On the Hierarchy of Block Deterministic Languages

Pascal Caron, Ludovic Mignot(✉), and Clément Miklarz

LITIS, Université de Rouen, 76801 Saint-Étienne du Rouvray Cedex, France
{pascal.caron,ludovic.mignot,clement.miklarz1}@univ-rouen.fr

Abstract. A regular language is k-block deterministic if it is specified by a k-block deterministic regular expression. This subclass of regular languages has been introduced by Giammarresi *et al.* as a possible extension of one-unambiguous regular languages defined and characterized by Brüggemann-Klein and Wood. We first show that each k-block deterministic regular language is the alphabetic image of some one-unambiguous regular language. Moreover, we show that the conversion from a minimal DFA of a k-block deterministic regular language to a k-block deterministic automaton not only requires state elimination, and that the proof given by Han and Wood of a proper hierarchy in k-block deterministic languages based on this result is erroneous. Despite these results, we show by giving a parameterized family that there is a proper hierarchy in k-block deterministic regular languages.

1 Introduction

A Document Type Definition (DTD) containing a grammar is used to know whether an XML file fits some specification. These grammars are made of rules whose right-hand part is a restricted regular expression. Brüggemann-Klein and Wood have formalized these regular expressions and have shown that the set of languages specified is strictly included in the set of regular ones. The distinctive aspect of such expressions is the one-to-one correspondence between each letter of the input word and a unique position in them. The resulting Glushkov automaton is deterministic. The languages specified are called one-unambiguous regular languages.

Several extensions of one-unambiguous expressions have been considered:

- k-block deterministic regular expressions [4] are such that while reading an input word, there is a one-to-one correspondence between the next at most k input symbols and the same number of symbols of the expression. These expressions have particular Glushkov automata. The transitions of these automata can be labeled by words of length at most k and for every couple of words labeling two output transitions of a single state, these words are not prefix from each other.
- k-lookahead regular expressions form another generalization. This time, the reading of the next k symbols of the input word allows one to know the next position in the expression. This extension has been proposed in [6].

F. Drewes (Ed.): CIAA 2015, LNCS 9223, pp. 63–75, 2015.
DOI: 10.1007/978-3-319-22360-5_6

– (k,l)-unambiguous regular expressions [3] is another extension of one-unambiguity, where the next k symbols may induce several paths, but with at most one common state.

These three families of expressions fit together as families of languages in the way that a language is k-block deterministic (resp. k-lookahead deterministic, (k,l)-unambiguous) if there exists a k-block deterministic (resp. k-lookahead deterministic, (k,l)-unambiguous) expression to represent it.

Preliminaries are gathered in Sect. 2. In Sect. 3, we recall several results from [4,6] on which we question their truthfulness. Indeed, we show in Sect. 4 that, due to an erroneous statement of Lemma 4, the witness family given as a proof of Theorem 3 is invalid; and present an alternative family, proving the infinite hierarchy of k-block deterministic regular languages w.r.t. k.

2 Preliminaries

2.1 Languages and Automata Basics

Let Σ be a non-empty finite *alphabet*. A *word* w *over* Σ is a finite sequence of symbols from Σ. The *length* of a word w is denoted by $|w|$, and the *empty word* is denoted by ε. The word x is a *prefix* of w if there exists a word u such that $w = xu$. The set of all prefixes of w is denoted by $\mathrm{Pref}(w)$.

Let Σ^* denote the set of all words over Σ. A *language over* Σ is a subset of Σ^*. Let L and L' be two languages over Σ. The following operations are defined:

– *the union*: $L \cup L' = \{w \mid w \in L \vee w \in L'\}$
– *the concatenation*: $L \cdot L' = \{w \cdot w' \mid w \in L \wedge w' \in L'\}$
– *the Kleene star*: $L^* = \bigcup_{k\in\mathbb{N}} L^k$ with $L^0 = \{\varepsilon\}$ and $L^{k+1} = L \cdot L^k$

A *regular expression over* Σ is built from $\emptyset$ (the empty set), ε, and symbols in Σ using the binary operators $+$ and $\cdot$, and the unary operator *. The *language* $\mathrm{L}(E)$ *specified by a regular expression* E is defined as follows:

$$\mathrm{L}(\emptyset) = \emptyset, \qquad \mathrm{L}(\varepsilon) = \{\varepsilon\}, \qquad \mathrm{L}(a) = \{a\},$$
$$\mathrm{L}(F+G) = \mathrm{L}(F) \cup \mathrm{L}(G), \qquad \mathrm{L}(F \cdot G) = \mathrm{L}(F) \cdot \mathrm{L}(G), \qquad \mathrm{L}(F^*) = \mathrm{L}(F)^*,$$

with $a \in \Sigma$, and F, G some regular expressions over Σ. Given a language L, if there exists a regular expression E such that $\mathrm{L}(E) = L$, then L is a *regular language*.

