RAIRO-Theor. Inf. Appl. 48 (2014) 281–305
DOI: 10.1051/ita/2014010

Available online at:
www.rairo-ita.org

A GENERAL FRAMEWORK FOR THE DERIVATION OF REGULAR EXPRESSIONS

Pascal Caron[1], Jean-Marc Champarnaud[1] and Ludovic Mignot[1]

Abstract. The aim of this paper is to design a theoretical framework that allows us to perform the computation of regular expression derivatives through a space of generic structures. Thanks to this formalism, the main properties of regular expression derivation, such as the finiteness of the set of derivatives, need only be stated and proved one time, at the top level. Moreover, it is shown how to construct an alternating automaton associated with the derivation of a regular expression in this general framework. Finally, Brzozowski's derivation and Antimirov's derivation turn out to be a particular case of this general scheme and it is shown how to construct a DFA, a NFA and an AFA for both of these derivations.

Mathematics Subject Classification. 68Q45.

1. Introduction

The (left) quotient of a language L over an alphabet Σ with respect to a word w in Σ^* is the language obtained by stripping the leading w from the words in L that are prefixed by w. The quotient operation plays a fundamental role in language theory and is especially involved in two main issues. First, checking whether a word w belongs to a language L turns out to be equivalent to checking whether the empty word belongs to the quotient of L w.r.t. w. Secondly it was proved by Myhill [17] and Nerode [18] that a language is regular if and only if the set of its quotients w.r.t. all the words in Σ^* is finite. In the case of a regular language L,

Keywords and phrases. Regular expressions, alternating automata, derivation, partial derivation.

[1] LITIS, Université de Rouen, 76801 Saint-Étienne du Rouvray Cedex, France.
{pascal.caron,jean-marc.champarnaud,ludovic.mignot}@univ-rouen.fr

Article published by EDP Sciences

© EDP Sciences 2014

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

the different quotients are the states of the so-called quotient automaton of L that is isomorphic to its minimal deterministic automaton.

Since the equality of two languages amounts to the isomorphism of their minimal deterministic automata, the construction of the quotient automaton *via* the computation of the left quotients is intractable. The seminal work of Brzozowski [4] that introduced the notion of a word derivative of a regular expression and the construction of the associated DFA, gave rise to a long series of studies (see for example [1, 3, 7, 9, 13]) that are all based on the simulation of the computation of a language quotient by the one of an expression derivative. In all these research works, the first rule is that if the expression E denotes the language L, then the derivative of E w.r.t. w, for any w, denotes the quotient of L w.r.t. the word w. Thus, checking whether a word w belongs to the language denoted by the expression E is equivalent to checking if the empty word belongs to the language denoted by the derivative of E w.r.t. w. The second rule is that as far as the set D of all the derivatives of E is finite, a finite automaton recognizing the language denoted by E can be constructed, admitting D as a set of states.

Let us notice that Brzozowski derivatives [4] handle unrestricted regular expressions and provide a deterministic automaton; Antimirov derivatives [1] only address simple regular expressions and provide both a deterministic automaton and a non-deterministic one; Antimirov derivatives have been recently extended to regular expressions [7] and this extension provides a deterministic automaton, a non-deterministic one and, as shown in this paper, an alternating automaton. Berry and Sethi continuations [3] are based on the linearization of the (simple) input expression and allow the construction of its Glushkov (non-deterministic) automaton. Champarnaud and Ziadi c-continuations [9] and Ilie and Yu derivatives [13] allow both the construction of the Glushkov automaton and of the Antimirov non-deterministic automaton. Let us mention that derivation has been extended to expressions with multiplicity [8, 15].

As mentioned by Antimirov [1], derivatives of regular expressions have proved to be a productive concept to investigate theoretical topics such as the algebra of regular expressions [11] or of K-regular expressions [14], the systems of language equations [6], the equivalence of simple regular expressions [12] or of regular expressions [2]. More recently, Brzozowski introduced a new approach for finding upper bounds for the state complexity of regular languages, based on the counting of their quotients (or of their derivatives) [5].

Moreover, derivatives provide a useful tool to implement regular matching algorithms: Brzozowski's DFA and Antimirov's NFA turn out to be competitive matching automata [21], compared for instance with Thompson's ε-automaton [22]. The derivative-based techniques are well-suited to functional languages, that are characterized by a good support for symbolic term manipulation. As an example, two derivative-based scanner generators have been recently developed, one for PLT Scheme and one for Standard ML, as reported in [19]. Similarly, Brzozowski's derivatives are used in the implementation of the XML schema language RELAX NG [10]. Finally, let us notice that derivatives can be extended to context-free

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

grammars, seen as recursive regular expressions, yielding a system for parsing context-free grammars [16].

The aim of this paper is to design a general framework where the computation of the set of derivatives of a regular expression, called derivation, is performed over a space of generic structures. Of course Brzozowski's derivation and Antimirov's one appear as particular cases of this general scheme. A first benefit of this formalism is that the properties inherent to the mechanism of derivation, such as the equality between the language denoted by a derivative and the corresponding quotient, the finiteness of the set of derivatives and the way for constructing the associated automata, need only be stated and proved one time, at the top level. A second benefit is that the general framework allows us to design the construction of an AFA from the set of derivatives. As a consequence, we show how to construct a DFA, a NFA and an AFA for any finite derivation, including both Brzozowski's one and Antimirov's one.

The next section is a preliminary section; it gathers classical notions concerning regular languages, regular expressions and finite automata, as well as boolean formulas and alternating automata. The notion of a regular expression derivation *via* a support is defined in Section 3, and the properties of the corresponding derivatives are investigated. Section 4 is devoted to the construction of the alternating automaton associated with the derivation of a regular expression *via* a support. This construction is illustrated in Section 5, where the support is based on the set of clausal forms over the alphabet of the regular expressions.

2. Preliminaries

Let $\mathbb{B} = \{0, 1\}$. A *boolean formula* ϕ over a set X is inductively defined by $\phi = x$ where $x \in X$, or $\phi = \mathrm{f}_{\mathbb{B}}(\phi_1, \ldots, \phi_k)$, where $\mathrm{f}_{\mathbb{B}}$ is the operator associated to the k-ary boolean function f from $\mathbb{B}^k$ to $\mathbb{B}$ and $\phi_1, \ldots, \phi_k$ are boolean formulas over X. The set of the boolean formulas over X is denoted by $\mathrm{BoolForm}(X)$. Let v be a function from X to $\mathbb{B}$. The *evaluation of* ϕ *with respect to* v is the boolean $\mathrm{eval}_{\mathrm{v}}(\phi)$ inductively defined by: $\mathrm{eval}_{\mathrm{v}}(x) = \mathrm{v}(x)$, $\mathrm{eval}_{\mathrm{v}}(\mathrm{f}_{\mathbb{B}}(\phi_1, \ldots, \phi_k)) = \mathrm{f}(\mathrm{eval}_{\mathrm{v}}(\phi_1), \ldots, \mathrm{eval}_{\mathrm{v}}(\phi_k))$. The set $\mathrm{Atom}(\phi)$ is the subset of X inductively defined by $\mathrm{Atom}(x) = \{x\}$ and $\mathrm{Atom}(\mathrm{f}_{\mathbb{B}}(\phi_1, \ldots, \phi_k)) = \bigcup_{1 \leq j \leq k} \mathrm{Atom}(\phi_j)$.

Let Σ be an alphabet, w be a word in Σ^* and L be a language over Σ. The empty word is denoted by ε. Deciding whether w belongs to L is called the *membership problem for the language* L. We denote by $\mathrm{r}_w(L)$ the boolean equal to 1 if $w \in L$, 0 otherwise. Let $L_1, \ldots, L_k$ be k languages. We denote by $\cdot$, * and f_L for any k-ary boolean function f the operators defined as follows: $L_1 \cdot L_2 = \{w_1 \cdot w_2 \in \Sigma^* \mid r_{w_1}(L_1) \wedge \mathrm{r}_{w_2}(L_2) = 1\}$, $L_1^* = \{\varepsilon\} \cup \{w_1 \cdots w_n \in \Sigma^* \mid n \in \mathbb{N} \wedge \forall k \in \{1, \ldots, n\}, \mathrm{r}_{w_k}(L_1) = 1\}$, $\mathrm{f}_L(L_1, \ldots, L_k) = \{w \in \Sigma^* \mid \mathrm{f}(\mathrm{r}_w(L_1), \ldots, \mathrm{r}_w(L_k)) = 1\}$. The *quotient of* a regular language L with respect to a word w, that is defined as the set $w^{-1}(L) = \{w' \in \Sigma^* \mid \mathrm{r}_{ww'}(L) = 1\}$ can be inductively computed as follows: $\varepsilon^{-1}(L) = L$, and for $a \in \Sigma$,

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

$$a^{-1}(L) = \begin{cases} a^{-1}(L_1)\cdot L_2 \cup a^{-1}(L_2) & \text{if} \quad L = L_1\cdot L_2 \wedge \varepsilon \in L_1, \\ a^{-1}(L_1)\cdot L_2 & \text{if} \quad L = L_1\cdot L_2 \wedge \varepsilon \notin L_1, \\ a^{-1}(L_1)\cdot L_1^* & \text{if} \quad L = L_1^*, \\ \mathrm{f}_L(a^{-1}(L_1),\ldots,a^{-1}(L_n)) & \text{if} \quad L = \mathrm{f}_L(L_1,\ldots,L_n), \\ \{\varepsilon\} & \text{if} \quad L = \{a\}, \\ \emptyset & \text{otherwise.} \end{cases}$$

$(a\cdot w')^{-1}(L) = w'^{-1}(a^{-1}(L))$ for $w' \in \Sigma^+$.

An *Alternating Automaton* (**AA**) is a 5-tuple $A = (\Sigma, Q, I, F, \delta)$ where Σ is an *alphabet*, Q is a set of states, I is a boolean formula over Q, F is a function from Q to $\mathbb{B}$ and δ is a function from $Q\times\Sigma$ to BoolForm(Q). The function δ is extended from BoolForm $(Q)\times\Sigma^*$ to BoolForm(Q) as follows:

$$\begin{aligned} \delta(\phi,\varepsilon) &= \phi, \\ \delta(\phi,aw) &= \delta(\delta(\phi,a),w), \\ \delta(\mathrm{f}_{\mathbb{B}}(\phi_1,\ldots,\phi_k),a) &= \mathrm{f}_{\mathbb{B}}(\delta(\phi_1,a),\ldots,\delta(\phi_k,a)), \end{aligned}$$

where a is any symbol in Σ, w is any word in Σ^* and $\phi_1,\ldots,\phi_k$ are any k boolean formulas over Q.

The *accessible part of* A is the AA $(\Sigma, Q', I, F', \delta')$ defined by:

$$\begin{gathered} Q' = \{q\in Q \mid \exists w\in\Sigma^*, q\in \mathrm{Atom}(\delta(I,w))\}; \\ \forall q\in Q',\ F'(q) = F(q); \\ \forall a\in\Sigma, \forall q\in Q', \delta'(q,a) = \delta(q,a). \end{gathered}$$

The *language recognized* by the alternating automaton A is the subset $L(A)$ of Σ^* defined by $L(A) = \{w\in\Sigma^* \mid \mathrm{eval}_F(\delta(I,w)) = 1\}$. Whenever Q is a finite set, A is said to be an *Alternating Finite state Automaton* (**AFA**).

A *regular expression* E over an alphabet Σ is inductively defined by: $E = a$, $E = 1$, $E = 0$, $E = (E_1)\cdot(E_2)$, $E = (E_1)^*$ or $E = \mathrm{f}_{\mathrm{e}}(E_1,\ldots,E_n)$, where $\forall k\in\mathbb{N}$, E_k is a regular expression, $a\in\Sigma$ and f_{e} is the operator associated to the n-ary boolean function f, *e.g.* $+$ is the operator associated to $\vee$. Parenthesis are omitted when there is no ambiguity. In the following, we assume that the regular expression operators as well as the boolean formula operators have no specific algebraic properties, unlike the boolean functions. For instance, the operator $+$ is not associative, not commutative, nor idempotent. A regular expression is said to be *simple* if the only boolean operator used is the sum.

