorem 2.1 *A language is first-order definable if and only if it is TLT.*

In fact, this result is a particular instance of the general fact that first-order formulas can express only local properties [10,28,29] and the main argument of the proof is to find a winning strategy for an appropriate Fraïssé-Ehrenfeucht game (see also [24]). A similar result was proved in [2,3] for $\mathcal{B}\Sigma_1$-formulas.

Theorem 2.2 *A language is $\mathcal{B}\Sigma_1$-definable if and only if it is STLT.*

3 Another combinatorial description

In this section, we give an alternative combinatorial description of the TLT and STLT languages.

Let A be a finite alphabet. If u is a word of length $\geq k$, we denote by $p_k(u)$ and $s_k(u)$, respectively, the prefix and suffix of length k of u. If u and x are two words, we denote by $\begin{bmatrix} u \\ x \end{bmatrix}$ the number of occurrences of the factor x in u. For instance $\begin{bmatrix} abababa \\ aba \end{bmatrix} = 3$, since aba occurs in three different places in $abababa$: $\underline{aba}baba$, $ab\underline{aba}ba$, $abab\underline{aba}$.

Let t be a positive integer, called the threshold. In the threshold t counting, all numbers $\geq t$ are identified. Thus threshold t counting can be viewed as a formalisation of children counting: zero, one, two, three, ..., many. This gives rise to a preorder on $\mathbb{N}$ defined formally as follows:

$$x \leq_t y \text{ if and only if either } x \leq y \text{ or } (x \geq t \text{ and } y \geq t)$$

The associated equivalence relation is defined by

$$x \equiv_t y \text{ if and only if either } (x < t \text{ and } x = y) \text{ or } (x \geq t \text{ and } y \geq t)$$

For instance the equivalence classes of $\equiv_4$ are $\{0\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4, 5, 6, 7, \ldots\}$. For each positive integer k, we define an equivalence relation $\equiv_{k,t}$ by setting

$$u \equiv_{k,t} v \text{ if and only if, for every word } x \text{ of length} \leq k,\ \begin{bmatrix} u \\ x \end{bmatrix} \equiv_t \begin{bmatrix} v \\ x \end{bmatrix}$$

For instance, $abababab \equiv_{2,3} abababa$ since $abababab$ contains 4 ($\equiv$ 3 threshold 3) occurrences of ab and 3 ($\equiv$ 3 threshold 3) occurrences of ba, and no occurrences of aa (respectively bb).

We also define an equivalence relation $\sim_{k,t}$ on A^+ by setting $u \sim_{k,t} v$ if and only if $u = v$ or $|u|, |v| \geq k$ and

(1) u and v have the same prefixes (resp. suffixes) of length $< k$,
(2) $u \equiv_{k,t} v$.

The following result was proved in [3].

Proposition 3.1

(1) *A subset of A^+ is TLT if and only if it is union of $\sim_{k,t}$-classes for some k and t.*
(2) *A subset of A^+ is STLT if and only if it is union of $\equiv_{k,t}$-classes for some k and t.*

These results complete the combinatorial description of the Σ_n-hierarchy, but do not solve the decidability questions: given a finite deterministic automaton $\mathcal{A}$, is it decidable whether the language accepted by $\mathcal{A}$ is first-order definable, $\mathcal{B}\Sigma_1$-definable?

4 The semigroup approach

This problem can be solved positively by semigroup-theoretic methods. If S is a semigroup, we denote by S^1 the monoid equal to S if S has an identity, and to $S \cup \{1\}$, where 1 is a new identity, otherwise. Recall that an element e of S is idempotent if $e^2 = e$. The set of idempotents of a semigroup S is denoted by $E(S)$. It is a well-known fact that in a finite semigroup, the subsemigroup generated by an element x contains a unique idempotent, denoted by x^ω.

Let L be a language of A^+. The *syntactic congruence of* L is the congruence $\sim_L$ on A^+ defined by $u \sim_L v$ if and only if, for every $x, y \in A^*$,

$$xuy \in L \Longleftrightarrow xvy \in L$$

The quotient semigroup $S(L) = A^+/{\sim_L}$ is called the *syntactic semigroup* of L. It is also equal to the transition semigroup of the minimal automaton of $\mathcal{A}$. It follows that a language is recognizable if and only if its syntactic semigroup is finite. The quotient morphism $\eta : A^+ \to S(L)$ is called the *syntactic morphism* and the subset $P = \eta(L)$ of $S(L)$ is the *syntactic image* of L.

Recall that a finite semigroup S is *aperiodic* if there exists an integer $n \geq 0$ such that, for each $s \in S$, $s^n = s^{n+1}$. It is equivalent to require, that for each $s \in S$, $s^\omega s = s^\omega$.

Another important property, introduced by Thérien and Weiss [26], is easier to state in terms of categories (there are also good mathematical reasons to do so). The *Cauchy category* of a finite semigroup S is defined as follows: the objects are the idempotents of S and, if $e, f \in E(S)$, the arrows from e to f are the triples (e, s, f), such that $s = es = sf$. Composition of arrows is defined by $(e, s, f)(f, t, g) = (e, st, g)$. The property of Thérien and Weiss states that for each $e, f \in E(S)$, and each $r, s, t \in S$, $erfsetf = etfserf$. It can be simply written

$$pqr = rqp \qquad (C)$$

where p and r are coterminal arrows, say, from e to f, and q is an arrow from f to e (see Figure 4.1). Thérien and Weiss did not explicitely mention the TLT languages in their paper but nevertheless gave the main argument of the proof of the following theorem.

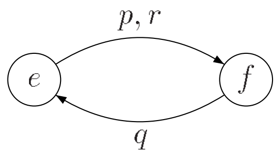


Figure 4.1. The condition $pqr = rqp$.

Theorem 4.1 *A language is TLT if and only if its syntactic semigroup S is aperiodic and satisfies (C).*

The link between the papers [27] and [26] was first observed in [2]. A complete proof of both results can also be found in the elegant book of Straubing on circuit complexity [24].

The problem of finding an algebraic description for the classes of STLT languages was left open in [2]. The solution to this problem is the main result of this paper and requires a few auxiliary definitions. Let S be a finite semigroup. The relations $\mathcal{R}$, $\mathcal{L}$ and $\mathcal{J}$ are defined as follows. Two elements r and s of S are $\mathcal{J}$-equivalent if they generate the same ideal, that is, if there exist $x, y, u, v \in S^1$ such that $ysv = r$ and $xru = s$. They are $\mathcal{R}$-equivalent (resp. $\mathcal{L}$-equivalent) if they generate the same right (resp. left) ideal, that is, if there exist $u, v \in S^1$ such that $sv = r$ and $ru = s$ (resp. $vs = r$ and $ur = s$). Let $\equiv$ be the coarsest equivalence relation on S satisfying the two following conditions

(1) for all $r, s \in S$, $r \mathcal{J} s$ implies $r \equiv s$,
(2) for all $e, f \in E(S)$ and $r, s \in S$, $erfse \equiv fserf$.

