Fundamenta Informaticae, , vol. 90, no. 1-2, (2009)
IOS Press

An efficient computation of the equation $\mathbb{K}$-automaton of a regular $\mathbb{K}$-expression

Jean-Marc Champarnaud
University of Rouen, LITIS
76800 Saint-Etienne du Rouvray – France
jean-marc.champarnaud@univ-rouen.fr

Faissal Ouardi
University of Rouen, LITIS
76800 Saint-Etienne du Rouvray – France
faissal.ouardi@univ-rouen.fr

Djelloul Ziadi
University of Rouen, LITIS
76800 Saint-Etienne du Rouvray – France
djelloul.ziadi@univ-rouen.fr

Abstract. The aim of this paper is to describe a quadratic algorithm for computing the equation $\mathbb{K}$-automaton of a regular $\mathbb{K}$-expression as defined by Lombardy and Sakarovitch. Our construction is based on an extension to regular $\mathbb{K}$-expressions of the notion of c-continuation that we introduced to compute the equation automaton of a regular expression as a quotient of its position automaton.

Keywords: regular $\mathbb{K}$-expression, $\mathbb{K}$-automaton, equation $\mathbb{K}$-automaton , c-continuation

1. Introduction

The conversion of a regular expression into an automaton is a rather old problem. The first algorithms appeared in the sixties, respectively based on the notion of position [22, 16], prebase [23] and ε-transition [26]. In the nineties, different techniques were developed to design quadratic[1] algorithms for

Address for correspondence: University of Rouen, LITIS, 76800 Saint-Etienne du Rouvray – France
[1] In the following, complexity depends on the size of the expression.

the construction of the position automaton: the star normal form [3], the compressed normal NFA [14] and the ZPC-structure [27]. Moreover the notion of partial derivative of a regular expression introduced by Antimirov [2] raised several challenging problems that boosted the research in this topic.

First, what is the relation between these different constructions? It was proved in [10] that the notions of prebase and of partial derivative lead to an identical automaton, the equation automaton[2]. Based on the notion of c-continuation it was shown in [11] that the equation automaton is a quotient of the c-continuation automaton that is itself isomorphic to the position automaton. A new construction based on the follow relation was introduced in [18] and the follow automaton was proved to be a quotient of the position automaton. It was shown in [7] that the follow automaton can be constructed from the ZPC-structure. Finally, as mentioned in [28], the deep relation that exists between position, equation and follow automata can be easily understood through the ZPC-structure and the c-continuation computation.

Second, how to compare the performance of the algorithms that yield a quotient of the position automaton? The equation automaton [11] and the follow automaton [18, 7] are both computed in quadratic time and space. Comparing the number of states of these automata is a more intricated issue. A new approach for this problem appears in [8, 9]: a class of normalized expressions is defined such that any expression can be turned into an equivalent normalized one in linear time, and it is proved that the equation automaton of a normalized expression is always smaller than its follow automaton.

Last, how to extend these constructions to the case of $\mathbb{K}$-expressions? Concerning the position $\mathbb{K}$-automaton, the first algorithm, based on an inductive construction, is described in [5], and the first quadratic one, based on a generalization of the notion of ZPC-structure appears in [28, 6]. As for the extension of the equation automaton, the notion of $\mathbb{K}$-derivative is used in [20, 21, 24] to show that the automaton of derived terms of a regular $\mathbb{K}$-expression is the weighted equivalent of the equation automaton. More recently, a unified frame is presented in [1] for the construction of position, equation, and follow automata and $\mathbb{K}$-automata, based on Thompson ε-automaton construction, epsilon-removal and minimization.

This paper addresses the construction of the equation $\mathbb{K}$-automaton. Our algorithm is based on the computation of a $\mathbb{K}$-covering [21] from the c-continuation $\mathbb{K}$-automaton onto the equation $\mathbb{K}$-automaton. It therefore provides a good understanding of this automaton. The structure of the automaton is derived from a set of c-continuations that is computed straightforwardly via the boolean case algorithm, while the set of weights is computed apart, leading to an overall quadratic complexity, i.e. as efficient as in the boolean case.

Let us mention that this algorithm has been implemented inside VAUCANSON platform [15], using the data structure proposed in [12] for the computation of the c-continuations. Implementation is precisely one of our main motivations for designing efficient algorithms that deal with weighted finite-state automata. Indeed these automata are widely used in many domains: pattern matching, string processing, image compression, information retrieval and extraction and machine learning among others.

The work reported in this paper was partially presented at DLT'2007 [13]. It is organized as follows. The next section contains useful preliminaries and Section 3 is a reminder of the fundamental results presented in [21]. Section 4 generalizes the computation of c-derivatives to regular $\mathbb{K}$-expressions, describes the construction of the equation $\mathbb{K}$-automaton from the c-continuation one and gives an analysis of its complexity.

[2]This name refers to the systems of expression equations used by Mirkin [23]. Other names are partial derivative automaton and Antimirov automaton.

2. Preliminaries

Let A be a finite alphabet, and $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ be a semiring (commutative or not). The star operator $\circledast$ can be partially defined over $\mathbb{K}$ as follows [17, 19]: the scalar $y^{\circledast} \in \mathbb{K}$ is the unique solution (if it exists) of the equations $y \otimes x \oplus \overline{1} = x$ and $x \otimes y \oplus \overline{1} = x$, with $\overline{0}^{\circledast} = \overline{1}$. In this paper, examples come from the semiring $(\mathbb{Q}, +, \times)$.

Definition 2.1. A (non-commutative) formal series S with coefficients in $\mathbb{K}$ and variables in A is a mapping from the free monoid A^* to $\mathbb{K}$ that associates a coefficient $\langle S, w\rangle \in \mathbb{K}$ with the word $w \in A^*$.

A formal series is usually written as an infinite sum: $S = \sum_{u \in A^*} \langle S, u\rangle u$. The *support* of the formal series S is the language $\mathrm{supp}(S) = \{u \in A^* \mid \langle S, u\rangle \neq \overline{0}\}$. The set of formal series over A with coefficients in $\mathbb{K}$ is denoted by $\mathbb{K}\langle\langle A\rangle\rangle$. A structure of semiring is defined on $\mathbb{K}\langle\langle A\rangle\rangle$ as follows [19]:
– $\langle S \oplus T, u\rangle = \langle S, u\rangle \oplus \langle T, u\rangle$,
– $\langle S \otimes T, u\rangle = \bigoplus_{u_1 u_2 = u} \langle S, u_1\rangle \otimes \langle T, u_2\rangle$, with $S, T \in \mathbb{K}\langle\langle A\rangle\rangle$.

The star of a series S is defined by: $S^* = \bigoplus_{n \geq 0} S^n$ with $S^0 = \varepsilon,\ S^n = S^{n-1} \otimes S$ if $n > 0$. For clarity the symbol S^* is used for series, whereas the symbol $\circledast$ is kept for the coefficient semiring $\mathbb{K}$. The star of a formal series does not always exist. The *proper series* S_p associated with a formal series S is defined by $\langle S_p, \varepsilon\rangle = 0$ and $\langle S_p, u\rangle = \langle S, u\rangle$ for any word $u \in A^+$. The star of a proper series always exists. We will use the following construction for the star of a formal series.

Proposition 2.1. *[19]* The star of a formal series $S \in \mathbb{K}\langle\langle A\rangle\rangle$ is defined if and only if $\langle S, \varepsilon\rangle^{\circledast}$ is defined in $\mathbb{K}$. In this case: $S^* = \langle S, \varepsilon\rangle^{\circledast}(S_p \langle S, \varepsilon\rangle^{\circledast})^*$.

A polynomial is a formal series with finite support. The set of polynomials is denoted by $\mathbb{K}\langle A\rangle$ that is a subsemiring of $\mathbb{K}\langle\langle A\rangle\rangle$.