A *finite automaton* A is a 5-tuple $(\Sigma, Q, I, F, \delta)$ where: Q is a finite set of states, $I \subset Q$ is the set of initial states, $F \subset Q$ is the set of final states, and $\delta \subset Q \times \Sigma \times Q$ is a set of transitions. The set δ is equivalent to a function of $Q \times \Sigma \to 2^Q : (p,a,q) \in \delta \iff q \in \delta(p,a)$. This function can be extended to $2^Q \times \Sigma^* \to 2^Q$ as follows: for any subset $Q' \subset Q$, for any symbol $a \in \Sigma$, for any word $w \in \Sigma^*$: $\delta(Q', \varepsilon) = Q'$, $\delta(Q', a) = \bigcup_{q\in Q'} \delta(q,a)$, $\delta(Q', a \cdot w) = \delta(\delta(Q', a), w)$; finally, we set $\delta(q,w) = \delta(\{q\}, w)$. The *language* $\mathrm{L}(A)$ *recognized by* A is the set $\{w \in \Sigma^* \mid \delta(I,w) \cap F \neq \emptyset\}$. Two automata are *equivalent* if they

recognize the same language. The *right language of a state q of A* is denoted by $\mathrm{L}_q(A) = \{w \in \Sigma^* \mid \delta(q, w) \cap F \neq \emptyset\}$. Two states are *equivalent* if they have the same right language.

An automaton $A = (\Sigma, Q, I, F, \delta)$ is *standard* if $|I| = 1$ and $\forall q \in Q, \forall a \in \Sigma, \delta(q, a) \cap I = \emptyset$. If A is not a standard automaton, then it is possible to compute an equivalent standard automaton $(\Sigma, Q_s, I_s, F_s, \delta_s)$ as follows:

- $Q_s = Q \cup \{i_s\}$ with $i_s \notin Q$
- $I_s = \{i_s\}$
- $F_s = F \cup \{i_s\}$ if $I \cap F \neq \emptyset$, F otherwise
- $\delta_s = \delta \cup \{(i_s, a, q) \mid \exists i \in I, (i, a, q) \in \delta\}$

This operation is called *standardization*.

An automaton $A = (\Sigma, Q, I, F, \delta)$ is *deterministic* if $|I| = 1$ and $\forall t_1 = (p, a, q_1),\ t_2 = (p, b, q_2) \in \delta, (t_1 \neq t_2) \implies (a \neq b)$. If A is not deterministic, it is possible to compute an equivalent deterministic automaton by using the powerset construction described in [10].

A deterministic automaton $A = (\Sigma, Q_A, \{i_A\}, F_A, \delta_A)$ is *minimal* if there is no equivalent deterministic automaton $B = (\Sigma, Q_B, \{i_B\}, F_B, \delta_B)$ such that $|Q_B| < |Q_A|$. If A is not minimal, it is possible to compute an equivalent minimal deterministic automaton by merging equivalent states [7,9]. Notice that two equivalent minimal deterministic automata are isomorphic.

Kleene's Theorem [8] asserts that the set of the languages specified by regular expressions is the same as the set of languages recognized by finite automata. The conversion of regular expressions into automata has been deeply studied, *e.g.* by Glushkov [5]. To differentiate each occurence of the same symbol in a regular expression, a *marking* of all the symbols of the alphabet is performed by indexing them with their relative position in the expression. The marking of a regular expression E produces a new regular expression denoted by $E^\sharp$ over the alphabet of indexed symbols denoted by Π_E where each indexed symbol occurs at most once in $E^\sharp$. The reverse of marking is the *dropping* of subscripts, denoted by $^\flat$, such that if $x \in \Pi_E$ and $x = a_k$, then $x^\flat = a$.

Let E be a regular expression over an alphabet Σ. The following functions are defined:

- $\mathrm{Null}(E) = \{\varepsilon\}$ if $\varepsilon \in \mathrm{L}(E)$, $\emptyset$ otherwise
- $\mathrm{First}(E) = \{x \in \Sigma \mid \exists w \in \Sigma^*,\ xw \in \mathrm{L}(E)\}$
- $\mathrm{Last}(E) = \{x \in \Sigma \mid \exists w \in \Sigma^*,\ wx \in \mathrm{L}(E)\}$
- $\mathrm{Follow}(E, x) = \{y \in \Sigma \mid \exists u, v \in \Sigma^*,\ uxyv \in \mathrm{L}(E)\}, \forall x \in \Sigma$

From these functions, an automaton recognizing $\mathrm{L}(E)$ can be computed:

Definition 1. *The* Glushkov automaton of a regular expression E over an alphabet Σ *is denoted by* $G_E = (\Sigma, Q_E, I_E, F_E, \delta_E)$ *with:*

- $Q_E = \Pi_E \cup \{i\}$
- $I_E = \{i\}$
- $F_E = \mathrm{Last}(E^\sharp) \cup \{i\}$ *if* $\mathrm{Null}(E^\sharp) = \{\varepsilon\}$, $\mathrm{Last}(E^\sharp)$ *otherwise*

– $\delta_E = \{(x, a, y) \in \Pi_E \times \Sigma \times \Pi_E \mid y \in \text{Follow}(E^\sharp, x) \wedge a = y^\flat\}$
$\cup \{(i, a, y) \in \{i\} \times \Sigma \times \Pi_E \mid y \in \text{First}(E^\sharp) \wedge a = y^\flat\}$

Finally, an automaton is a *Glushkov automaton* if it is the Glushkov automaton of a regular expression E.