The set of the regular expressions over an alphabet Σ is denoted by Exp(Σ). We denote by $\mathrm{P_F}(\mathrm{Exp}(\Sigma))$ the set of ordered finite subsets of Exp(Σ). By extension we denote by $P_F(X)$ the set of ordered finite subsets of the set X. The *language denoted by* E is the subset $L(E)$ of Σ^* inductively defined as follows: $L(E\cdot F) = L(E)\cdot L(F)$, $L(E^*) = (L(E))^*$, $L(\mathrm{f}_{\mathrm{e}}(E_1,\ldots,E_n)) = \mathrm{f}_L(L(E_1),\ldots,L(E_n))$, $L(a) = \{a\}$, $L(0) = \emptyset$, and $L(1) = \{\varepsilon\}$ with E and F any two regular expressions, a any symbol in Σ and f_L the operator associated with f (*e.g.* $\cup$ is associated to $\vee$). Whenever two expressions E_1 and E_2 denote the same language, E_1 and E_2 are said

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

to be *equivalent* (denoted by $E_1 \sim E_2$). In the following, we denote by $\mathrm{r}_w(E)$ the boolean $\mathrm{r}_w(L(E))$. Notice that the boolean $r_\varepsilon(E)$ is straightforwardly computed as follows:

$$\forall \alpha \in \Sigma \cup \{1,0\},\ r_\varepsilon(\alpha) = \begin{cases} 1 & \text{if } \alpha = 1, \\ 0 & \text{otherwise,} \end{cases}$$

$$r_\varepsilon(\mathrm{f_e}(E_1,\ldots,E_k)) = \mathrm{f}(r_\varepsilon(E_1),\ldots,r_\varepsilon(E_k)),$$

$$r_\varepsilon(E_1 \cdot E_2) = r_\varepsilon(E_1) \wedge r_\varepsilon(E_2),$$

$$r_\varepsilon(E_1^*) = 1.$$

Given a regular expression E over an alphabet Σ and two distinct symbols a, b in Σ, the *Brzozowski derivative* of E w.r.t. a is the expression $\frac{\mathrm{d}}{\mathrm{d}_a}(E)$ inductively defined by:

$$\frac{\mathrm{d}}{\mathrm{d}_a}(a) = 1,\ \frac{\mathrm{d}}{\mathrm{d}_a}(b) = \frac{\mathrm{d}}{\mathrm{d}_a}(1) = \frac{\mathrm{d}}{\mathrm{d}_a}(0) = 0,$$

$$\frac{\mathrm{d}}{\mathrm{d}_a}(\mathrm{f}_e(E_1,\ldots,E_k)) = \mathrm{f}_e\left(\frac{\mathrm{d}}{\mathrm{d}_a}(E_1),\ldots,\frac{\mathrm{d}}{\mathrm{d}_a}(E_k)\right),\ \frac{\mathrm{d}}{\mathrm{d}_a}(E_1^*) = \frac{\mathrm{d}}{\mathrm{d}_a}(E_1)\cdot E_1^*,$$

$$\frac{\mathrm{d}}{\mathrm{d}_a}(E_1 \cdot E_2) = \begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}_a}(E_1)\cdot E_2 + \dfrac{\mathrm{d}}{\mathrm{d}_a}(E_2) & \text{if } \mathrm{r}_\varepsilon(E_1) = 1, \\ \dfrac{\mathrm{d}}{\mathrm{d}_a}(E_1)\cdot E_2 & \text{otherwise,} \end{cases}$$

where b is any symbol in $\Sigma \setminus \{a\}$, $E_1,\ldots,E_k$ are any k regular expressions over Σ, and f_e is the operator associated with the k-ary boolean function f. Notice that Brzozowski defines the *dissimilar derivative* of E as the expression $\frac{\mathrm{d}'}{\mathrm{d}'_a}(E)$ inductively computed by substituting the operator $+_{ACI}$ to the $+$ operator in the derivative formulas, where $+_{ACI}$ is the associative, commutative and idempotent version of $+$.

Given a simple regular expression E over an alphabet Σ and two distinct symbols a, b in Σ, the *Antimirov partial derivative* of E w.r.t. a is the set of expression $\frac{\partial}{\partial_a}(E)$ inductively defined by:

$$\frac{\partial}{\partial_a}(a) = \{1\},\ \frac{\partial}{\partial_a}(b) = \frac{\partial}{\partial_a}(1) = \frac{\partial}{\partial_a}(0) = \emptyset,$$

$$\frac{\partial}{\partial_a}(E_1 + E_2) = \frac{\partial}{\partial_a}(E_1) \cup \frac{\partial}{\partial_a}(E_2),\ \frac{\partial}{\partial_a}(E_1^*) = \frac{\partial}{\partial_a}(E_1)\cdot E_1^*,$$

$$\frac{\partial}{\partial_a}(E_1 \cdot E_2) = \begin{cases} \dfrac{\partial}{\partial_a}(E_1)\cdot E_2 \cup \dfrac{\partial}{\partial_a}(E_2) & \text{if } \mathrm{r}_\varepsilon(E_1) = 1, \\ \dfrac{\partial}{\partial_a}(E_1)\cdot E_2 & \text{otherwise,} \end{cases}$$

where b is any symbol in $\Sigma \setminus \{a\}$, E_1 and E_2 are any two regular expressions over Σ, and for any finite set $\mathcal{E}$ of regular expression, for any regular expression F, $\mathcal{E}\cdot F = \bigcup_{E\in\mathcal{E}}\{E\cdot F\}$. See [20] for a survey of expression to automaton construction using partial derivatives.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

3. Derivation *via* a support

We now introduce the notion of a derivation *via* a support. Recall that a Brzozowski derivative is an expression, whereas an Antimirov derivative is a set of expressions. In our framework, a derivative is more generally an element of an arbitrary set, called the structure set. For example, a structure can be an expression, a set of expressions or a set of set of expressions. A support is essentially made of a structure set equipped with operators, and of a mapping that transforms a structure into an expression.

Definition 3.1. Let Σ be an alphabet. Let $\mathbb{E}$ be a set and h be a mapping from $\mathbb{E}$ to $\mathrm{Exp}(\Sigma)$. Let $\mathcal{O}$ be a set containing:

- for any k-ary boolean function f, an operator $\mathrm{f}_{\mathbb{E}}$ from $\mathbb{E}^k$ to $\mathbb{E}$;
- an operator $\cdot_{\mathbb{E}}$ from $\mathbb{E} \times \mathrm{Exp}(\Sigma)$ to $\mathbb{E}$.

Let $1_{\mathbb{E}}$ and $0_{\mathbb{E}}$ be two elements in $\mathbb{E}$. The 6-tuple $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$ is said to be a *support* if the three following conditions are satisfied:

(1) for any k elements $\mathcal{E}_1, \ldots, \mathcal{E}_k$ in $\mathbb{E}$:
$$\mathrm{h}(\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k)) \sim \mathrm{f}_e(\mathrm{h}(\mathcal{E}_1), \ldots, \mathrm{h}(\mathcal{E}_k));$$
(2) for any element $\mathcal{E}$ in $\mathbb{E}$, for any expression E in $\mathrm{Exp}(\Sigma)$:
$$\mathrm{h}(\mathcal{E} \cdot_{\mathbb{E}} E) \sim \mathrm{h}(\mathcal{E}) \cdot E;$$
(3) $\mathrm{h}(1_{\mathbb{E}}) \sim 1$ and $\mathrm{h}(0_{\mathbb{E}}) \sim 0$.

Notice that the expressions $\mathrm{h}(\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k))$ and $\mathrm{f}_e(\mathrm{h}(\mathcal{E}_1), \ldots, \mathrm{h}(\mathcal{E}_k))$ need not to be identical. They are only required to define the same language. A support is based on a set of generic structures that can be used to handle regular expressions. We now define the notion of regular expression derivation *via* a support.

Definition 3.2. Let $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$ be a support. The *derivation via* $\mathbb{S}$ is the mapping D from $\Sigma^+ \times \mathrm{Exp}(\Sigma)$ to $\mathbb{E}$ inductively defined for any $a \in \Sigma$, for any word u in Σ^+ and for any expression E in $\mathrm{Exp}(\Sigma)$ by:

$$\mathrm{D}(a, E) = \begin{cases} \mathrm{f}_{\mathbb{E}}(\mathrm{D}(a, E_1), \ldots, \mathrm{D}(a, E_k)) & E = \mathrm{f}_e(E_1, \ldots, E_k), \\ (\mathrm{D}(a, E_1) \cdot_{\mathbb{E}} E_2) \vee_{\mathbb{E}} \mathrm{D}(a, E_2) & E = E_1 \cdot E_2 \ \wedge \ \varepsilon \in L(E_1), \\ \mathrm{D}(a, E_1) \cdot_{\mathbb{E}} E_2 & E = E_1 \cdot E_2 \ \wedge \ \varepsilon \notin L(E_1), \\ \mathrm{D}(a, E_1) \cdot_{\mathbb{E}} E_1^* & E = E_1^*, \\ 1_{\mathbb{E}} & E = a, \\ 0_{\mathbb{E}} & E \in \Sigma \setminus \{a\} \cup \{1, 0\}. \end{cases}$$
$$\mathrm{D}(au, E) = \mathrm{D}(u, \mathrm{h}(\mathrm{D}(a, E))).$$

Lemma 3.3. *Let* D *be the derivation* via *a support* $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$. *Let* a *be any symbol in* Σ *and* u *a word in* Σ^+. *Then:*
$$\mathrm{D}(ua, E) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E))).$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Proof. By recurrence over the length of u.

- If $u = b$ with $b \in \Sigma$ the by definition, $\mathrm{D}(ba, E) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(b, E)))$.
- If $u = bv$ then $\mathrm{D}(bva, E) = \mathrm{D}(va, \mathrm{h}(\mathrm{D}(b, E)))$. According to the recurrence hypothesis, $\mathrm{D}(va, \mathrm{h}(\mathrm{D}(b, E))) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(v, \mathrm{h}(\mathrm{D}(b, E)))))$. By definition, $\mathrm{D}(v, \mathrm{h}(\mathrm{D}(b, E))) = \mathrm{D}(bv, E)$. Finally it holds that $\mathrm{D}(bva, E) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(bv, E)))$ □

As a direct consequence of this lemma, we will use, as a reference of $\mathrm{D}(w, E)$ both Definition 3.2 or its direct lemma (Lem. 3.3).

Furthermore, if for all expression E in $\mathrm{Exp}(\Sigma)$, the set $\{D(w, E) \mid w \in \Sigma^+\}$ is finite, the derivation D is said to be a *finite derivation*.

3.1. Classical derivations are derivations *via* a support

This subsection illustrates the fact that both Antimirov's derivation and Brzozowski's one are derivations *via* a support.

Definition 3.4. We denote by $\mathbb{S}_A = (\Sigma, \mathbb{E} = \mathrm{P_F}(\mathrm{Exp}(\Sigma)), \mathrm{h}_A, \mathcal{O}_A, \{1\}, \emptyset)$ the 6-tuple defined by:

- for any $\mathcal{E} \in \mathbb{E}$, $\mathrm{h}_A(\mathcal{E}) = \sum_{E \in \mathcal{E}} E$,
- $\mathcal{O}_A = \{\mathrm{f}_{\mathbb{E}} \mid \mathrm{f}$ is a k-ary boolean function$\} \cup \{\cdot_{\mathbb{E}}\}$ where for any elements $\mathcal{E}_1, \ldots, \mathcal{E}_k$ in $\mathbb{E}$,
  - $\mathcal{E} \cdot_{\mathbb{E}} F = \bigcup_{E \in \mathcal{E}} \{E \cdot F\}$,
  - $\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k) = \mathcal{E}_1 \cup \mathcal{E}_2$ if $k = 2$ and $\mathrm{f} = \vee$,
  - $\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k) = \{\mathrm{f}_e(\mathrm{h}_A(\mathcal{E}_1), \ldots, \mathrm{h}_A(\mathcal{E}_k))\}$ otherwise.

Proposition 3.5. *The* 6*-tuple* $\mathbb{S}_A$ *is a support. Furthermore, for any simple regular expression* E *over* Σ*, for any symbol* a*, it holds* $\mathrm{D}_A(a, E) = \frac{\partial}{\partial_a}(E)$*, where* D_A *is the derivation* via $\mathbb{S}_A$.