Definition 2.2. The semiring of regular series $\mathbb{K}\,\mathrm{Rat}(A^*) \subset \mathbb{K}\langle\langle A\rangle\rangle$ is the smallest set of $\mathbb{K}\langle\langle A\rangle\rangle$ that contains the semiring $\mathbb{K}\langle A\rangle$ of polynomials, and that is stable under the operations of addition, product and star (when it is defined).

Definition 2.3. A regular $\mathbb{K}$-expression over an alphabet A is inductively defined as follows:
– $a \in A, k \in \mathbb{K}$ are regular $\mathbb{K}$-expressions that respectively denote the regular series $S_a = a$ and $S_k = \overline{k}$,
– if F, G and H are regular $\mathbb{K}$-expressions that respectively denote the regular series S_{F}, S_{G} and S_{H} (such that ${S_{\mathrm{H}}}^*$ exists), then $(\mathrm{F} + \mathrm{G})$, $(\mathrm{F} \cdot \mathrm{G})$ and (H^*) are regular $\mathbb{K}$-expressions that respectively denote the regular series $S_{\mathrm{F}} \oplus S_{\mathrm{G}}$, $S_{\mathrm{F}} \otimes S_{\mathrm{G}}$ and ${S_{\mathrm{H}}}^*$.

Let E be a $\mathbb{K}$-expression. We will denote by A_{E} the alphabet of E, and by $|\,\mathrm{E}\,|$ its size that is equal to the size of the syntax tree of E. The linearized version $\overline{\mathrm{E}}$ of E is the $\mathbb{K}$-expression deduced from E by associating with every occurrence of a symbol a of A_{E} its rank i in E. Subscripted symbols a_i are called positions. Let h be the mapping that associates with a position $a_i \in A_{\overline{\mathrm{E}}}$ the symbol $a \in A_{\mathrm{E}}$. Given an expression F over a set of positions, we denote by $h(\mathrm{F})$ the expression obtained by replacing every position x in F by $h(x)$. We write $\mathrm{E} \equiv \mathrm{F}$ if E and F graphically coincid.

An element of $\mathbb{K}$ can be seen as a $\mathbb{K}$-expression (written k) or as a scalar (written $\overline{k}$). It induces a morphism from the semiring of $\mathbb{K}$-expressions with no occurrence of symbol of A_{E} to the semiring $\mathbb{K}$.

We denote by $\overline{F}$ the scalar[3] associated with such an expression F. Following [6], the null term $\lambda(\mathrm{E})$ of a $\mathbb{K}$-expression E is the $\mathbb{K}$-expression obtained from E by replacing each occurrence of a symbol of A_{E} by the symbol 0 (associated with the scalar $\overline{0}$ of $\mathbb{K}$). For example, if $\mathrm{E} = (\frac{1}{2} \cdot a^* + \frac{1}{3} \cdot b^*)^* \cdot a^*$, we get $A_{\mathrm{E}} = \{a, b\}$, $\overline{\mathrm{E}} = (\frac{1}{2} \cdot a_1^* + \frac{1}{3} \cdot b_2^*)^* \cdot a_3^*$, $A_{\overline{\mathrm{E}}} = \{a_1, b_2, a_3\}$ and $|\overline{\mathrm{E}}| = 13$. It comes $\lambda(\mathrm{E}) = (\frac{1}{2} \cdot 0^* + \frac{1}{3} \cdot 0^*)^* \cdot 0^*$ and $\overline{\lambda(\mathrm{E})} = \overline{6}$.

Definition 2.4. A $\mathbb{K}$-automaton $\mathcal{A} = \langle Q, A, q_0, \delta, \gamma, \mu \rangle$ is defined as follows[4]:

- Q is a finite set of states; A is the alphabet; q_0 is the initial state,
- $\delta \subseteq Q \times A \times Q$ is the set of transitions,
- $\gamma : \delta \to \mathbb{K}$ (resp. $\mu : Q \to \mathbb{K}$) is the transition (resp. output) weight function.

A path p from a state q_0 to a state q_n is a sequence of transitions $p = (p_1, p_2, \cdots, p_n)$ with $p_i = (q_{i-1}, a_i, q_i)$ for $1 \leq i \leq n$. Its label is the word $w(p) = a_1 a_2 \cdots a_n$. We denote by $\mathrm{coef}(p)$ the weight of the path p in $\mathcal{A}$, with $\mathrm{coef}(p) = \gamma(p_1) \otimes \gamma(p_2) \otimes \cdots \otimes \gamma(p_n) \otimes \mu(q_n)$. Let $\mathcal{C}_{\mathcal{A}}$ be the set of all paths in $\mathcal{A}$ starting from q_0. The $\mathbb{K}$-automaton $\mathcal{A}$ *realizes* the series $S_{\mathcal{A}}$ defined by:

$$S_{\mathcal{A}} = \sum_{u \in A^*} \langle S_{\mathcal{A}}, u \rangle u, \text{ where } \langle S_{\mathcal{A}}, u \rangle = \bigoplus_{p \in \mathcal{C}_{\mathcal{A}}, w(p) = u} \mathrm{coef}(p)$$

A series $S \in \mathbb{K}\langle\langle A \rangle\rangle$ is recognizable if there exists a $\mathbb{K}$-automaton that realizes it.

Theorem 2.1. *(Schützenberger [25])* A formal series is recognizable if and only if it is regular.

Definition 2.5. *[21]* Let $\mathcal{A}$ and $\mathcal{B}$ be two $\mathbb{K}$-automata. Let $\Delta(q, q') = \bigoplus_{(q,a,q') \in \delta} \gamma(q, a, q')a$. A surjective mapping φ from the set of states of $\mathcal{A}$ onto the set of states of $\mathcal{B}$ induces a $\mathbb{K}$-covering from $\mathcal{A}$ onto $\mathcal{B}$ if $\mathcal{A}$ is such that:

$$\forall p, q \in Q_{\mathcal{A}},\ \varphi(p) = \varphi(q) \quad \Rightarrow \quad \begin{cases} 1) \quad \mu(p) = \mu(q) \\ 2) \quad \forall r \in Q_{\mathcal{A}}, \ \bigoplus_{s \in \varphi^{-1}\varphi(r)} \Delta_{\mathcal{A}}(p, s) = \bigoplus_{s \in \varphi^{-1}\varphi(r)} \Delta_{\mathcal{A}}(q, s) \end{cases}$$

and if $\mathcal{B}$ satisfies the following conditions:

$$\begin{array}{lll} 3) \quad \forall r \in Q_{\mathcal{B}}, & \mu(r) = \mu(p) & \text{for any } p \in \varphi^{-1}(r) \\ 4) \quad \forall (r, s) \in Q_{\mathcal{B}}^2, & \Delta_{\mathcal{B}}(r, s) = \bigoplus_{q \in \varphi^{-1}(s)} \Delta_{\mathcal{A}}(p, q) & \text{for any } p \in \varphi^{-1}(r) \end{array}$$

Proposition 2.2. *[21]* Let $\mathcal{A}$ and $\mathcal{B}$ be two $\mathbb{K}$-automata. If $\varphi : \mathcal{A} \longrightarrow \mathcal{B}$ is a $\mathbb{K}$-covering, then $S_{\mathcal{A}} = S_{\mathcal{B}}$.

[3] Actually there is no conflict with the notation $\overline{\mathrm{E}}$ used for the linearized version of E.
[4] A more general definition is given in [4].

