On the star height of rational languages

A new presentation for two old results*

Sylvain Lombardy Jacques Sakarovitch †

Abstract

The *star height* of a rational language, introduced by Eggan in 1963, has proved to be the most puzzling invariant defined for rational languages. Here, we give a new proof of Eggan's theorem on the relationship between the cycle rank of an automaton and the star height of an expression that describes the language accepted by the automaton. We then present a new method for McNaughton's result on the star height of pure-group language; it is based on the definition of a (finite) automaton which can be canonically associated to every (rational) language and which we call *universal*. In contrast with the minimal automaton, the universal automaton of a pure-group language has the property that it contains a subautomaton of minimal cycle rank that recognizes the language.

Résumé

La *hauteur d'étoile* d'un langage rationnel, invariant défini par Eggan en 1963, s'est révélé être un des aspects les plus difficiles de l'étude de cette famille de langages. Dans cet article, nous donnons une nouvelle preuve du théorème d'Eggan sur le lien entre le rang cyclique d'un automate et la hauteur d'étoile d'une expression qui dénote le langage accepté par l'automate. Nous présentons ensuite une nouvelle méthode pour établir le résultat de McNaughton sur la calculabilité de la hauteur d'étoile des langages à groupes. Elle est fondée sur la considération de ce que nous appelons l'*automate universel* d'un langage, automate canoniquement associé à tout langage, fini quand le langage est rationnel. Contrairement à l'automate minimal, l'automate universel d'un langage à groupe a la propriété qu'il contient un sous-automate de rang cyclique minimum qui reconnaît le langage.

*to appear in the proceedings of the 3rd International Conference on Words, Languages, and Combinatorics, Kyoto (Japan), March 2 000. This confrence has also been presented at the DCAGRS 2000 Workshop, London (Ontario, Canada), July 2000

†Laboratoire Traitement et Communication de l'Information (C. N. R. S. URA 820), E. N. S. T., Paris.

The star height of a rational language is the infimum of the star height of the rational expressions that denote the language. The star height has been defined in 1963 by Eggan (in [8]) who basically proved one thing and asked two questions.

Eggan showed that the star height of a rational expression is related to another quantity that is defined on a finite automaton which produces the expression, a quantity which he called *rank* and which we call here *loop complexity.* He proved that there are rational languages of arbitrary large star height, provided that an arbitrary large number of letters are available. And he stated the following two problems.

- Is the star height of a rational language computable?

- Does there exist, on a fixed finite alphabet, rational languages of arbitrary large star height?

For a long time, the first one was considered as one of the most difficult problem in the theory of automata and eventually solved (positively) by Hashiguchi in 1988 ([9]).

The second problem, much easier, was solved in 1966 by Dejean and Schützenberger, positively as well ([7]). Soon afterwards, in 1967, McNaughton published a paper, entitled "The loop complexity of pure-group languages" ([10]) where he gave a conceptual proof of what Dejean and Schützenberger had established by means of combinatorial virtuosity (one of the "jewels" in formal language theory [12]). He proved that the loop complexity, and thus the star height, of 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 belong to that family).

The purpose of this communication is to give a new, and hopefully enlightening, presentation of Eggan's and McNaughton's results. We first give a new proof of Eggan's theorem, by describing an explicit correspondence between the computation that yields the loop complexity of an automaton and the computation of an expression that denotes the language accepted by the automaton. We then present a new method for McNaughton's result on the star height of pure-group language; it is based on the definition of a (finite) automaton which can be canonically associated to every (rational) language and which we call *universal.* In contrast with the minimal automaton, the universal automaton of a pure-group language has the property that it contains a subautomaton of minimal cycle rank that recognizes the language.

We mostly use the classical terminology, notation and results for automata and languages. We give explicit notes when we depart from the standard ones.

1 Eggan's Theorem

1.1 Star height and loop complexity

Rational expressions (over A^*) are the well formed formulae built from the atomic formulae that are 0, 1 and the elements of A and using the binary operators $+$ and $\cdot$ and the unary operator $*$.

The operator $*$ is the one that "gives access to infinity". Hence the idea of measuring the complexity of an expression as the largest number of nested calls to that operator in the expression. This number is called the *star height* of the expression, denoted by $\mathsf{h}[\mathsf{E}]$ and defined recursively by:

$$\text{if } \mathsf{E} = \mathsf{0},\ \mathsf{E} = \mathsf{1} \text{ or } \mathsf{E} = a \in A, \qquad \mathsf{h}[\mathsf{E}] = 0 \ , \tag{1}$$

$$\text{if } \mathsf{E} = \mathsf{E}' + \mathsf{E}'' \text{ or } \mathsf{E} = \mathsf{E}' \cdot \mathsf{E}'', \qquad \mathsf{h}[\mathsf{E}] = \mathsf{max}(\mathsf{h}[\mathsf{E}'], \mathsf{h}[\mathsf{E}'']) \ , \tag{2}$$

$$\text{if } \mathsf{E} = \mathsf{F}^*, \qquad \mathsf{h}[\mathsf{E}] = 1 + \mathsf{h}[\mathsf{F}] \ . \tag{3}$$

Examples 1 : i) $\mathsf{h}[(a+b)^*] = 1$; $\mathsf{h}[a^*\,(b\,a^*)^*] = 2$.

ii) $\mathsf{h}[a^* + a^*b(ba^*b)^*ba^* + a^*b(ba^*b)^*a(b + a(ba^*b)^*a)^*a(ba^*b)^*ba^*] = 3$,

$\mathsf{h}[(a + b(ba^*b)^*b)^*] = 3$; $\mathsf{h}[a^*b(ab^*a + ba^*b)^*ba^*] = 2$. □

These examples show that two equivalent expressions may have different star heights. The following definition is then natural.

Definition 1 *The star height of a rational language L of A^*, denoted by $\mathsf{h}[L]$, is the minimum of the star height of the expressions that denote*[1] *the language L:*

$$\mathsf{h}[L] = \mathsf{min}\{\mathsf{h}[\mathsf{E}] \mid \mathsf{E} \in \mathsf{RatE}\,A^* \quad |\mathsf{E}| = L\} \ .$$

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*[2] of a graph: a ball in a graph is a strongly connected component that contains at least one arc (*cf.* Figure 1).

[1] We write $|\mathsf{E}|$ for the language denoted by the expression E. Similarly, we write $|\mathcal{A}|$ for the language accepted by the automaton $\mathcal{A}$. $\mathsf{RatE}\,A^*$ is the set of rational expressions over the alphabet A.