Proof. Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be two sets of simple regular expressions and E be a simple regular expression. The condition **(1)** of Definition 3.1 is satisfied since: p

(a) if $k = 2$ and $\mathrm{f} = \vee$,

$$\begin{aligned} L\left(\mathrm{h_A}\left(\mathcal{E}_1 \cup \mathcal{E}_2\right)\right) &= L\left(\textstyle\sum_{E \in \mathcal{E}_1 \cup \mathcal{E}_2} E\right) \\ &= \textstyle\bigcup_{E \in \mathcal{E}_1 \cup \mathcal{E}_2} L(E) \\ &= \textstyle\bigcup_{E_1 \in \mathcal{E}_1} L(E_1) \cup \bigcup_{E_2 \in \mathcal{E}_2} L(E_2) \\ &= L\left(\textstyle\sum_{E_1 \in \mathcal{E}_1} E_1 + \sum_{E_2 \in \mathcal{E}_2} E_2\right) \\ &= L\left(\mathrm{h_A}(\mathcal{E}_1) + \mathrm{h_A}(\mathcal{E}_2)\right) \end{aligned}$$

(b) and otherwise,

$$\begin{aligned} L(\mathrm{h_A}(\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k)) &= L\left(\mathrm{h_A}\left(\{\mathrm{f}_e(\mathrm{h_A}(\mathcal{E}_1), \ldots, \mathrm{h_A}(\mathcal{E}_k))\}\right)\right) \\ &= L\left(\mathrm{f}_e(\mathrm{h_A}(\mathcal{E}_1), \ldots, \mathrm{h_A}(\mathcal{E}_k))\right) \end{aligned}$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

The condition **(2)** of Definition 3.1 is satisfied since:

$$\begin{aligned} L(\mathrm{h}_A(\mathcal{E}_1 \cdot_{\mathbb{E}} E)) &= L(\mathrm{h}_A(\textstyle\bigcup_{E_1\in\mathcal{E}_1}\{E_1 \cdot E\})) \\ &= L(\textstyle\sum_{E_1\in\mathcal{E}_1} E_1 \cdot E) \\ &= L(\textstyle\sum_{E_1\in\mathcal{E}_1} E_1) \cdot L(E) \\ &= L(\mathrm{h}_A(\mathcal{E}_1)) \cdot L(E). \end{aligned}$$

The condition **(3)** of Definition 3.1 is satisfied since:

$$L(\mathrm{h}_A(\{1\})) = L(1)$$

$$L(\mathrm{h}_A(\emptyset)) = \emptyset = L(0).$$

Consequently, $\mathbb{S}_A$ is a support. Moreover, since the operators $\cup$ and $\cdot_{\mathbb{E}}$ in $\mathcal{O}_A$ are the operations used in partial derivation, it can be shown by induction that for any simple regular expression E over Σ and for any symbol a, $\mathrm{D}_A(a, E) = \frac{\partial}{\partial_a}(E)$. □

Definition 3.6. We denote by $\mathbb{S}_B = (\Sigma, \mathbb{E}' = \mathrm{Exp}(\Sigma), \mathrm{h}_B, \mathcal{O}_B, 1, 0)$ the 6-tuple defined by:

- for any $E \in \mathrm{Exp}(\Sigma)$, $\mathrm{h}_B(E) = E$;
- $\mathcal{O}_B = \{\mathrm{f}_{\mathbb{E}'} \mid \mathrm{f} \text{ is a } k\text{-ary boolean function}\} \cup \{\cdot_{\mathbb{E}'}\}$, where for any $E_1, \ldots, E_k$ elements in $\mathrm{Exp}(\Sigma)$;
  - $E \cdot_{\mathbb{E}'} F = E \cdot F$;
  - $\mathrm{f}_{\mathbb{E}'}(E_1, \ldots, E_k) = E_1 + E_2$ if $k = 2$ and $\mathrm{f} = \vee$;
  - $\mathrm{f}_{\mathbb{E}'}(E_1, \ldots, E_k) = \mathrm{f}_e(E_1, \ldots, E_k)$ otherwise.

Proposition 3.7. *The* 6*-tuple* $\mathbb{S}_B$ *is a support. Furthermore, for any regular expression* E *over* Σ*, for any symbol* a*, it holds* $\mathrm{D}_B(a, E) = \frac{\mathrm{d}}{\mathrm{d}_a}(E)$*, where* D_B *is the derivation* via $\mathbb{S}_B$*.*

Proof. Since h_B is the identity and $\mathbb{E}' = \mathrm{Exp}(\Sigma)$, it is obvious that $\mathbb{S}_B$ is a support. Moreover, since the operators in $\mathcal{O}_B$ are the operators of regular expressions, it can be shown by induction that for any regular expression E over Σ and for any symbol a, $\mathrm{D}_B(a, E) = \frac{\mathrm{d}}{\mathrm{d}_a}(E)$. □

Let us notice that, by definition, the derivation D_A more generally addresses unrestricted expressions ; therefore it provides a natural extension for Antimirov derivation. See [7] and Section 5 for alternative extensions.

3.2. Main properties of supports

We now show that the language denoted by the expression associated with any derivative $D(w, E)$ is equal to the corresponding quotient.

Proposition 3.8. *Let* D *be the derivation* via *a support* $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$*. Then for any word* w *in* Σ^+*, for any expression* E *in* $\mathrm{Exp}(\Sigma)$*, it holds:*

$$L(\mathrm{h}(\mathrm{D}(w, E))) = w^{-1}(L(E)).$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Proof. By recurrence over the length of w.

(1) Let $w = a \in \Sigma$. By induction over the structure of E.

$$L(\mathrm{h}(\mathrm{D}(a,a))) = L(\mathrm{h}(1_{\mathbb{E}})) = \{\varepsilon\} = a^{-1}(L(a))$$

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(a,b))) &= L(\mathrm{h}(\mathrm{D}(a,1))) = L(\mathrm{h}(\mathrm{D}(a,0))) \\ &= L(\mathrm{h}(0_{\mathbb{E}})) = \emptyset = a^{-1}(L(b)) = a^{-1}(L(1)) = a^{-1}(L(0)) \end{aligned}$$

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(a,E_1^*))) &= L(\mathrm{h}(\mathrm{D}(a,E_1) \cdot_{\mathbb{E}} E_1^*)) \\ &= L(\mathrm{h}(\mathrm{D}(a,E_1))) \cdot L(E_1^*) = a^{-1}(L(E_1)) \cdot L(E_1^*) = a^{-1}(L(E_1^*)) \end{aligned}$$

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(a,\mathrm{f}_e(E_1,\ldots,E_k)))) &= L(\mathrm{h}(\mathrm{f}_{\mathbb{E}}(\mathrm{D}(a,E_1),\ldots,\mathrm{D}(a,E_k)))) \\ &= \mathrm{f}_L(L(\mathrm{h}(\mathrm{D}(a,E_1))),\ldots,L(\mathrm{h}(\mathrm{D}(a,E_k)))) \\ &= \mathrm{f}_L(a^{-1}(L(E_1)),\ldots,a^{-1}(L(E_k))) \\ &= a^{-1}(\mathrm{f}_L(L(E_1),\ldots,L(E_k))) = a^{-1}(L(\mathrm{f}_e(E_1,\ldots,E_k))). \end{aligned}$$

Let us consider that $\varepsilon \in L(E_1)$:

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(a,E_1 \cdot E_2))) &= L(\mathrm{h}((\mathrm{D}(a,E_1) \cdot_{\mathbb{E}} E_2) \vee_{\mathbb{E}} \mathrm{D}(a,E_2))) \\ &= L(\mathrm{h}(\mathrm{D}(a,E_1) \cdot_{\mathbb{E}} E_2)) \cup L(\mathrm{h}(\mathrm{D}(a,E_2))) \\ &= L(\mathrm{h}(\mathrm{D}(a,E_1))) \cdot L(E_2) \cup L(\mathrm{h}(\mathrm{D}(a,E_2))) \\ &= a^{-1}(L(E_1)) \cdot L(E_2) \cup a^{-1}(L(E_2)) \\ &= a^{-1}(L(E_1) \cdot L(E_2)) = a^{-1}(L(E_1 \cdot E_2)). \end{aligned}$$

Let us consider that $\varepsilon \notin L(E_1)$:

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(a,E_1 \cdot E_2))) &= L(\mathrm{h}(\mathrm{D}(a,E_1) \cdot_{\mathbb{E}} E_2)) = \\ L(\mathrm{h}(\mathrm{D}(a,E_1))) \cdot L(E_2) &= a^{-1}(L(E_1)) \cdot L(E_2) = \\ a^{-1}(L(E_1) \cdot L(E_2)) &= a^{-1}(L(E_1 \cdot E_2)). \end{aligned}$$

(2) Let $w = au$ with $a \in \Sigma$ and $u \in \Sigma^+$. According to the recurrence hypothesis,

$$\begin{aligned} L(\mathrm{h}(\mathrm{D}(w,E))) &= L(\mathrm{h}(\mathrm{D}(u,\mathrm{h}(\mathrm{D}(a,E))))) = \\ &= u^{-1}(L(\mathrm{h}(\mathrm{D}(a,E)))) = u^{-1}(a^{-1}(L(E))) = (au)^{-1}(L(E)). \quad \square \end{aligned}$$

From Proposition 3.8 we deduce that $r_\varepsilon(h(D(w,E))) = r_\varepsilon(w^{-1}L(E))$. This property does not depend whether the derivation is finite or not and since the boolean $r_\varepsilon(E)$ can be inductively computed for any regular expression E, any support defines a syntactical solution of the membership problem of the language $L(E)$.

Corollary 3.9. *For a given regular expression E, any derivation* via *a support can be used to solve the membership problem for $L(E)$.*

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

As an example, the support $\mathbb{S}_B$ of Definition 3.6 can be used to solve the membership test, even if the associated derivation is not finite.

Given an expression, the finiteness of the set of its derivatives is a necessary condition for the construction of an associated finite automaton. It is well-known that the set of Brzozowski's derivatives is not necessarily finite whereas the set of dissimilar derivatives and the set of Antimirov's derived terms are finite sets. We now give two sufficient conditions of finiteness in the general case. The first one has already been stated in the case of Brzozowski derivatives [4]. The second one is related to the mapping h of the support, that needs to satisfy specific properties.

Proposition 3.10. *Let* D *be the derivation* via *a support* $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$. *The following set of conditions is sufficient for the mapping* D *to be a finite derivation:*

(1) $\vee_{\mathbb{E}}$ *is associative, commutative and idempotent (***H1***);*
(2) *for any k-ary boolean function* f, *for any k elements* $\mathcal{E}_1, \ldots, \mathcal{E}_k$ *in* $\mathbb{E}$,
$$\mathrm{D}(a, \mathrm{h}(\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k))) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}(a, \mathrm{h}(\mathcal{E}_1)), \ldots, \mathrm{D}(a, \mathrm{h}(\mathcal{E}_k))) \text{ (}\mathbf{H2}\text{).}$$

Proof. Let us show by induction over the structure of regular expressions that for any expression E in $\mathrm{Exp}(\Sigma)$, the set $\{\mathrm{D}(w, E) \mid w \in \Sigma^+\}$ is finite.

If $\mathbf{E} \in \mathbf{\Sigma} \cup \{\mathbf{1}, \mathbf{0}\}$: According to the definition of derivation, the proposition holds.

If $\mathbf{E} = \mathbf{E_1^*}$: Let w be a word in Σ^+. Let us show by recurrence over the length of w that $\mathrm{D}(w, E_1^*)$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_1) \cdot_{\mathbb{E}} E_1^* \mid w' \neq \varepsilon \text{ is a suffix of } w\}$.

(1) Let $w = a \in \Sigma$. Since $\mathrm{D}(a, E_1^*) = \mathrm{D}(a, E_1) \cdot_{\mathbb{E}} E_1^*$, the property holds.
(2) Let $w = ua$ with $a \in \Sigma$ and $u \in \Sigma^+$. By definition, $\mathrm{D}(ua, E_1^*) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1^*)))$. By the recurrence hypothesis, $\mathrm{D}(u, E_1^*)$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_1) \cdot_{\mathbb{E}} E_1^* \mid w' \neq \varepsilon \text{ is a suffix of } u\}$. According to hypothesis **H2**, $\mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1^*)))$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(a, \mathrm{h}(\mathrm{D}(w', E_1))) \cdot_{\mathbb{E}} E_1^* \mid w' \neq \varepsilon \text{ is a suffix of } u\} \cup \{\mathrm{D}(a, E_1) \cdot_{\mathbb{E}} E_1^*\}$. So, $\mathrm{D}(ua, E_1^*)$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_1) \cdot_{\mathbb{E}} E_1^* \mid w' \neq \varepsilon \text{ is a suffix of } ua\}$.

As a consequence, since the set $\{\mathrm{D}(w, E_1) \mid w \in \Sigma^+\}$ is a finite set by induction hypothesis, since $\mathrm{Card}(\bigcup_{w \in \Sigma^+} \{\mathrm{D}(w', E_1) \mid w' \neq \varepsilon \text{ is a suffix of } w\}) = \mathrm{Card}(\bigcup_{w \in \Sigma^+} \{\mathrm{D}(w, E_1)\})$ and since $\vee_{\mathbb{E}}$ is associative, commutative and idempotent, we get:
$$\mathrm{Card}(\{\mathrm{D}(w, E_1^*) \mid w \in \Sigma^+\}) \leq 2^{\mathrm{Card}(\{\mathrm{D}(w, E_1) \mid w \in \Sigma^+\})}.$$

If $\mathbf{E} = \mathrm{f_e}(\mathbf{E_1}, \ldots, \mathbf{E_k})$: Let w be a word in Σ^+. Let us show by recurrence over the length of w that $\mathrm{D}(w, \mathrm{f}_e(E_1, \ldots, E_k)) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}(w, E_1), \ldots, \mathrm{D}(w, E_k))$.