We say that a subset P of S *saturates* a relation $\sim$ if, for all $s, r \in S$, $s \in P$ and $s \sim r$ imply $r \in P$.

We are now ready to state our main result.

Theorem 4.2 *Let L be a recognizable language, S its syntactic semigroup and P its syntactic image. The following conditions are equivalent:*

(1) *L is $\mathcal{B}\Sigma_1$-definable,*
(2) *L is STLT,*
(3) *S is aperiodic and satisfies (C), and P saturates the relation $\equiv$.*

The proof of this result is given in the next section.

5 Proof of the main theorem

Our proof is inspired in part by the proof of Wilke [30], who gave a characterization of the TLT languages of infinite words. We first introduce some combinatorial definitions.

5.1 Graphs and Networks

Let $G = (V, E)$ be a graph. If p is a path, v a vertex and e an edge, $|p|_v$ (resp. $|p|_e$) denotes the number of occurrences of v (resp. e) in p. More generally, if V' (resp. E') is a set of vertices (resp. edges), we set

$$|p|_{V'} = \sum_{v \in V'} |p|_v \qquad |p|_{E'} = \sum_{e \in E'} |p|_e$$

In particular, $|p|_V = |p|_E + 1$. A *simple path* is a path p that goes at most once through each edge, that is, such that $|p|_e \leq 1$ for every $e \in E$.

If U is a subset of V, we set

$$\begin{aligned} In(U) &= \{e \in E \mid e = (v, w) \text{ for some } v \notin U \text{ and } w \in U\} \\ Out(U) &= \{e \in E \mid e = (v, w) \text{ for some } v \in U \text{ and } w \notin U\} \end{aligned}$$

Let us remind an elementary result due to Euler, in which $1_{\mathcal{P}}$ denotes the characteristic function of a property $\mathcal{P}$:

$$1_{\mathcal{P}} = \begin{cases} 1 & \text{if } \mathcal{P} \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Proposition 5.1 *Let $G = (V, E)$ be a graph and let p be a path from a vertex v to a vertex w. Then for every subset U of V, $|p|_{In(U)} - |p|_{Out(U)} = 1_{v \in U} - 1_{w \in U}$.*

A *network* $N = (V, E, c)$ is given by

(1) a graph (V, E),
(2) an initial vertex and a final vertex,
(3) a map $c : E \to \mathbb{N}$, called the *capacity map*.

A *path in the network* N is a path p of the underlying graph such that, for every edge $e \in E$, $|p|_e \leq c(e)$. The *capacity of a path* p is the integer

$$c(p) = \min \{|p|_e \mid e \in E, |p|_e > 0\}$$

In particular, a path has a capacity $\geq c$ if it goes at least c times through each of its edges. A path is called *Eulerian* if it exhausts its capacity, that is, for each edge e, $|p|_e = c(e)$. A network is called *Eulerian* if it contains an Eulerian path visiting all edges.

Let t be a non negative integer. Consider the graph $G_t = (V, E_t)$ where

$$E_t = \{e \in E \mid c(e) \geq t\}$$

The strongly connected components of the graph G_t are called the *t-components* of the network.

Proposition 5.2 *Let (V, E, c) be an Eulerian network and let $e = (v, w)$ be an edge of E. If $c(e) \geq 2 + t(|E|!)$, then v and w are in the same t-component.*

Proof. Let $k = c(e)$. Since the network is Eulerian, there exists a path p that goes k times through e. Therefore, there exists a factorization

$$p = p_0 e p_1 \cdots p_{k-1} e p_k$$

where, for $1 \leq i \leq k-1$, p_i is a path from w to v.

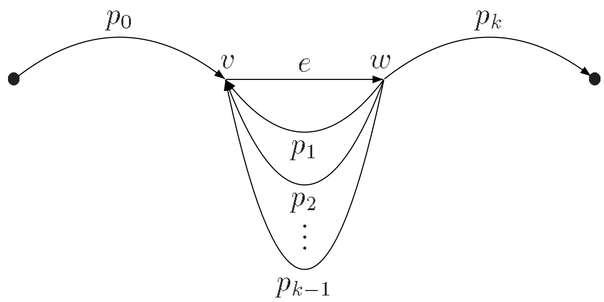


Figure 5.1. The path p.

By removing the loops from the $p_i's$, we may assume that the $p_i's$ are simple paths from w to v. Now, since a simple path never goes twice through the same edge, there are at most $|E|(|E|-1)\cdots(|E|-n+1)$ simple paths of length n, and therefore, the number of simple paths from w to v is bounded by $|E|!$. Since $k - 1 \geq 1 + t(|E|!)$, one of the simple paths p_i, say q, is used at least $t+1$ times. Now, the path $(qe)^t q$ (resp. $(eq)^t e$) is a path "extracted" from p. Therefore, for each edge a occurring in qe,

$$t \leq |(qe)^t q|_a \leq |p|_a \leq c(a)$$
$$t \leq |(eq)^t e|_a \leq |p|_a \leq c(a)$$

and thus $(qe)^t q$ (resp. $(eq)^t e$) is a path in the graph G_t. It follows that v and w are in the same t-component. $\square$

5.2 *Proof of Theorem 4.2*

The equivalence of (1) and (2) follows from Theorem 2.2. We now prove that (2) implies (3). Let L be a STLT language. By Proposition 3.1, L is union of $\equiv_{k,t}$-classes for some k and t. Let $\eta : A^+ \to S$ be the syntactic morphism of L and let P be the syntactic image of L. Since L is STLT, it is also TLT and thus, by Theorem 4.1, S is aperiodic and satisfies (C). It remains to see that P saturates the equivalence relation $\equiv$. Let s and r be two $\mathcal{J}$-equivalent elements of S and suppose that $s \in P$. Then there exist $x, y, u, v \in S^1$ such that $usx = r$ and $vry = s$. Since η is onto, each element $s \in S^1$ is the image under η of a word of A^*. We keep the same notation s for this word, so that $\eta(s) = s$. The context will make it clear whether we are in working in S^1 or in A^*.