Example 1. Let $E = (a + b)^*a + \varepsilon$. Then $E^\sharp = (a_1 + b_2)^*a_3 + \varepsilon$ with $\Pi_E = \{a_1, b_2, a_3\}$, and G_E is given in Fig. 1.

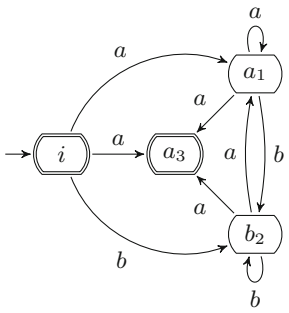

Fig. 1. The Glushkov automaton G_E of $E = (a + b)^*a + \varepsilon$

2.2 One-Unambiguous Regular Languages

We present the notion of one-unambiguity introduced in [1].

Definition 2. *A regular expression E is* one-unambiguous *if G_E is deterministic. A regular language is* one-unambiguous *if it is specified by some one-unambiguous regular expression.*

Brüggemann-Klein and Wood showed that the one-unambiguity of a regular language is stucturally decidable over its minimal DFA. This decision procedure is related to the strongly connected components of the underlying graph and to their links with the remaining parts.

Let $A = (\Sigma, Q, I, F, \delta)$ be a deterministic automaton. A set $O \subset Q$ is called an *orbit* if it is a strongly connected component. An orbit is *trivial* if it consists of only one state and there is no transition from it to itself in A. *The orbit of a state* q, denoted by $\text{O}(q)$ is the orbit to which q belongs. The set of orbits of A is denoted by $\mathcal{O}_A$. Let $O \in \mathcal{O}_A$ be an orbit and $p \in O$ be a state. The state p is a *gate of* O if $(p \in F) \vee (\exists a \in \Sigma, \exists q \in (Q \setminus O), q \in \delta(p, a))$. The set of gates of O is denoted by $\text{G}(O)$. The automaton A has the *orbit property* if all the gates of each orbit have identical connections to the outside. More formally:

Definition 3. *An automaton $A = (\Sigma, Q, I, F, \delta)$ has the orbit property if, for any orbit O in $\mathcal{O}_A$, for any two states (p, q) in $\text{G}(O)$, the two following conditions are satisfied:*

– $p \in F \Longrightarrow q \in F$,
– $\forall r \in (Q \setminus O), \forall a \in \Sigma, r \in \delta(p,a) \Longrightarrow r \in \delta(q,a)$.

Let $q \in Q$ be a state. The *orbit automaton* A_q *of the state* q *in* A is the automaton obtained by restricting the states and the transitions of A to $\mathrm{O}(q)$ with initial state q and final states $\mathrm{G}(\mathrm{O}(q))$. For any state $q \in Q$, the languages $\mathrm{L}(A_q)$ are called the *orbit languages of* A. A symbol $a \in \Sigma$ is *A-consistent* if there exists a state $q_a \in Q$ such that all final states of A have a transition labelled by a to q_a. A set S of symbols is A-consistent if each symbol in S is A-consistent. The *S-cut* A_S of A is constructed from A by removing, for each $a \in S$, all transitions labelled by a that leave a final state of A. All these notions can be used to characterize one-unambiguous regular languages:

Theorem 1 ([1]). *Let M be a minimal deterministic automaton and S be a M-consistent set of symbols. Then, $\mathrm{L}(M)$ is one-unambiguous if and only if:*

1. *the S-cut M_S of M has the orbit property*
2. *all orbit languages of M_S are one-unambiguous.*

Furthermore, if M consists of a single non-trivial orbit and $\mathrm{L}(M)$ is one-unambiguous, M has at least one M-consistent symbol.

This theorem suggests an inductive algorithm to decide, given a minimal deterministic automaton M whether $\mathrm{L}(M)$ is one-unambiguous: the *BKW test*. Furthermore, the theorem defines a sufficient condition over non-minimal deterministic automaton:

Lemma 1 ([1]). *Let A be a deterministic automaton and M be its equivalent minimal deterministic automaton.*

1. *If A has the orbit property, then so does M*
2. *If all orbit languages of A are one-unambiguous, then so are all orbit languages of M.*

Consequently, the BKW test is extended to deterministic automata which are not minimal. Reinterpreting the results in [1], it can be shown that

Lemma 2. *The Glushkov automaton of a one-unambiguous regular expression passes the BKW test.*

2.3 Block Deterministic Regular Languages

We present the notion of block determinism introduced in [4].

