Star Height of Reversible Languages and Universal Automata

Sylvain Lombardy and Jacques Sakarovitch

Laboratoire Traitement et Communication de l'Information (CNRS / ENST)
Ecole Nationale Supérieure des Télécommunications
46, rue Barrault 75634 Paris Cedex 13, France
{lombardy,sakarovitch}@enst.fr

Abstract. The *star height* of a regular language is an invariant that has been shown to be effectively computable in 1988 by Hashiguchi. But the algorithm that corresponds to his proof leads to impossible computations even for very small instances. Here we solve the problem (of computing star height) for a special class of regular languages, called *reversible languages*, that have attracted much attention in various areas of formal language and automata theory in the past few years. These reversible languages also strictly extend the classes of languages considered by McNaughton, Cohen, and Hashiguchi for the same purpose, and with different methods.
Our method is based upon the definition (inspired by the reading of Conway's book) of an automaton that is effectively associated to every language — which we call the *universal automaton* of the language — and that contains the image of any automaton that accepts the language. We show that the universal automaton of a reversible language contains a subautomaton where the star height can be computed.

Key words: Finite automata, regular expressions, star height, reversible automata, reversible languages, universal automata.

Introduction

Among all invariants attached to regular languages, the *star height* introduced by Eggan in 1963 proved to be the most puzzling one. The formal definition of star height will be recalled below but, in one word, one can say that the star height of a (regular) language K is the minimum of *nested* star operations that has to be used in order to describe K by a regular expression. As this star operation is the only[1] regular operation that goes from finite to infinite, star height is a very sensible complexity measure.

In a manner of a parallel to Kleene's theorem, Eggan ([7]) showed that the star height of a rational expression is related to another quantity that is defined

[1] Regular languages are *closed* under complement but complement is not considered as a regular operation.

on a finite automaton which produces the expression, a quantity which he called *rank* and which was later called *loop complexity.* And he stated the following two problems:

i) Does there exist, on a fixed finite alphabet, regular languages of arbitrary large star height?

ii) Is the star height of a regular language computable?

The first problem was solved, positively, in 1966 by Dejean and Schützenberger ([6]). Soon afterwards, in 1967, McNaughton ([14]) gave a conceptual proof of what Dejean and Schützenberger had established by means of combinatorial virtuosity (one of the "jewels" of formal language theory, as stated in [17]). He proved that the loop complexity, and thus the star height, of a *pure-group language*, *i.e.* a language whose syntactic monoid is a finite group, is computable. And that this family contains languages of arbitrary large loop complexity (the languages considered by Dejean and Schützenberger were — of course (?) — pure-group languages).

In addition, McNaughton's gave simple evidence of the fact that the star height of a regular language *is not a syntactic invariant*, or, which is roughly equivalent, the minimal automaton of a language is not of minimal loop complexity. As we explain below, the work of McNaughton was pushed further on, and, to some extend, our present result can be seen as its ultimate stage, which is reached by new methods.

Before that, we have to mention first that the second problem (the computability of star height) remained open for long and was considered as one of the most difficult in the theory of automata. It was eventually solved (positively) by Hashiguchi in 1988 ([10]). The method of Hashiguchi is completely different from the previous work and can be (briefly) described as follow. Given an expression E that denotes a language L on A^*, a bound M (super-exponential in the size of E) is computed. If L is of star height 1, it is denoted by an expression of height 1 which is shorter than M. If no such expression denotes L, then *all languages* denoted by those expressions are taken as *letters of a new alphabet*; and if L is of star height 2, it is denoted by an expression of star height 1 on this new alphabet and which is shorter than M. And so on. The process stops at the latest when the star height of E is reached.

Although very impressive and recognized as a tour de force, this solution presents unsatisfactory aspects and this opinion can be explained in two ways. First, the algorithm that this solution embodies is not only of high complexity (it is known to be PSPACE-hard) but also leads to computations that are by far impossible, even for very small examples. For instance, if L is accepted by a 4 state automaton of loop complexity 3 (and with a small 10 element transition monoid), then a *very low minorant* of the number of languages to be compared with L for equality is:

$$\left(10^{10^{10}}\right)^{\left(10^{10^{10}}\right)^{\left(10^{10^{10}}\right)}} .$$

Second, and this is not independent from the previous observation, the algorithm does not deal with *any structural aspect* of the language, *i.e.* the data are just *numbers*, here the number of elements of the syntactic monoid, but not the monoid itself, nor the automaton, whose form or properties could be taken into account. This is the reason why we think that the problem deserves to be considered again.

We address here the problem of computing the star height of the family of reversible languages, that are a very natural generalization of pure-group languages. A regular language is *reversible* if it is recognized by a *reversible automaton*. An automaton is reversible if every letter, and thus every word, induces a 1-1 partial mapping on the set of states of the automaton, *i.e.* if its transition function is both deterministic and co-deterministic. These reversible languages have already attracted much attention in automata theory, and more widely in theoretical computer science. They have been studied in connexion with inverse monoids (Silva [18]) and for their topological properties (Pin [15], Héam [13]).

It is noteworthy that the minimal automaton of a reversible language is not necessarily a reversible automaton — a part of the originality and the intricacy of our result lays in that discrepancy. Pin has shown in [15] that it is decidable whether a regular language is reversible or not. This makes the study of reversible languages meaningful.