Since S is finite, there is an integer n such that, for any $s \in S$, s^n is idempotent. We may assume that $n \geq kt$. Then

$$(vu)^n s(xy)^n \equiv_{k,t} u(vu)^n s(xy)^n x$$

But $\eta((vu)^n s(xy)^n) = s \in P$ and thus $(vu)^n s(xy)^n \in L$. It follows that $u(vu)^n s(xy)^n x \in L$ and thus $\eta(u(vu)^n s(xy)^n x) = r \in P$.

Let now $e, f \in E(S)$ and suppose that $esfre \in P$. Let, as before, e and f be words such that $\eta(e) = e$ and $\eta(f) = f$. Then, for $n \geq kt$,

$$e^n s f^n r e^n \equiv_{k,t} f^n r e^n s f^n$$

But $\eta(e^n s f^n r e^n) = esfre \in P$ and thus $e^n s f^n r e^n \in L$. Therefore $f^n r e^n s f^n \in P$ and thus $\eta(f^n r e^n s f^n) = fresf \in P$. Thus P saturates $\equiv$.

(3) implies (2). Suppose that S is aperiodic and satisfies (C), and that P saturates $\equiv$. By Theorem 4.1, L is TLT, and thus is union of $\sim_{k,t}$-classes for some $k, t > 0$. Let

$$T = 2 + t(|A|^k)! \quad \text{and} \quad T' = (1 + |A|)T$$

We claim that L saturates $\equiv_{k,T'}$. The claim will show, by Proposition 3.1, that L is STLT. Let us associate to each word $u \in A^+$ of length $\geq k$ the network $N(u)$ defined as follows:

(1) the vertices are the factors of length $k-1$ of u,
(2) the edges are the pairs $(p_{k-1}(x), s_{k-1}(x))$, where x is a factor of length k of u,
(3) the initial vertex is $p_{k-1}(x)$ and the final vertex is $s_{k-1}(x)$,
(4) the capacity of the edge $(p_{k-1}(x), s_{k-1}(x))$ is $\left[\begin{smallmatrix} u \\ x \end{smallmatrix}\right]$

Note that the number of edges of this network is bounded by $|A|^k$, the number of words of length k. This network is Eulerian since, if $u = a_1 a_2 \cdots a_n$, the path

$$p(u) = (a_1 \cdots a_{k-1}, a_2 \cdots a_k) \ \cdots \ (a_{n-k+1} \cdots a_{n-1}, a_{n-k+2} \cdots a_n)$$

exhausts its capacity. Four such networks are represented in Figure 5.2 at the end of this section.

Let u and u' be two words such that $u \equiv_{k,T'} u'$ and $u \in L$. Our aim is to show that $u' \in L$. The result is clear if $|u| < T'$ or $|u'| < T'$, since the relation $u \equiv_{k,T'} u'$ implies $u = u'$ in this case. Thus we may assume $|u| \geq T'$ and $|u'| \geq T'$. We claim that $\eta(u) \equiv \eta(u')$. Since L saturates $\equiv$, this will ensure that $u' \in L$.

The proof of the claim requires several steps. The reader is encouraged to follow the main arguments on Example 5.1 given below. We first establish some properties of the networks $N(u)$ and $N(u')$. First, they have the same underlying graph G. For each edge e of G, denote by $c(e)$ (resp. $c'(e)$) the capacity of e in $N(u)$ (resp. $N(u')$). Then, by hypothesis,

$$c(e) \equiv_{T'} c'(e)$$

In particular, since $T' \geq t$, $N(u)$ and $N(u')$ have the same t-components.

Lemma 5.3 *Let C be a t-component of $N(u)$. Then, for every edge $e \in In(C) \cup Out(C)$, $|p(u)|_e = |p(u')|_e < T$.*

Proof. Since $p(u)$ is an Eulerian path, $|p(u)|_e = c(e)$.

If $|p(u)|_e \geq T$, then $c(e) \geq 2 + t(A^k)!$, and by Proposition 5.2, the two extremities of e are in the same t-component, a contradiction. Therefore $|p(u)|_e < T$ and since $|p(u)|_e \equiv_{T'} |p(u')|_e$, $|p(u)|_e = |p(u')|_e$. $\square$

Let p and s (resp. p' and s') be the initial and final vertices of $N(u)$ (resp. $N(u')$). Let $C(p)$, $C(p')$, $C(s)$ and $C(s')$ be the t-components of p, p', s and s', respectively.

Lemma 5.4 *Either $C(p) = C(p')$ and $C(s) = C(s')$, or $C(p) = C(s)$ and $C(p') = C(s')$.*

Proof. Let C be a t-component of $N(u)$. By Proposition 5.1,

$$|p(u)|_{In(C)} - |p(u)|_{Out(C)} = 1_{p \in C} - 1_{s \in C}$$
$$|p(u')|_{In(C)} - |p(u')|_{Out(C)} = 1_{p' \in C} - 1_{s' \in C}$$

and by Lemma 5.3,

$$|p(u)|_{In(C)} - |p(u)|_{Out(C)} = |p(u')|_{In(C)} - |p(u')|_{Out(C)}$$

Therefore

$$1_{p \in C} - 1_{s \in C} = 1_{p' \in C} - 1_{s' \in C}$$

If $s \in C(p)$, then p and s are in the same t-component. In particular $C(p) = C(s)$ and $1_{p \in C(p')} = 1_{s \in C(p')}$. Since $1_{p \in C(p')} - 1_{s \in C(p')} = 1_{p' \in C(p')} - 1_{s' \in C(p')}$, it follows $1_{p' \in C(p')} = 1_{s' \in C(p')}$ and thus $s' \in C(p')$ and $C(p') = C(s')$.

If $s \notin C(p)$, then $1 = 1_{p \in C(p)} - 1_{s \in C(p)} = 1_{p' \in C(p)} - 1_{s' \in C(p)}$ and $-1 = 1_{p \in C(s)} - 1_{s \in C(s)} = 1_{p' \in C(s)} - 1_{s' \in C(s)}$. It follows $p' \in C(p)$ and $s' \in C(s)$, whence $C(p) = C(p')$ and $C(s) = C(s')$. □

We now consider the two cases separately.