[2] Like in *a ball of wool.*

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

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

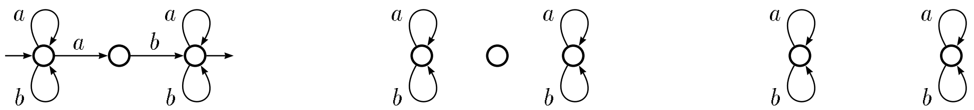


Figure 1: An automaton, its strongly connected components, and its balls.

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

Theorem 1 [8] *The loop complexity of a trim automaton $\mathcal{A}$ is equal to the infimum of the star height of the expressions (denoting $|\mathcal{A}|$) that are obtained by the different possible runs of the McNaughton-Yamada algorithm on $\mathcal{A}$.*

There is an infimum "hidden" in the definition of the loop complexity and the theorem states that it is equal to another infimum. We shall relate more closely the two quantities, star height and loop complexity, by defining them relative to an order of the states of the automaton and showing that they are equal in that setting. The equality of the two minima will follow then obviously.

1.2 The Brzozowski and McCluskey's algorithm

McNaughton-Yamada's algorithm is probably the best known algorithm for computing a rational expression that denotes the language accepted by an automaton. For our purpose however, it is convenient to use a variant of it, due to Brzozowski and McCluskey ([2]), which is completely equivalent[4]. This algorithm has been described in [13]. It uses *generalized automata* and processes by deleting state after state.

Let us call *generalized* an automaton $\mathcal{A} = \langle Q, A, E, I, T \rangle$ in which the labels of the transitions are not *letters* anymore but expressions, that is the elements of E are triples (p, e, q) with p and q in Q and $e \in \mathsf{RatE}\,A^*$. The label of a computation is, as

[3]Eggan [8] calls it "cycle rank". McNaughton [10] 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 differnt meanings.

[4]This statement can be made precise and meaningful: an expression obtained by one algorithm can be transformed into an expression computed by the other by using the axiom $\mathsf{E}^* = 1 + \mathsf{E}\,\mathsf{E}^*$ (*cf.* [11]).

usual, *the product* of the labels of the transitions that constitute this computation and the language accepted by $\mathcal{A}$ is *the union* of the labels of the successful computations of $\mathcal{A}$.

Starting from a (generalized) automaton $\mathcal{A}$, the Brzozowski and McCluskey's algorithm — or BMC algorithm — consists in building a generalized automaton $\mathcal{C}$ which can be called trivial: an initial state i, a final state t (distinct from i) and a single transition from i to t and labelled by a rational expression E which denotes the language accepted by $\mathcal{A}$ (*cf.* Figure 2).

The first phase consists in building a kind of "normalized" automaton $\mathcal{B}$ by adding to $\mathcal{A} = \langle Q, A, E, I, T \rangle$ two distinct states i and t, and a transition labelled by 1_{A^*} from i to every initial state of $\mathcal{A}$, and a transition labelled by 1_{A^*} from every final state of $\mathcal{A}$ to t. The state i is the unique initial state of $\mathcal{B}$, the state t its unique final state: $\mathcal{B}$ is equivalent to $\mathcal{A}$. As $\mathcal{A}$, and then $\mathcal{B}$, are finite, one can assume — after some finite unions on the labels of the transitions — that there is at most one transition from p to q for every pair (p, q) of states of $\mathcal{B}$.

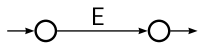


Figure 2: The result of the BMC algorithm .

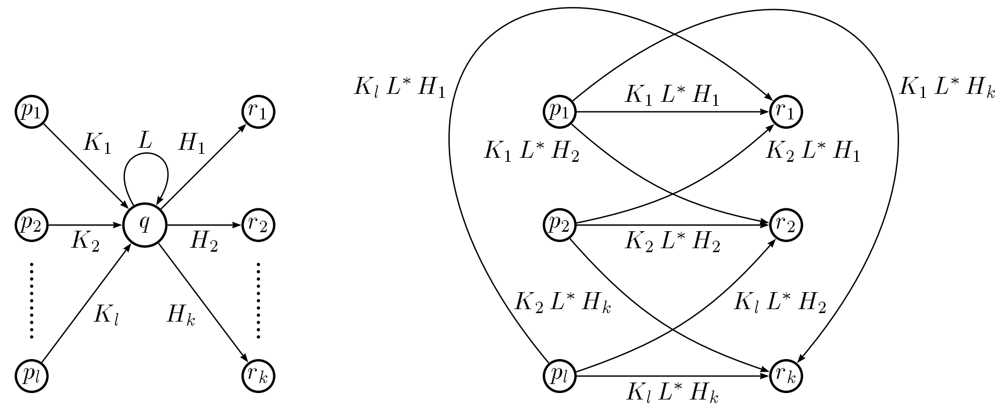
The second phase has as many steps as there are states in $\mathcal{A}$. It consists in successively removing states from $\mathcal{B}$ (but i and t) and to update the transitions in such a way that at every step an equivalent automaton is computed whose labels are obtained from those of the preceding one by union, product and star.

More precisely, let q be an element of Q; let $p_1, p_2, \ldots, p_l$ be the states of $\mathcal{B}$ which are the origin of a transition whose end is q, and $K_1, K_2, \ldots, K_l$ the labels of these transitions; let $r_1, r_2, \ldots, r_k$ be the states of $\mathcal{B}$ which are the end of a transition whose origin is q, and $H_1, H_2, \ldots, H_k$ the labels of these transitions — some of the r_j may coincide with some of the p_h. Let L be the label of the transition whose origin and end is q, if it exists; otherwise, $L = 1_{A^*}$.

Let $\mathcal{B}'$ be the automaton obtained from $\mathcal{B}$ by removing q and all the transitions adjacent to q, and by adding, for every pair of states (p_h, r_j), $1 \leqslant h \leqslant l$ and $1 \leqslant j \leqslant k$, the transition $(p_h, K_h\, L^*\, H_j, r_j)$ (*cf.* Figure 3).

The automata $\mathcal{B}$ and $\mathcal{B}'$ are equivalent. By iterating this construction n times, $(n = \|Q\|)$[5], an automaton $\mathcal{C}$ is obtained that contains no states of the automaton $\mathcal{A}$ and which is of the prescribed form.