The starting point of our work is the definition of the *universal automaton* of a language. The universal automaton $\mathcal{U}_L$ of a language L is finite if and only if L is regular (if the minimal automaton of L has n states, $\mathcal{U}_L$ is effectively computable and has between n and 2^n states) and has the property that any automaton $\mathcal{A}$ which recognizes L has a morphic image in $\mathcal{U}_L$. Somehow, $\mathcal{U}_L$ plays the same role with respect to *any* automaton which recognizes L as the role played by the *minimal automaton* of L with respect to *any deterministic* automaton which recognizes L. In particular, $\mathcal{U}_L$ contains as a subautomaton any minimal automaton (even non-deterministic ones) that recognizes L. The definition of the universal automaton is directly derived from the one of the *factor matrix* of a language, given by Conway in [5, Chap. 6]. The definition of the universal automaton has also been mentionned in [1].

The aim of this paper is the presentation and the proof of the following result.

Theorem 1. *The universal automaton of a reversible language K contains a subautomaton of minimal loop complexity that recognizes K.*

As the universal automaton $\mathcal{U}_K$ is effectively computable, this result directly yields, by means of a simple inspection of all its subautomata and their loop complexity, an algorithm computing the star height of K which is far more faster than the one derived from Hashiguchi's method. The mere statement of that algorithm yields a procedure which has a doubly exponential complexity in the number of the states of the minimal automaton recognizing K.

Theorem 1 clearly extends the new version we have given to McNaughton's theorem by means of the universal automaton ([12]). Along the same line as

McNaughton, star height was shown to be computable by Cohen [3] for "(special) reset-free events" and by Hashiguchi [9] for "reset-free events" in general. As reset-free events are reversible languages whose minimal automaton is reversible (the special ones are those for which this minimal automaton has only one final state), Theorem 1 stricly encompasses these developments as well. It should be noted also that the way Theorem1 is proved is reverse to the one used in McNaughton's or Hashiguchi's proofs. They start from an automaton which has the desired properties (it is the minimal automaton of the language) and they show they are able to build from it an automaton of minimal loop complexity for the language. We do not (need to) know explicitely the reversible automaton that accepts the language but we show that an automaton of minimal loop complexity can be found in an automaton that we can compute from the language (it is the universal automaton).

The paper is organized as follow. We first recall the definitions of star height and loop complexity, and give in Section 2 the one of universal automaton of a language. In Section 3, we present McNaughton result on pure-group languages within our framework, as it is an introduction to the main theorem. In the next section, we define the reversible languages and state the main property of their universal automaton by means of the subset expansion. The proof of Theorem1 is given in the last section. Due to space limitation, proofs are somewhat sketchy.

1 From Star Height of Expressions to Loop Complexity of Automata

We follow [8] for the standard definitions and notation for automata.

Regular expressions (over A^*) are the well-formed formulae built from the atomic formulae that are $\mathsf{0}$, 1 and the letters of A and using the binary operators "+" and "·" and the unary operator "$*$" . The *star height* of an expression E, denoted by $\mathsf{h}(\mathsf{E})$, is defined recursively by:

$$\begin{array}{ll} \text{if } \mathsf{E} = \mathsf{0},\ \mathsf{E} = 1 \text{ or } \mathsf{E} = a \in A, & \mathsf{h}(\mathsf{E}) = 0\,, \\ \text{if } \mathsf{E} = \mathsf{E}' + \mathsf{E}'' \text{ or } \mathsf{E} = \mathsf{E}' \cdot \mathsf{E}'', & \mathsf{h}(\mathsf{E}) = \max(\mathsf{h}(\mathsf{E}'), \mathsf{h}(\mathsf{E}''))\,, \\ \text{if } \mathsf{E} = \mathsf{F}^*, & \mathsf{h}(\mathsf{E}) = 1 + \mathsf{h}(\mathsf{F})\ . \end{array}$$

Example 1. The expressions $(a+1)(a^2+b)^*a+1$ and $(b^*a+1)\,(a\,b^*a)^*$ have star height 1 and 2 respectively. As they both denote the language K_1 accepted by the automaton $\mathcal{A}_1$, this shows that two equivalent expressions may have different star heights.

Definition 1. *The star height of a regular language K of A^*, which we note as* $\mathsf{h}(\mathrm{K})$*, is the minimum of the star height of the expressions that denote*[2] *the*

[2] We write $|\mathsf{E}|$ for the language denoted by the expression E. Similarly, we write $|\mathcal{A}|$ for the language accepted by the automaton $\mathcal{A}$.

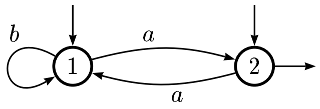

Fig. 1. The automaton $\mathcal{A}_1$

language K:

$$\mathsf{h}(\mathrm{K}) = \min\{\mathsf{h}(\mathsf{E}) \mid \mathsf{E} \in \mathsf{Rat}\, A^* \quad |\mathsf{E}| = K\} .$$

The star height of an expression reflects also a structural property of an automaton (more precisely, of the underlying graph of an automaton) which corresponds to that expression. In order to state it, we define the notion of *a ball* of a graph: a ball in a graph is a strongly connected component that contains at least one arc.