(1) Let $w = a \in \Sigma$. According to the definition of D, the property holds.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

(2) Let $w = ua$ with $a \in \Sigma$ and $u \in \Sigma^+$. By definition, $\mathrm{D}(ua, \mathrm{f}_e(E_1, \ldots, E_k)) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, \mathrm{f}_e(E_1, \ldots, E_k))))$. By the recurrence hypothesis, $\mathrm{D}(u, \mathrm{f}_e(E_1, \ldots, E_k)) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}(u, E_1), \ldots, \mathrm{D}(u, E_k))$. According to hypothesis **H2**,

$$\mathrm{D}(a, \mathrm{h}(\mathrm{f}_{\mathbb{E}}(\mathrm{D}(u, E_1), \ldots, \mathrm{D}(u, E_k)))) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1))), \ldots, \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_k)))) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}(ua, E_1), \ldots, \mathrm{D}(ua, E_k)).$$

As a consequence, since for all integer j in $\{1, \ldots, k\}$ the set $\{\mathrm{D}(w, E_j) \mid w \in \Sigma^+\}$ is finite by induction hypothesis, we get:

$$\mathrm{Card}(\{\mathrm{D}(w, \mathrm{f}_e(E_1, \ldots, E_k)) \mid w \in \Sigma^+\}) \leq \mathrm{Card}(\{\mathrm{D}(w, E_1) \mid w \in \Sigma^+\}) \times \cdots \times \mathrm{Card}(\{\mathrm{D}(w, E_k) \mid w \in \Sigma^+\}).$$

If $\mathbf{E} = \mathbf{E_1} \cdot \mathbf{E_2}$: Let w be a word in Σ^+. Let us show by recurrence over the length of w that either $\mathrm{D}(w, E_1 \cdot E_2) = \mathrm{D}(w, E_1) \cdot_{\mathbb{E}} E_2$ or $\mathrm{D}(w, E_1 \cdot E_2) = (\mathrm{D}(w, E_1) \cdot_{\mathbb{E}} E_2) \vee_{\mathbb{E}} \mathcal{E}$ where $\mathcal{E}$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_2) \mid w' \neq \varepsilon$ is a suffix of $w\}$.

(1) Let $w = a \in \Sigma$. According to the definition of D, the property holds.

(2) Let $w = ua$ with $a \in \Sigma$ and $u \in \Sigma^+$.
By definition, $\mathrm{D}(ua, E_1 \cdot E_2) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1 \cdot E_2)))$.
Two cases have to be considered:

(a) $\mathrm{D}(u, E_1 \cdot E_2) = \mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2$. Either $\mathrm{D}(ua, E_1 \cdot E_2) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2))$, or $\mathrm{D}(ua, E_1 \cdot E_2) = (\mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2))) \vee_{\mathbb{E}} \mathrm{D}(a, \mathrm{h}(E_2))$. According to the recurrence hypothesis, both of these cases satisfy the proposition.

(b) $\mathrm{D}(u, E_1 \cdot E_2) = (\mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2) \vee_{\mathbb{E}} \mathcal{E}$ where $\mathcal{E}$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_2) \mid w' \neq \varepsilon$ is a suffix of $u\}$. Either $\mathrm{D}(ua, E_1 \cdot E_2) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2)) \vee_{\mathbb{E}} \mathrm{D}(a, \mathrm{h}(\mathcal{E}))$ or $\mathrm{D}(ua, E_1 \cdot E_2) = \mathrm{D}(a, \mathrm{h}(\mathrm{D}(u, E_1) \cdot_{\mathbb{E}} E_2)) \vee_{\mathbb{E}} \mathrm{D}(a, \mathrm{h}(E_2)) \vee_{\mathbb{E}} \mathrm{D}(a, \mathrm{h}(\mathcal{E}))$. According to hypothesis **H2**, $\mathcal{E}' = \mathrm{D}(a, \mathrm{h}(\mathcal{E}))$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(a, \mathrm{h}(\mathrm{D}(w', E_2))) \mid w' \neq \varepsilon$ is a suffix of $u\}$, set that equals $\{\mathrm{D}(w', E_2) \mid w' \neq \varepsilon$ is a suffix of $ua\}$.

Consequently, $\mathrm{D}(ua, E_1 \cdot E_2) = (\mathrm{D}(ua, E_1) \cdot_{\mathbb{E}} E_2) \vee_{\mathbb{E}} \mathcal{E}$ where $\mathcal{E}$ is a finite $\vee_{\mathbb{E}}$-combination of elements in the set $\{\mathrm{D}(w', E_2) \mid w' \neq \varepsilon$ is a suffix of $ua\}$.

As a consequence, since the sets $\{\mathrm{D}(w, E_1) \mid w \in \Sigma^+\}$ and $\{\mathrm{D}(w, E_2) \mid w \in \Sigma^+\}$ are finite by induction hypothesis and since $\vee_{\mathbb{E}}$ is associative, commutative and idempotent, we get:

$$\mathrm{Card}(\{\mathrm{D}(w, E_1 \cdot E_2) \mid w \in \Sigma^+\}) \leq \mathrm{Card}(\{\mathrm{D}(w, E_1) \mid w \in \Sigma^+\}) \times 2^{\mathrm{Card}(\{\mathrm{D}(w, E_2) \mid w \in \Sigma^+\})}.$$ □

The derivation D_A of Definition 3.4 is an example of finite derivation since $\cup$ is associative, commutative and idempotent, and since for any k-ary boolean function f and for any k elements $\mathcal{E}_1, \ldots, \mathcal{E}_k$ in $\mathrm{P_F}(\mathrm{Exp}(\Sigma))$, we have:

$$\mathrm{D}_A(a, \mathrm{h}_A(\mathrm{f}_{\mathbb{E}}(\mathcal{E}_1, \ldots, \mathcal{E}_k))) = \mathrm{f}_{\mathbb{E}}(\mathrm{D}_A(a, \mathrm{h}_A(\mathcal{E}_1)), \ldots, \mathrm{D}_A(a, \mathrm{h}_A(\mathcal{E}_k))).$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

On the opposite, since + is not an ACI law, Proposition 3.10 does not allow us to conclude for the derivation D_B of Definition 3.6. Brzozowski showed in [4] that it is possible to compute a finite set of dissimilar derivatives using a quotient of the expressions w.r.t. an ACI sum. It can be achieved by considering the support $\mathbb{S}'_B$ defined as follows:

Definition 3.11. We denote by $\mathbb{S}'_B = (\Sigma, \mathbb{E}'' = \mathrm{P_F}(\mathrm{Exp}(\Sigma)), \mathrm{h}_A, \mathcal{O}'_B, \{1\}, \{0\})$ the 6-tuple defined by:

- for any $\mathcal{E} \in \mathbb{E}''$, $\mathrm{h}_A(\mathcal{E}) = \sum_{E\in\mathcal{E}} E$ (Def. 3.4);
- $\mathcal{O}'_B = \{\mathrm{f}_{\mathbb{E}''} \mid \mathrm{f}$ is a k-ary boolean function$\} \cup \{\cdot_{\mathbb{E}''}\}$, where for any $\mathcal{E}_1, \ldots, \mathcal{E}_k$ elements in $\mathbb{E}''$;
  - $\mathcal{E}_1 \cdot_{\mathbb{E}''} F = \left\{\left(\sum_{E\in\mathcal{E}_1} E\right) \cdot F\right\}$;
  - $\mathrm{f}_{\mathbb{E}''}(\mathcal{E}_1, \ldots, \mathcal{E}_k) = \mathcal{E}_1 \cup \mathcal{E}_2$ if $k = 2$ and $\mathrm{f} = \vee$;
  - $\mathrm{f}_{\mathbb{E}''}(\mathcal{E}_1, \ldots, \mathcal{E}_k) = \{\mathrm{f}_e(\mathrm{h}_A(\mathcal{E}_1), \ldots, \mathrm{h}_A(\mathcal{E}_k))\}$ otherwise.

Proposition 3.12. *The* 6*-tuple* $\mathbb{S}'_B$ *is a support. Furthermore, for any regular expression* E *and for any symbol* a*, it holds that* $\mathrm{h}_A(\mathrm{D}'_B(a, E))$ *is the dissimilar derivative of* E *w.r.t.* a*, where* D'_B *is the derivation* via *the support* $\mathbb{S}'_B$*.*

Proof. According to Definition 3.4, Definition 3.11 and Proposition 3.5, the conditions **(1)** and **(3)** of Definition 3.1 are satisfied by $\mathbb{S}'_B$.

The condition **(2)** of Definition 3.1 is satisfied since:

$$\begin{aligned} L(\mathrm{h}_A(\mathcal{E}_1 \cdot_{\mathbb{E}''} E)) &= L(\mathrm{h}_A(\{(\textstyle\sum_{F\in\mathcal{E}_1} F) \cdot E\})) \\ &= L((\textstyle\sum_{F\in\mathcal{E}_1} F) \cdot E) \\ &= L(\textstyle\sum_{F\in\mathcal{E}_1} F) \cdot L(E) \\ &= L(\mathrm{h}_A(\mathcal{E}_1)) \cdot L(E). \end{aligned}$$

Consequently, $\mathbb{S}'_B$ is a support.

Moreover, since the operator $\cup$ is an ACI law and since the catenation product $\cdot_{\mathbb{E}''}$ returns a singleton, it can be shown by induction that for any regular expression E and for any symbol a, it holds that $\mathrm{h}_A(\mathrm{D}'_B(a, E))$ is the dissimilar derivative of E w.r.t. a. □

4. From derivation *via* a support to automata

Computing the set of derivatives of a regular expression E w.r.t. a derivation D is similar to computing the transition function δ of an automaton, where $\delta(E, w) = \mathrm{h}(\mathrm{D}(w, E))$. As far as alternating automata are concerned, the resulting expression $\mathrm{h}(\mathrm{D}(w, E))$ needs to be transformed into a boolean formula. This computation is performed through a *base function* defined as follows.

Definition 4.1. Let Σ be an alphabet. A *base function* B is a mapping from $\mathrm{Exp}(\Sigma)$ to $\mathrm{BoolForm}(\mathrm{Exp}(\Sigma))$ such that for any expression E and for any word w in Σ^*:

$$w \in L(E) \Leftrightarrow \mathrm{eval}_{r_w}(B(E)) = 1.$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Definition 4.2. Let B be a base function and D be the derivation *via* the support $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$. Let E be an expression in $\mathrm{Exp}(\Sigma)$. Let $A = (\Sigma, Q, I, F, \delta)$ be the automaton defined by:

- $Q = \mathrm{Exp}(\Sigma)$;
- $I = E$;
- $\forall q \in Q,\ F(q) = \begin{cases} 1 & \text{if } \varepsilon \in L(q); \\ 0 & \text{otherwise;} \end{cases}$
- $\forall a \in \Sigma, \forall q \in Q, \delta(q, a) = \mathrm{B}(\mathrm{h}(\mathrm{D}(a, q)))$.

The accessible part of A is said to be the (D, B)-*alternating automaton of* E. Notice that there may exist an infinite number of states.

Theorem 4.3. *The* (D, B)*-alternating automaton of a regular expression* E *recognizes* $L(E)$.

Proof. Let $A = (\Sigma, Q, I, F, \delta)$ be the (D, B)-alternating automaton of E. Let D be the derivation *via* the support $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_{\mathbb{E}}, 0_{\mathbb{E}})$. Let w be a word in Σ^+. Let us show by recurrence over the length of w that for any boolean formula $\phi = \mathrm{f}_{\mathbb{B}}(q_1, \ldots, q_k)$ in $\mathrm{FormBool}(Q)$, the following **(P)** proposition holds:

$$\mathrm{eval}_F(\delta(\phi, w)) = \mathrm{eval}_{r_\varepsilon}(\mathrm{f}_{\mathbb{B}}(\mathrm{B}(\mathrm{h}(\mathrm{D}(w, q_1))), \ldots, \mathrm{B}(\mathrm{h}(\mathrm{D}(w, q_k))))).$$

(1) If $w = a \in \Sigma$, by definition of the transition function δ, $\delta(\phi, a) = \mathrm{f}_{\mathbb{B}}(\delta(q_1, a), \ldots, \delta(q_k, a))$. By construction, for any integer j in $\{1, \ldots, k\}$, $\delta(q_j, a) = \mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_j)))$.
Since for any state $q \in Q$, $F(q) = 1 \Leftrightarrow \varepsilon \in L(q)$, the following proposition holds: $\mathrm{eval}_F(\delta(\phi, a)) = \mathrm{eval}_{r_\varepsilon}(\mathrm{f}_{\mathbb{B}}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_1))), \ldots, \mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_k)))))$.