[5]The cardinal of a set Q is denoted by $\|Q\|$



(a) Before the deletion of q (b) After the deletion of q

Figure 3: A step of the BMC algorithm .

1.3 The Eggan–Brzozowski index

In order to prove Theorem 1, we define the *EB index* of an automaton $\mathcal{A}$, which is at the same time a generalization and a refinement of the loop complexity.

If $\mathcal{A}$ is a generalized automaton, we call *EB index of a transition* e of $\mathcal{A}$, denoted $\mathsf{i}_{\mathrm{EB}}(e)$, the star height of the label of e:

$$\mathsf{i}_{\mathrm{EB}}(e) = \mathsf{h}[|e|] \enspace .$$

If $\mathcal{A} = \langle Q, A, E, I, T \rangle$ is a "classical" automaton over A:

$$\forall e \in E \qquad \mathsf{i}_{\mathrm{EB}}(e) = 0 \enspace .$$

Then, we define the EB index of an automaton $\mathcal{A}$, not absolutely but relative to a total order ω on the set Q of the states of $\mathcal{A}$, that order which is implicit in the BMC algorithm.

We use the following notation and convention. If ω is an order on Q, we denote by $\overline{\omega}$ *the largest element* of Q for ω. If $\mathcal{S}$ is a subautomaton of $\mathcal{A}$, we still denote ω the trace of the order ω on the set R of the states of $\mathcal{S}$ and, in such a context, $\overline{\omega}$ is the largest element of R for ω.

We call then *EB index of $\mathcal{A}$ relative to* ω, and we note $\mathsf{i}_{\mathrm{EB}}(\mathcal{A}, \omega)$, the integer defined by the following algorithm (called EB algorithm):

- If $\mathcal{A}$ is not a ball:

$$\mathsf{i}_{\mathrm{EB}}(\mathcal{A}, \omega) = \mathsf{max}\big(\{\mathsf{i}_{\mathrm{EB}}(e) \mid e \text{ does not belong to a ball of } \mathcal{A}\} \cup \{\mathsf{i}_{\mathrm{EB}}(\mathcal{P}, \omega) \mid \mathcal{P} \text{ is ball of } \mathcal{A}\}\big) \qquad (4)$$

• If $\mathcal{A}$ is a ball:

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) = 1 + \max\bigl(\{\mathsf{i}_{\mathrm{E\,B}}(e) \mid e \text{ is adjacent to } \overline{\omega}\}, \mathsf{i}_{\mathrm{E\,B}}(\mathcal{A}\setminus\overline{\omega},\omega)\bigr) \tag{5}$$

If $\mathcal{A}$ is a "classical" automaton, (4) and (5) become respectively:

• If $\mathcal{A}$ is not a ball:

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) = \max\bigl(\{\mathsf{i}_{\mathrm{E\,B}}(\mathcal{P},\omega) \mid \mathcal{P} \text{ is ball of } \mathcal{A}\}\bigr) \tag{6}$$

• If $\mathcal{A}$ is a ball:

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) = 1 + \mathsf{i}_{\mathrm{E\,B}}(\mathcal{A}\setminus\overline{\omega},\omega) \tag{7}$$

to which the base of the recurrence has to be added:

• If $\mathcal{A}$ does not contain any ball, or is empty:

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) = 0 \ . \tag{8}$$

From which it is directly derived:

Property 1 $\mathsf{lc}(\mathcal{A}) = \min\{\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) \mid \ \omega$ *order on* Q }. ■

We denote by $\mathsf{E}_{\mathrm{B\,C}}(\mathcal{A},\omega)$ the rational expression obtained by running the BMC algorithm on $\mathcal{A}$ with the order ω, that is by deleting the states of $\mathcal{A}$ *the smallest* first. It should be noted that it follows from these definitions that, once the order ω is fixed, the order of deletion of states in the BMC algorithm is the reverse order of the "deletion" of states in the computation of the EB index. Theorem 1 is then the consequence of the following.

Proposition 1 *Let ω be a total order on the set of states of an automaton $\mathcal{A}$. The EB index of $\mathcal{A}$ relative to ω is equal to the star height of the rational expression obtained by running the BMC algorithm on $\mathcal{A}$ with the order ω, i.e.*

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{A},\omega) = \mathsf{h}[\mathsf{E}_{\mathrm{B\,C}}(\mathcal{A},\omega)] \ .$$

Proof. By induction on the number of states of $\mathcal{A}$. By convention, the states i and t that have been added are larger than all the states of $\mathcal{A}$ in the order ω and are not deleted in the BMC algorithm .

The base of the induction is thus a generalized automaton with 3 states, like in the Figure 4 a) or b).

In case a), $\mathcal{B}$ contains no ball and it holds:

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{B},\omega) = \max\bigl(\mathsf{h}[\mathsf{E}],\mathsf{h}[\mathsf{F}],\mathsf{h}[\mathsf{H}]\bigr) = \mathsf{h}[\mathsf{E}+\mathsf{F}\cdot\mathsf{H}] = \mathsf{h}[\mathsf{E}_{\mathrm{B\,C}}(\mathcal{B},\omega)] \ .$$

In case b), the unique state of $\mathcal{B}$ which is neither initial nor final is a ball whose EB index is $1 + \mathsf{h}[\mathsf{G}]$, and it holds:

$$\mathsf{i}_{\mathrm{EB}}(\mathcal{B}, \omega) = \mathsf{max}\big(\mathsf{h}[\mathsf{E}], \mathsf{h}[\mathsf{F}], \mathsf{h}[\mathsf{H}], (1 + \mathsf{h}[\mathsf{G}])\big) = \mathsf{h}[\mathsf{E} + \mathsf{F} \cdot \mathsf{G}^* \cdot \mathsf{H}] = \mathsf{h}[\mathsf{E}_{\mathrm{BC}}(\mathcal{B}, \omega)] \ .$$

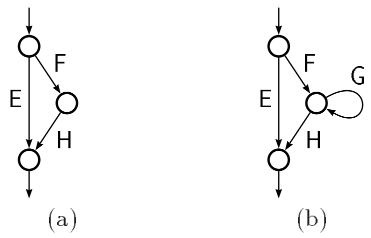


Figure 4: Base of the induction

Let now $\mathcal{B}$ be an automaton of the prescribed form and with $n+2$ states, q the smallest state in the order ω, and $\mathcal{B}'$ the automaton after the first step of the BMC algorithm applied to $\mathcal{B}$ — that is, after deletion of q. Since the adjacency relations (for the other states than q) are the same in $\mathcal{B}$ and in $\mathcal{B}'$, and as q is *the smallest element* in the order ω, the EB algorithm runs in the same way in $\mathcal{B}$ and in $\mathcal{B}'$, *i.e.* the succession of balls build in both cases is identical, up to the processing of q in $\mathcal{B}$ excluded. It remains to show that the computed values are identical as well.