Definition 2. *The* loop complexity[3] *of a graph* $\mathcal{G}$ *is the integer* $\mathsf{lc}(\mathcal{G})$ *recursively defined by:*

$\mathsf{lc}(\mathcal{G}) = 0$ *if* $\mathcal{G}$ *contains no ball (in particular, if* $\mathcal{G}$ *is empty);*
$\mathsf{lc}(\mathcal{G}) = \max\{\mathsf{lc}(\mathcal{P}) \mid \mathcal{P}$ *ball of* $\mathcal{G}\}$ *if* $\mathcal{G}$ *is not a ball itself;*
$\mathsf{lc}(\mathcal{G}) = 1 + \min\{\mathsf{lc}(\mathcal{G} \setminus \{\mathrm{s}\}) \mid s$ *vertex of* $\mathcal{G}\}$ *if* $\mathcal{G}$ *is a ball.*

As Eggan showed, star height and loop complexity are the two faces of the same notion:

Theorem 2. *[7] The star height of a language K is equal to the minimal loop complexity of an automaton that recognizes K.*

In a previous paper [12], we showed an even stronger connection between star height of an expression and loop complexity of an automaton.

Proposition 1. *The loop complexity of a trim automaton* $\mathcal{A}$ *is equal to the infimum of the star height of the expressions that are obtained by the different possible runs of the McNaughton-Yamada algorithm on* $\mathcal{A}$*.*

Theorem 2 allows to deal with automata instead of expressions, and to look for automata of minimal loop complexity instead of expressions of minimal star height. This is what we do in the sequel.

[3] Eggan [7] as well as Cohen [3] and Hashiguchi [9] call it “cycle rank”, Bchi calls it “feedback complexity”. McNaughton [14] calls loop complexity of a language the minimum cycle rank of an automaton that accepts the language. We have taken this terminology and made it parallel to star height, for “rank” is a word of already many different meanings.

2 The Universal Automaton of a Language

Let $\mathcal{A} = \langle Q, M, E, I, T\rangle$ be an automaton[4] over a monoid M. For any state q of $\mathcal{A}$ let us call "*past of* q *(in* $\mathcal{A}$*)* " the set of labels of computations that go from an initial state of $\mathcal{A}$ to q, let us denote it by $\mathsf{Past}_{\mathcal{A}}(q)$; *i.e.*

$$\mathsf{Past}_{\mathcal{A}}(q) = \{m \in M \mid \exists i \in I \quad i \xrightarrow[\mathcal{A}]{m} q\}.$$

In a dual way, we call "*future of* q *(in* $\mathcal{A}$*)*" the set of labels of computations that go from q to a final state of $\mathcal{A}$, and we denote it by $\mathsf{Fut}_{\mathcal{A}}(q)$; *i.e.*

$$\mathsf{Fut}_{\mathcal{A}}(q) = \{m \in M \mid \exists t \in T \quad q \xrightarrow[\mathcal{A}]{m} t\}.$$

For every q in Q it then obviously holds:

$$[\mathsf{Past}_{\mathcal{A}}(q)]\ [\mathsf{Fut}_{\mathcal{A}}(q)] \subseteq |\mathcal{A}|\ . \tag{1}$$

Moreover, if one denotes by $\mathsf{Trans}_{\mathcal{A}}(p,q)$ the set of labels of computations that go from p to q, it then holds:

$$[\mathsf{Past}_{\mathcal{A}}(p)]\ [\mathsf{Trans}_{\mathcal{A}}(p,q)]\ [\mathsf{Fut}_{\mathcal{A}}(q)] \subseteq |\mathcal{A}|\ . \tag{2}$$

It can also be observed that a state p of $\mathcal{A}$ is initial (resp. final) if and only if 1_{A^*} belongs to $\mathsf{Past}_{\mathcal{A}}(p)$ (resp. to $\mathsf{Fut}_{\mathcal{A}}(p)$).

Hence, if K is the subset of M recognized by $\mathcal{A}$, every state of $\mathcal{A}$ induces a *subfactorization* of K: this is how equation (1) will be called. It is an idea due to J. Conway [5, chap. 6] to take the converse point of view, that is to build an automaton from the factorizations of a subset (in any monoid).

More specifically, let K be any subset of a monoid M and let us call *factorization* of K a pair (L, R) of subsets of M such that $L\,R \subseteq K$ and (L, R) is *maximal*[5] for that property in $M \times M$. We denote by Q_K the set of factorizations of K and for every p, q in Q_K *the factor* $F_{p,q}$ of K is the subset of M such that

$$L_p\, F_{p,q}\, R_q \subseteq K$$

and $F_{p,q}$ is *maximal* for that property in M. If $\alpha\colon M \longrightarrow N$ is a morphism that recognizes K, *i.e.* $K\alpha\alpha^{-1} = K$, and if (L, R) is a factorization and F is a factor of K then:

i) $L = L\alpha\alpha^{-1}$, $R = R\alpha\alpha^{-1}$, and $F = F\alpha\alpha^{-1}$;

ii) $(L\alpha, R\alpha)$ is factorization and $F\alpha$ is factor of $K\alpha$;

or, in other words, factorizations and factors are *syntactic objects* with respect to K. As a consequence, Q_K is finite if and only if K is recognizable.

In [5], the $F_{p,q}$ are organized as a $Q_K \times Q_K$-matrix of subsets of A^*, called the *factor matrix* of the language K. A further step consists in building an

[4] An automaton over M is a labelled graph where the label of edges are taken in M.