Let Σ be an alphabet and k be an integer. The *set of blocks* $B_{\Sigma,k}$ is the set $\{w \mid w \in \Sigma^* \wedge 1 \leq |w| \leq k\}$. The notions of regular expression and automaton can be extended to ones over set of blocks. Let E be a regular expression over Γ and $A = (\Gamma, Q, I, F, \delta)$ be an automaton. Let Σ be an alphabet and k be an integer, if $\Gamma \subset B_{\Sigma,k}$ then E and A are (Σ, k)-*block*. And since $\Gamma \subset B_{\Sigma,k} \subset \Sigma^*$, a

language over Γ is also a language over Σ. To distinguish blocks as syntactic components in a regular expression, we write them between square brackets. Those are omitted for one letter blocks. The notion of determinism can be extended to block-determinism.

Definition 4. *An automaton* $A = (\Gamma, Q, I, F, \delta)$ *is* k-block deterministic *if the following conditions hold:*

- *there exists an alphabet* Σ *such that* A *is* (Σ, k)*-block,*
- $|I| = 1$,
- $\forall t_1 = (p, b_1, q_1), t_2 = (p, b_2, q_2) \in \delta, (t_1 \neq t_2) \Longrightarrow (b_1 \notin \mathrm{Pref}(b_2))$.

Since $\Sigma = B_{\Sigma,1}$, regular expressions and automata can be considered as ones over a set of blocks. Moreover, the blocks can be treated as single symbols, as we do when we refer to the elements of an alphabet. With this assumption, the marking of block regular expressions induces the construction of a Glushkov automaton from a block regular expression, and the usual automaton transformations such as determinization and minimization can be easily performed.

Example 2. Let $E = [aa]^*([ab]b + ba)b^*$. Then $E^\sharp = [aa]_1^*([ab]_2 b_3 + b_4 a_5)b_6^*$, and G_E is given in Fig. 2.

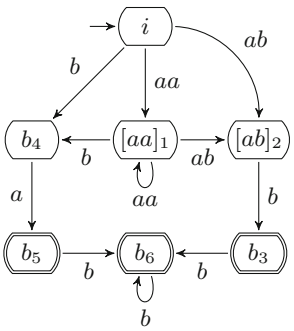

Fig. 2. The $(\Sigma, 2)$-block Glushkov automaton G_E

Finally, the block determinism of a Glushkov automaton can be used to extend the block determinism to block expression:

Definition 5. *A block regular expression* E *is* k*-block deterministic if* G_E *is* k*-block deterministic. A regular language is* k*-block deterministic if it is specified by some* k*-block deterministic regular expressions.*

Example 3. Since the Glushkov automaton in Fig. 2 is 2-block deterministic, $\mathrm{L}([aa]^*([ab]b + ba)b^*)$ is 2-block deterministic.

Let $A = (\Sigma, Q, I, F, \delta)$ be an automaton and Γ be a set. Then the automaton $B = (\Gamma, Q, I, F, \delta')$ is *an alphabetic image of* A if there exists an injection ϕ from Σ to Γ such that $\delta' = \{(p, \phi(a), q) \mid (p, a, q) \in \delta\}$. In this case, we set $B = \phi(A)$. Caron and Ziadi showed in [2] that an automaton is a Glushkov one if and only if the two conditions hold:

- it is homogeneous (for any state q, for any two transitions (p, a, q) and (r, b, q), the symbols a and b are the same);
- it satisfies some structural properties over the transition structure.

One can check that any injection ϕ from Σ to Γ preserves such conditions, since the alphabetical image preserves the transition structure by only changing the symbol labeling a transition. Therefore

Lemma 3. *The alphabetic image of an automaton A is a Glushkov automaton if and only if A is a Glushkov automaton.*

Let us show that the BKW test can be used to characterize the k-block determinism of a regular language:

Theorem 2. *A regular language L is k-block deterministic if and only if it is recognized by a k-block deterministic automaton K such that K is the alphabetic image of a deterministic automaton which passes the BKW test.*

Proof. Let us show the double implication.