Lemma 5.5 *If $C(p) = C(p')$ and $C(s) = C(s')$, then $\eta(u) \mathcal{J} \eta(u')$.*

Proof. Since p and p' are in the same t-component of $N(u)$, there exist two words v and v' such that $p_{k-1}(v) = p$, $p_{k-1}(v') = p'$ and, for each factor x (resp. y) of length k of vp' (resp. $v'p$), $\binom{u}{x} \geq t$ (resp. $\binom{u}{y} \geq t$). Similarly, since s and s' are in the same t-component of $N(u)$, there exist two words w and w' such that $s_{k-1}(w) = s$, $s_{k-1}(w') = s'$ and, for each factor x (resp. y) of length k of $s'w$ (resp. sw'), $\binom{u}{x} \geq t$ (resp. $\binom{u}{y} \geq t$). It follows that $vu'w \sim_{k,t} u$ and $v'uw' \sim_{k,t} u'$ and thus $\eta(vu'w) = \eta(u')$ and $\eta(v'uw') = \eta(u')$. Therefore $\eta(u) \mathcal{J} \eta(u')$. □

We are left with the case $C(p) = C(s)$ and $C(p') = C(s')$. We first reduce this case to the case $p = s$ and $p' = s'$. Since p and s are in the same t-component of $N(u)$, either $p = s$ or there exist two paths of capacity $\geq t$ from p to s and from s to p. In the latter case, there exist two words v and w such that $p_{k-1}(v) = s$, $p_{k-1}(w) = p$ and, for each factor x (resp. y) of length k of vp (resp. ws), $\binom{u}{x} \geq t$ (resp. $\binom{u}{y} \geq t$). It follows that $uvw \sim_{k,t} u$ and thus $\eta(uvw) = \eta(u)$ and $\eta(u) \mathcal{J} \eta(uvp)$. Since $u \in L$, $\eta(u) \in P$ and thus $\eta(uvp) \in P$ and $uvp \in L$. Therefore, we may now substitute uvp for u, that is, we may assume that $p = s$. Similarly, we may suppose that $p' = s'$. We may also assume that $p \neq p'$, for otherwise, Lemma 5.5 can be applied. We now produce a loop of capacity t around p.

Lemma 5.6 *There exists a non-trivial loop of capacity $\geq t$ around p.*

Proof. If $C(p)$ is not reduced to $\{p\}$, there exists another vertex q in the same t-component as p and thus there exist two paths of capacity $\geq t$ from p to q and from q to p. Concatenating these two paths gives the desired loop.

Suppose now that $C(p) = \{p\}$. The path $p(u)$ goes from p to p and $p(u')$ from p' to p'. We now count the number of occurrences of p in both $p(u)$ and $p(u')$. Each occurrence of p, except for the last occurrence of p in $p(u)$, is the origin of an edge of the given path of the form $e = (p, q)$. Actually e is either the edge (p, p) or an edge of $Out(C(p))$. Therefore

$$|p(u)|_p = |p(u)|_{Out(C(p))} + |p(u)|_{(p,p)} + 1$$
$$|p(u')|_p = |p(u')|_{Out(C(p))} + |p(u')|_{(p,p)}$$

But since $u \equiv_{k,T'} u'$, u and u' have the same factors of length $k-1$, counted threshold T'. Thus

$$|p(u)|_p \equiv_{T'} |p(u')|_p$$

Now, by Lemma 5.3, for every edge $e \in In(C(p)) \cup Out(C(p))$, $|p(u)|_e = |p(u')|_e < T$. Therefore, since $|Out(C(p))| \leq |A|$,

$$|p(u)|_{Out(C(p))} = |p(u')|_{Out(C(p))} < |A|T$$

Now, combining the previous relations and in view of the choice of T', we get

$$|p(u)|_{(p,p)} + 1 \equiv_T |p(u')|_{(p,p)}$$

whence $|p(u)|_{(p,p)} \geq T$. Therefore, the edge (p, p) is a loop [2] of capacity at least equal to t. $\square$

Since $p(u)$ is a path from p to p that goes through p', there exists a factorization $u = u_1p'u_2$ such that $p_{k-1}(u_1p') = p$ and $s_{k-1}(u_2) = p$. Furthermore, by Lemma 5.6, there exists a word v such that $s_{k-1}(v) = p$ and, for each factor x of length k of pv, $\left[\begin{smallmatrix} u \\ x \end{smallmatrix}\right] \geq t$. Similarly, there exists a word w such that $s_{k-1}(w) = p'$ and, for each factor x of length k of $p'w$, $\left[\begin{smallmatrix} u \\ x \end{smallmatrix}\right] \geq t$. Since S is finite, there exists an integer π such that all elements of the form s^π, where $s \in S$, are idempotent.

Now $(pv)^\pi u_1 (p'w)^\pi u_2 (pv)^\pi \sim_{k,t} u$ and $(p'w)^\pi u_2 (pv)^\pi u_1 (p'w)^\pi \sim_{k,t} u'$. Setting $e = \eta(pv)^\pi$, $f = \eta(p'w)^\pi$, $s = \eta(u_1)$ and $t = \eta(u_2)$, one gets $\eta(u) = esfte$ and $\eta(u') = ftesf$ and thus $\eta(u) \equiv \eta(u')$, proving the claim and the theorem. $\square$

Example 5.1 In Figure 5.2 below, four networks are represented. The parameters are $k = 3$ and $t = 3$. The graph on the left hand side corresponds to

[2] Actually, this case can only occur if $p = a^{k-1}$ for some letter a such that $\left[\begin{smallmatrix} u \\ a^k \end{smallmatrix}\right] \geq T$.

the words $u = (ab)^4(cb)^4abcb$ and $u' = b(ab)^4(cb)^4abc$. The graph on the right hand side corresponds to the words $u = (ab)^4(cb)^4a$ and $u' = b(cb)^4(ab)^4cb$. The initial and final vertices of u (resp. u') are represented by full (resp. dotted) unlabelled arrows.

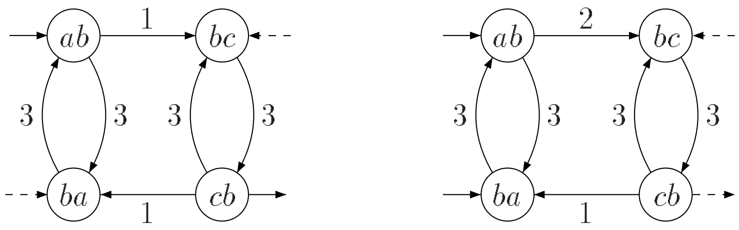


Figure 5.2. Four networks.