Let E be a regular $\mathbb{K}$-expression over an alphabet A and consider the language $\mathrm{L}(\overline{\mathrm{E}})$ associated with the linearized version of E. In the weighted case, $\mathrm{First}(\mathrm{E})$, $\mathrm{Last}(\mathrm{E})$ and $\mathrm{Follow}(x,\mathrm{E})$ with x a position of E, are polynomials in $\mathbb{K}\langle A_{\overline{\mathrm{E}}}\rangle$ that can be computed as follows [6]:

$$\begin{aligned}
\mathrm{First}(k) &= \overline{0} \text{ for all } k \in \mathbb{K}\\
\mathrm{First}(a) &= \overline{1}a_i \text{ (a_i is the position associated with a in } A_{\overline{\mathrm{E}}})\\
\mathrm{First}(\mathrm{F}+\mathrm{G}) &= \mathrm{First}(\mathrm{F}) \oplus \mathrm{First}(\mathrm{G})\\
\mathrm{First}(\mathrm{F}\cdot\mathrm{G}) &= \mathrm{First}(\mathrm{F}) \oplus \overline{\lambda(\mathrm{F})}\,\mathrm{First}(\mathrm{G})\\
\mathrm{First}(\mathrm{F}^*) &= \overline{\lambda(\mathrm{F})}^{\circledast}\,\mathrm{First}(\mathrm{F})
\end{aligned}$$

Similar rules hold for Last except for $\mathrm{Last}(\mathrm{F}\cdot\mathrm{G}) = \mathrm{Last}(\mathrm{G}) \oplus \overline{\lambda(\mathrm{G})}\,\mathrm{Last}(\mathrm{F})$.

$$\begin{aligned}
\mathrm{Follow}(x,k) &= \overline{0} \text{ for all } k \in \mathbb{K}\\
\mathrm{Follow}(x,a) &= \overline{0} \text{ for all } a \in A\\
\mathrm{Follow}(x,\mathrm{F}+\mathrm{G}) &= \mathrm{Follow}(x,\mathrm{F}) \oplus \mathrm{Follow}(x,\mathrm{G})\\
\mathrm{Follow}(x,\mathrm{F}\cdot\mathrm{G}) &= \mathrm{Follow}(x,\mathrm{F}) \oplus \langle \mathrm{Last}(\mathrm{F}), x\rangle\,\mathrm{First}(\mathrm{G}) \oplus \mathrm{Follow}(x,\mathrm{G})\\
\mathrm{Follow}(x,\mathrm{F}^*) &= \mathrm{Follow}(x,\mathrm{F}) \oplus \langle \mathrm{Last}(\mathrm{F}^*), x\rangle\,\mathrm{First}(\mathrm{F})
\end{aligned}$$

These polynomials lead to the definition of the position $\mathbb{K}$-automaton of E, that realizes the series denoted by E.

Definition 2.6. The position $\mathbb{K}$-automaton $\mathcal{P}_{\mathrm{E}} = \langle Q, A_{\mathrm{E}}, q_0, \delta, \gamma, \mu\rangle$ is defined by:

- $Q = \{q_0\} \cup A_{\overline{\mathrm{E}}}$, with $q_0 \notin A_{\overline{\mathrm{E}}}$,
- $(q,a,p) \in \delta \Leftrightarrow h(p) = a$ and $\begin{cases} \langle \mathrm{First}(\mathrm{E}), p\rangle \neq \overline{0} & \text{if } q = q_0,\\ \langle \mathrm{Follow}(q,\mathrm{E}), p\rangle \neq \overline{0} & \text{otherwise.}\end{cases}$
- For all $(p,a,q) \in \delta$, it holds:
  $\gamma(q,a,p) = \begin{cases} \langle \mathrm{First}(\mathrm{E}), p\rangle & \text{if } q = q_0,\\ \langle \mathrm{Follow}(q,\mathrm{E}), p\rangle & \text{otherwise.}\end{cases}$
- $\mu(q) = \begin{cases} \overline{\lambda}(\mathrm{E}) & \text{if } q = q_0,\\ \langle \mathrm{Last}(\mathrm{E}), q\rangle & \text{otherwise.}\end{cases}$

3. From $\mathbb{K}$-derivatives to the equation $\mathbb{K}$-automaton

In this section, we recall the main results reported in [21] about the computation of the set of $\mathbb{K}$-derivatives of a regular $\mathbb{K}$-expression. Such a $\mathbb{K}$-derivative is defined as a polynomial, which leads to a generalization of the notion of partial derivative [2].

Definition 3.1. Let E be a regular $\mathbb{K}$-expression[5] and $a \in A$. The $\mathbb{K}$-derivative of E w.r.t. a is the polynomial $\partial_a(\mathrm{E})$ inductively defined as follows:

$$\partial_a(k) = \overline{0} \qquad \partial_a(b) = \begin{cases} \overline{1} & \text{if } b = a \\ \overline{0} & \text{otherwise} \end{cases}$$

$$\partial_a(\mathrm{E}+\mathrm{F}) = \partial_a(\mathrm{E}) \oplus \partial_a(\mathrm{F})$$
$$\partial_a(\mathrm{E}\cdot\mathrm{F}) = \partial_a(\mathrm{E})\cdot\mathrm{F} \oplus \overline{\lambda(\mathrm{E})}\partial_a(\mathrm{F})$$
$$\partial_a(\mathrm{E}^*) = \overline{\lambda(\mathrm{E})}^{\circledast}(\partial_a(\mathrm{E})\cdot\mathrm{E}^*)$$

By linearity, the $\mathbb{K}$-derivative of a polynomial is given by: $\partial_a(\bigoplus_{i\in I} \overline{k_i}\,\mathrm{E}_i) = \bigoplus_{i\in I} \overline{k_i}\partial_a(\mathrm{E}_i)$.
The $\mathbb{K}$-derivative of a regular $\mathbb{K}$-expression E w.r.t. a word $u \in A^+$ is defined by: $\forall u \in A^+,\ \forall a \in A,\ \partial_{ua}(\mathrm{E}) = \partial_a(\partial_u(\mathrm{E}))$.

Example 3.1. The $\mathbb{K}$-derivatives w.r.t. a and b of the regular $\mathbb{K}$-expression $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$ are the polynomials $\partial_a(\mathrm{E}) = \overline{\frac{1}{2}}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$ and $\partial_b(\mathrm{E}) = \overline{\frac{1}{2}}b^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$.

Proposition 3.1. *[21]* Let E and F be regular $\mathbb{K}$-expressions. Let $u \in A^+$ and $k \in \mathbb{K}$. Then it holds:
1) $\partial_u(\mathrm{E}+\mathrm{F}) = \partial_u(\mathrm{E}) \oplus \partial_u(\mathrm{F})$,
2) $\partial_u(\mathrm{E}\cdot\mathrm{F}) = \partial_a(\mathrm{E})\cdot\mathrm{F} \oplus \bigoplus_{u=vw} \langle\partial_v(\mathrm{E}), 1\rangle\partial_w(\mathrm{F})$,
3) $\partial_u(\mathrm{E}^*) = \bigoplus_{u=u_1\cdots u_n} \overline{\lambda(\mathrm{E})}^{\circledast}\langle\partial_{u_1}(\mathrm{E}), 1\rangle\overline{\lambda(\mathrm{E})}^{\circledast}\cdots\langle\partial_{u_{n-1}}(\mathrm{E}), 1\rangle\overline{\lambda(\mathrm{E})}^{\circledast}(\partial_{u_n}(\mathrm{E})\cdot\mathrm{E}^*)$.

There exists a set of regular $\mathbb{K}$-expressions that plays a specific role in the computation of the $\mathbb{K}$-derivatives of E: the set of derived terms of E.