1. Let L be a k-block deterministic regular language over Σ. Then there exists a k-block deterministic Glushkov automaton $K = (B_{\Sigma,k}, Q, \{i\}, F, \delta_K)$ that recognizes L. Let $\Pi = \{[b] \mid b \in B_{\Sigma,k}\}$ be an alphabet, $\varphi : \Pi \to B_{\Sigma,k}$ be the bijection such that for every $[b] \in \Pi, \varphi([b]) = b$. Let $A = (\Pi, Q, \{i\}, F, \delta_A)$ be a Glushkov automaton such that $K = \varphi(A)$. Let us suppose that A is not deterministic. Then, there exist two transitions $(p, a, q), (p, a, r) \in \delta_A$ such that $q \neq r$. Thus, $(p, \varphi(a), q), (p, \varphi(a), r) \in \delta_K$, which contradicts the fact that K is k-block deterministic. So, A is a deterministic Glushkov automaton, and therefore passes the BKW test following Lemma 2.
2. Let $A = (\Pi, Q_A, \{i_A\}, F_A, \delta_A)$ be a deterministic automaton which passes the BKW test, $K = \{\Gamma, Q_A, \{i_A\}, F_A, \delta_K)$ be a k-block deterministic automaton, and $\varphi : \Pi \to \Gamma$ be an injection such that $K = \varphi(A)$. Now, $\varphi : \Pi \to \Gamma$ is extended into the morphism $\varphi : \Pi^* \to \Gamma^*$ such that for every letter $a \in \Pi$ and every word $w \in \Pi^*$ we have $\varphi(a \cdot w) = \varphi(a) \cdot \varphi(w)$ and $\varphi(\varepsilon) = \varepsilon$. In this case, $\mathrm{L}(K) = \varphi(\mathrm{L}(A))$. Since A passes the BKW test, there exists an equivalent deterministic Glushkov automaton $G = (\Pi, Q_G, \{i_G\}, F_G, \delta_G)$. Following Lemma 3, there also exists a Glushkov automaton $H = (\Gamma, Q_G, \{i_G\}, F_G, \delta_H)$ such that $H = \varphi(G)$ and $\mathrm{L}(H) = \varphi(\mathrm{L}(G))$. Since A and G are equivalent deterministic automata, $\varphi(\mathrm{L}(G)) = \varphi(\mathrm{L}(A))$. And so $\mathrm{L}(H) = \mathrm{L}(K)$. Let us suppose that H is not k-block deterministic, then there exist two transitions $(p_H, \varphi(a), q_H), (p_H, \varphi(b), r_H) \in \delta_H$ such that either $(\varphi(a) = \varphi(b)) \wedge (q_H \neq r_H)$

or $(\varphi(a) \neq \varphi(b)) \wedge (\varphi(a) \in \mathrm{Pref}(\varphi(b)))$. By definition, $(p_H, a, q_H), (p_H, b, r_H) \in \delta_G$. But since G and A are equivalent deterministic automata, there exist two transitions (p_A, a, q_A), $(p_A, b, r_A) \in \delta_A$, and by definition, $(p_A, \varphi(a), q_A)$, $(p_A, \varphi(b), r_A) \in \delta_K$. Let us suppose that $(\varphi(a) = \varphi(b)) \wedge (q_h \neq r_h)$. Since φ is an injection, $(a = b) \wedge (q_h \neq r_h)$, which contradicts the fact that G is deterministic. So let us suppose that $(\varphi(a) \neq \varphi(b)) \wedge (\varphi(a) \in \mathrm{Pref}(\varphi(b)))$, it contradicts the fact that K is k-block deterministic. Therefore, H is a k-block deterministic Glushkov automaton, and $\mathrm{L}(K)$ is k-block deterministic. □

It has been proved that one-unambiguous regular languages are a proper subfamily of k-block deterministic regular languages. Therefore one can wonder whether there exists an infinite hierarchy in k-block deterministic regular languages regarding k. That has been achieved by Han and Wood [6], but with an invalid assumption.

3 Previous Results on Block-Deterministic Languages

In [4], a method is presented for creating from a block automaton an equivalent block automaton with larger blocks by eliminating states while preserving the right language of every other states.

Let $A = (\Gamma, Q, I, F, \delta)$ be a block automaton. The *state elimination of q in A* creates a new block automaton, denoted by $\mathcal{S}(A, q)$, computed as follows: first, the state q and all transitions going in and out of it are removed; second, for every two transitions (r, u, q) and (q, v, s) in δ, the transition (r, uv, s) is added. This transformation is illustrated in Fig. 3.

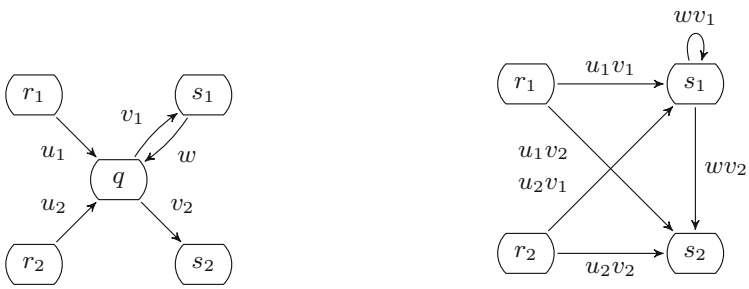

Fig. 3. The state elimination of the state q

Definition 6. *Let $A = (\Gamma, Q, I, F, \delta)$ be a block automaton. A state $q \in Q$ satisfies the* state elimination precondition *if it is neither an initial state nor a final state and it has no self-loops.*

The state elimination is extended to a set $S \subset Q$ of states if every state in S satisfies the state elimination precondition, and the subgraph induced by S is

acyclic. In this case, we can eliminate the states in S in any order. Giammarresi *et al.* [4] suggest that state elimination is sufficient to decide the k-block determinism of a regular language.