[5] In the partial order induced by the inclusion in M.

automaton, which we call *the universal automaton* of K, denoted by $\mathcal{U}_K$, and based on the factorizations and the factors of K:

$$\mathcal{U}_K = \langle Q_K, A, E_K, I_K, T_K \rangle ,$$

$$\text{where} \qquad I_K = \{p \in Q_K \mid 1_{A^*} \in L_p\} , \quad T_K = \{q \in Q_K \mid 1_{A^*} \in R_q\}$$

$$\text{and} \qquad E_K = \{(p, a, q) \in Q_K \times A \times Q_K \mid p, q \in Q_K , \ a \in F_{p,q}\} ,$$

and, obviously, $|\mathcal{U}_K| = K$. What makes $\mathcal{U}_K$ *universal* is expressed in the following result.

Theorem 3. *[16] If $\mathcal{A} = \langle Q, A, E, I, T \rangle$ is a trim automaton that accepts K, then there exists an* automaton morphism *from $\mathcal{A}$ into $\mathcal{U}_K$, and $\mathcal{U}_K$ is minimal for this property.*

In particular, $\mathcal{U}_K$ contains as a subautomaton every *minimal* automaton (deterministic, or non deterministic) that recognizes K.

Example 2. The factorizations of $K_1 = |\mathcal{A}_1|$ are (they have to be computed in the syntactic monoid):

$$(a^*, (1 + b^*a)(ab^*a)^*),$$
$$(a^*b^*(a^2 + b)^*, (a^2 + b)^*a),$$
$$(1 + a^*b^*(a^2 + b)^*a, (ab^*a)^*).$$

The universal automaton of K_1 is shown opposite.

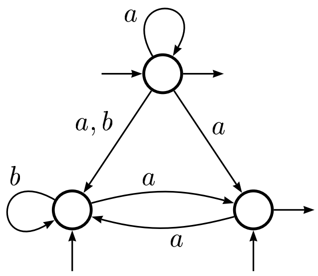

Fig. 2. The universal automaton of K_1

3 The Star Height of Pure-Group Languages

Before dealing with reversible languages in whole generality, we present the statement and the proof of McNaughton's theorem on pure-group languages by means of the universal automaton (*cf.* [12] for a complete exposition).

Theorem 4. *The universal automaton of a pure-group language K contains a subautomaton of minimal loop complexity that recognizes K.*

For any automaton $\mathcal{B}$ that recognizes a language K — and in particular for one of minimal loop complexity — there exists a morphism from $\mathcal{B}$ into $\mathcal{U}_K$. If an (automaton) morphism were preserving loop complexity or, at least, were not increasing it, the theorem would follow immediately, and not only for group languages but for any language. But this is not the case, by far. With that idea in mind, one has to consider morphisms of special kind.

Definition 3. *A morphism $\varphi \colon \mathcal{B} \to \mathcal{A}$ is said to be* conformal[6] *if any computation in $\mathcal{A}$ is the image of (at least) one computation in $\mathcal{B}$.*

[6] McNaughton call them "*pathwise*" (*in* [14]) but his definition of morphism is slightly different from ours.

Theorem 5. *[14, Theorem 3] If $\varphi\colon \mathcal{B} \to \mathcal{A}$ is a conformal morphism, then the loop complexity of $\mathcal{B}$ is greater than or equal to the one of $\mathcal{A}$: $\mathsf{lc}(\mathcal{B}) \geqslant \mathsf{lc}(\mathcal{A})$.* □

Even in the case of a pure-group language K, the morphism φ from an automaton $\mathcal{B}$ (that recognizes K) into $\mathcal{U}_K$ is not necessarily conformal. The proof of Theorem 5 boils down to show that, nevertheless, φ is conformal on those balls of $\mathcal{B}$ that are crucial for the loop complexity. This is proved *via* the following two results, and this is where our method differs from McNaughton's proof.

Proposition 2. *The balls of the universal automaton of a pure-group language are deterministic and complete*[7]*.*

The result follows from the fact that every state of $\mathcal{U}_K$ can be identified with a subset of the syntactic group G of K. The balls of $\mathcal{U}_K$ are exactly the orbits of these subsets under the action of the group G.[8]

Lemma 1. *Let K be a language of A^* whose syntactic monoid is a group G. For every g in G, let H_g be the set of words whose image in G is g and let $W = H_{1_G}$. Let $\mathcal{A}_K = \langle Q, A, E, \{i\}, T\rangle$ be the minimal automaton of K and $\mathcal{B}$ any equivalent automaton.*

For every g in the image of K in G, there exists a state r in $\mathcal{B}$ such that:

i) $W \cap \mathsf{Past}_{\mathcal{B}}(r) \neq \emptyset$ and $H_g \cap \mathsf{Fut}_{\mathcal{B}}(r) \neq \emptyset$.

ii) For every loop[9]*, labelled by a word y, around the initial state i in $\mathcal{A}_K$, there exists a loop, labelled by a word $x\,y\,z$, around r in $\mathcal{B}$, where x is in W.*

Proof. Let k be the order of G and n the number of states of $\mathcal{B}$. Let l be an integer and C_l the set of words of length smaller than l and labelling a loop around i in $\mathcal{A}_K$. Let w_l be the product of all k-th power of words in C_l:

$$w_l = \prod_{v \in C_l} v^k \ .$$

Every v^k is in W and so is w_l. The image of w_l in G is 1_G. Therefore, for every u in H_g, ${w_l}^n u$ is in the language K. Hence, there is a successful computation, labelled by ${w_l}^n u$ in $\mathcal{B}$. As $\mathcal{B}$ has only n states, there is a loop labelled by a power of w_l around a state r_l of $\mathcal{B}$. For r_l, i) holds and ii) holds for y shorter than l.