In these two diagrams, the t-components are $\{ab, ba\}$ and $\{bc, cb\}$. In the diagram on the left, $C(p) = C(p')$ and $C(s) = C(s')$. According to Lemma 5.5, the elements $\eta(u)$ and $\eta(u')$ are $\mathcal{J}$-equivalent. Indeed $b(ab)^4(cb)^4abc \equiv_{k,t} b[(ab)^4(cb)^4abcb]c$ and $(ab)^4(cb)^4abcb \equiv_{k,t} a[b(ab)^4(cb)^4abc]b$.

In the diagram on the right, $C(p) = C(s)$ and $C(p') = C(s')$. One can verify that $\eta((ab)^4(cb)^4a) \equiv \eta(b(cb)^4(ab)^4cb)$.

6 Complexity issues

In this section, we give polynomial time algorithms to decide whether the language accepted by a deterministic n-state automaton is expressible by a first-order sentence (respectively a $\mathcal{B}\Sigma_1$-sentence). The idea of the algorithm is of course to use Theorem 4.1, but a direct translation fails, since testing for aperiodicity is PSPACE-complete [6]. The trick is that, for semigroups satisfying (C), aperiodicity is equivalent to much more constraint conditions. There is actually some freedom in the selection of a new constraint, and our choice was motivated by algorithmic concerns.

Recall that a semigroup S is *$\mathcal{R}$-trivial* if, for every $x, y \in S$, $x \mathrel{\mathcal{R}} y$ implies $x = y$ and *locally $\mathcal{R}$-trivial* if each of its local semigroup is $\mathcal{R}$-trivial. It is equivalent to state (see [15] for more details) that, for each $e \in E(S)$, and every $x, y \in S$,

$$(exeye)^\omega = (exeye)^\omega exe \qquad (1)$$

We can now modify Theorem 4.1 as follows.

Theorem 6.1 *A semigroup satisfying (C) is aperiodic if and only if it is locally $\mathcal{R}$-trivial.*

Proof. Let S be a semigroup. If S is locally $\mathcal{R}$-trivial, then it satisfies (1). In particular, for $e = y = x^\omega$, one gets $x^\omega = x^\omega x$ and thus S is aperiodic.

We claim that if S satisfies (C), then it satisfies the identity

$$exeye = eyexe \tag{2}$$

for each $e \in E(S)$ and $x, y \in S$. Indeed, it suffices to substitute in (C) e for f and q, exe for p and eye for r.

Suppose now that S is aperiodic and satisfies (C). Then it satisfies (2) and thus

$$(exeye)^\omega exe = (exe)^\omega exe(eye)^\omega = (exe)^\omega(eye)^\omega = (exeye)^\omega$$

Thus S satisfies (1) and is locally $\mathcal{R}$-trivial. □

Corollary 6.2 *A language is TLT if and only if its syntactic semigroup is locally $\mathcal{R}$-trivial and satisfies (C).*

We now study these two conditions separately. We first introduce a convenient definition, already used in [23,7], but we adopt the formulation proposed by Wilke. A *pattern* is a graph whose vertices are state variables and whose edges are labeled by word variables. In addition, a pattern comes with side conditions stating which state variables are to be interpreted as distinct states, as the initial state or as final states. An A^+-labeled graph *matches a pattern* if there is an assignment to the variables obeying the type constraints and the side conditions such that the graph obtained be replacing each variable by the value assigned to it is a subgraph of the given graph. To apply this definition to automata, we consider an automaton $\mathcal{A}$ with alphabet A as an A^+-labeled graph, for which there is an edge from state q to state q' labeled by the word u if and only if there is a path labeled by u from q to q' in $\mathcal{A}$. The following result is proved in [7].

Proposition 6.3 *The syntactic semigroup of a language is locally $\mathcal{R}$-trivial if and only if its minimal automaton does not match the following pattern, with $q_1 \neq q_2$.*

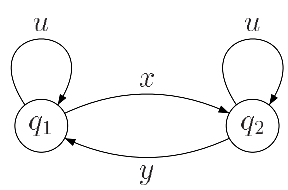


Figure 6.1. The forbidden pattern for locally $\mathcal{R}$-trivial semigroups.

A similar result holds for (C). In the sequel, we adopt the following notation: L is a language over A, $\eta : A^+ \to S$ its syntactic semigroup and

$\mathcal{A} = (Q, A, \cdot, q_0, F)$ its minimal automaton. There is a natural action from S on Q, defined by $q \cdot s = q \cdot u$, where u is any word such that $\eta(u) = s$ (this clearly does not depend on the choice of u).

Proposition 6.4 *The syntactic semigroup of a language satisfies (C) if and only if its minimal automaton does not match the following pattern, with $q_4 \neq q_7$.*

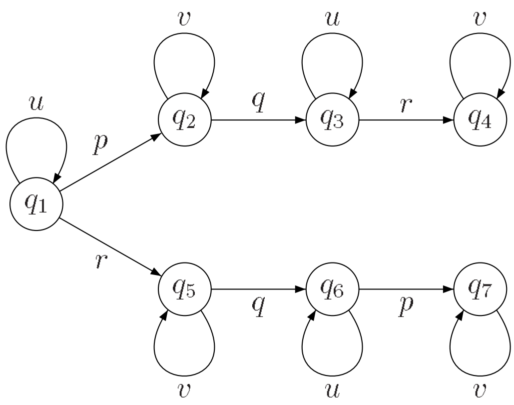


Figure 6.2. The forbidden pattern for (C).

Proof. If $\mathcal{A}$ matches the pattern, Condition (C) cannot be satisfied, since the words $u^\omega p v^\omega q u^\omega r v^\omega$ and $u^\omega r v^\omega q u^\omega p v^\omega$ are not $\sim_L$-equivalent.

Conversely, if (C) is not satisfied, there exist words x, y, f, g and h of A^+ such that

$$\eta(x^\omega f y^\omega g x^\omega h y^\omega) \neq \eta(x^\omega h y^\omega g x^\omega f y^\omega)$$

Therefore, there exist words s and t in A^* such that $q_0 \cdot s x^\omega f y^\omega g x^\omega h y^\omega t \in F$ and $q_0 \cdot s x^\omega h y^\omega g x^\omega f y^\omega t \notin F$. To recover the pattern given in Figure 6.2, it suffices to set $u = x^\omega$, $v = y^\omega$, $p = x^\omega f y^\omega$, $q = y^\omega g x^\omega$, $r = x^\omega h y^\omega$, $q_1 = q_0 \cdot s x^\omega$, $q_2 = q_1 \cdot p$, $q_3 = q_2 \cdot q$, $q_4 = q_3 \cdot r$, $q_5 = q_1 \cdot r$, $q_6 = q_5 \cdot q$, $q_7 = q_6 \cdot p$. Furthermore, $q_4 \neq q_7$, since $q_4 \cdot t \in F$ and $q_7 \cdot t \notin F$. $\square$

Corollary 6.5 *A language is TLT if and only if its minimal automaton does not match the patterns represented in Figure 6.1 and 6.2.*

The condition "P saturates $\equiv$" can be decomposed into three subconditions

(1) P saturates the relation $\mathcal{R}$,
(2) P saturates the relation $\mathcal{L}$,
(3) for each $s, t \in S$ and $e, f \in E(S)$, $esfte \in p$ if and only if $ftesf \in P$.