Lemma 4 ([4,6]). *Let M be a minimal deterministic automaton of a k-block-deterministic regular language. We can transform M to a k-block deterministic automaton that satisfies the orbit property using state elimination.*

Using this lemma, Han and Wood stated that:

Theorem 3 ([6]). *There is a proper hierarchy in k-block-deterministic regular languages.*

Proof. Han and Wood exhibited the family of languages L_k specified by regular expressions $E_k = ([a^k])^*([a^{k-1}b]b+ba)b^*$ whose minimal deterministic automata M_k are represented in Fig. 4. Following Lemma 4, there is no other choice but to eliminate states q_1 to q_{k-1}, in any order, to have the orbit property. Thus, L_k is k-block deterministic and not $(k-1)$-block deterministic. □

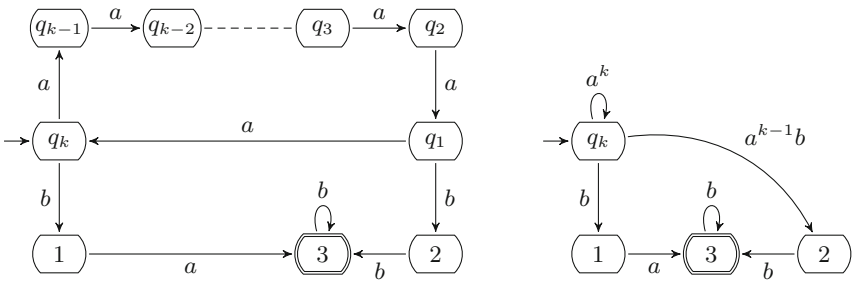

Fig. 4. The minimal deterministic automaton M_k and its equivalent k-block deterministic automaton after having eliminated states q_1 to q_{k-1}

4 A Witness for the Infinite Hierarchy

In this section, we exhibit a counter-example for Lemma 4. We can find a k-block deterministic language with a minimal deterministic automaton from which we cannot get any k-block deterministic automaton that satisfies the orbit property. In Fig. 5, the leftmost automaton is minimal and none of its states can be eliminated. However, by applying standardization, we create an equivalent deterministic automaton from which we can eliminate the state i to get the rightmost equivalent 2-block deterministic automaton.

This clearly shows that the only action of state elimination is not enough to decide whether a language is k-block deterministic. Using this operation, we show that:

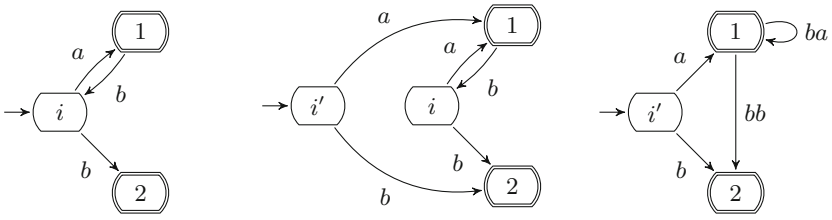

Fig. 5. The counter-example

Proposition 1. *$\forall k \in \mathbb{N} \setminus \{0\}$, the language L_k is 2-block deterministic.*

Proof. As shown in Fig. 6, we can always standardize M_k, proceed to the state elimination of q_k and get a 2-block deterministic automaton which respects the conditions stated in Theorem 2. Thus, L_k is 2-block deterministic and is specified by the regular expressions $F_k = (a^{k-1}([aa]a^{k-2})^*([ab]a + bb) + ba)b^*$. □

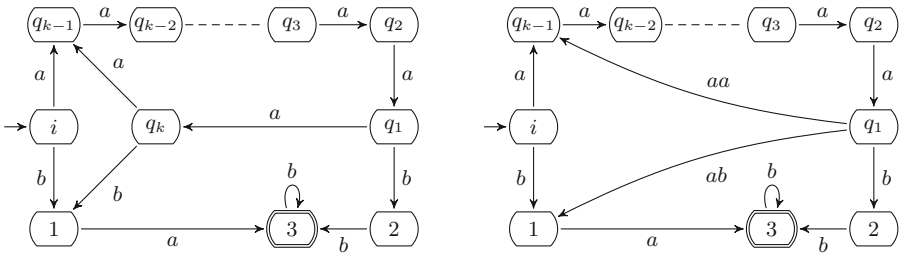

Fig. 6. The standardization of M_k followed by the state elimination of q_k

However, Theorem 3 is still correct since we can give proper details about the proof with our own parameterized family of languages. Let $k \in \mathbb{N} \setminus \{0\}$ be an integer and $A_k = (\Sigma, Q_k, I_k, F_k, \delta_k)$ be the automaton (given in Fig. 7) such that:

- $\Sigma = \{a, b, c\}$
- $Q_k = \{f\} \cup \{\alpha_j, \beta_j \mid 1 \leq j \leq k\}$
- $I_k = \{\beta_k\}$
- $F_k = \{f\} \cup \{\alpha_k, \beta_k\}$
- $\delta_k = \Delta_k \cup \Gamma_k$ with:
  - $\Delta_k = \{(\beta_k, a, \alpha_k), (\beta_1, b, f), (\alpha_k, a, \alpha_k), (\alpha_1, b, f), (\alpha_1, c, \beta_k)\}$
  - $\Gamma_k = \{(\alpha_j, b, \alpha_{j-1}), (\beta_j, b, \beta_{j-1}) \mid 2 \leq j \leq k\}$