Definition 3.2. The set $\mathrm{dt}(\mathrm{E})$ of the derived terms of a regular $\mathbb{K}$-expression E is inductively defined as follows:

$$\mathrm{dt}(k) = \emptyset \qquad \mathrm{dt}(a) = \{1\} \qquad \mathrm{dt}(\mathrm{F}+\mathrm{G}) = \mathrm{dt}(\mathrm{F}) \cup \mathrm{dt}(\mathrm{G})$$

$$\mathrm{dt}(\mathrm{F}\cdot\mathrm{G}) = \bigcup_{\mathrm{F}_i\in\mathrm{dt}(\mathrm{F})} (\mathrm{F}_i\cdot\mathrm{G}) \cup \mathrm{dt}(\mathrm{G})$$
$$\mathrm{dt}(\mathrm{F}^*) = \bigcup_{\mathrm{F}_i\in\mathrm{dt}(\mathrm{F})} (\mathrm{F}_i\cdot\mathrm{F}^*)$$

Notice that the set of derived terms of a regular $\mathbb{K}$-expression E plays the same function as the prebase of a regular expression, as introduced by Mirkin [23].

Example 3.2. For the regular $\mathbb{K}$-expression $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$, we have:
$$\mathrm{dt}(\mathrm{E}) = \{\frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*, a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*, b^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*\}.$$

The computation of the $\mathbb{K}$-derivatives of a regular $\mathbb{K}$-expression E leads to the construction of the equation $\mathbb{K}$-automaton[6] that realizes the series denoted by E.

[5] In [21] a different definition of a $\mathbb{K}$-regular expression is used, where k is not an expression. Hence a slightly different formulation of the $\mathbb{K}$-derivative.
[6] Also called the automaton of derived terms in [21].

Definition 3.3. The equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}} = \langle Q, A_{\mathrm{E}}, q_0, \delta, \gamma, \mu \rangle$ of a regular $\mathbb{K}$-expression E is defined as follows:

- $Q = \mathrm{dt}(\mathrm{E}) \cup \{\mathrm{E}\}$; $q_0 = \{\mathrm{E}\}$,
- $(\mathrm{E}_i, a, \mathrm{E}_j) \in \delta \Leftrightarrow \langle \partial_a(\mathrm{E}_i), \mathrm{E}_j \rangle \neq \overline{0}$ for all $\mathrm{E}_i, \mathrm{E}_j \in Q$,
- $\gamma(\mathrm{E}_i, a, \mathrm{E}_j) = \langle \partial_a(\mathrm{E}_i), \mathrm{E}_j \rangle$ for all $\mathrm{E}_i, \mathrm{E}_j \in Q$; $\mu(\mathrm{E}_i) = \overline{\lambda(\mathrm{E}_i)}$.

Example 3.3. (Ex. 3.1 continued)

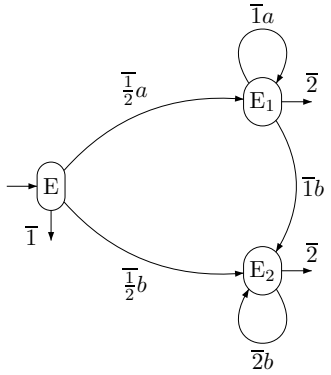

Figure 1. The equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}}$ associated with $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$.

Theorem 3.1. *[21]* There exists a $\mathbb{K}$-covering from the position $\mathbb{K}$-automaton onto the equation $\mathbb{K}$-automaton of a regular $\mathbb{K}$-expression.

4. From c-derivatives to the equation $\mathbb{K}$-automaton

We now define the notions of c-derivative, c-continuation and c-continuation automaton for a regular $\mathbb{K}$-expression. The main point here is that, following Definition 2.3, a regular $\mathbb{K}$-expression E can be seen as a regular expression over the alphabet $A_{\mathrm{E}} \cup B_{\mathrm{E}}$, where B_{E} is the set of elements of $\mathbb{K}$ occurring in E. Therefore, for any symbol $x \in A_{\overline{\mathrm{E}}}$, the c-continuation w.r.t. x of the regular $\mathbb{K}$-expression E graphically coincids with the c-continuation w.r.t. x of the associated regular expression. As a consequence, the set of states and the set of transitions of the c-continuation $\mathbb{K}$ automaton of E are computed straightforwardly via the boolean case algorithm described in [12].

We also define the notion of coefficient of a regular $\mathbb{K}$-expression w.r.t. a symbol and show how the weights of the transitions are deduced from the coefficients of the c-continuations. We assume that, for E a regular $\mathbb{K}$-expression, the following identities are satisfied: $0 \cdot \mathrm{E} = \mathrm{E} \cdot 0 = 0$, $0 + \mathrm{E} = \mathrm{E} + 0 = \mathrm{E}$, $1 \cdot \mathrm{E} = \mathrm{E} \cdot 1 = \mathrm{E}$.

4.1. From c-derivatives to the c-continuation $\mathbb{K}$-automaton

Definition 4.1. The c-derivative of a regular $\mathbb{K}$-expression E w.r.t. a symbol a, written $d_a(\mathrm{E})$, is defined by:

$$
\begin{aligned}
d_a(k) &= 0 \\
d_a(x) &= \begin{cases} 1 & \text{if } a = x \\ 0 & \text{otherwise} \end{cases} \\
d_a(\mathrm{F}+\mathrm{G}) &= \begin{cases} d_a(\mathrm{F}) & \text{if } d_a(\mathrm{F}) \neq 0 \\ d_a(\mathrm{G}) & \text{otherwise} \end{cases} \\
d_a(\mathrm{F}\cdot\mathrm{G}) &= \begin{cases} d_a(\mathrm{F})\cdot\mathrm{G} & \text{if } d_a(\mathrm{F}) \neq 0 \\ d_a(\mathrm{G}) & \text{if } d_a(\mathrm{F}) = 0 \text{ and } \lambda(\mathrm{F}) \neq 0 \\ 0 & \text{otherwise} \end{cases} \\
d_a(\mathrm{F}^*) &= d_a(\mathrm{F})\cdot\mathrm{F}^*
\end{aligned}
$$

The c-derivative of E w.r.t. a word u is defined by: $d_\varepsilon(\mathrm{E}) = \mathrm{E}$, and $d_{u_1\ldots u_n}(\mathrm{E}) = d_{u_2\ldots u_n}(d_{u_1}(\mathrm{E}))$.

The main property of c-derivatives still holds for regular $\mathbb{K}$-expressions, leading to the notion of c-continuation.

Theorem 4.1. If E is linear, for every symbol $a \in A_{\overline{\mathrm{E}}}$ there exists a $\mathbb{K}$-expression $c_a(\mathrm{E})$, called the c-continuation of E w.r.t. a, such that for every word $u \in A^*_{\overline{\mathrm{E}}}$, the c-derivative $d_{ua}(\mathrm{E})$ is either 0 or $c_a(\mathrm{E})$.

Example 4.1. (Ex. 3.1 continued)
We have $d_{b_2b_2}(\overline{\mathrm{E}}) = b_2^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^*$ and $d_{b_3b_2}(\overline{\mathrm{E}}) = b_2^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^*$.

Proposition 4.1. For every symbol a of a linear expression E, the c-continuation $c_a(\mathrm{E})$ can be computed as follows:

$$
\begin{aligned}
c_a(a) &= 1 \\
c_a(\mathrm{F}+\mathrm{G}) &= \begin{cases} c_a(\mathrm{F}) & \text{if } c_a(\mathrm{F}) \text{ exists} \\ c_a(\mathrm{G}) & \text{otherwise} \end{cases}
\end{aligned}
\qquad
\begin{aligned}
c_a(\mathrm{F}\cdot\mathrm{G}) &= \begin{cases} c_a(\mathrm{F})\cdot\mathrm{G} & \text{if } c_a(\mathrm{F}) \text{ exists} \\ c_a(\mathrm{G}) & \text{otherwise} \end{cases} \\
c_a(\mathrm{F}^*) &= c_a(\mathrm{F})\cdot\mathrm{F}^*
\end{aligned}
$$

In the following we will write c_x instead of $c_x(\mathrm{E})$ when there is no ambiguity.