(2) Let $w = au$ with $a \in \Sigma$ and $u \in \Sigma^+$. Then it holds:

$$\begin{aligned}
&\mathrm{eval}_F(\delta(\phi, au)) \\
&= \mathrm{eval}_F(\delta(\delta(\phi, a), u)) && \text{(Definition of } \delta\text{)} \\
&= \mathrm{eval}_F(\delta(\mathrm{f}_{\mathbb{B}}(\delta(q_1, a), \ldots, \delta(q_k, a))), u)) && \text{(Definition of } \delta(\phi, a)\text{)} \\
&= \mathrm{eval}_F(\mathrm{f}_{\mathbb{B}}(\delta(\delta(q_1, a), u), \ldots, \delta(\delta(q_k, a), u))) && \text{(Definition of } \delta\text{)} \\
&= \mathrm{eval}_F(\mathrm{f}_{\mathbb{B}}(\delta(\mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_1))), u), \ldots, \delta(\mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_k))), u))) \\
& && \text{(Construction of } \delta\text{)} \\
&= \mathrm{f}(\mathrm{eval}_F(\delta(\mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_1))), u)), \ldots, \mathrm{eval}_F(\delta(\mathrm{B}(\mathrm{h}(\mathrm{D}(a, q_k))), u))) \\
& && \text{(Definition of eval}_F\text{)} \\
&= \mathrm{f}(\mathrm{eval}_{r_\varepsilon}(\mathrm{B}(\mathrm{h}(\mathrm{D}(u, \mathrm{h}(\mathrm{D}(a, q_1)))))), \\
&\qquad \ldots, \mathrm{eval}_{r_\varepsilon}(\mathrm{B}(\mathrm{h}(\mathrm{D}(u, \mathrm{h}(\mathrm{D}(a, q_k))))))) \\
& && \text{(Induction hypothesis)} \\
&= \mathrm{eval}_{r_\varepsilon}(\mathrm{f}_{\mathbb{B}}(\mathrm{B}(\mathrm{h}(\mathrm{D}(u, \mathrm{h}(\mathrm{D}(a, q_1))))), \ldots, \mathrm{B}(\mathrm{h}(\mathrm{D}(u, \mathrm{h}(\mathrm{D}(a, q_k))))))) \\
& && \text{(Definition of eval}_F\text{)} \\
&= \mathrm{eval}_{r_\varepsilon}(\mathrm{f}_{\mathbb{B}}(\mathrm{B}(\mathrm{h}(\mathrm{D}(au, q_1))), \ldots, \mathrm{B}(\mathrm{h}(\mathrm{D}(au, q_k))))) && \text{(Definition of D)}
\end{aligned}$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Finally, it holds:

$$\begin{aligned} w \in L(E) &\Leftrightarrow \varepsilon \in L(\mathrm{h}(\mathrm{D}(w,E))) && \text{(Prop. 3.8)}\\ &\Leftrightarrow \mathrm{eval}_{r_\varepsilon}(\mathrm{B}((\mathrm{h}(\mathrm{D}(w,E))))) = 1 && \text{(Def. 4.1)}\\ &\Leftrightarrow \mathrm{eval}_F(\delta(E,w)) = 1 && \text{(Proposition (}\mathbf{P}\text{))}\\ &\Leftrightarrow w \in L(A) && \text{(Definition of the language of an } \mathbf{AA}\text{)} \end{aligned}$$

□

Previous definitions and properties address non necessarily finite automata. We now give sufficient conditions for the finiteness of the automaton. The basic idea is that if it is equivalent to derive an expression E or to derive its atoms $\mathrm{Atom}(\mathrm{B}(E))$, then the set of atoms obtained by a finite derivation is finite.

Definition 4.4. Let $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_\mathbb{E}, 0_\mathbb{E})$ be a support, let D be the derivation *via* $\mathbb{S}$ and B be a base function. The couple (D, B) satisfies the *atom-derivability property* if for any expression E in $\mathrm{Exp}(\Sigma)$ and for any symbol a in Σ:

$$\mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a,E)))) = \bigcup_{E' \in \mathrm{Atom}(\mathrm{B}(E))} \mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a,E')))).$$

Theorem 4.5. *Let A be the (D,B)-alternating automaton of a regular expression E. If* D *is finite and if* (D,B) *satisfies the atom-derivability property, then:*

A is an AFA.

Proof. Let D be the derivation *via* the support $\mathbb{S} = (\Sigma, \mathbb{E}, \mathrm{h}, \mathcal{O}, 1_\mathbb{E}, 0_\mathbb{E})$. Let w be a word in Σ^+.

(1) Let us show by recurrence over the length of w that for any expression q in Q, $\mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(w,q)))) = \mathrm{Atom}(\delta(q,w))$.
   (a) If $w = a \in \Sigma$, $\delta(q,a) = \mathrm{B}(\mathrm{h}(\mathrm{D}(a,q)))$. Then, $\mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a,q)))) = \mathrm{Atom}(\delta(q,a))$.
   (b) If $w = ua$ with $a \in \Sigma$ and $u \in \Sigma^+$, by the recurrence hypothesis, $\mathrm{Atom}(\delta(q,u)) = \mathrm{Atom}(\mathrm{B}(\mathrm{D}(u,q)))$.

$$\begin{aligned} &\mathrm{Atom}(\delta(q,ua))\\ &= \mathrm{Atom}(\delta(\delta(q,u),a)) && \text{(Definition of } \delta\text{)}\\ &= \mathrm{Atom}(\delta(\mathrm{f}_\mathbb{B}(q'_1,\ldots,q'_j),a))\\ &\qquad \text{(Definition of } \delta(q,u) \text{ with } \mathrm{Atom}(\delta(q,u)) = \{q'_1,\ldots,q'_j\}\text{)}\\ &= \mathrm{Atom}(\mathrm{f}_\mathbb{B}(\delta(q'_1,a),\ldots,\delta(q'_j,a))) && \text{(Definition of } \delta\text{)}\\ &= \textstyle\bigcup_{q' \in \{q'_1,\ldots,q'_j\}} \mathrm{Atom}(\delta(q',a)) && \text{(Definition of Atom)}\\ &= \textstyle\bigcup_{q' \in \mathrm{Atom}(\delta(q,u))} \mathrm{Atom}(\delta(q',a)) && \text{(Definition of } \{q'_1,\ldots,q'_j\}\text{)}\\ &= \textstyle\bigcup_{q' \in \mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(u,q))))} \mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a,q'))))\\ &\qquad \text{(Induction hypothesis and construction of } \delta\text{)}\\ &= \mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(a,\mathrm{h}(\mathrm{D}(u,q)))))) && \text{(Atom-derivability property)} \end{aligned}$$

(2) As a direct consequence of the previous point, since the set $\{\mathrm{D}(w,E) \mid w \in \Sigma^+\}$ is finite, so are the sets $\{\mathrm{B}(\mathrm{h}(\mathrm{D}(w,E))) \mid w \in \Sigma^+\}$ and $\bigcup_{w\in\Sigma^+} \mathrm{Atom}(\mathrm{B}(\mathrm{h}(\mathrm{D}(w,q))))$. Finally, the set Q, that is equal to $\bigcup_{w\in\Sigma^+} \mathrm{Atom}(\delta(q,w))$, is a finite set. □

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Example 4.6. Let D_A and D'_B be the derivations of Definitions 3.4 and 3.11. Let B_A be the base inductively defined for any expression by $\mathrm{B}_A(E+F) = \mathrm{B}_A(E) \vee_{\mathbb{B}} \mathrm{B}_A(F)$, $\mathrm{B}_A(E) = E$ otherwise. Let B_B the base defined for any expression by $\mathrm{B}_B(E) = E$. It can be shown that any couple in $\{\mathrm{D}_A, \mathrm{D}'_B\} \times \{\mathrm{B}_A, \mathrm{B}_B\}$ satisfies the atom-derivability property.

Furthermore, the $(\mathrm{D}_A, \mathrm{B}_A)$-AFA of E can be straightforwardly transformed into the derived term NFA of E defined by Antimirov in [1]; the $(\mathrm{D}_A, \mathrm{B}_B)$-AFA of E can be transformed into the partial derivative DFA of E defined by Antimirov in [1]; the $(\mathrm{D}'_B, \mathrm{B}_B)$-AFA of E can be transformed into the dissimilar derivative DFA of E defined by Brzozowski in [4]. Notice that the $(\mathrm{D}'_B, \mathrm{B}_A)$-AFA of E can be transformed into a NFA that is different from the NFA of Antimirov.

Finally, the base B_C inductively defined by $\mathrm{B}_C(\mathrm{f}_e(E_1, \ldots, E_k) = \mathrm{f}_{\mathbb{B}}(\mathrm{B}_C(E_1), \ldots, \mathrm{B}_C(E_k))$ for any operator f_e, $\mathrm{B}_C(E) = E$ otherwise, provides an AFA construction both from D_A and D'_B.

5. Derivation *via* the set of clausal forms

In this section, we show that the set of clausal forms over the set of regular expressions and equipped with the right operators is a derivation support and that the associated $\mathrm{D}_{\mathbb{C}}$ derivation is finite. Furthermore we prove that the atom-derivability property is satisfied whenever the $\mathrm{D}_{\mathbb{C}}$ derivation is associated with a base function in the set $\{\mathrm{B}_A, \mathrm{B}_B, \mathrm{B}_C\}$. Finally we illustrate these results by the construction of the $(\mathrm{D}_{\mathbb{C}}, \mathrm{B}_C)$-AFA of the expression $E = ((ab)^*a)\mathrm{XOR}_e((abab)^*a)$.

Let us first recall some definitions about clausal forms and their operators. A *clausal form* over a set X is an element in $\mathcal{C}(X) = P_F(P_F(X \cup \overline{X}))$ where $\overline{X} = \{\overline{x} \mid x \in X\}$. Let $\oplus$ and $\otimes$ be the two mappings from $\mathcal{C}(X) \times \mathcal{C}(X)$ to $\mathcal{C}(X)$ and $\ominus$ be the function from $\mathcal{C}(X)$ to $\mathcal{C}(X)$ defined for any $\mathcal{C}_1, \mathcal{C}_2$ in $\mathcal{C}(X)$ by:

- $\mathcal{C}_1 \oplus \mathcal{C}_2 = \mathcal{C}_1 \cup \mathcal{C}_2$;
- $\mathcal{C}_1 \otimes \mathcal{C}_2 = \bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} \{C_1 \cup C_2\}$;
- $\ominus(\mathcal{C}_1) = \begin{cases} \{\emptyset\} & \text{if } \mathcal{C}_1 = \emptyset, \\ \emptyset & \text{if } \mathcal{C}_1 = \{\emptyset\}, \\ \otimes_{C \in \mathcal{C}_1} \oplus_{c \in C} \{\{\mathrm{n}(c)\}\} & \text{otherwise,} \end{cases}$

  where $n(c) = \begin{cases} c' & \text{if } c = \overline{c'}, \\ \overline{c} & \text{otherwise.} \end{cases}$

It can be shown that for any element $x \in X \cup \overline{X}$, for any clause $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ in $\mathcal{C}(X)$, the following conditions are satisfied:

- $\mathcal{C}_1 \otimes (\mathcal{C}_2 \oplus \mathcal{C}_3) = (\mathcal{C}_1 \otimes \mathcal{C}_2) \oplus (\mathcal{C}_1 \otimes \mathcal{C}_3)$;
- $\ominus \ominus \{\{x\}\} = \{\{x\}\}$;
- $\ominus(\mathcal{C}_1 \oplus \mathcal{C}_2) = \ominus(\mathcal{C}_1) \otimes \ominus(\mathcal{C}_2)$;
- $\ominus(\mathcal{C}_1 \otimes \mathcal{C}_2) = \ominus(\mathcal{C}_1) \oplus \ominus(\mathcal{C}_2)$.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Furthermore, let us notice that there exist clauses $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ such that $\ominus\ominus\mathcal{C}_1 \neq \mathcal{C}_1$ or $\mathcal{C}_1 \oplus (\mathcal{C}_2 \otimes \mathcal{C}_3) \neq (\mathcal{C}_1 \oplus \mathcal{C}_2) \otimes (\mathcal{C}_1 \oplus \mathcal{C}_3)$.