Indeed, in a finite semigroup, the relation $\mathcal{J}$ is the join of $\mathcal{R}$ and $\mathcal{L}$. Therefore, if a relation saturates $\mathcal{R}$ and $\mathcal{L}$, it saturates $\mathcal{J}$. We now treat these conditions separately in the next propositions.

Proposition 6.6 *The image of a language in its syntactic semigroup saturates $\mathcal{R}$ if and only if its minimal automaton does not match the following pattern, with $p_0 \in F$ and $p_1 \notin F$.*

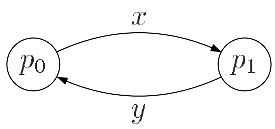


Figure 6.3. The forbidden pattern for saturating the $\mathcal{R}$-classes.

Proof. Let u and v be two $\mathcal{R}$-equivalent elements with $u \in P$ and $v \notin P$. Then there exist elements $x, y \in S^1$ such that $ux = v$ and $vy = u$. Now, setting $p_0 = q_0{\cdot}u$ and $p_1 = p_0{\cdot}x$, we have $p_1{\cdot}y = p_0$. Furthermore, since $u \in P$ and $v \notin P$, $p_0 \in F$ and $p_1 \notin F$. Thus the automaton matches the given pattern.

Conversely, if the pattern occurs, let s be a word such that $p_0 = q_0{\cdot}s$. Consider now the elements $u = \eta(s(xy)^\omega)$ and $v = \eta(s(xy)^\omega x)$. They are $\mathcal{R}$-equivalent since $v = u\eta(x)$ and $u = v\eta(y(xy)^{\omega-1})$. Furthermore, $q_0{\cdot}u = p_0 \in F$ and $q_0{\cdot}v = p_1 \notin F$. Thus $u \in P$, $v \notin P$ and P does not saturate $\mathcal{R}$. $\square$

Proposition 6.7 *The image of a language in its syntactic semigroup saturates $\mathcal{L}$ if and only if its minimal automaton does not match the following pattern, where q_0 is the initial state, and $q_2 \neq q_3$.*

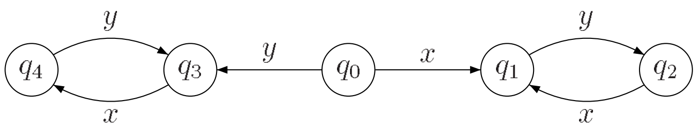


Figure 6.4. Forbidden pattern for saturating the $\mathcal{L}$-classes.

Proof. Let v and w be two $\mathcal{L}$-equivalent elements with $u \in P$ and $v \notin P$. Then there exist elements $s, t \in S^1$ such that $sv = u$ and $tu = v$. Setting $x = s(ts)^\omega$, $y = t(st)^{2\omega-1}$, we obtain $xy = (st)^\omega$ and $yx = (ts)^\omega$. Now, setting $q_1 = q_0{\cdot}x$, $q_2 = q_1{\cdot}y$, $q_3 = q_0{\cdot}y$ and $q_4 = q_3{\cdot}x$, we have $q_2{\cdot}x = q_1$ and $q_4{\cdot}y = q_3$. Furthermore, since $v \in P$ and $u \notin P$, the states $q_0{\cdot}u$ and $q_0{\cdot}v$ are necessarily distinct. But as $u = xyu$ and $v = yu$, $q_0{\cdot}u = q_0{\cdot}xyu = q_2{\cdot}u$ and $q_0{\cdot}v = q_0{\cdot}yu = q_3{\cdot}u$ and hence q_2 and q_3 are distinct. Thus the minimal automaton matches the given pattern.

Conversely, suppose that the minimal automaton matches this pattern. Since $q_2 \neq q_3$, there exists a word u such that $q_2{\cdot}u \in F$ and $q_3{\cdot}u \notin F$ (or, dually, $q_2{\cdot}u \notin F$ and $q_3{\cdot}u \in F$). Now, the elements $u' = \eta((xy)^\omega u)$ and

$v' = \eta(y(xy)^\omega u)$ are $\mathcal{L}$-equivalent, $q_0 \cdot (xy)^\omega u \in F$ and $q_0 \cdot y(xy)^\omega u \notin F$. Thus $u' \in P$, $v' \notin P$ and P does not saturate $\mathcal{L}$. □

Using similar arguments, it is not difficult to prove the following result.

Proposition 6.8 *The image P of a language in its syntactic semigroup S satisfies the condition*

$$esfte \in P \Leftrightarrow ftese \in P \quad \text{for each } s, t \in S \text{ and } e, f \in E(S).$$

if and only if its minimal automaton does not match the pattern below, where q_0 is the initial state, $p_0 \in F$ and $p_1 \notin F$.

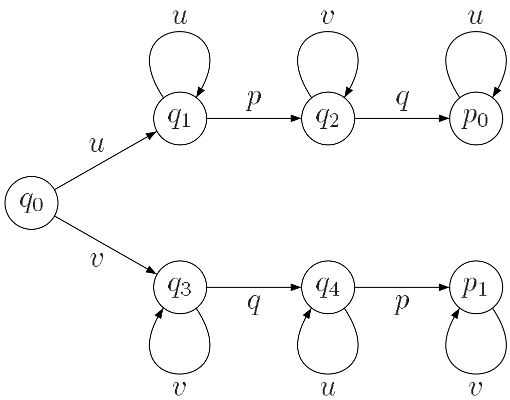


Figure 6.5. The third forbidden pattern for saturating the $\equiv$-classes.

Corollary 6.9 *A language is STLT if and only if its minimal automaton does not match any of the patterns represented in Figures 6.1 to 6.5.*

We now analyse the complexity of our algorithms. In general, checking whether an automaton matches a given pattern can be done in polynomial time. We give a precise algorithm for the pattern represented in Figure 6.2. The other cases are similar and simpler.