Definition 4.2. The coefficient of a linear $\mathbb{K}$-expression E w.r.t. a symbol a is the scalar $k_a(\mathrm{E})$ inductively defined as follows:

$$
\begin{aligned}
k_a(k) &= \overline{0} \\
k_a(b) &= \begin{cases} \overline{1} & \text{if } b = a \\ \overline{0} & \text{otherwise} \end{cases}
\end{aligned}
\qquad
\begin{aligned}
k_a(\mathrm{F}+\mathrm{G}) &= k_a(\mathrm{F}) \oplus k_a(\mathrm{G}) \\
k_a(\mathrm{F}\cdot\mathrm{G}) &= k_a(\mathrm{F}) \oplus \overline{\lambda(\mathrm{F})} \otimes k_a(\mathrm{G}) \\
k_a(\mathrm{F}^*) &= \overline{\lambda(\mathrm{F})}^{\circledast} \otimes k_a(\mathrm{F})
\end{aligned}
$$

The coefficient of E w.r.t. a word u is defined by: $k_\varepsilon(\mathrm{E}) = \overline{\lambda(\mathrm{E})}$, and $k_{u_1\ldots u_n}(\mathrm{E}) = k_{u_1}(\mathrm{E}) \otimes k_{u_2\ldots u_n}(d_{u_1}(\mathrm{E}))$.

Definition 4.3. The c-continuation automaton $\mathcal{C}_{\mathrm{E}} = \langle Q_{\mathcal{C}}, A_{\mathrm{E}}, q_{\mathcal{C}}, \delta_{\mathcal{C}}, \gamma_{\mathcal{C}}, \mu_{\mathcal{C}} \rangle$ of a regular $\mathbb{K}$-expression E is defined by:

- $Q_{\mathcal{C}} = \{(x, c_x) \mid x \in A_{\overline{\mathrm{E}}} \cup \{0\}\}$; $q_{\mathcal{C}} = (0, c_0)$,
- $((x, c_x), a, (y, c_y)) \in \delta_{\mathcal{C}} \Leftrightarrow h(y) = a$ and $d_y(c_x) \equiv c_y$,
- $\gamma_{\mathcal{C}}((x, c_x), a, (y, c_y)) = k_y(c_x)$; $\mu_{\mathcal{C}}(c_x) = \overline{\lambda(c_x)}$.

Notice that if $k_y(c_x) = \overline{0}$ then the transition is not considered.

Example 4.2. (Ex. 3.1 continued)

$$
\begin{aligned}
c_0(\mathrm{E}) &= \overline{\mathrm{E}} \\
c_{a_1}(\mathrm{E}) &= a_1^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^* \\
c_{b_2}(\mathrm{E}) &= b_2^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^* \\
c_{b_3}(\mathrm{E}) &= b_3^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^*
\end{aligned}
$$

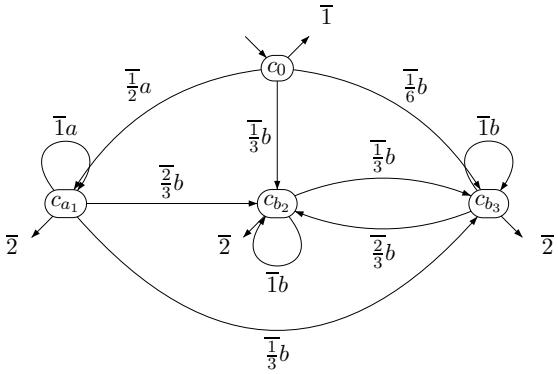

Figure 2. The c-continuation $\mathbb{K}$-automaton associated with $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$.

4.2. From the c-continuation $\mathbb{K}$-automaton to the equation $\mathbb{K}$-automaton

As for the boolean case, we can relate the c-continuation $\mathbb{K}$-automaton to both the position and the equation $\mathbb{K}$-automata.

Proposition 4.2. Let E be a regular $\mathbb{K}$-expression. Then the following equalities hold: $\mathrm{First}(\mathrm{E}) = \bigoplus_{x \in A_{\overline{\mathrm{E}}}} k_x(\overline{\mathrm{E}})x$, $\mathrm{Last}(\mathrm{E}) = \bigoplus_{x \in A_{\overline{\mathrm{E}}}} \overline{\lambda(c_x)}x$ and, for all position x of E, $\mathrm{Follow}(x, \mathrm{E}) = \bigoplus_{y \in A_{\overline{\mathrm{E}}}} k_y(c_x)y$.

Theorem 4.2. The c-continuation and position $\mathbb{K}$-automata of a regular $\mathbb{K}$-expression are isomorphic.

As a corollary and according to Proposition 2.2, the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}$ realizes the series denoted by E. Following the boolean track [11], we now consider the equivalence $\sim$ defined by:

$$\forall (x, c_x), (z, c_z) \in Q_{\mathcal{C}}, (x, c_x) \sim (z, c_z) \Leftrightarrow h(c_x) \equiv h(c_z)$$

We will denote by $[x]$ the class of the state (x, c_x) and we will write z instead of (z, c_z) whenever there is no ambiguity (for instance $z \in [x]$). We associate with every class $[x]$ the expression $C_{[x]}$ such that, for all $z \in [x]$, $C_{[x]} = h(c_z)$.

We now define the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim} = \langle Q_{\sim}, A_{\mathrm{E}}, I_{\sim}, \delta_{\sim}, \gamma_{\sim}, \mu_{\sim} \rangle$ whose states are the $\sim$-classes and that will be proved to be equivalent to $\mathcal{C}_{\mathrm{E}}$. The relation $\sim$ being right-invariant [11], the function $\delta_{\sim}$ is well-defined. The soundness of the weight functions $\gamma_{\sim}$ and $\mu_{\sim}$ will be proved by exhibiting a $\mathbb{K}$-covering from $\mathcal{C}_{\mathrm{E}}$ onto $\mathcal{C}_{\mathrm{E}}/_{\sim}$.

Definition 4.4. The $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim} = \langle Q_{\sim}, A_{\mathrm{E}}, q_{\sim}, \delta_{\sim}, \gamma_{\sim}, \mu_{\sim} \rangle$ is defined by:

- $Q_{\sim} = \{C_{[x]} \mid x \in A_{\overline{\mathrm{E}}} \cup \{0\}\}$; $q_{\sim} = C_{[0]}$,
- $(C_{[x]}, a, C_{[y]}) \in \delta_{\sim} \Leftrightarrow$ for any $z \in [x]$, $\exists t \in [y] \mid h(t) = a$ and $(c_z, a, c_t) \in \delta_{\mathcal{C}}$,
- $\gamma_{\sim}(C_{[x]}, a, C_{[y]}) = \bigoplus_{t \in [y], h(t)=a} \gamma_{\mathcal{C}}((z, c_z), a, (t, c_t))$, for any $z \in [x]$,
- $\mu_{\sim}(C_{[x]}) = \overline{\lambda(c_z)}$, for any $z \in [x]$.

Theorem 4.3. Let E be a regular $\mathbb{K}$-expression. Consider the surjective mapping $\varphi : Q_{\mathcal{C}} \longrightarrow Q_{\sim}$ defined by: for all $x \in A_{\overline{\mathrm{E}}} \cup \{0\}$, $\varphi(x, c_x) = C_{[x]}$. Then the mapping φ defines a $\mathbb{K}$-covering from $\mathcal{C}_{\mathrm{E}}$ onto $\mathcal{C}_{\mathrm{E}}/_{\sim}$.

Proof:
Notice that we have $\varphi(x, c_x) = \varphi(z, c_z) \Leftrightarrow (x, c_x) \sim (z, c_z) \Leftrightarrow h(c_x) \equiv h(c_z)$.