If we consider the infinite sequence $r_0, r_1, r_2, \ldots$, we can find a state r that occurs infinitely often. Thus i) and ii) are met by this state r. □

[7] *i.e.* for every state p in every ball, there exists for every letter of the alphabet exactly one transition whose origin is p, that is labelled by a, and whose end belongs to the same ball as p.

[8] Let us note that we give here a statement which is sligtly different from the corresponding lemma (Lemma 6) in [12]. The forthcomming Lemma 3 appears thus as a natural generalization of Lemma 1 for reversible languages.

[9] We call loop around s any path whose origin and end are s.

Proof. (of Theorem 4) Let φ be a morphism from $\mathcal{B}$ into $\mathcal{U}_K$, $\mathcal{C}_g$ the ball of $\mathcal{B}$ containing r and $\mathcal{D}_g$ the ball of $\mathcal{U}_K$ containing $r\varphi$. Proposition 2 and Lemma 1 imply together that the restriction of φ to $\mathcal{C}_g$ is *conformal* and *surjective* onto $\mathcal{D}_g$ and that $\mathcal{D}_g$ accepts all words in H_g. The union of the $\mathcal{D}_g$, for g in the image of K in G recognizes K and its loop complexity is smaller than or equal to the one of $\mathcal{B}$ (which could have been chosen of minimal loop complexity). $\square$

4 Reversible Languages and their Universal Automata

Definition 4. *An automaton (on a free monoid) is* reversible *if its transition function is deterministic and codeterministic. A regular language is* reversible *if it is accepted by a finite reversible automaton.*

The minimal automaton of a reversible language is not necessarily reversible, *cf.* on Fig. 3, the minimal automaton of the reversible automaton $\mathcal{A}_1$. However, it is decidable whether a given regular language is reversible by performing some computations on its syntactic monoid (*cf.* [15]).

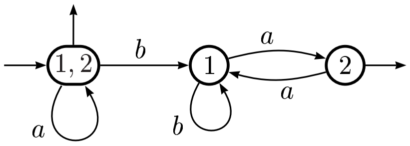

Fig. 3. The minimal automaton of $\mathcal{A}_1$

By comparison, the "reset-free events" of Hashiguchi ([9]) are those reversible languages whose minimal automaton is reversible and the "special[10] reset-free events" of Cohen ([3]) are those for which this minimal automaton has only one final state.

As for group languages, the proof of Theorem 1 will be based on a property of universal automata.

Proposition 3. *The balls of the universal automaton of a reversible language are reversible.*

The proof of Proposition 3 is far more elaborate than the corresponding property for group language. It is first based on the construction of a (good) approximation of $\mathcal{U}_K$ from an automaton $\mathcal{A}$ which accepts K without computing its transition monoid. The states of this new automaton will be *sets of subsets* of the state set of $\mathcal{A}$.

Let $\mathcal{A} = \langle Q, A, E, I, T\rangle$ be an automaton that accepts K and (λ, μ, ν) the corresponding Boolean representation:

$$\begin{aligned} & \lambda \in \mathbb{B}^Q \text{ is a row vector: } \lambda_p = 1 \Leftrightarrow p \in I\ , \\ & \nu \in \mathbb{B}^Q \text{ is a column vector: } \nu_p = 1 \Leftrightarrow p \in T\ , \\ \text{and} \quad & \mu\colon A^* \to \mathbb{B}^{Q\times Q} \text{ is a morphism: } (a\mu)_{p,q} = 1 \Leftrightarrow (p,a,q) \in E\ . \end{aligned}$$

[10] They are not called special in [3] for they are the only ones considered.

For every u in A^*, $\lambda{\cdot}u\mu$ is a (row) vector of $\mathbb{B}^Q$ and thus represents a subset of Q. A set of vectors, *i.e.* a set of subsets of Q, is an *antichain* if its elements are incomparable (for the inclusion order). For instance, the set of antichains in $\mathfrak{P}(\mathbb{B}^{\{1,2\}})$ is $\Big\{\{(0,0)\},\{(1,0)\},\{(0,1)\},\{(1,1)\},\{(1,0),(0,1)\}\Big\}$.

One can associate to any factorization (L,R) of K, indeed to L, an antichain of $\mathfrak{P}(\mathbb{B}^Q)$: if u and v are in L and R respectively, then: $\lambda{\cdot}u\mu{\cdot}v\mu{\cdot}\nu = 1$; if u' is such that $\lambda{\cdot}u'\mu = \lambda{\cdot}u\mu$, then $u' \in L$ since (L,R) is maximal. Moreover, if $\lambda{\cdot}u\mu \subseteq \lambda{\cdot}u'\mu$, then u' is in L as well and therefore L is characterized by an antichain.