Let $\mathcal{P}$ be the smallest ball of $\mathcal{B}$ that strictly contains q — and if such a ball does not exist, let $\mathcal{P} = \mathcal{B}$ — and let $\mathcal{P}'$ be "the image" of $\mathcal{P}$ in $\mathcal{B}'$. Two cases are possible. If q is not the origin (and the end) of a loop — case a) —, the transitions of $\mathcal{P}'$ are either identical to those of $\mathcal{P}$ or labelled by products $\mathsf{F} \cdot \mathsf{H}$, where F and H are labels of transitions of $\mathcal{P}$. It then comes:

$$\begin{aligned}\mathsf{i}_{\mathrm{EB}}(\mathcal{P}', \omega) &= \mathsf{max}\big(\mathsf{max}\{\mathsf{i}_{\mathrm{EB}}(e) \mid e \text{ does not belong to a ball of } \mathcal{P}'\},\\ &\qquad\qquad\qquad\qquad \mathsf{max}\{\mathsf{i}_{\mathrm{EB}}(\mathcal{Q}, \omega) \mid \mathcal{Q} \text{ is a ball of } \mathcal{P}'\}\big)\\ &= \mathsf{max}\big(\mathsf{max}\{\mathsf{i}_{\mathrm{EB}}(e) \mid e \text{ does not belong to a ball of } \mathcal{P}\},\\ &\qquad\qquad\qquad\qquad \mathsf{max}\{\mathsf{i}_{\mathrm{EB}}(\mathcal{Q}, \omega) \mid \mathcal{Q} \text{ is a ball of } \mathcal{P}\}\big)\\ &= \mathsf{i}_{\mathrm{EB}}(\mathcal{P}, \omega) \ . \end{aligned} \tag{9}$$

If q is the origin (and the end) of a loop labelled by G — case b) —, *i.e.* q is a ball of $\mathcal{B}$ by itself, the transitions of $\mathcal{P}'$ are either identical to those of $\mathcal{P}$ or labelled by products $\mathsf{F}\cdot\mathsf{G}^*\cdot\mathsf{H}$. It then comes, since $\mathsf{i}_{\mathrm{E\,B}}(\{q\},\omega) = 1 + \mathsf{h}[\mathsf{G}]$:

$$\begin{aligned}
\mathsf{i}_{\mathrm{E\,B}}(\mathcal{P}',\omega) &= \max\bigl(\max\{\mathsf{i}_{\mathrm{E\,B}}(e) \mid e \text{ does not belong to a ball of } \mathcal{P}'\}, \\
&\qquad\qquad\qquad \max\{\mathsf{i}_{\mathrm{E\,B}}(\mathcal{Q},\omega) \mid \mathcal{Q} \text{ is a ball of } \mathcal{P}'\}\bigr) \\
&= \max\bigl(\max\{\mathsf{i}_{\mathrm{E\,B}}(e) \mid e \text{ does not belong to a ball of } \mathcal{P}\}, \\
&\qquad\qquad (1+\mathsf{h}[\mathsf{G}]), \max\{\mathsf{i}_{\mathrm{E\,B}}(\mathcal{Q},\omega) \mid \mathcal{Q} \text{ is a ball of } \mathcal{P}'\}\bigr) \\
&= \max\bigl(\max\{\mathsf{i}_{\mathrm{E\,B}}(e) \mid e \text{ does not belong to a ball of } \mathcal{P}\}, \\
&\qquad\quad \mathsf{i}_{\mathrm{E\,B}}(\{q\},\omega), \\
&\qquad\quad \max\{\mathsf{i}_{\mathrm{E\,B}}(\mathcal{Q},\omega) \mid \mathcal{Q} \text{ ball of } \mathcal{P}, \text{ different from } \{q\}\}\bigr) \\
&= \mathsf{i}_{\mathrm{E\,B}}(\mathcal{P},\omega) \ .
\end{aligned} \tag{9'}$$

If $\mathcal{P} = \mathcal{B}$ (and $\mathcal{P}' = \mathcal{B}'$), the equalities (9) and (9)' become

$$\mathsf{i}_{\mathrm{E\,B}}(\mathcal{B}',\omega) = \mathsf{i}_{\mathrm{E\,B}}(\mathcal{B},\omega) \tag{10}$$

which yields the induction and then the proposition. Otherwise, and without any induction on the number of nested balls that contain q, (10) is obtained from (9) by noting that the transitions of $\mathcal{B}'$ are either identical to those of $\mathcal{B}$ or correspond to transitions that are adjacent to q.

In case a), the labels of these transitions are products of the labels of transitions of $\mathcal{B}$, their index is obtained by taking a maximum and (10) follows from the relation $\max(a,b,c) = \max(a,\max(b,c))$.

In case b), the labels of these transitions are, as above, of the form $\mathsf{F}\cdot\mathsf{G}^*\cdot\mathsf{H}$, of index $\max(\mathsf{h}[\mathsf{F}],\mathsf{h}[\mathsf{H}],1+\mathsf{h}[\mathsf{G}])$. The corresponding transition in $\mathcal{B}$ has label F (or H); it is processed by the E B algorithm when the index of the transition of label H (or F) and the one of *the ball* $\{q\}$, whose index is $1+\mathsf{h}[\mathsf{G}]$, are already computed. The result, that is (10), follows then, for the same reason as above. ■

1.4 No rush to conclusion

After Theorem 1 that shows that the correspondance between automata and expressions can be carried on to a correspondance between loop complexity and star height, one could have thought that to the *minimal automaton* would correspond an expression of *minimal star height*. There is no such thing of course (or the star height of a language would not be mysterious anymore). The following example describes one of the simplest language whose minimal automaton is not of minimal loop complexity.