Indeed, for example we have for any distinct $x, y, z \in X$

$$\begin{aligned}\ominus\ominus\{\{x,y\},\{z\}\} &= \ominus\{\{\overline{x},\overline{z}\},\{\overline{y},\overline{z}\}\} = \{\{x,y\},\{x,z\},\{y,z\},\{z\}\}\\ &\neq \{\{x,y\},\{z\}\}.\end{aligned}$$

or

$$\begin{aligned}&\{\{x\}\} \oplus (\{\{y\}\} \otimes \{\{z\}\}) &&= \{\{x\},\{y,z\}\}\\ \neq\ &(\{\{x\}\} \oplus \{\{y\}\}) \otimes (\{\{x\}\} \oplus \{\{z\}\}) &&= \{\{x\},\{y\}\} \otimes \{\{x\},\{z\}\}\\ &&&= \{\{x\},\{x,z\},\{x,y\},\{y,z\}\}.\end{aligned}$$

From now on, we will consider the set $\mathbb{C} = \mathcal{C}(\mathrm{Exp}(\Sigma))$ of the clausal forms over the set of regular expressions. We now explain how a clausal form over the set of regular expressions is transformed into a regular expression. Let us consider the function $\mathrm{h}_{\mathbb{C}}$ defined from $\mathbb{C}$ to $\mathrm{Exp}(\Sigma)$ for any element $\mathcal{C}$ in $\mathbb{C}$ by:

$$\mathrm{h}_{\mathbb{C}}(\mathcal{C}) = \begin{cases} 0 & \text{if } \mathcal{C} = \emptyset,\\ \sum_{C\in\mathcal{C}} \begin{cases} \neg_e 0 & \text{if } C = \emptyset,\\ \bigwedge_{e\, E\in C} c(E) & \text{otherwise,}\end{cases} & \text{otherwise.}\end{cases}$$

$$\text{where } c(E) = \begin{cases} \neg_e(E') & \text{if } E = \overline{E'},\\ E & \text{otherwise.}\end{cases}$$

Let us now give the definition of the support operators. For any k-ary boolean function f, let $\mathrm{f}_{\mathbb{C}}$ be the operator from $(\mathbb{C})^k$ to $\mathbb{C}$ associated with f defined by:

- $\mathrm{f}_{\mathbb{C}}(\mathcal{C}_1, \ldots, \mathcal{C}_k) = \oplus_{b=(b_1,\ldots,b_k)\in\mathbb{B}^k | \mathrm{f}(b)=1} \otimes_{1\leq j\leq k} \mathrm{g}(b_j, \mathcal{C}_j)$;
- where $\mathrm{g}(b_j, \mathcal{C}_j) = \begin{cases} \mathcal{C}_j & \text{if } b_j = 1;\\ \ominus\mathcal{C}_j & \text{otherwise.}\end{cases}$

The operator $\cdot_{\mathbb{C}}$ from $\mathbb{C} \times \mathrm{Exp}(\Sigma)$ to $\mathbb{C}$ is defined, for any clause $\mathcal{C}$ in $\mathbb{C}$ and for any expression F in $\mathrm{Exp}(\Sigma)$, by:

$$\mathcal{C} \cdot_{\mathbb{C}} F = \bigcup_{C\in\mathcal{C}} \{\mathrm{h}_{\mathbb{C}}(\{C\}) \cdot F\}.$$

Finally, we consider the operation set $\mathcal{O}_{\mathbb{C}}$ defined by

$$\mathcal{O}_{\mathbb{C}} = \{\oplus, \cdot_{\mathbb{C}}\} \cup \{\mathrm{f}_{\mathbb{C}} \mid \mathrm{f} \text{ is a } k\text{-ary boolean function different from } \vee\}.$$

Let us notice that, by definition, the operator $\ominus$ (resp. $\otimes$) is equal to the operator $\neg_{\mathbb{C}}$ (resp. $\wedge_{\mathbb{C}}$) whereas the operator $\oplus$ is different from $\vee_{\mathbb{C}}$, since:

$$\mathcal{C}_1 \vee_{\mathbb{C}} \mathcal{C}_2 = ((\ominus(\mathcal{C}_1) \otimes \mathcal{C}_2) \oplus (\mathcal{C}_1 \otimes \ominus(\mathcal{C}_2))) \oplus (\mathcal{C}_1 \otimes \mathcal{C}_2).$$

There exist several expressions (combinations of $\oplus$, $\otimes$ and $\ominus$) for a given $\mathrm{f}_{\mathbb{C}}$ operator. As an example, with the ternary $\vee^3(b_1, b_2, b_3) = b_1 \vee b_2 \vee b_3$ is associated an operator $\vee^3_{\mathbb{C}}$ that can be expressed as the combination of two $\oplus$ operators:

$$\vee^3_{\mathbb{C}}(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3) = (\mathcal{C}_1 \oplus \mathcal{C}_2) \oplus \mathcal{C}_3.$$

Reduced expressions can be found using Karnaugh maps for instance.

We now consider the 6-tuple $\mathbb{S}_{\mathbb{C}} = (\Sigma, \mathbb{C}, \mathcal{O}_{\mathbb{C}}, \mathrm{h}_{\mathbb{C}}, \{\{1\}\}, \emptyset)$ and show that it is a support.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Proposition 5.1. *The* 6*-tuple* $\mathbb{S}_{\mathbb{C}}$ *is a support.*

Proof. Properties of support are trivially checked for the clauses $\{\{1\}\}$, $\emptyset$ and $\{\emptyset\}$. Let us consider that $\mathcal{C}, \mathcal{C}_1, \ldots, \mathcal{C}_k$ are elements in $\mathbb{C} \setminus \{\emptyset, \{\emptyset\}\}$. According to the definitions of $\oplus$ and $\otimes$:

$$\begin{aligned}
L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1 \oplus \mathcal{C}_2)) &= L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1 \cup \mathcal{C}_2)) \\
&= L(\textstyle\sum_{C \in \mathcal{C}_1 \cup \mathcal{C}_2} \bigwedge_{e\, E \in C} \mathrm{c}(E)) \\
&= \textstyle\bigcup_{C \in \mathcal{C}_1 \cup \mathcal{C}_2} L(\bigwedge_{e\, E \in C} \mathrm{c}(E)) \\
&= \textstyle\bigcup_{C \in \mathcal{C}_1} L(\bigwedge_{e\, E \in C} \mathrm{c}(E)) \cup \bigcup_{C \in \mathcal{C}_2} L(\bigwedge_{e\, E \in C} \mathrm{c}(E)) \\
&= L(\textstyle\sum_{C \in \mathcal{C}_1} \bigwedge_{e\, E \in C} \mathrm{c}(E)) \cup L(\sum_{C \in \mathcal{C}_2} \bigwedge_{e\, E \in C} \mathrm{c}(E)) \\
&= L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1)) \cup L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2)) \\
&= L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1) + \mathrm{h}_{\mathbb{C}}(\mathcal{C}_2)) \\
L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1 \otimes \mathcal{C}_2)) &= L(\mathrm{h}_{\mathbb{C}}(\textstyle\bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} \{C_1 \cup C_2\}) \\
&= \textstyle\bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} L(\mathrm{h}_{\mathbb{C}}(\{C_1 \cup C_2\})) \\
&= \textstyle\bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} L(\wedge_{e C \in C_1 \cup C_2} c(E)) \\
&= \textstyle\bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} L(\wedge_{e C \in C_1} c(E)) \cap L(\wedge_{e C \in C_2} c(E)) \\
&= \textstyle\bigcup_{(C_1, C_2) \in \mathcal{C}_1 \times \mathcal{C}_2} L(\mathrm{h}_{\mathbb{C}}(\{C_1\})) \cap L(\mathrm{h}_{\mathbb{C}}(\{C_2\})) \\
&= \textstyle\bigcup_{C \in \mathcal{C}_1} L(\mathrm{h}_{\mathbb{C}}(\{C\})) \cap \bigcup_{C \in \mathcal{C}_2} L(\mathrm{h}_{\mathbb{C}}(\{C\})) \\
&= L(\textstyle\sum_{C \in \mathcal{C}_1} \mathrm{h}_{\mathbb{C}}(\{C\})) \cap L(\sum_{C \in \mathcal{C}_2} \mathrm{h}_{\mathbb{C}}(\{C\})) \\
&= L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1) \wedge_e \mathrm{h}_{\mathbb{C}}(\mathcal{C}_2)).
\end{aligned}$$

Moreover,

$$\begin{aligned}
L(\mathrm{h}_{\mathbb{C}}(\ominus(\{\{c\}\}))) &= L(\mathrm{h}_{\mathbb{C}}(\{\{n(c)\}\})) \\
&= \begin{cases} L(\mathrm{h}_{\mathbb{C}}(\{\{\overline{E}\}\})) & \text{if } c = E \\ L(\mathrm{h}_{\mathbb{C}}(\{\{E\}\})) & \text{if } c = \overline{E} \end{cases} \\
&= \begin{cases} L(\neg_e E) & \text{if } c = E \\ L(E) & \text{if } c = \overline{E} \end{cases} \\
&= \begin{cases} \neg_L(E) & \text{if } c = E \\ \neg_L(L(\neg_e E)) & \text{if } c = \overline{E} \end{cases} \\
&= \begin{cases} \neg_L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\})) & \text{if } c = E \\ \neg_L(L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\}))) & \text{if } c = \overline{E} \end{cases} \\
&= \neg_L(L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\}))).
\end{aligned}$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Consequently:

$$\begin{aligned}
L(\mathrm{h}_{\mathbb{C}}(\ominus(\mathcal{C}_1))) &= L(\mathrm{h}_{\mathbb{C}}(\otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \{\{\mathrm{n}(c)\}\})) \\
&= \bigcap_{C_1\in\mathcal{C}_1} L(\mathrm{h}_{\mathbb{C}}(\oplus_{c\in C}\{\{\mathrm{n}(c)\}\})) \\
&= \bigcap_{C_1\in\mathcal{C}_1} \bigcup_{c\in C} L(\mathrm{h}_{\mathbb{C}}(\{\{\mathrm{n}(c)\}\})) \\
&= \bigcap_{C_1\in\mathcal{C}_1} \bigcup_{c\in C} L(\mathrm{h}_{\mathbb{C}}(\ominus(\{\{c\}\}))) \\
&= \bigcap_{C_1\in\mathcal{C}_1} \bigcup_{c\in C} \neg_L(L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\}))) \\
&= \bigcap_{C_1\in\mathcal{C}_1} \neg_L(\bigcap_{c\in C} L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\}))) \\
&= \neg_L(\bigcup_{C_1\in\mathcal{C}_1} \bigcap_{c\in C} L(\mathrm{h}_{\mathbb{C}}(\{\{c\}\}))) \\
&= \neg_L(L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))) \\
&= L(\neg_e(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))).
\end{aligned}$$

Hence, according to definition of $\mathrm{f}_{\mathbb{C}}$:

$$\begin{aligned}
L(\mathrm{h}_{\mathbb{C}}(\mathrm{f}_{\mathbb{C}}(\mathcal{C}_1,\ldots,\mathcal{C}_k))) &= L(\mathrm{h}_{\mathbb{C}}(\oplus_{b=(b_1,\ldots,b_k)|\mathrm{f}(b)=1} \otimes_{1\le j\le k} \mathrm{g}(b_j,\mathcal{C}_j))) \\
&= \bigcup_{b=(b_1,\ldots,b_k)|\mathrm{f}(b)=1} \bigcap_{1\le j\le k} L(\mathrm{h}_{\mathbb{C}}(\mathrm{g}(b_j,\mathcal{C}_j))) \\
&= \bigcup_{b=(b_1,\ldots,b_k)|\mathrm{f}(b)=1} \bigcap_{1\le j\le k} \begin{cases} L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_j)) & \text{if } b_j = 1 \\ L(\mathrm{h}_{\mathbb{C}}(\ominus\mathcal{C}_j)) & \text{if } b_j = 0 \end{cases} \\
&= \bigcup_{b=(b_1,\ldots,b_k)|\mathrm{f}(b)=1} \bigcap_{1\le j\le k} \begin{cases} L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_j)) & \text{if } b_j = 1 \\ \neg_L L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_j)) & \text{if } b_j = 0 \end{cases} \\
&= \mathrm{f}_L(L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1)),\ldots,L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_k))) \\
&= L(\mathrm{f}_e(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1),\ldots,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_k))).
\end{aligned}$$

Furthermore,

$$\begin{aligned}
L(\mathrm{h}_{\mathbb{C}}(\mathcal{C}\cdot_{\mathbb{C}} F)) &= L\left(\mathrm{h}_{\mathbb{C}}\left(\bigcup_{C\in\mathcal{C}}\left\{\left(\bigwedge_{e\ E\in C} c(E)\right)\cdot F\right\}\right)\right) \\
&= \bigcup_{C\in\mathcal{C}} L\left(\left(\bigwedge_{e\ E\in C} c(E)\right)\cdot F\right) \\
&= \bigcup_{C\in\mathcal{C}} L\left(\bigwedge_{e\ E\in C} c(E)\right)\cdot L(F) \\
&= \left(\bigcup_{C\in\mathcal{C}} L\left(\bigwedge_{e\ E\in C} c(E)\right)\right)\cdot L(F) \\
&= L\left(\sum_{C\in\mathcal{C}}\bigwedge_{e\ E\in C} c(E)\right)\cdot L(F) \\
&= L\left(\mathrm{h}_{\mathbb{C}}(\mathcal{C})\right)\cdot L(F) \\
&= L\left(\mathrm{h}_{\mathbb{C}}(\mathcal{C})\cdot F\right)
\end{aligned}$$

□

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

We now study the properties of the derivation $\mathrm{D}_{\mathbb{C}}$ associated with the support $\mathbb{S}_{\mathbb{C}}$.