Definition 5. *Let* (λ,μ,ν) *be the Boolean representation of an automaton* $\mathcal{A}$ *on* A^**. Let* S *be the set of subsets of* $\lambda{\cdot}A^*\mu$ *that are antichains. We call* subset expansion *of* $\mathcal{A}$ *the automaton* $\mathcal{V}_\mathcal{A} = \langle S,A,F,J,U\rangle$ *defined by*

$$J = \{Y \in S \mid \exists\xi \in Y \quad \xi \subseteq \lambda\}\,, \qquad U = \{X \in S \mid \forall\theta \in X \quad \theta{\cdot}\nu = 1\}$$
$$F = \{(X,a,Y) \in S \times A \times S \mid \forall\theta \in X\,,\ \exists\xi \in Y \quad \xi \subseteq \theta{\cdot}a\mu\}\,.$$

It is immediate that $\mathcal{V}_\mathcal{A}$ is equivalent to $\mathcal{A}$, and the following lemma is not difficult to prove.[11]

Lemma 2. *If an automaton* $\mathcal{A}$ *accepts* K*,* $\mathcal{U}_K$ *is a subautomaton of* $\mathcal{V}_\mathcal{A}$*.* □

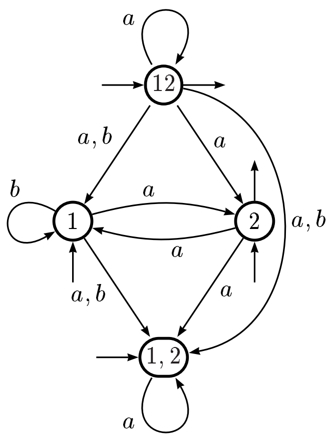

Fig. 4. The subset expansion of $\mathcal{A}_1$

[11] It can be noted that the definition of the subset expansion does not come completely out of the blue: in the case where $\mathcal{A}$ is the minimal automaton of a language K, $\mathcal{V}_\mathcal{A}$ is "the subset automaton of order 0 of K" described by Cohen and Brzozowski in [4].

Proposition 4. *The balls of the subset expansion of a reversible automaton are deterministic.*

Proof. Let $\mathcal{V}_\mathcal{A}$ be the subset expansion of $\mathcal{A}$ represented by (λ, μ, ν) of dimension Q. Let X and Y be two states in a given ball of $\mathcal{V}_\mathcal{A}$. There exist then u in $\mathsf{Trans}_{\mathcal{V}_\mathcal{A}}(X, Y)$ and v in $\mathsf{Trans}_{\mathcal{V}_\mathcal{A}}(Y, X)$. Recall that X and Y are sets of subsets of Q; we denote by X_i (resp. by Y_i) the subset of elements of X (resp. of Y) which contain i states of Q.

Claim: for every integer i, there is a bijection from X_i into Y_i (resp. from Y_i into X_i) induced by u (resp. by v). Suppose the claim does not hold and let i be the smallest integer such that u does not induce a bijection from X_i into Y_i.

i) If there exists θ in X_i such that $\theta{\cdot}u\mu$ is not in Y_i, there exist $j < i$, ξ in Y_j and θ' in X_j such that $\theta'{\cdot}u\mu = \xi \subseteq \theta{\cdot}u\mu$. As $u\mu$ is a 1-to-1 mapping, $\theta' \subseteq \theta$, which is a contradiction because θ and θ' are incomparable. Thus u induces a function from X_i into Y_i.

ii) If there exist θ and θ' in X_i such that $\theta{\cdot}u\mu = \theta'{\cdot}u\mu$, then, again since $u\mu$ is a 1-to-1 mapping, $\theta = \theta'$. Thus u induces an injective function from X_i into Y_i.

iii) For the same reason, v induces an injective function from Y_i into X_i. Thus both u and v induce a bijection between X_i and Y_i.

Therefore, the state that can be reach in a ball by a path labelled by u from a state X is completely defined by $\{\theta{\cdot}u\mu \mid \theta \in X\}$. The balls of $\mathcal{V}_\mathcal{A}$ are deterministic.

Proposition 3 is then the direct consequence of Lemma 2 and of Proposition 4. Indeed, if the balls of $\mathcal{V}_\mathcal{A}$ are deterministic, so are those of $\mathcal{U}_K$. If K is reversible, $\widetilde{K}$, the mirror image of K is reversible as well, and the balls of $\mathcal{U}_{\widetilde{K}}$, which are *the transposed* of those of $\mathcal{U}_K$, are deterministic. Therefore, the balls of $\mathcal{U}_K$ are codeterministic, thus reversible. □

5 Star Height of Reversible Languages

We can now proceed to the proof of Theorem 1. We begin with a property which is the equivalent of Lemma 1 for reversible languages. In order to state that property, we define a decomposition of a language according to an automaton that accepts it; this is an adaptation of a method devised by Hashiguchi in [9].

Let $\mathcal{A} = \langle Q, A, E, I, T\rangle$ be an automaton that accepts K. Every computation in $\mathcal{A}$ can be decomposed, in a unique way, into a sequence:

$$i \xrightarrow{v_0} p_1 \xrightarrow{u_1} q_1 \xrightarrow{v_1} p_2 \xrightarrow{u_2} q_2 \longrightarrow q_{m-1} \xrightarrow{v_{m-1}} p_m \xrightarrow{u_m} q_m \xrightarrow{v_m} t$$

where i (resp. t) is an initial (resp. final) state, p_i (resp. q_i) is the first (resp. the last) state of the i-th ball crossed by the path, the words v_i label paths between balls and the words u_i, possibly empty, label paths in balls.

Let W_{p_i} be the set of words whose image in the transition monoid of $\mathcal{A}$ is an idempotent *and* that label a loop around p_i; let G_i be the set of words whose image in this monoid is the same as u_i, and let $H_i = W_{p_i} G_i$. The language $v_0\, H_1\, v_1\, H_2 \ldots v_{m-1}\, H_m\, v_m$ is a subset of K that consists of words which label paths with the same "behaviour" in the automaton. We call such a set an $\mathcal{A}$-*constituent* of K. The states p_1, q_1, p_2, ... ,q_m are *the markers* of this $\mathcal{A}$-constituent. There are finitely many distinct $\mathcal{A}$-constituents of K and their union is equal to K.