First of all, let us notice that the word $b^j \in \mathrm{L}(A_k)$ if and only if $j = k$. Thus, for all $k \neq k'$, $\mathrm{L}(A_k) \neq \mathrm{L}(A_{k'})$. Furthermore,

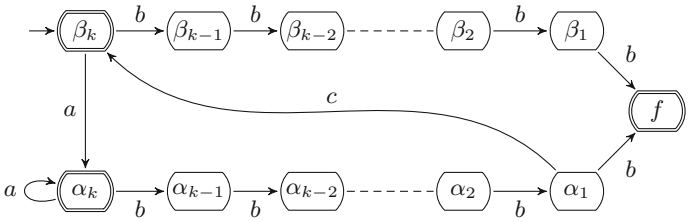

Fig. 7. The k-block deterministic automaton A_k

Proposition 2. *$\forall k \in \mathbb{N} \setminus \{0\}$, $\mathrm{L}(A_k)$ is k-block deterministic.*

Proof. By construction, for all k, A_k is trimmed and deterministic. So, any automaton that we can get from eliminating states such that the state elimination precondition is respected is a block deterministic automaton.

For any integer k in $\mathbb{N} \setminus \{0\}$, we can eliminate the set of states $\{\alpha_j, \beta_j \mid 1 \leq j \leq k-1\}$ because none of these states are initial or final and their induced subgraph is acyclic. Thus, we can get a k-block deterministic automaton B_k, such that $\mathrm{L}(B_k) = \mathrm{L}(A_k)$, shown in Fig. 8. Obviously B_k respects the conditions stated in Theorem 2, so $\mathrm{L}(A_k)$ is k-block deterministic. Furthermore, it can be checked that $\mathrm{L}(A_k)$ is specified by the k-block deterministic regular expression $(a(\varepsilon + [b^{k-1}c]))^*(\varepsilon + [b^k])$. □

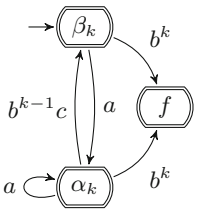

Fig. 8. The k-block deterministic automaton B_k

Finally, let us show that the index cannot be reduced:

Proposition 3. *$\forall k \in \mathbb{N} \setminus \{0\}$, $\mathrm{L}(A_k)$ is not $(k-1)$-block deterministic.*

Proof. Let $B = (B_{\Sigma,k-1}, Q_B, \{i_B\}, F_B, \delta_B)$ be a $(k-1)$-block deterministic automaton equivalent to A_k.

We first show that there exists a non-trivial orbit $O \subset Q_B$ and two states $\alpha, \beta \in O$ such that $\mathrm{L}_\alpha(B) = \mathrm{L}_{\alpha_k}(A_k)$ and $\mathrm{L}_\beta(B) = \mathrm{L}_{\beta_k}(A_k)$. Let us consider the following state sequences: $(\alpha_{k,j})_{j\in\mathbb{N}} \subset F_B$ and $(\beta_{k,j})_{j\in\mathbb{N}} \subset F_B$, such that $\beta_{k,0} = i_B$, $\delta_B(\beta_{k,j}, a) = \alpha_{k,j}$ and $\delta_B(\alpha_{k,j}, b^{k-1}c) = \beta_{k,j+1}$. It follows that $\delta_B(i_B, (ab^{k-1}c)^j) = \beta_{k,j}$ and $\delta_B(i_B, (ab^{k-1}c)^j a) = \alpha_{k,j}$. Notice that the existence of $\alpha_{k,j}$ and $\beta_{k,j}$ is ensured by the fact that $\mathrm{L}(B) = \mathrm{L}(A_k)$. Let us

suppose that there exists $j \in \mathbb{N}$ such that $\mathrm{L}_{\beta_{k,j}}(B) \neq \mathrm{L}_{\beta_k}(A_k)$. Then there exists $w \in \Sigma^*$ such that $w \in \mathrm{L}_{\beta_{k,j}}(B) \triangle \mathrm{L}_{\beta_k}(A_k)$, where for any two sets X and Y, $X \triangle Y = (X \setminus Y) \cup (Y \setminus X)$. And since $\delta_k(\beta_k, (ab^{k-1}c)^j) = \beta_k$, $(ab^{k-1}c)^j \cdot w \in \mathrm{L}(B) \triangle \mathrm{L}(A_k)$. Thus, $\mathrm{L}(B) \neq \mathrm{L}(A_k)$ which is contradictory. So, for every $j \in \mathbb{N}$, we have $\mathrm{L}_{\beta_{k,j}}(B) = \mathrm{L}_{\beta_k}(A_k)$. The proof that for every $j \in \mathbb{N}$, we have $\mathrm{L}_{\alpha_{k,j}}(B) = \mathrm{L}_{\alpha_k}(A_k)$, is done in the same way. Now, let us suppose that for every $j \neq j' \in \mathbb{N}$, we have $\alpha_{k,j} \neq \alpha_{k,j'}$ and $\beta_{k,j} \neq \beta_{k,j'}$. Then Q_B would be infinite, which would contradict the fact that B is a finite automaton. So, there exist $j < j' \in \mathbb{N}$ such that $\alpha_{k,j} = \alpha_{k,j'}$ or $\beta_{k,j} = \beta_{k,j'}$. Thus, either there exists a path going from $\beta_{k,j}$ to $\alpha_{k,j}$ and a path going from $\alpha_{k,j}$ to $\beta_{k,j'} = \beta_{k,j}$, and $\beta_{k,j}$ and $\alpha_{k,j}$ belong to the same orbit; or there exists a path going from $\alpha_{k,j}$ to $\beta_{k,j+1}$ and a path going from $\beta_{k,j+1}$ to $\alpha_{k,j'} = \alpha_{k,j}$, and $\alpha_{k,j}$ and $\beta_{k,j+1}$ belong to the same orbit.