Example 2 : Let F_2 and F_3 be the languages of $A^* = \{a,b\}^*$ consisting of words whose number of a's is congruent to the number of b's plus 1 modulo 2 and 3 respectively and let F_6 be their union:

$$F_2 = \{f \mid |f|_a - |f|_b \equiv 1 \mod 2\}\,, \quad F_3 = \{f \mid |f|_a - |f|_b \equiv 1 \mod 3\}$$
$$\text{and} \qquad F_6 = \{f \mid |f|_a - |f|_b \equiv 1, 3, 4 \text{ or } 5 \mod 6\}\ .$$

The minimal automaton of F_6 is the "double ring" of length 6, whose loop complexity is 3. The minimal automata of F_2 and F_3 have loop complexity 1 and 2, hence the star height of F_6 is at most 2 (*cf.* Figure 5). □

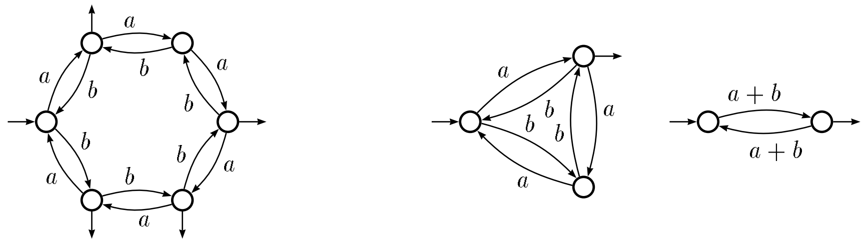


Figure 5: An automaton of minimal loop complexity (right) which is not the minimal automaton (left) for F_6.

2 Conway's universal automaton

The new interpretation of McNaughton's theorem we are aiming at makes use of a construction which is basically due to Conway.

Let $\mathcal{A} = \langle Q, M, E, I, T \rangle$ be an automaton 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:

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

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:

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

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 every automaton, and in every automaton, every state induces a set of *factorizations* — this will be how equations such as $(*)$ or $(**)$ will be called — of the subset accepted by the automaton. It is an idea essentially due to J. Conway [5, chap. 6], and that proved to be extremely fruitful, 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* for that property in $M \times M$.[6] We denote by Q_K the set of factorizations of K. For every p, q in Q_K *the factor* $F_{p,q}$ of K is the maximal subset of M such that

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

It is easy to verify that if $\alpha\colon M \longrightarrow N$ is a surjective morphism that recognizes K, *i.e.* $K\alpha\alpha^{-1} = K$, and if (L, R) is a factorization and F 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], J. Conway defines the $F_{p,q}$, organized as a $Q_K{\times}Q_K$-matrix, as the *factor matrix* of the language K, subset of A^*. A further step consists in building an automaton, which we call *the universal automaton* of K, denoted by $\mathcal{U}_K$, and based on the factorizations and the factors of K:

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

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

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

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

Theorem 2 *If $\mathcal{A} = \langle\, Q, A, E, I, T \,\rangle$ is an automaton that accepts K, then there exists an* automaton *morphism from $\mathcal{A}$ into $\mathcal{U}_K$: $\varphi\colon \mathcal{A} \longrightarrow \mathcal{U}_K$, and $\mathcal{U}_K$ is minimal for this property. Moreover, if $\mathcal{A}$ is* minimal[7] *then φ is* injective.

[6] Maximal in the order induced by the inclusion in M.

[7] With respect to K: no state of $\mathcal{A}$ can be deleted without making $|\mathcal{A}|$ smaller, no two states of $\mathcal{A}$ can be merged without making $|\mathcal{A}|$ larger.

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

Example 3 : Let $K_1 = A^*a\,b\,A^*$ be the language of words that contain at least one factor $a\,b$. Easy computations show that K_1 has 3 factorizations:

$$u = (A^*, A^*a\,b\,A^*)\,, \qquad v = (A^*a\,A^*, A^*b\,A^*)\,, \qquad \text{and} \qquad w = (A^*a\,b\,A^*, A^*)\,,$$

which yield the universal automaton represented at Figure 6. □

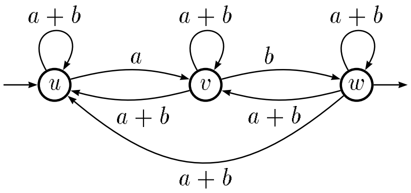
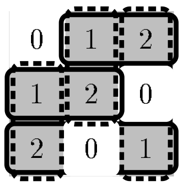


Figure 6: The universal automaton of $K_1 = A^*a\,b\,A^*$

Example 4 : Let E_3 be the language of $A^* = \{a, b\}^*$ consisting of words whose number of a's is not congruent to the number of b's modulo 3:

$$E_3 = \{f \mid |f|_a \not\equiv |f|_b \mod 3\}\ .$$

The 3 factorizations of E_3 are best seen on its syntactic monoid $\mathbb{Z}/3\,\mathbb{Z}$ as represented opposite. The universal automaton of E_3 is then represented, in two ways, at Figure 7. □



The factorizations of E_3.

The construction of the universal automaton by means of factorization has been more or less given in [6] and [1] (where it is refered also to [3]) without reference to the work of Conway. Theorem 2 was stated in [4].

3 McNaughton's Theorem

With the previous definitions, McNaughton's Theorem on pure-group language becomes:

Theorem 3 *The universal automaton of a pure-group language K contains as a subautomaton an automaton of minimal loop complexity that recognizes K.*

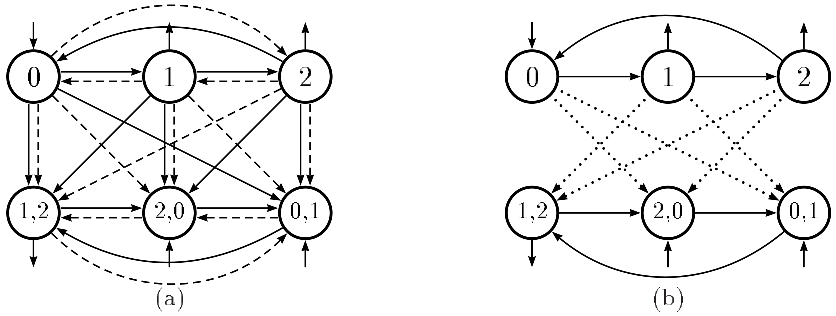


Figure 7: The universal automaton of E_3. On subfigure (a), the transitions labelled by a are represented by solid arrows, the transitions labelled by b by dashed ones. The subfigure (b) is a simplification of the previous one, in preparation of Figure 8 that would be unreadable without these conventions. The arrows labelled by b are removed for they are opposite to the ones labelled by a inside the two levels. The solid and dashed arrows between the two levels are replaced by dotted arrows that can be considered as labelled by 1_{A^*} and that play the role of two previous ones: for instance, the dotted arrow between 0 and $\{2,0\}$ represents the solid arrow between 0 and $\{0,1\}$ and the dashed one beween 0 and $\{1,2\}$ in subfigure (a).