Proposition 6.10 *There is an $O(n^7)$ algorithm to test whether a given n-state minimal automaton matches the pattern given in Figure 6.2.*

Proof. For each $m > 0$, consider the graph G_m with Q^m as set of vertices and with edges of the form $\big((q_1, \ldots, q_m), (q_1 \cdot a, \ldots, q_m \cdot a)\big)$, for each letter $a \in A$. By construction G_m has n^m vertices and $|A|n^m$ edges. Thus the total size of G_m is in $O(n^m)$. It is well known that computing the strongly connected components of a graph can be achieved in time linear in the size of the graph. Similarly, one can compute in linear time the vertices which are the origin of a non-trivial loop.

The search for the pattern is done by marking the elements of Q^7 in different fashions. We first compute in $O(n^4)$ the set S_v of non-trivial loops of G_4. Then we set a red flag on the 7-tuples $(q_1, \ldots, q_7)$ such that (q_2, q_4, q_5, q_7) belongs to S_v. This step is done in $O(n^7)$ time. Intuitively, this red flag indicates whether it is possible to have a loop labeled by the same word v around the states q_2, q_4, q_5 and q_7. We proceed in the same way to compute in $O(n^3)$ the set S_u of non-trivial loops of G_3. Next, we set a blue flag on the 7-tuples such that (q_1, q_3, q_6) belongs to S_u. This takes care of the loops labeled by u in the pattern.

Now, we work in G_2 to compute in $O(n^4)$ the set S of pairs $((p, q), (p', q'))$ for which there is a path in G_2 from (p, q) to (p', q'). Then we set a green (respectively yellow and magenta) flag on the 7-tuples such that $((q_1, q_6), (q_2, q_7))$ (resp. $((q_2, q_3), (q_5, q_6))$, $((q_3, q_4), (q_1, q_5))$) belongs to S.

Now the automaton matches the pattern if and only if there exists a 7-tuple having all possible flags. $\square$

Theorem 6.11 *There is an $O(n^7)$-time algorithm to decide whether the language recognized by a minimal n-state automaton is first-order definable (resp. $\mathcal{B}\Sigma_1$-definable).*

Proof. The proof of Proposition 6.10 can be readily adapted to check whether a minimal automaton matches a given pattern. Since the patterns involved in the characterizations of TLT and STLT languages have at most 7 vertices, the complexity is in $O(n^7)$. $\square$

Theorem 6.11 is in contrast with the corresponding result for the first-order logic of the binary relation $<$, interpreted as the natural order on the integers. For this logic, McNaughton and Papert [12] gave a combinatorial description (the star-free languages) and Schützenberger [21] gave an algebraic characterization (the syntactic semigroup is aperiodic), but it was shown in [6] that the corresponding algorithm is PSPACE-complete.

7 Two examples

We conclude this paper by giving two examples illustrating our main results.

Example 7.1 Let $A = \{a, b, c\}$, and let $L = c(ab)^* \cup c(ab)^*a$. Then L is recognized by the following automaton.

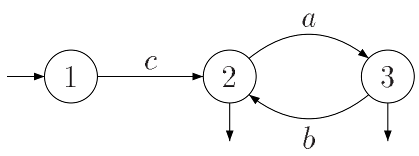


Figure 7.1. The minimal automaton of L (sink state omitted).

The transitions and the relations defining the syntactic semigroup S of L are given in the following tables

	a	b	c	aa	ab	ba	ca
1	–	–	2	–	–	–	3
2	3	–	–	–	2	–	–
3	–	2	–	–	–	3	–

$$a^2 = b^2 = c^2 = ac = bc = cb = 0$$
$$aba = a$$
$$bab = b$$
$$cab = c$$

The $\mathcal{J}$-class structure is represented in the following diagram, where the grey box is the image of L.

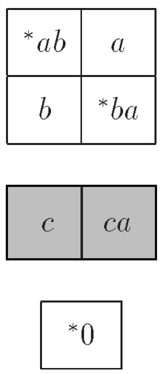


Figure 7.2. The $\mathcal{J}$-class structure.

Thus P saturates $\mathcal{J}$, and L is SLT. In fact, $L = A^*cA^* \setminus (A^*aaA^* \cup A^*acA^* \cup A^*bbA^* \cup A^*bcA^* \cup A^*cbA^* \cup A^*ccA^*)$.

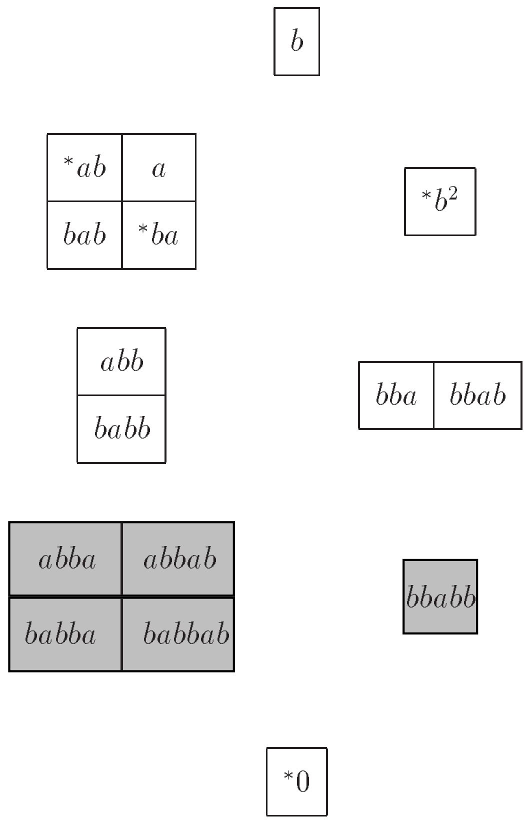
Example 7.2 Let $A = \{a, b\}$, and let $L = (1 + b)a(ba)^*b^2b^*a(ba)^*(1 + b) \cup b^2b^*a(ba)^*b^2b^*$. The transitions and the relations defining the syntactic semigroup of L are given in the following tables

Elements	1	2	3	4	5	6	7	8	9	10	11
a	6	10	0	7	10	0	0	6	7	0	6
b	11	2	3	3	0	8	4	2	9	5	9
aa	0	0	0	0	0	0	0	0	0	0	0
ab	8	5	0	4	5	0	0	8	4	0	8
ba	6	10	0	0	0	6	7	10	7	10	7
bb	9	2	3	3	0	2	3	2	9	0	9
abb	2	0	0	3	0	0	0	2	3	0	2
bab	8	5	0	0	0	8	4	5	4	5	4
bba	7	10	0	0	0	10	0	10	7	0	7
$abba$	10	0	0	0	0	0	0	10	0	0	10
$babb$	2	0	0	0	0	2	3	0	3	0	3
$bbab$	4	5	0	0	0	5	0	5	4	0	4
$abbab$	5	0	0	0	0	0	0	5	0	0	5
$babba$	10	0	0	0	0	10	0	0	0	0	0
$bbabb$	3	0	0	0	0	0	0	0	3	0	3
$babbab$	5	0	0	0	0	5	0	0	0	0	0