Finally, let us focus on such an orbit O with two gates α and β such that $\mathrm{L}_\alpha(B) = \mathrm{L}_{\alpha_k}(A_k)$ and $\mathrm{L}_\beta(B) = \mathrm{L}_{\beta_k}(A_k)$. We know that for every $i \in \mathbb{N}$ such that $1 \leq i < k$, we have $\delta_k(\beta_k, b^i) = \beta_{k-i}$ with $|\mathrm{L}_{\beta_{k-i}}(A_k)| < \infty$. Since $\mathrm{L}_\beta(B) = \mathrm{L}_{\beta_k}(A_k)$ and B is $(k-1)$-block deterministic, there exist $j \in \mathbb{N}$ and $p \in Q_B$ such that $1 \leq j < k$, $\delta_B(\beta, [b^j]) = p$ and $\mathrm{L}_p(B) = \mathrm{L}_{\beta_{k-j}}(A_k)$. This means that $|\mathrm{L}_p(B)| < \infty$, so $p \notin O$. Now, if there does not exist a state $q \in Q_B$ such that $\delta_B(\alpha, [b^j]) = q$, then B does not have the orbit property. So, let us suppose that such a state exists. We know that for every $i \in \mathbb{N}$ such that $1 \leq i < k$, we have $\delta_k(\alpha_k, b^i) = \alpha_{k-i}$ with $|\mathrm{L}_{\alpha_{k-i}}(A_k)| = \infty$. Since $\mathrm{L}_\alpha(B) = \mathrm{L}_{\alpha_k}(A_k)$, we have $\mathrm{L}_q(B) = \mathrm{L}_{\alpha_{k-j}}(A_k)$ and $|\mathrm{L}_q(B)| = \infty$. So $p \neq q$ and B does not have the orbit property.

Since $\mathrm{L}(A_k)$ cannot be recognized by a $(k-1)$-block deterministic alphabetic image of an automaton passing the BKW test, following Theorem 2 it holds that $\mathrm{L}(A_k)$ is not $(k-1)$-block deterministic. □

References

1. Brüggemann-Klein, A., Wood, D.: One-unambiguous regular languages. Inf. Comput. **140**(2), 229–253 (1998). http://dx.doi.org/10.1006/inco.1997.2688
2. Caron, P., Ziadi, D.: Characterization of Glushkov automata. Theoret. Comput. Sci. **233**(1–2), 75–90 (2000)
3. Caron, P., Flouret, M., Mignot, L.: (k,l)-unambiguity and quasi-deterministic structures: an alternative for the determinization. In: Dediu, A.-H., Martín-Vide, C., Sierra-Rodríguez, J.-L., Truthe, B. (eds.) LATA 2014. LNCS, vol. 8370, pp. 260–272. Springer, Heidelberg (2014)
4. Giammarresi, D., Montalbano, R., Wood, D.: Block-deterministic regular languages. In: Restivo, A., Ronchi Della Rocca, S., Roversi, L. (eds.) ICTCS 2001. LNCS, vol. 2202, pp. 184–196. Springer, Heidelberg (2001)
5. Glushkov, V.M.: The abstract theory of automata. Russ. Math. Surv. **16**, 1–53 (1961)
6. Han, Y.S., Wood, D.: Generalizations of 1-deterministic regular languages. Inf. Comput. **206**(9–10), 1117–1125 (2008)

7. Hopcroft, J.E.: An n log n algorithm for minimizing the states in a finite automaton. In: Kohavi, Z. (ed.) The Theory of Machines and Computations, pp. 189–196. Academic Press, New York (1971)
8. Kleene, S.: Representation of events in nerve nets and finite automata. In: Shannon, C., McCarthy, J. (eds.) Automata Studies, pp. 3–41. Princeton University Press, Princeton (1956). Annals of Mathematics Studies 34
9. Moore, E.F.: Gedanken experiments on sequential machines. In: Shannon, C., McCarthy, J. (eds.) Automata Studies, pp. 129–153. Princeton University Press, Princeton (1956)
10. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. **3**(2), 115–125 (1959)