As the universal automaton of a rational language is finite, it is possible to enumerate its subautomata, to keep only those which recognize the language, and to distinguish among them those of minimal loop complexity. Hence:

Corollary 2 [10] *The star height of a rational pure-group language is computable.* ■

On the other hand, the same theorem yields directly what had been established by Dejean and Schützenberger by means of subtle and sophisticated combinatorial arguments:

Corollary 3 [7] *Let W_q be the language of $\{a,b\}^*$ consisting of words whose number of a's is congruent to the number of b's modulo 2^q. Then the star height of W_q is q.*

Proof. The syntactic monoid of W_q is the group $\mathbb{Z}/2^q\mathbb{Z}$ and the image of W_q in this group is the identity. It is an immediate computation that the universal automaton of the identity of a group is the group itself and that (the Cayley graph of) $\mathbb{Z}/2^q\mathbb{Z}$ has a loop complexity equal to q. ■

The proof of Theorem 3 follows indeed the original proof by McNaughton. For any automaton $\mathcal{B}$ that accepts 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 it is not the case, by far. With that idea in mind, one has to consider morphisms of a special kind.

Example 2 (continued): The universal automaton of $F_6 = \{f \mid |f|_a - |f|_b \equiv 1, 3, 4 \text{ or } 5 \bmod 6\}$ is represented at Figure 8. Two of its balls form the automaton shown above and that accepts F_6 with minimal loop complexity. □

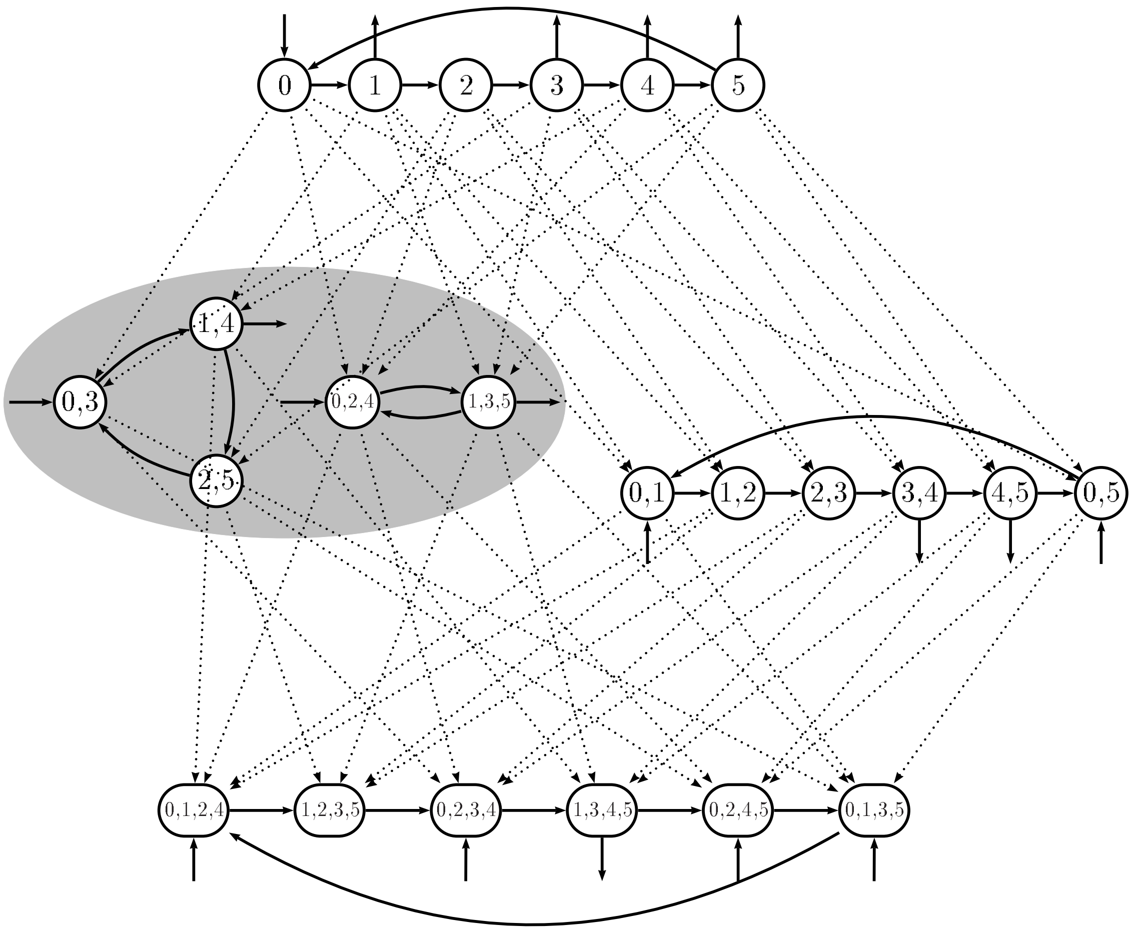


Figure 8: The universal automaton of F_6.

3.1 Conformal morphisms

Definition 3 *A morphism $\varphi\colon \mathcal{B} \to \mathcal{A}$ is said to be* conformal[8] *if any computation in $\mathcal{A}$ is the image of (at least) one computation in $\mathcal{B}$.*

A morphism is not necessarily conformal, as shown by the example of Figure 9.

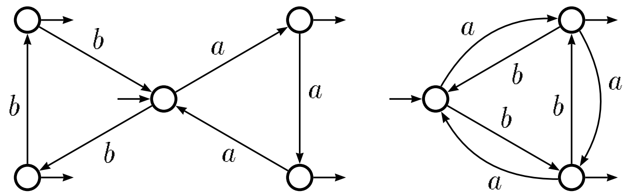


Figure 9: A non conformal morphism (the horizontal is the morphism).

The notion of conformal morphism allows to put into relation the loop complexity of an automaton and the one of its image by a morphism.

Theorem 4 [10] *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 $\mathcal{A}$: $\mathsf{lc}(\mathcal{B}) \geqslant \mathsf{lc}(\mathcal{A})$.* ■

We first state and prove a lemma.