Condition 1 of the Definition 2.5 says that two equivalent states in $\mathcal{C}_{\mathrm{E}}$ should have the same output weight. It can be rewritten: $\forall (x, c_x), (z, c_z) \in Q_{\mathcal{C}}$, $(x, c_x) \sim (z, c_z) \Rightarrow \mu_{\mathcal{C}}(x, c_x) = \mu_{\mathcal{C}}(z, c_z)$. This condition is satisfied since $\mu_{\mathcal{C}}(x, c_x) = \overline{\lambda(c_x)}$ and $(x, c_x) \sim (z, c_z) \Rightarrow h(c_x) \equiv h(c_z) \Rightarrow \lambda(c_x) = \lambda(c_z)$. Moreover, since $(x, c_x) \sim (z, c_z) \Rightarrow \lambda(c_x) = \lambda(c_z)$, $\mu_{\sim}(C_{[x]})$ can be computed as $\overline{\lambda(c_z)}$, for any $z \in [x]$. Hence the Condition 3 is also satisfied.

Condition 2 ensures that the transition weights in $\mathcal{C}_{\mathrm{E}}/_{\sim}$ can be computed from the weights of the transitions outgoing from any state in the origin class. Let us set $S_x = \bigoplus_{t \in [y], h(t)=a} \gamma_{\mathcal{C}}((x, c_x), a, (t, c_t))$. The transition weight function $\gamma_{\mathcal{C}}$ must be such that: $\forall (x, c_x), (z, c_z) \in Q_{\mathcal{C}}$, $(x, c_x) \sim (z, c_z) \Rightarrow \forall (y, c_y) \in Q_{\mathcal{C}}$, $\forall a \in A_{\mathrm{E}}$, $S_x = S_z$. We have $\gamma_{\mathcal{C}}((x, c_x), a, (t, c_t)) = k_t(c_x)$. Let $t = a_i$ (resp. $t = a_j$) be the k^{th} symbol occurring in c_x (resp. c_z). Since $h(c_x) \equiv h(c_z)$, we have $h(a_i) = h(a_j)$. Moreover, since $h(d_{a_i}(c_x)) = h(d_{a_j}(c_z))$, we get $h(c_{a_i}) \equiv h(c_{a_j})$, and thus $(a_i, c_{a_i}) \sim (a_j, c_{a_j})$. Hence for all position $t = a_i$ occurring in the sum S_x there is a corresponding position $t = a_j$ occurring in the sum S_z. Finally, by a simple induction we get that $k_{a_i}(c_x) = k_{a_j}(c_z)$. Consequently, the two sums S_x and S_z are equal and Condition 2 is satisfied. Moreover, since the sum S_z is independent from the choice of z in $[x]$, $\gamma_{\sim}(C_{[x]}, a, C_{[y]})$ can be computed as S_z for any $z \in [x]$. Hence the Condition 4 is also satisfied. □

As a corollary and according to Proposition 2.2, the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim}$ realizes the series denoted by E.

We now show that the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim}$ and the equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}}$ are isomorphic.

Lemma 4.1. Let E be a regular $\mathbb{K}$-expression. Then it holds: $\mathrm{dt}(\mathrm{E}) = \bigcup_{x \in A_{\overline{\mathrm{E}}}} h(c_x(\mathrm{E}))$.

Lemma 4.2. The following equality holds for all positions x in $A_{\overline{\mathrm{E}}} \cup \{0\}$:

$$\partial_a(h(c_x(\mathrm{E}))) = \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{E})) h(d_{a_i}(c_x(\mathrm{E})))$$

Proof:
By induction on the size of E. If E is k or a_i the proposition holds. Let $\mathrm{E} = \mathrm{F} + \mathrm{G}$. We have $c_x(\mathrm{E}) = c_x(\mathrm{F})$ if $c_x(\mathrm{F})$ exists and $c_x(\mathrm{G})$ otherwise. Assume that $c_x(\mathrm{F})$ is defined. In this case we have:

$$\begin{aligned}
\partial_a(h(c_x(\mathrm{F}+\mathrm{G})) &= \partial_a(h(c_x(\mathrm{F})) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{F}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{F}))h(d_{a_i}(c_x(\mathrm{F}))) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{E}))h(d_{a_i}(c_x(\mathrm{E})))
\end{aligned}$$

Let us consider now the case when $\mathrm{E} = \mathrm{F} \cdot \mathrm{G}$. We have $c_x(\mathrm{E}) = c_x(\mathrm{F}) \cdot \overline{\mathrm{G}}$ if $c_x(\mathrm{F})$ exists and $c_x(\mathrm{G})$ otherwise. Assume that $c_x(\mathrm{F})$ is defined. In this case we have:

$$\begin{aligned}
\partial_a(h(c_x(\mathrm{F}\cdot\mathrm{G}))) &= \partial_a(h(c_x(\mathrm{F}) \cdot \overline{\mathrm{G}})) \\
&= \partial_a(h(c_x(\mathrm{F})))h(\overline{\mathrm{G}}) \oplus \overline{\lambda(c_x(\mathrm{F}))}\partial_a(h(\overline{\mathrm{G}})) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{F}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{F}))h(d_{a_i}(c_x(\mathrm{F})) \cdot \overline{\mathrm{G}}) \oplus \\
&\quad \bigoplus_{a_i \in A_{\overline{\mathrm{G}}}, h(a_i)=a} \overline{\lambda(c_x(\mathrm{F}))}k_{a_i}(\overline{\mathrm{G}})h(d_{a_i}(\overline{\mathrm{G}})) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} (k_{a_i}(c_x(\mathrm{F})) \oplus \overline{\lambda(c_x(\mathrm{F}))}k_{a_i}(\overline{\mathrm{G}}))h(d_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{G}})) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{G}})h(d_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{G}}))
\end{aligned}$$

Let us consider the case when $\mathrm{E} = \mathrm{F}^*$. We have $c_x(\mathrm{E}) = c_x(\mathrm{F}) \cdot \overline{\mathrm{F}}^*$, for any $x \in A_{\overline{\mathrm{E}}}$. In this case we have:

$$\begin{aligned}
\partial_a(h(c_x(\mathrm{F}^*)) &= \partial_a(h(c_x(\mathrm{F}) \cdot \overline{\mathrm{F}}^*)) \\
&= \partial_a(h(c_x(\mathrm{F})))h(\overline{\mathrm{F}}^*) \oplus \overline{\lambda(c_x(\mathrm{F}))}\partial_a(h(\overline{\mathrm{F}}^*)) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} (k_{a_i}(c_x(\mathrm{F})) \oplus \overline{\lambda(c_x(\mathrm{F}))}k_{a_i}(\overline{\mathrm{F}}^*))h(d_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{F}}^*)) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{F}}^*)h(d_{a_i}(c_x(\mathrm{F}) \cdot \overline{\mathrm{F}}^*)) \\
&= \bigoplus_{a_i \in A_{\overline{\mathrm{E}}}, h(a_i)=a} k_{a_i}(c_x(\mathrm{F}^*))h(d_{a_i}(c_x(\mathrm{F}^*)))
\end{aligned}$$

□

Corollary 4.1. The following equality holds for all positions x, y in $A_{\overline{\mathrm{E}}} \cup \{0\}$:

$$\langle \partial_a(h(c_x(\mathrm{E})), h(c_y(\mathrm{E}))\rangle = \bigoplus_{a_i \in [y], h(a_i)=a} k_{a_i}(c_x(\mathrm{E}))$$

Theorem 4.4. Let E be a regular $\mathbb{K}$-expression. The $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim}$ and the equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}}$ are isomorphic.

Proof:
By Lemma 4.1, the $\mathbb{K}$-automata $\mathcal{C}_{\mathrm{E}}/_{\sim}$ and $\mathcal{E}_{\mathrm{E}}$ have identical sets of states and identical output weight functions. By Lemma 4.2, they have identical sets of transitions, and by Corollary 4.1, identical transition weight functions. □

Finally, let us notice that combining Theorem 4.2, Theorem 4.3 and Theorem 4.4 provides a proof of Theorem 3.1.