We are now in a position to state the generalization of Lemma 1.

Lemma 3. *Let $\mathcal{A}$ be a reversible automaton and $\mathcal{B}$ any equivalent automaton. Let $v_0\, H_1\, v_1\, H_2 \ldots v_{m-1}\, H_m\, v_m$ be any $\mathcal{A}$-constituent of $|\mathcal{A}| = K$ and p_1, q_1, p_2, ..., q_m its markers. Then there exist m states r_1, r_2, ... ,r_m in $\mathcal{B}$ such that:*

i) $v_0\, W_{p_1} \cap \mathsf{Past}_{\mathcal{B}}(r_1) \neq \emptyset$, $\quad H_m\, v_m \cap \mathsf{Fut}_{\mathcal{B}}(r_m) \neq \emptyset$,
and, $\quad \forall i \in [1; m-1] \quad (H_i\, v_i\, W_{p_i}) \cap \mathsf{Trans}_{\mathcal{B}}(r_i, r_i + 1) \neq \emptyset$.

ii) For every i in $[1; m]$, for every loop around p_i labelled by a word v, there exists a loop around r_i, labelled by a word $u\,v\,w$, where u is in W_{p_i} and w in A^*.

Proof. There exists an integer k such that, for every word $v \in A^*$, the image of v^k in the transition monoid of $\mathcal{A}$ is an idempotent. Let n be the number of states of $\mathcal{B}$. Let l be an integer that will silently index the sets we define now. For every $i \in [1; m]$, let C_i be the set of words of length smaller than l that label a loop around p_i in $\mathcal{A}$. Let w_i be the product of all k-th power of words in C_i:

$$w_i = \prod_{v \in C_i} v^k \,.$$

For every $v_0 u_1 v_1 \cdots v_m$ in the $\mathcal{A}$-constituent,

$$w = v_0 (w_1)^n u_1 v_1 (w_2)^n u_2 \ldots (w_m)^n u_m v_m$$

is in the $\mathcal{A}$-constituent as well. Hence, there is a successful computation labelled by w in $\mathcal{B}$. As $\mathcal{B}$ has only n states, this path contains, for every i, a loop labelled by a power of w_i around a state r_i of $\mathcal{B}$. The m-tuple $r^{(l)} = (r_1, r_2, \ldots, r_m)$ verifies i) and ii) for y shorter than l. If we consider the infinite sequence $r^{(1)}, r^{(2)}, \ldots,$ we can find an m-tuple that occurs infinitly often and that verifies the lemma. $\square$

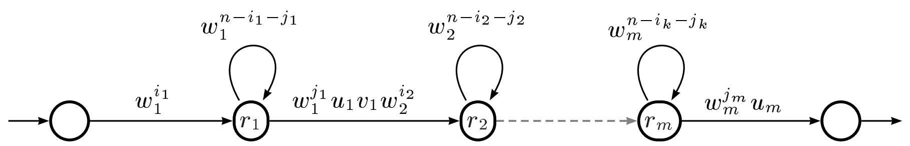

Fig. 5. A witness word for a $\mathcal{A}$-constituent.

Proof. (of Theorem 1, Sketch) Let $\mathcal{A}$ be a reversible automaton that accepts K, $\mathcal{B}$ an equivalent automaton with minimal loop complexity, and $\mathcal{U}_K$ the universal automaton of K.

For every $\mathcal{A}$-constituent of K — with markers $p_1, q_1, p_2, \ldots, q_m$ —, there exist m states $r_1, r_2, \ldots, r_m$ in $\mathcal{B}$ verifying the Lemma 3 and a morphism φ from $\mathcal{B}$ into $\mathcal{U}_K$ that maps $r_1, r_2, \ldots, r_m$ onto $s_1, s_2, \ldots, s_m$.

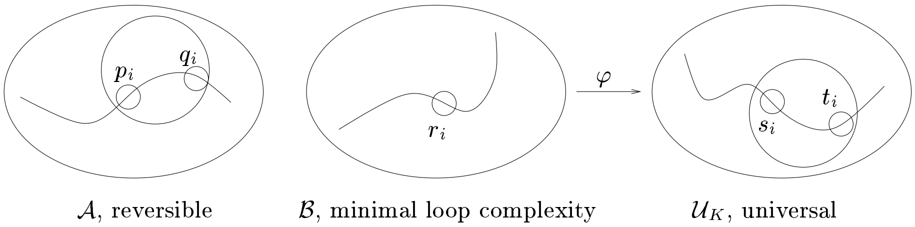

Fig. 6. Proof of Theorem 1

For every i, let $\mathcal{S}_i$ be the ball of $\mathcal{U}_K$ that contains s_i. For every path of $\mathcal{S}_i$, there exists a loop around s_i that contains this path and that is labelled by an idempotent[12] y.

As $\mathcal{U}_K$ is a subautomaton of $\mathcal{V}_\mathcal{A}$, s_i can be seen as a set of subsets of Q. One of these subsets contains p_i. Since there is a bijection from these sets into themselves induced by y, and as y is an idempotent, this bijection is an identity. Therefore, there is a loop labelled by y around p_i.