Lemma 4 *Let $\varphi\colon \mathcal{B} \to \mathcal{A}$ be a conformal morphism. For every ball $\mathcal{P}$ of $\mathcal{A}$, there exists a ball $\mathcal{Q}$ of $\mathcal{B}$ such that the restriction of φ to $\mathcal{Q}$ is a conformal morphism from $\mathcal{Q}$ onto $\mathcal{P}$.*

Proof. This lemma (as the theorem) is indeed a statement on graphs and not on automata, that is we can forget the labels on the transitions. But the proof will be simpler if we make use of the notion of automata, that is of labelled graphs — not with the original labels, but with labels that are convenient for the proof. Every transition of $\mathcal{A}$ is considered as having a *distinct label* and every state of $\mathcal{A}$ is considered as both initial and final.

The words of the language accepted by $\mathcal{A}$ (resp. by a subautomaton $\mathcal{P}$ of $\mathcal{A}$) characterize the pathes in the graph $\mathcal{A}$ (resp. in the graph $\mathcal{P}$). The transitions of $\mathcal{B}$ are labelled in such a way that φ is a morphism and every state of $\mathcal{B}$ is both initial and final.

Let $\mathcal{P}$ be a ball of $\mathcal{A}$ and $\mathcal{R} = \mathcal{P}\varphi^{-1}$. Let $n = \|\mathcal{R}\|$ and $m = \|\mathcal{P}\|$ be the numbers of states of $\mathcal{R}$ and of $\mathcal{P}$ respectively. Let w be a circuit (then a word) that contains all the pathes of $\mathcal{P}$ of length smaller than 2^{n+m}. The circuit w^n is a path in $\mathcal{P}$ which is lifted into a path in $\mathcal{R}$ (as φ is conformal). By the pigeon hole principle, there exists a k such that a factor w^k is the label of a circuit in $\mathcal{R}$; let $\mathcal{Q}$ be the ball in $\mathcal{R}$, and then in $\mathcal{B}$,

[8] McNaughton call them "*pathwise*" [10] but his definition of morphism is slightly different from ours.

which contains this circuit. By construction, $\mathcal{Q}$ accepts all the words of length smaller than 2^{n+m} of the language accepted by $\mathcal{P}$, $\mathcal{Q}$ is thus equivalent[9] to $\mathcal{P}$, then all the pathes of $\mathcal{P}$ are lifted in $\mathcal{Q}$: the restriction of φ from $\mathcal{Q}$ onto $\mathcal{P}$ is conformal. ■

Proof of Theorem 4. By contradiction. Among all automata for which the proposition does not hold, let $\mathcal{B}$ be an automaton with minimal loop complexity d, and let c be the loop complexity of $\mathcal{A}$: $c > d$.

If $d = 0$, the length of the pathes in $\mathcal{B}$ is bounded and it is impossible for φ to be conformal, then $d > 0$.

By definition, there exists a ball $\mathcal{P}$ in $\mathcal{A}$ of loop complexity c and, by Lemma 4, a ball $\mathcal{Q}$ of $\mathcal{B}$ whose image by φ is $\mathcal{P}$. This ball is of loop complexity at most d but it is as well, by minimality of d, of loop complexity at least d. There exists then a state q of $\mathcal{Q}$ such that

$$\mathsf{lc}(\mathcal{Q} \setminus \{q\}) = d - 1 \ .$$

Let $p = q\varphi$, $\mathcal{P}' = \mathcal{P} \setminus \{p\}$ and $\mathcal{Q}' = \mathcal{Q} \setminus \{p\varphi^{-1}\}$; it holds $\mathsf{lc}(\mathcal{Q}') \leqslant \mathsf{lc}(\mathcal{Q} \setminus \{q\}) = d - 1$ and $\mathsf{lc}(\mathcal{P}') \geqslant c - 1 > d - 1$.

Any path of $\mathcal{P}'$ is a path of $\mathcal{P}$ which does not go through p; such a path is the image of a path of $\mathcal{Q}$ which does not go through any of the states in $p\varphi^{-1}$, that is, the image of a path of $\mathcal{Q}'$: φ is a conformal morphism from $\mathcal{Q}'$ onto $\mathcal{P}'$, a contradiction with the minimality of d. ■

3.2 Proof of Theorem 3

In the sequel, $K \subset A^*$ is a pure-group language, $\alpha\colon A^* \to G$ is the syntactic morphism, $P = K\alpha$ and $\mathcal{A}_K = \langle G, A, \delta, 1_G, P \rangle$ is the minimal automaton of K. For w in A^* and g in G, we note $g \cdot w$ for $g\,(w\alpha)$, the multiplication being taken in G.

Even in the case of a pure-group language K, the morphism φ from an automaton $\mathcal{B}$ (that accepts K) into the universal automaton $\mathcal{U}_K$ is not necessarily conformal. The proof of the theorem boils down to show that nevertheless φ is conformal on those balls of $\mathcal{B}$ that are crucial for the loop complexity. This goes via two properties of the balls of the universal automaton $\mathcal{U}_K$ of a pure-group language K that we establish first.

Lemma 5 *The balls of $\mathcal{U}_K$ are deterministic and complete.*

□ It follows from the definition of the universal automaton that if

$$(L_1, R_1) \xrightarrow[\mathcal{U}_K]{a} (L_2, R_2)$$

[9] As two automata with n and m states respectively are equivalent if they coincide on all words of length smaller than 2^{n+m}. This is the argument which makes the use of automata instead of graphs powerful.

is a transition of $\mathcal{U}_K$, then $L_1\,(a\alpha)\,R_2 \subseteq P$ and then both $L_1 \cdot a \subseteq L_2$ and $(a\alpha)\,R_2 \subseteq R_1$ hold.

Let (L_1, R_1) and (L_2, R_2) be two states of $\mathcal{U}_K$ in a same ball. There exist u and v in A^* such that $L_1 \cdot u \subseteq L_2$ and $L_2 \cdot v \subseteq L_1$. As G is a group, the action of every element is injective. Then $\|L_1\| \leqslant \|L_2\| \leqslant \|L_1\|$, hence $\|L_1\| = \|L_2\|$ and $L_1 \cdot u = L_2$. Which means that L_2 is uniquely determined by L_1 and u: the ball is deterministic.