4.3. An efficient algorithm for converting a regular $\mathbb{K}$-expression into its equation $\mathbb{K}$-automaton

We now present an efficent algorithm for computing the equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}}$ of a regular $\mathbb{K}$-expression E. Following Theorem 4.4, it is based on the construction of the $\mathbb{K}$-automata $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{E}}/_{\sim}$. We first show how to compute the structure (states and transitions) of these two $\mathbb{K}$-automata by simply running the boolean case algorithm *AlgoCtoE* [12] for converting a regular expression into its equation automaton.

Let A_{E} be the alphabet of the regular $\mathbb{K}$-expression E and B_{E} be the set of elements of $\mathbb{K}$ occurring in E. Then, according to Definition 2.3 the $\mathbb{K}$-expression E can be seen as a regular expression over the alphabet $A_{\mathrm{E}} \uplus B_{\mathrm{E}}$. Let us denote this expression by $\hat{\mathrm{E}}$. We have: $\mathrm{E} \equiv \hat{\mathrm{E}}$. Then from Definition 4.1 it comes:

Proposition 4.3. For any symbol $x \in A_{\mathrm{E}}$, the c-continuation of the regular $\mathbb{K}$-expression E w.r.t. x is equal to the c-continuation of the regular expression $\hat{\mathrm{E}}$ w.r.t. x.

As a consequence, the set of c-continuations of the regular $\mathbb{K}$-expression E and the structure (the set of states and the set of transitions) of the $\mathbb{K}$-automata $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{E}}/_{\sim}$ can be directly determined by applying the boolean case algorithm *AlgoCtoE* [12] to the regular expression $\hat{\mathrm{E}}$.

Example 4.3. Let us illustrate this fact with the regular $\mathbb{K}$-expression $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$. Notice that the $\mathbb{K}$-automata $\mathcal{C}_{\mathrm{E}}$ and $\mathcal{C}_{\mathrm{E}}/_{\sim}$ are respectively shown in Figure 2 and Figure 1.

The syntax tree of E is equipped with links (represented by dashed arrows in Figure 3) that allow to compute a c-continuation as a product of subexpressions of E. For each concatenation node the left son is linked to the right one and for each star node the son is linked to its parent.

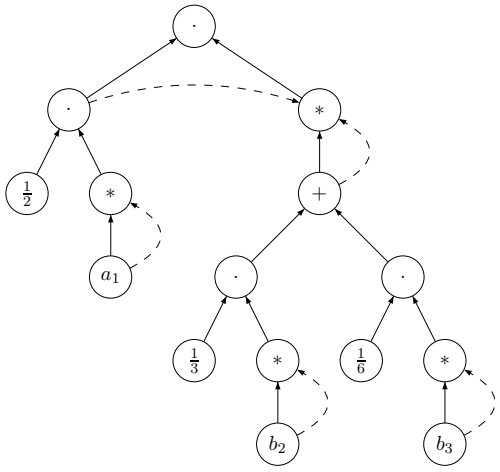

Figure 3. The syntax tree of $\mathrm{E} = \frac{1}{2}a^*(\frac{1}{3}b^* + \frac{1}{6}b^*)^*$.

Then for each symbol x in $A_{\overline{\mathrm{E}}}$, the c-continuation $c_x(\mathrm{E})$ is the concatenation of the targets of the links collected on the path up to the root. Adding the expression $c_0(\mathrm{E})$ we obtain:

$$\begin{aligned}
c_0(\mathrm{E}) &= \frac{1}{2}a_1^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^* \\
c_{a_1}(\mathrm{E}) &= a_1^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^* \\
c_{b_2}(\mathrm{E}) &= b_2^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^* \\
c_{b_3}(\mathrm{E}) &= b_3^*(\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^*
\end{aligned}$$

We first preprocess the star subexpressions of E. There are 4 star subexpressions:

$$\begin{aligned}
s_0 &= a_1^* \\
s_1 &= b_2^* \\
s_2 &= b_3^* \\
s_3 &= (\frac{1}{3}b_2^* + \frac{1}{6}b_3^*)^*
\end{aligned}$$

The alphabet of the strings $h(s_i)$ is: $\{a, b, +, \cdot, *\} \cup B_{\mathrm{E}}$. After identification of the strings $h(s_i)$ (that is sorting in lexicographic order and comparing one string to the following one), there are 3 classes:

$$\begin{aligned}
S_0 &= \{s_0\} &&= a^* \\
S_1 &= \{s_1, s_2\} &&= b^* \\
S_2 &= \{s_3\} &&= (\frac{1}{3}b^* + \frac{1}{6}b^*)^*
\end{aligned}$$

We then compute the pseudo-continuations:

$$\begin{aligned}
\ell_0(\mathrm{E}) &= \frac{1}{2}S_0 \cdot S_2 \\
\ell_1(\mathrm{E}) &= S_0 \cdot S_2 \\
\ell_2(\mathrm{E}) &= S_1 \cdot S_2 \\
\ell_3(\mathrm{E}) &= S_1 \cdot S_2
\end{aligned}$$

The alphabet of the strings $h(\ell_i)$ is: $\{a, b, +, \cdot, *\} \cup B_{\mathrm{E}} \cup \{S_0, S_1, S_2\}$. After identification, there are 3 classes:

$$\begin{aligned} L_0(\mathrm{E}) &= \{\ell_0\} &&= \frac{1}{2} S_0.S_2 \\ L_1(\mathrm{E}) &= \{\ell_1\} &&= S_0.S_2 \\ L_2(\mathrm{E}) &= \{\ell_2, \ell_3\} &&= S_1.S_2 \end{aligned}$$

The two identification steps are performed by calling the Identification procedure detailed in [12]. The only difference with the classical boolean case is that the alphabet of the strings $h(s_i)$ and the alphabet of the strings $h(\ell_i)$ are augmented with the elements of B_{E}. Since these alphabets have still an $O(|E|)$ complexity, the $O(|E|^2)$ space and time complexity of the Identification procedure is not modified.

Let us now consider the computation of the set of transitions of $\mathcal{C}_{\mathrm{E}}$.
For every x in $A_{\overline{\mathrm{E}}}$, we consider the set $T_x = \{y \mid y \in A_{\overline{\mathrm{E}}} \text{ and } d_y(c_x) \neq 0\}$. The set of transitions of $\mathcal{C}_{\mathrm{E}}$ is easily computed from the sets T_x since $y \in T_x \Leftrightarrow ((x, c_x), h(y), (y, c_y)) \in \delta_{\mathcal{C}}$. We first compute the $First$ sets of $\hat{\mathrm{E}}$. It amounts to construct the $First$ forest in the ZPC-structure of $\hat{\mathrm{E}}$ [27] as far as $Null(k)$ is set to 1, for all $k \in B_{\mathrm{E}}$. For each symbol x in $A_{\overline{\mathrm{E}}}$, the set T_x is the union of the $First$ sets of the origins of the links collected on the path up to the root of the syntax tree of $\hat{\mathrm{E}}$. In our example, we get that: $T_0 = \{a_1, b_2, b_3\}$, $T_1 = \{a_1, b_2, b_3\}$, $T_2 = \{b_2, b_3\}$ and $T_3 = \{b_2, b_3\}$.

As in the classical boolean case, the set T_x can be computed in O(|E|) time as a disjoint union of $First$ sets using the ZPC technique (in particular the elimination of redundant follow links). Similarly the computation of the set of transitions of $\mathcal{C}_{\mathrm{E}}/_{\sim}$ can be performed in the same way as in the boolean case, that is by choosing an arbitrary element in each $\sim$-class of states.