By Lemma 3, there is a loop in $\mathcal{B}$ labelled by $x\,y\,z$ around r_i, with x idempotent. The image by φ of this loop is a loop around s_i in $\mathcal{U}_K$. Since x is an idempotent, it labels a loop around s_i (in $\mathcal{S}_i$) by itself; thus, there is a path in $\mathcal{S}_i$, beginning in s_i and labelled by y which is the image of a path in the ball of r_i. As the balls of $\mathcal{U}_K$ are deterministic, this path is the one which contains the path chosen in $\mathcal{S}_i$ which is therefore the image by φ of a path of the ball of r_i.

Hence, the morphism φ is conformal from the ball of r_i onto $\mathcal{S}_i$, whose loop complexity is smaller than or equal to the loop complexity of the ball of r_i.

Then it can easily be verified that there exists a state t_i in $\mathcal{S}_i$ such that: i) H_i is a subset of $\mathsf{Trans}_{\mathcal{U}_K}(s_i, t_i)$; ii) if $i < m$, there is a path in $\mathcal{U}_K$ without any loop between t_i and s_{i+1} labelled by v_i, and if $i = m$, t_i is terminal.

This establishes that every $\mathcal{A}$-constituent is accepted by a subautomaton of $\mathcal{U}_K$ with a loop complexity smaller than or equal to the minimal loop complexity for K and this subautomaton contains entire balls of $\mathcal{U}_K$. Thus, the union of these subautomata in $\mathcal{U}_K$ accepts K with minimal loop complexity. $\square$

[12] of the transition monoid of $\mathcal{A}$.

6 A word about complexity

Theorem 1 yields naturally an algorithm which computes the star height of a reversible language.

As the size of the universal automaton of a language K can be exponential in the size of the minimal automaton of K, this algorithm can not be of a lower complexity than exponential. For reasons we shall briefly explain below, the algorithm is indeed doubly exponential. However, the properties that have been established in order to prove Theorem 1 may be taken into account in order to avoid expensives operations (such as, for instance, the enumeration of all subautomata of the universal automaton).

Theorem 1 can be restated as follows: the universal automaton $\mathcal{U}_K$ of a reversible language K contains a subautomaton

i) which recognizes K,

ii) of minimal loop complexity,

ii) whose balls are balls of $\mathcal{U}_K$.

And this statement can be translated into the following algorithm.

First start from $\mathcal{A}_K$, the minimal automaton of K with n states and test whether K is reversible or not; this operation has polynomial complexity [15]. Next, build the universal automaton $\mathcal{U}_K$, identify every ball of $\mathcal{U}_K$ and compute the loop complexity of each of them.

Let k be the loop complexity of $\mathcal{A}_K$. For every integer i from 1 to k, build the largest subautomaton $\mathcal{D}_i$ of $\mathcal{U}_K$ which contains no state of any ball of $\mathcal{U}_K$ of loop complexity larger than i, and test whether $\mathcal{D}_i$ accepts K. The first i for which this condition holds is the star height of K.

There are two critical sections in that algorithm. The first one is the computation of the loop complexity of the balls of $\mathcal{U}_K$. The size of the balls can be exponential in n and the computation of the loop complexity of an automaton is exponential in the size of the automaton. The second critical section is the test for equivalence of $\mathcal{D}_i$. This requires *a priori* the determinization of $\mathcal{D}_i$, which is already of exponential size.

References

1. Arnold A., Dicky A., and Nivat M., A Note about Minimal Non-deterministic Automata. *Bull. of E.A.T.C.S.* **47** (1992), 166–169.
2. Büchi J. R., *Finite Automata, their Algebras and Grammars: Toward a Theory of formal Expressions*. D. Siefkes (Ed.). Springer-Verlag, 1989.
3. Cohen R., Star height of certain families of regular events. *J. Computer System Sci.* **4** (1970), 281–297.
4. Cohen R. and Brzozowski R., General properties of star height of regular events. *J. Computer System Sci.* **4** (1970), 260–280.
5. Conway J. H., *Regular algebra and finite machines*. Chapman and Hall, 1971.
6. Dejean F. and Schützenberger M. P., On a question of Eggan. *Inform. and Control* **9** (1966), 23–25.
7. Eggan L. C., Transition graphs and the star-height of regular events. *Michigan Mathematical J.* **10** (1963), 385–397.

8. Eilenberg S., *Automata, Languages and Machines* vol. A, Academic Press, 1974.
9. Hashiguchi K., The star height of reset-free events and strictly locally testable events. *Inform. and Control* **40** (1979), 267–284.
10. K. Hashiguchi, Algorithms for determining relative star height and star height. *Inform. and Computation* **78** (1988), 124–169.
11. Lombardy S., On the construction of reversible automata for reversible languages, submitted.
12. Lombardy S. and Sakarovitch J., On the star height of rational languages: a new version for two old results, *Proc. 3rd Int. Col. on Words, Languages and Combinatorics*, (M. Ito, Ed.) World Scientific, to appear. Available at the URL: `www.enst.fr/~jsaka`.
13. Héam P.-C., Some topological properties of rational sets. *J. of Automata, Lang. and Comb.*, to appear.
14. McNaughton R., The loop complexity of pure-group events. *Inform. and Control* **11** (1967), 167–176.
15. Pin J.-E., On reversible automata. In *Proc. 1st LATIN Conf., (I. Simon, Ed.), Lecture Notes in Comput. Sci.* **583** (1992), 401–416.
16. Sakarovitch J., *Eléments de théorie des automates.* Vuibert, to appear.
17. Salomaa A., *Jewels of formal language theory.* Computer Science Press, 1981.
18. Silva P., On free inverse monoid languages *Theoret. Informatics and Appl.* **30** (1996), 349–378.