On the other hand, if (L, R) is a factorization of P, $(L\,(u\alpha), (u\alpha)^{-1}R)$ is a factorization of P as well, for every u in A^* and there exists a transition labelled by u from the first one onto the second one. For every u, there exists v such that $(u\,v)\alpha = 1_G$, and then a transition labelled by v from $(L\,(u\alpha), (u\alpha)^{-1}R)$ onto (L, R). Then, $(L\,(u\alpha), (u\alpha)^{-1}R)$ belongs to the same ball as (L, R) and this ball is complete. □

Lemma 6 *For every integer k, there exists a word w_k in A^*, whose image in G is 1_G, and such that any computation of lenght k in any ball C of $\mathcal{U}_K$ is a sub-computation of any computation in C labelled by w_k.*

□ Every word whose image in G is 1_G, is the label of a circuit in every ball of $\mathcal{U}_K$ and for every state as starting point. For every ball, and every state of this ball, one can build a circuit that contains all computations of length k in that ball. Let z be the product of the labels of all these circuits. One can choose for w_k a power z^n of z such that its image in G is 1_G. □

Proof of Theorem 3. Let $\mathcal{B}$ be an automaton of minimal loop complexity that accepts K and n the number of states of $\mathcal{B}$. Let φ be a morphism from $\mathcal{B}$ onto $\mathcal{U}_K$.

Let g in P, thus a final state of $\mathcal{A}_K$, and let u_g, be a word of A^* that is mapped onto g by α. For every integer k, the word $(w_k)^n u_g$ is in K and then is accepted by $\mathcal{B}$. The block star lemma, applied to factors w_k, yields a state p_k of $\mathcal{B}$ which is the starting point of a circuit that is labelled by a certain power (w_k^l). In other words, a computation with label $(w_k)^n u_g$ can be factorized as follow:

$$r_k \xrightarrow[\mathcal{B}]{{w_k}^{l'}} p_k \xrightarrow[\mathcal{B}]{{w_k}^{l}} p_k \xrightarrow[\mathcal{B}]{{w_k}^{l''} u_g} s_k$$

Let $\mathcal{D}_k$ be the ball of $\mathcal{B}$ that contains p_k, and thus this circuit. A infinite sequence of balls $\mathcal{D}_k$ is obtained in that way, in which at least one ball $\mathcal{D}$ appears infinitely often.

Let $\mathcal{C}$ be the ball of $\mathcal{U}_K$ which contains the image of $\mathcal{D}$ by φ. For every path c of $\mathcal{C}$, there exist an integer k larger than the length of c, an integer l and a state p of $\mathcal{D}$ such that there exist a circuit of $\mathcal{D}$ of origin p and labelled by $(w_k)^l$. This same word $(w_k)^l$ is the label of a circuit in $\mathcal{C}$ that goes through all computations of length smaller than or equal to k; in particular, it contains c itself. Hence c is the image of a computation in $\mathcal{D}$.

The ball $\mathcal{C}$ is then the image of $\mathcal{D}$ by φ and the restriction of φ to $\mathcal{D}$ is *conformal*. By Theorem 4, $\mathsf{lc}(\mathcal{D}) \geqslant \mathsf{lc}(\mathcal{C})$ holds.

Let (L, R) be the factorization that is the image of p by φ — where p is the state defined above. As $(w_k)^{l'}$ is in $\mathsf{Past}_{\mathcal{B}}(p)$, 1_G is in $\mathsf{Past}_{\mathcal{U}_K}((L, R))$ and then 1_G is in L, that is, (L, R) is an initial state of $\mathcal{U}_K$. In the same way, $(w_k)^{l''} u_g$ is in $\mathsf{Fut}_{\mathcal{B}}(p)$ and g is in R. Every word u of A^* such that $u\alpha = g$ is the label of a computation in $\mathcal{C}$ that starts from (L, R) (initial state) and that ends at the state $(L\,g, g^{-1}R)$, which is a final state of $\mathcal{U}_K$ since $1_G \in g^{-1}R$; hence u is accepted by $\mathcal{C}$. The ball $\mathcal{C}$ is a subautomaton of $\mathcal{U}_K$ that accepts a language which contains $g\alpha^{-1}$ and which is contained in K.

The same construction can be repeated for every g in P and a set $\mathcal{E}$ of balls of $\mathcal{U}_K$ is obtained which accepts the whole language K. Every ball in $\mathcal{E}$ has a loop complexity that is smaller than or equal to the loop complexity of at least one ball of $\mathcal{B}$. The loop complexity of $\mathcal{E}$ is at most equal to the loop complexity of $\mathcal{B}$ which was supposed to be minimal. ■

Acknowledgements

It is a pleasure to thank J. Brzozowski for the fruitful discussions we had during his last visit to Europe.

The second author gratefully thanks D. Wood for his invitation and hospitality in November 1999 at the Hong Kong University for Science and Technology. The friendly athmosphere and the stimulating discussions we had then lead to the definition of the E B-index.

The second author is also grateful to M. Ito who invited him at the 3rd International Conference on Languages, Words and Combinatorics, and especially to H. Jürgensen who gave him the opportunity to present this work at the 2rd workshop on Descriptive Complexity of Automata, Grammars and Related Structures in July 2000, in London (Ontario), with the attendance of R. McNaughton.

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] J. A. Brzozowski, and E. J. McCluskey Signal flow graph techniques for sequential circuit state diagrams. *IEEE Transactions on Electronic Computers* **12** (1963), 67–76.

[3] Carrez, C. On the minimalization of non-deterministic automaton. Tech. rep. du laboratoire de calcul de la Faculté des Sciences de l'Université de Lille, 1970.

[4] Carton, O. Factorisations et morphismes. unpublished manuscript.

[5] J. H. Conway *Regular algebra and finite machines.* Chapman and Hall, 1971.

[6] Courcelle, B., Niwinski, D., and Podelski, A. A geometrical view of the determinization and minimization of finite-state automata. *Math. Systems Theory 24* (1991), 117–146.

[7] F. Dejean and M. P. Schützenberger. On a question of Eggan. *Inform. and Control* **9** (1966), 23–25.

[8] L. C. Eggan Transition graphs and the star-height of regular events. *Michigan Mathematical J.* **10** (1963), 385–397.

[9] Hashiguchi, K. Algorithms for determining relative star height and star height. *Inform. and Computation 78* (1988), 124–169.

[10] R. McNaughton The loop complexity of pure-group events. *Inform. and Control* **11** (1967), 167–176.

[11] J. Sakarovitch *Eléments de théorie des automates*, in preparation.

[12] Salomaa, A. *Jewels of formal language theory.* Computer Science Press, 1981.

[13] Wood, D. *Theory of computation.* Wiley, 1987.