Theorem 5.2. *Let E be an expression in $\mathrm{Exp}(\Sigma)$. Let (D,B) be a couple in $\{\mathrm{D}_{\mathbb{C}}\}\times\{\mathrm{B}_A,\mathrm{B}_B,\mathrm{B}_C\}$. Then:*

The (D,B)-automaton of E is an AFA recognizing $L(E)$.

Proof. **(I)** Let us show that the derivation $\mathrm{D}_{\mathbb{C}}$ satisfies the sufficient conditions for finiteness of Proposition 3.10. **(a)** Since $\oplus=\cup$, the function $\oplus$ is associative, commutative and idempotent (**H1**). **(b)** According to the definition of the operators $\oplus$, $\otimes$ and $\ominus$:

$$\begin{aligned}
\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1\oplus\mathcal{C}_2)) &= \mathrm{D}_{\mathbb{C}}\Big(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1\cup\mathcal{C}_2)\Big)\\
&= \mathrm{D}_{\mathbb{C}}\Big(a,\textstyle\sum_{C\in\mathcal{C}_1\cup\mathcal{C}_2}\begin{cases}\neg_e 0 & \text{if } C=\emptyset,\\ \bigwedge_{e\,E\in C}c(E) & \text{otherwise,}\end{cases}\Big)\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1\cup\mathcal{C}_2}\begin{cases}\mathrm{D}_{\mathbb{C}}(a,\neg_e 0) & \text{if } C=\emptyset,\\ \mathrm{D}_{\mathbb{C}}(a,\bigwedge_{e\,E\in C}c(E)) & \text{otherwise,}\end{cases}\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1}\begin{cases}\mathrm{D}_{\mathbb{C}}(a,\neg_e 0) & \text{if } C=\emptyset,\\ \mathrm{D}_{\mathbb{C}}(a,\bigwedge_{e\,E\in C}c(E)) & \text{otherwise,}\end{cases}\\
&\quad\cup\textstyle\bigcup_{C\in\mathcal{C}_2}\begin{cases}\mathrm{D}_{\mathbb{C}}(a,\neg_e 0) & \text{if } C=\emptyset,\\ \mathrm{D}_{\mathbb{C}}(a,\bigwedge_{e\,E\in C}c(E)) & \text{otherwise,}\end{cases}\\
&= \mathrm{D}_{\mathbb{C}}(a,\textstyle\sum_{C\in\mathcal{C}_1}\begin{cases}\neg_e 0 & \text{if } C=\emptyset,\\ \bigwedge_{e\,E\in C}c(E) & \text{otherwise,}\end{cases}\\
&\quad\cup\mathrm{D}_{\mathbb{C}}(a,\textstyle\sum_{C\in\mathcal{C}_2}\begin{cases}\neg_e 0 & \text{if } C=\emptyset,\\ \bigwedge_{e\,E\in C}c(E) & \text{otherwise,}\end{cases}\\
&= \mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))\oplus\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2)).
\end{aligned}$$

$$\begin{aligned}
\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1\otimes\mathcal{C}_2)) &= \mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2}\{C_1\cup C_2\})\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2}\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\{C_1\cup C_2\})\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2}\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\{C_1\}\cup\{C_2\})\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2}\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\{C_1\})\cup\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\{C_2\})\\
&= \mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))\otimes\mathrm{D}_{\mathbb{C}}(a,\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2)).
\end{aligned}$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

$$
\begin{aligned}
\mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\ominus(\mathcal{C}_1))) &= \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \{\{\mathrm{n}(c)\}\})) \\
&= \otimes_{C_1\in\mathcal{C}_1} \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\{\oplus_{c\in C}\{\mathrm{n}(c)\}\})) \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\{\{\mathrm{n}(c)\}\})) \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\begin{cases} c' & \text{if } c = \overline{c'}, \\ \overline{c} & \text{otherwise.} \end{cases})) \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \begin{cases} \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(c')) & \text{if } c = \overline{c'}, \\ \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\overline{c})) & \text{otherwise.} \end{cases} \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \begin{cases} \mathrm{D}_{\mathbb{C}}(a, c') & \text{if } c = \overline{c'}, \\ \mathrm{D}_{\mathbb{C}}(a, \neg_e c)) & \text{otherwise.} \end{cases} \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \begin{cases} \ominus\ominus \mathrm{D}_{\mathbb{C}}(a, c') & \text{if } c = \overline{c'}, \\ \ominus \mathrm{D}_{\mathbb{C}}(a, c)) & \text{otherwise.} \end{cases} \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \begin{cases} \ominus \mathrm{D}_{\mathbb{C}}(a, \neg_e c') & \text{if } c = \overline{c'}, \\ \ominus \mathrm{D}_{\mathbb{C}}(a, c)) & \text{otherwise.} \end{cases} \\
&= \otimes_{C_1\in\mathcal{C}_1} \oplus_{c\in C} \ominus \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\{\{c\}\})) \\
&= \ominus \oplus_{C_1\in\mathcal{C}_1} \otimes_{c\in C} \mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\{\{c\}\})) \\
&= \ominus \mathrm{D}_{\mathbb{C}}(a, \textstyle\sum_{C_1\in\mathcal{C}_1} \wedge_{e c\in C} \mathrm{h}_{\mathbb{C}}(\{\{c\}\})) \\
&= \ominus(\mathrm{D}_{\mathbb{C}}(a, \mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))).
\end{aligned}
$$

Finally, since any $\mathrm{f}_{\mathbb{B}}$ is defined as combination of $\oplus$, $\otimes$ and $\ominus$ operators, hypothesis **H2** holds. According to Proposition 3.10, $\mathrm{D}_{\mathbb{C}}$ is finite.

(II) Let us show that any couple in $\{\mathrm{D}_{\mathbb{C}}\} \times \{\mathrm{B}_A, \mathrm{B}_B, \mathrm{B}_C\}$ satisfies the atom-derivability property. **(a)** By definition of B_B, the atom-derivability property is satisfied by $(\mathrm{D}_{\mathbb{C}}, \mathrm{B}_B)$. **(b)** By induction over the structure of E. Let a be a symbol in Σ. **(i)** If $E \in \Sigma \cup \{1, 0\}$ or if $E = F \cdot G$ or if $E = F^*$, since $\mathrm{B}_A(E) = \mathrm{B}_C(E) = E$, $\mathrm{Atom}(\mathrm{B}_A(E)) = \mathrm{Atom}(\mathrm{B}_C(E)) = \{E\}$. **(ii)** Let f be a k-ary boolean function. Let us first prove that the equation of the atom-derivanility property is satisfied for the operators $\oplus$, $\otimes$ and $\ominus$. If $\mathcal{C}_1$ or $\mathcal{C}_2$ equal $\emptyset$ or $\{\emptyset\}$, equation is trivially satisfied. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two clauses different from $\emptyset$ and $\{\emptyset\}$.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

$$\begin{aligned}
\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1 \cup \mathcal{C}_2))) &= \mathrm{Atom}(\mathrm{B}_C(\textstyle\sum_{C\in\mathcal{C}_1\cup\mathcal{C}_2} \wedge_{e\,E\in C} c(E)))\\
&= \mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_1\cup\mathcal{C}_2}\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1\cup\mathcal{C}_2}\mathrm{Atom}(\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1}\mathrm{Atom}(\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&\quad \cup \textstyle\bigcup_{C\in\mathcal{C}_2}\mathrm{Atom}(\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&= \mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_1}\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&\quad \cup \mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_2}\mathrm{B}_C(\wedge_{e\,E\in C} c(E)))\\
&= \mathrm{Atom}(\mathrm{B}_C(\textstyle\sum_{C\in\mathcal{C}_1} \wedge_{e\,E\in C} c(E)))\\
&\quad \cup \mathrm{Atom}(\mathrm{B}_C(\textstyle\sum_{C\in\mathcal{C}_2} \wedge_{e\,E\in C} c(E)))\\
&= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))) \cup \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2))).
\end{aligned}$$

$$\begin{aligned}
\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1 \otimes \mathcal{C}_2))) &= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2}\{C_1 \cup C_2\})))\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_1 \cup C_2\})))\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_1\})))\\
&\quad \cup \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_2\})))\\
&= \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_1\})))\\
&\quad \cup \textstyle\bigcup_{(C_1,C_2)\in\mathcal{C}_1\times\mathcal{C}_2} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_2\})))\\
&= \textstyle\bigcup_{C_1\in\mathcal{C}_1} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_1\})))\\
&\quad \cup \textstyle\bigcup_{C_2\in\mathcal{C}_2} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{C_2\})))\\
&= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))) \cup \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2))).
\end{aligned}$$

$$\begin{aligned}
\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\ominus\mathcal{C}_1))) &= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{\{n(x)\}\})))\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1}\bigcup_{x\in C} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{\{n(x)\}\})))\\
&= \textstyle\bigcup_{C\in\mathcal{C}_1}\bigcup_{x\in C} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\{\{x\}\})))\\
&= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\textstyle\bigcup_{C\in\mathcal{C}_1}\bigcup_{x\in C}\{\{x\}\})))\\
&= \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1))).
\end{aligned}$$

Hence, since any operator $\mathrm{f}_{\mathbb{C}}$ is a composition of $\oplus$, $\otimes$ and $\ominus$:

$$\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{f}_{\mathbb{C}}(\mathcal{C}_1,\ldots,\mathcal{C}_k)))) = \textstyle\bigcup_{1\le j\le k} \mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_j))).$$

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Consequently:

$$\begin{aligned}
&\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{D}_{\mathbb{C}}(a,\mathrm{f}_e(E_1,\ldots,E_k)))))\\
&=\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{f}_{\mathbb{C}}(\mathrm{D}_{\mathbb{C}}(a,E_1),\ldots,\mathrm{D}_{\mathbb{C}}(a,E_k)))))\\
&=\bigcup_{1\le j\le k}\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{D}_{\mathbb{C}}(a,E_j))))\\
&=\bigcup_{1\le j\le k}\bigcup_{E'\in\mathrm{Atom}(\mathrm{B}_C(E_j))}\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{D}_{\mathbb{C}}(a,E'))))\\
&=\bigcup_{E'\in\mathrm{Atom}(\mathrm{B}_C(E))}\mathrm{Atom}(\mathrm{B}_C(\mathrm{h}_{\mathbb{C}}(\mathrm{D}_{\mathbb{C}}(a,E')))).
\end{aligned}$$

Furthermore, if $\mathrm{f}\neq\vee$, $\mathrm{B}_A(\mathrm{f}_e(E_1,\ldots,E_k))=\mathrm{f}_e(E_1,\ldots,E_k)$.

$$\mathrm{Atom}(\mathrm{B}_A(\mathrm{f}_e(E_1,\ldots,E_k)))=\{\mathrm{f}_e(E_1,\ldots,E_k)\}.$$

Finally,

$$\begin{aligned}
\mathrm{Atom}(\mathrm{B}_A(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1\cup\mathcal{C}_2)))&=\mathrm{Atom}(\mathrm{B}_A(\textstyle\sum_{C\in\mathcal{C}_1\cup\mathcal{C}_2}\wedge_{e\,E\in C}c(E)))\\
&=\mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_1\cup\mathcal{C}_2}\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&=\textstyle\bigcup_{C\in\mathcal{C}_1\cup\mathcal{C}_2}\mathrm{Atom}(\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&=\textstyle\bigcup_{C\in\mathcal{C}_1}\mathrm{Atom}(\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&\quad\cup\textstyle\bigcup_{C\in\mathcal{C}_2}\mathrm{Atom}(\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&=\mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_1}\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&\quad\cup\mathrm{Atom}(\vee_{\mathbb{B}C\in\mathcal{C}_2}\mathrm{B}_A(\wedge_{e\,E\in C}c(E)))\\
&=\mathrm{Atom}(\mathrm{B}_A(\textstyle\sum_{C\in\mathcal{C}_1}\wedge_{e\ E\in C}c(E)))\\
&\quad\cup\mathrm{Atom}(\mathrm{B}_A(\textstyle\sum_{C\in\mathcal{C}_2}\wedge_{e\ E\in C}c(E)))\\
&=\mathrm{Atom}(\mathrm{B}_A(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_1)))\cup\mathrm{Atom}(\mathrm{B}_A(\mathrm{h}_{\mathbb{C}}(\mathcal{C}_2))).
\end{aligned}$$

Consequently, any couple in $\{\mathrm{D}_{\mathbb{C}}\}\times\{\mathrm{B}_A,\mathrm{B}_B,\mathrm{B}_C\}$ satisfies the atom-derivability property.