Relations :

$$aa = 0 \qquad aba = a \qquad b^3 = b^2 \qquad abbabb = 0 \qquad bbabba = 0$$

The idempotents are ab, ba, bb and 0. The $\mathcal{J}$-class structure is represented in the following diagram:



The image of the language is $P = \{bbabb, abba, abbab, babba, babbab\}$. It appears in grey in the diagram. One can verify that P saturates $\equiv$. Notice in particular that $babbab = (ba)(bb)(ba)$. Since the elements $e = ba$ and $f = bb$ are idempotent, $efe \in P$ should imply $fef \in P$, since P saturates $\equiv$. Indeed, $fef = babba \in P$. In fact,

$$L = (F(ab^2, 1) \cap F(b^2a, 1)) \setminus (F(aa, 1) \cup F(ab^2, 2) \cup F(b^2a, 2))$$

and thus L is STLT.

References

[1] J. Almeida, *Finite semigroups and universal algebra*, Series in Algebra Vol 3, Word Scientific, Singapore, 1994.

[2] D. Beauquier and J.-E. Pin, Factors of words, in *Automata, Languages and Programming, (G. Ausiello, M. Dezani-Ciancaglini and S. Ronchi Della Rocca, eds.), Lecture Notes in Comput. Sci.* **372**, Springer, (1989), 63–79.

[3] D. Beauquier and J.-E. Pin, Languages and scanners, *Theoret. Comput. Sci.* **84**, (1991), 3–21.

[4] J.A. Brzozowski and I. Simon, Characterizations of locally testable languages, *Discrete Math.* **4**, (1973), 243-271.

[5] J.R. Büchi, On a decision method in restricted second-order arithmetic, in *Proc. 1960 Int. Congr. for Logic, Methodology and Philosophy of Science*, Stanford Univ. Press, Standford, (1962), 1–11.

[6] S. Cho and D.T. Huynh, Finite automaton aperiodicity is PSPACE-complete, *Theoret. Comput. Sci.* **88**, (1991), 99–116.

[7] J. Cohen, D. Perrin and J.-E. Pin, On the expressive power of temporal logic, *J. Comput. System Sci.* **46**, (1993), 271–294.

[8] H.D. Ebbinghaus and J. Flum, *Finite Model Theory*, Springer, (1995).

[9] S. Eilenberg, *Automata, languages and machines*, Vol. B, Academic Press, New York, 1976.

[10] H. Gaifman, On local and non-local properties, in *Proc. of the Herbrandt Symposium, Logic Colloquium'81 (J. Stern, ed.), Studies in Logic* **107**, North-Holland, Amsterdam, (1982), 105–135.

[11] R. McNaughton, Algebraic decision procedures for local testability, *Math. Syst. Theor.* **8**, (1974), 60-76.

[12] R. McNaughton and S. Pappert, *Counter-free Automata*, MIT Press, 1971.

[13] D. Perrin, *Automata*, Chapter 1 in Handbook of Theoretical Computer Science (Van Leeuwen, J. ed.), Vol B: Formal Models and Semantics, Elsevier (1990).

[14] D. Perrin and J.-E. Pin, First-order logic and star-free sets, *J. Comput. System Sci.* **32**, (1986), 393–406.

[15] J.-E. Pin, *Variétés de langages formels*, Masson, Paris, 1984. English translation: *Varieties of formal languages*, Plenum, New-York, 1986.

[16] J.-E. Pin, Finite semigroups and recognizable languages : an introduction, in NATO Advanced Study Institute *Semigroups, Formal Languages and Groups*, J. Fountain (ed.), Kluwer academic publishers, (1995), 1–32.

[17] J.-E. Pin, A variety theorem without complementation, *Izvestiya VUZ Matematika* **39** (1995) 80–90. English version, *Russian Mathem. (Iz. VUZ)* **39**, (1995), 74–83.

[18] J.-E. Pin, The expressive power of existential first-order sentences of Büchi's sequential calculus, *ICALP 1996, Lecture Notes in Comput. Sci.* **1099**, Springer, (1996) 300–311.

[19] J.-E. Pin, Logic, Semigroups and Automata on Words, *Annals of Mathematics and Artificial Intelligence* **16**, (1996), 343–384.

[20] J.-E. Pin and P. Weil, Polynomial closure and unambiguous product, *Theory Comput. Systems* **30**, (1997), 1–39.

[21] M.P. Schützenberger, On finite monoids having only trivial subgroups, *Information and Control* **8**, (1965), 190–194.

[22] J. Stern, Characterization of some classes of regular events, *Theoret. Comp. Sci.* **35**, (1985), 17–42.

[23] J. Stern, Complexity of some problems from the theory of automata, *Inform. and Control* **66**, (1985), 63–176.

[24] H. Straubing, *Finite automata, formal logic and circuit complexity*, Birkhäuser, 1994.

[25] H. Straubing, D. Thérien and W. Thomas, Regular Languages Defined with Generalized Quantifiers, in *Proc. 15th ICALP*, Springer Lecture Notes in Computer Science **317**, (1988), 561–575.

[26] D. Thérien and A. Weiss, Graph congruences and wreath products, *J. Pure Applied Algebra* **35**, (1985), 205–215.

[27] W. Thomas, Classifying regular events in symbolic logic, *J. Comput. Syst. Sci* **25**, (1982), 360–375.

[28] W. Thomas, On logics, tilings, and automata, *Proc. 18th ICALP, Madrid, (J. Leach Albert et al., eds.), Lect. Notes in Comp. Sci.* **510**, Springer, Berlin, (1991), 441–454.

[29] W. Thomas, On the Ehrenfeucht-Fraïssé Game in Theoretical Computer Science, *TAPSOFT'93, M.C. Gaudel, J.P. Jouannaud (Eds.), Lect. Notes in Comp. Sci.* **668**, Springer, Berlin, (1993), 559–568.

[30] Th. Wilke, Locally threshold testable languages of infinite words, in *STACS 93, P. Enjalbert, A. Finkel, K.W. Wagner (Eds.), Lect. Notes in Comp. Sci.* **665**, Springer, Berlin, (1993), 607–616.