(III) According to Theorem 4.5, from **(I)** and **(II)**, the theorem holds. □

To conclude this section, we give two examples enlightening our construction. Example 5.3 illustrates the computation of an AFA from a regular expression while Example 5.4 shows that in some cases, our construction leads to an automaton which is exponentially smaller than the one computed with the construction of Brzozowski. In order to improve readability, regular expressions are simplified according to the following rules:

$$\begin{aligned}
E+0\equiv 0+E\equiv E\\
E\cdot 0\equiv 0\cdot E\equiv 0\\
E\cdot 1\equiv 1\cdot E\equiv E.
\end{aligned}$$

Example 5.3. Let $E=((ab)^*a)\mathrm{XOR}_e((abab)^*a)$. We now construct the $(\mathrm{D}_{\mathbb{C}},\mathrm{B}_C)$-automaton of E, that is the AFA (Σ,Q,E,F,δ). We first compute the derivatives $\mathcal{C}$ of E w.r.t. any symbol in Σ, according to the derivation $\mathrm{D}_{\mathbb{C}}$ and then

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

compute the derivatives of the expressions that are atoms of the base of $\mathrm{h}_{\mathbb{C}}(\mathcal{C})$. This scheme is repeated until no more expression is produced.

$$\begin{aligned}
\mathrm{D}_{\mathbb{C}}(a,E) &= (\ominus(\mathrm{D}_{\mathbb{C}}(a,(ab)^*a)) \otimes \mathrm{D}_{\mathbb{C}}(a,(abab)^*a)) \\
&\quad \oplus(\mathrm{D}_{\mathbb{C}}(a,(ab)^*a) \otimes \ominus(\mathrm{D}_{\mathbb{C}}(a,(abab)^*a))) \\
&= (\ominus(\{\{b(ab)^*a\},\{1\}\}) \otimes \{\{bab(abab)^*a\},\{1\}\}) \\
&\quad \oplus(\{\{b(ab)^*a\},\{1\}\} \otimes \ominus(\{\{bab(abab)^*a\},\{1\}\})) \\
&= (\{\{\overline{b(ab)^*a},\overline{1}\}\} \otimes \{\{bab(abab)^*a\},\{1\}\}) \\
&\quad \oplus(\{\{b(ab)^*a\},\{1\}\} \otimes \{\{\overline{bab(abab)^*a},\overline{1}\}\}) \\
&= \{\{\overline{b(ab)^*a},\overline{1},bab(abab)^*a\},\{\overline{b(ab)^*a},\overline{1},1\}, \\
&\quad \{b(ab)^*a,\overline{bab(abab)^*a},\overline{1}\},\{1,\overline{bab(abab)^*a},\overline{1}\}\}
\end{aligned}$$

$$\begin{aligned}
\mathrm{D}_{\mathbb{C}}(b,E) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(a,b(ab)^*a) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(b,b(ab)^*a) &= \{\{(ab)^*a\}\} \\
\mathrm{D}_{\mathbb{C}}(a,(ab)^*a) &= \{\{b(ab)^*a\},\{1\}\} \\
\mathrm{D}_{\mathbb{C}}(b,(ab)^*a) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(a,1) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(b,1) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(a,bab(abab)^*a) &= \emptyset
\end{aligned}$$

$$\begin{aligned}
\mathrm{D}_{\mathbb{C}}(b,bab(abab)^*a) &= \{\{ab(abab)^*a\}\} \\
\mathrm{D}_{\mathbb{C}}(a,ab(abab)^*a) &= \{\{b(abab)^*a\}\} \\
\mathrm{D}_{\mathbb{C}}(b,ab(abab)^*a) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(a,b(abab)^*a) &= \emptyset \\
\mathrm{D}_{\mathbb{C}}(b,b(abab)^*a) &= \{\{(abab)^*a\}\} \\
\mathrm{D}_{\mathbb{C}}(a,(abab)^*a) &= \{\{bab(abab)^*a\},\{1\}\} \\
\mathrm{D}_{\mathbb{C}}(b,(abab)^*a) &= \emptyset.
\end{aligned}$$

From the computation of the derivatives, we deduce:

- the set $Q = \{q_1,\ldots,q_8\}$ of states:
  $q_1 = E$, $q_2 = b(ab)^*a$, $q_3 = 1$, $q_4 = bab(abab)^*a$;
  $q_5 = ab(abab)^*a$, $q_6 = b(abab)^*a$, $q_7 = (ab)^*a$, $q_8 = (abab)^*a$;
- the function F from Q to $\mathbb{B}$:
  $F(q_3) = F(q_7) = F(q_8) = 1$;
  $F(q_1) = F(q_2) = F(q_4) = F(q_5) = F(q_6) = 0$;
- and the function δ from $Q \times \Sigma$ to BoolForm(Q):

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
a	$\neg_{\mathbb{B}} q_2 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_3 \wedge_{\mathbb{B}} q_4 \vee_{\mathbb{B}} \neg_{\mathbb{B}} q_2 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_3 \wedge_{\mathbb{B}} q_3 \vee_{\mathbb{B}} q_2 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_4 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_3 \vee_{\mathbb{B}} q_3 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_4 \wedge_{\mathbb{B}} \neg_{\mathbb{B}} q_3$	0	0	0	q_6	0	$q_2 \vee_{\mathbb{B}} q_3$	$q_4 \vee_{\mathbb{B}} q_3$
b	0	q_7	0	q_5	0	q_8	0	0

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

Let us notice that in this example, substituting $\mathrm{B}_{\mathbb{C}}(E)$ to E in I would produce a smaller automaton.

Example 5.4. Let n_1 and n_2 be any two integers satisfying $n_1 < n_2$. Let us consider the expression $E = E_1 \cap E_2$ where $E_1 = (a+b)^*a(a+b)^{n_1}$ and $E_2 = (a+b)^*b(a+b)^{n_2}$. It can be checked that for any word w in Σ^*:

$$\frac{\mathrm{d}}{\mathrm{d}_w}(E) = \frac{\mathrm{d}}{\mathrm{d}_w}(E_1) \cap \frac{\mathrm{d}}{\mathrm{d}_w}(E_2).$$

For any integer j, let us set $F_j = (a+b)^j$ with $F_0 = 1$. It is a folk knowledge that the number of dissimilar derivatives of E_1 (resp. E_2) is exponential with respect to n_1 (resp. n_2). Consequently, the number of states of the $(\mathrm{D}_B, \mathrm{B}_B)$-automaton of E is exponential with respect to $\max(n_1, n_2)$.

Moreover, the number of derivated terms of E (using our extension of Antimirov partial derivation) is quadratic with respect to the number of occurrences of symbols of E. Indeed

$$\mathrm{D}_A(w, E_1 \cap E_2) \subset \bigcup_{(F,G)\in\{E_1,F_{n_1},F_{n_1-1},\ldots 1,0\}\times\{E_2,F_{n_2},F_{n_2-1},\ldots 1,0\}} \{F \cap G\}$$

consequently, the $(\mathrm{D}_A, \mathrm{B}_A)$-automaton of E has a quadratic number of states with respect to the number of occurrences of symbols of E.

Finally, the $(\mathrm{D}_{\mathbb{C}}, \mathrm{B}_{\mathbb{C}})$ automaton of E has a linear number of states since

$$\begin{array}{ll} \mathrm{D}_{\mathbb{C}}(a, E) = \{\{E_1, E_2\}, \{F_{n_1}, E_2\}\} & \mathrm{D}_{\mathbb{C}}(b, E) = \{\{E_1, E_2\}, \{E_1, F_{n_2}\}\}, \\ \mathrm{D}_{\mathbb{C}}(a, E_1) = \{\{E_1\}, \{F_{n_1}\}\} & \mathrm{D}_{\mathbb{C}}(b, E_1) = \{\{E_1\}\} \\ \mathrm{D}_{\mathbb{C}}(a, E_2) = \{\{E_2\}\}\} & \mathrm{D}_{\mathbb{C}}(b, E_2) = \{\{E_2\}, \{F_{n_2}\}\} \end{array}$$

$$\mathrm{D}_{\mathbb{C}}(a, F_j) = \mathrm{D}_{\mathbb{C}}(b, F_j) = \begin{cases} \{\{F_{j-1}\}\} & \text{if } j \geq 1 \\ \emptyset & \text{otherwise.} \end{cases}$$

6. Conclusion

This paper provides two main results. First, the theoretical scheme of derivations *via* a support allows us to formalize intrinsic properties of (unrestricted) regular expression derivations. As a by-product we obtain a kind of unification of the classical derivations that compute word derivatives, partial derivatives or extended partial derivatives. Secondly, the notion of base function that associates a boolean formula with a regular expression allows us to show how to deduce an alternating automaton equivalent to a given regular expression from the set of its derivatives *via* a given support. We are now investigating new features: it is possible, for example, to define morphisms from one support to another one in order to study the relations between the associated automata. An other perspective is to replace the derivation mapping by an other mapping (right derivation or left-and-right derivation for example, or any transformation with good properties). There also exist

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press

well-know algorithms to reduce boolean formulas (Karnaugh, Quine-McCluskey); we intend to investigate reduction techniques based on derivation. Finally we intend to extend the theoretical derivation scheme in order to handle the derivation of regular expression with multiplicities.

References

[1] V. Antimirov, Partial derivatives of regular expressions and finite automaton constructions. *Theoret. Comput. Sci.* **155** (1996) 291–319.

[2] V.M. Antimirov and P.D. Mosses, Rewriting extended regular expressions. *Theoret. Comput. Sci.* **143** (1995) 51–72.

[3] G. Berry and R. Sethi, From regular expressions to deterministic automata. *Theoret. Comput. Sci.* **48** (1986) 117–126.

[4] J.A. Brzozowski, Derivatives of regular expressions. *J. Assoc. Comput. Mach.* **11** (1964) 481–494.

[5] J.A. Brzozowski, Quotient complexity of regular languages. *J. Automata, Languages and Combinatorics* **15** (2010) 71–89.

[6] J.A. Brzozowski and E.L. Leiss, On equations for regular languages, finite automata, and sequential networks. *Theoret. Comput. Sci.* **10** (1980) 19–35.

[7] P. Caron, J.-M. Champarnaud and L. Mignot, Partial derivatives of an extended regular expression, in *LATA*, vol. 6638 of *Lect. Notes Comput. Sci.* Edited by A.H. Dediu, S. Inenaga and C. Martín-Vide. Springer (2011)179–191.

[8] J.-M. Champarnaud, F. Ouardi and D. Ziadi, An efficient computation of the equation $\mathbb{K}$-automaton of a regular $\mathbb{K}$-expression. *Fundam. Inform.* **90** (2009) 1–16.

[9] J.-M. Champarnaud and D. Ziadi, Canonical derivatives, partial derivatives, and finite automaton constructions. *Theoret. Comput. Sci.* **239** (2002) 137–163.

[10] J. Clarke, An algorithm for RELAX NG validation. Implementation Report (2002).

[11] J.-H. Conway, *Regular algebra and finite machines*. Chapman and Hall (1971).

[12] A. Ginzburg, A procedure for checking equality of regular expressions. *J. ACM* **14** (1967) 355–362.

[13] L. Ilie and S. Yu, Follow automata. *Inf. Comput.* **186** (2003) 140–162.

[14] D. Krob, Differentation of K-rational expressions. *Internat. J. Algebra Comput.* **2** (1992) 57–87.

[15] S. Lombardy and J. Sakarovitch, Derivatives of rational expressions with multiplicity. *Theoret. Comput. Sci.* **332** (2005) 141–177.

[16] M. Might, D. Darais and D. Spiewak, Parsing with derivatives: a functional pearl, in *ICFP*, edited by M.M.T. Chakravarty, Zh. Hu and O. Danvy. ACM (2011) 189–195.

[17] J. Myhill, Finite automata and the representation of events. *WADD* **TR-57-624** (1957) 112–137.

[18] A. Nerode, Linear automata transformation, in *Proc. of AMS* **9** (1958) 541–544.

[19] S. Owens, J.H. Reppy and A. Turon, Regular-expression derivatives re-examined. *J. Funct. Program.* **19** (2009) 173–190.

[20] J. Sakarovitch, The language, the expression, and the (small) automaton, in *CIAA*, vol. 3845 of *Lect. Notes Comput. Sci.* Edited by J. Farré, I. Litovsky and S. Schmitz. Springer (2005) 15–30.

[21] M. Sulzmann and K.Z.M. Lu, Partial derivative regular expression pattern matching. Manuscript (2007).

[22] K. Thompson, Regular expression search algorithm. *Comm. ACM* **11** (1968) 419–422.

Communicated by Ch. Choffrut.
Received March 14, 2014. Accepted March 14, 2014.

https://doi.org/10.1051/ita/2014010 Published online by Cambridge University Press