The Algorithm *AlgoKExptoEq* described below converts a regular $\mathbb{K}$-expression E into its equation $\mathbb{K}$-automaton $\mathcal{E}_{\mathrm{E}}$. It first constructs the structure of the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim}$ by running the boolean case algorithm *AlgoCtoE* over the regular expression $\hat{\mathrm{E}}$, and then it computes the sets of weights $\mu_{\sim}$ and $\gamma_{\sim}$ according to Definition 4.4.

Algorithm 1 AlgoKExptoEq(E)

Input: a regular $\mathbb{K}$-expression E
Output: the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}/_{\sim}$
Compute the set of c-continuations of E and the $\mathbb{K}$-automaton $\mathcal{C}_{\mathrm{E}}$
Compute $Q_{\sim}$ as the quotient $Q_{\mathcal{C}}/\sim$
Set $\delta_{\sim}$ to $\emptyset$ and $\gamma_{\sim}$ to $\overline{0}$
for all $C_{[x]} \in Q_{\sim}$ **do**
    Choose an arbitrary element (z, c_z) in $C_{[x]}$.
    Set $\mu_{\sim}(C_{[x]}) = \mu_{\mathcal{C}}((z, c_z))$.
    for all $((z, c_z), h(y), (y, c_y)) \in \delta_{\mathcal{C}}$ **do**
        $\delta_{\sim} = \delta_{\sim} \cup (C_{[x]}, h(y), C_{[y]})$
        $\gamma_{\sim}(C_{[x]}, h(y), C_{[y]}) = \gamma_{\sim}(C_{[x]}, h(y), C_{[y]}) \oplus \gamma_{\mathcal{C}}((z, c_z), h(y), (y, c_y))$
    end for
end for

The complexity of this algorithm is as follows. Let n (resp. $\tilde{n}$) be the number of states in $\mathcal{C}_{\mathrm{E}}$ (resp. $\mathcal{C}_{\mathrm{E}}/_{\sim}$). We assume that $O(n) = O(|\,\mathrm{E}\,|)$. Although $O(\tilde{n}) = O(n)$, time complexity will be expressed

as far as possible w.r.t. $\tilde{n}$ for more accuracy.

On the one hand, the computation of the set of states of $\mathcal{C}_{\mathrm{E}}/_{\sim}$ (Step 1: Lines 3–4) and the computation of its set of transitions (Step 2: Lines 6,7,9,10) are carried out via the boolean case algorithm *AlgoCtoE* [12]. As a straightforward consequence, Step 1 can be implemented in $O(n^2)$ time over the ZPC-structure [27] of the regular expression associated with E, and Step 2 in $O(\tilde{n}n)$ time. Notice that any optimisation of Step 1 or of Step 2, would lead to an improvement of both boolean and weighted algorithms.

On the other hand, the computation of the transition weight function (Step 4: Lines 6,7,9,11) can be implemented in $O(\tilde{n}n)$ time. Indeed, the weight $\gamma_{\sim}(C_{[x]}, a, C_{[y]})$ only depends on the weights of the transitions outgoing from one arbitrarily chosen element in $[x]$. Moreover, each weight $\gamma_{\mathcal{C}}((z, c_z), h(y), (y, c_y))$ is involved in the computing of at most one weight $\gamma_{\sim}(C_{[x]}, a, C_{[y]})$. Notice that the output weight function (Step 3: Lines 6–8) is computed in $O(\tilde{n})$ time since $\mu_{\sim}(C_{[x]}) = \mu_{\mathcal{C}}((z, c_z))$ for any $z \in [x]$.

Finally it comes an $O(n^2 + \tilde{n}n)$ complexity. Let us put emphasis on the fact that it is actually the use of the ZPC-structure that allows us to get this quadratic complexity.

Theorem 4.5. Let E be a regular $\mathbb{K}$-expression. The Algorithm *AlgoKExptoEq* computes the equation $\mathbb{K}$-automaton of E with a time complexity $O(|Q_{\mathcal{C}}^2| + |Q_{\sim}||Q_{\mathcal{C}}|)$, that is a quadratic time complexity w.r.t. the size of E.

5. Conclusion

The algorithm we described for converting a regular $\mathbb{K}$-expression into its equation $\mathbb{K}$-automaton makes complete the general approach based on the notion of ZPC-structure and of c-continuation computation that we already used to handle boolean constructions as well as the one of the position $\mathbb{K}$-automaton. Its main advantadge is its robustness: it is straightforwardly deduced from our algorithm for constructing the equation automaton. The role of the set of c-continuations is easy to understand, its computation and its partitioning are well-studied procedures, leading to a quadratic time complexity in both boolean and weighted cases.

References

[1] C. Allauzen and M. Mohri. A Unified Construction of the Glushkov, Follow, and Antimirov Automata. in MFCS'2006, *Lecture Notes in Computer Science*, R. Kralovic and P. Urzyczyn eds., 4162:110-121, 2006.

[2] V. Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *Theoret. Comput. Sci.*, 155:291-319, 1996.

[3] A. Brüggemann-Klein, Regular Expressions into Finite Automata. *Theoret. Comput. Sci.*, 120(1993), 197–213.

[4] J. Berstel and C. Reutenauer, Les séries rationnelles et leurs langages, *Études et recherches en informatique*. Masson, Paris, 1984. English version: Rational series and their languages, Springer, 1988.

[5] P. Caron and M. Flouret, Glushkov construction for series : The non commutative case, *Intern. Journ. Comput. Maths*, 80(4): 457-472, 2003.

[6] J.-M. Champarnaud, E. Laugerotte, F. Ouardi, and D. Ziadi, From Regular Weighted Expressions to Finite Automata, *Intern. Journ. of Found. Comput. Sci.*, (15)5: 687-700, 2004.

[7] J.-M. Champarnaud, F. Nicart and D. Ziadi. From the $\mathcal{ZPC}$ -structure of a regular expression to its follow automaton, *Intern. Journ. of Alg. and Comp.*, 16(1): pp 17-34, 2006.

[8] J.-M. Champarnaud, F. Ouardi and D. Ziadi. Follow automaton versus equation automaton, In: DCFS'2004, *Descriptional Complexity of Formal Systems Workshop*, Proceedings, L. Ilie and D. Wotschke (eds.), pp. 145-153, 2004.

[9] J.-M. Champarnaud, F. Ouardi and D. Ziadi. Normalized expressions and finite automata, *Intern. Journ. of Alg. and Comp.*, 17–1(2007), 141–154.

[10] J.-M. Champarnaud and D. Ziadi. From Mirkin's Prebases to Antimirov's Word Partial Derivatives. *Informatica Fundamentae*, 45(3):195-205, 2001.

[11] J.-M. Champarnaud and D. Ziadi. Canonical derivatives and finite automaton constructions, *Theoret. Compt. Sci.*, 289:137-163, 2002.

[12] J.-M. Champarnaud and D. Ziadi. From c-continuations to new quadratic algorithms for automaton synthesis, *Intern. Journ. of Alg. and Comp.*, 11(6), 707-735, 2001.

[13] J.-M. Champarnaud, F. Ouardi and D. Ziadi. An efficient computation of the equation K-automaton of a regular K-expression, in DLT'2007, *Lecture Notes in Computer Science*, T. Harju, J. Karhumäki and A. Lepistö eds., Springer-Verlag Berlin Heidelberg, 4588(2007), 145–156.

[14] C.-H. Chang and R. Paige. From Regular Expressions to DFA's Using Compressed NFA's, *Theoret. Comput. Sci.*, 178, 1-36, 1997.

[15] T. Claveirole, S. Lombardy, S. O'Connor, L.-N. Pouchet, and J. Sakarovitch, Inside Vaucanson, Proc. of CIAA'2005 (J. Farré, I. Litovsky and S. Schmitz eds.), *Lect. Notes in Comp. Sci.*, 3845, Springer (2006), 116-128.

[16] V.-M. Glushkov. The abstract theory of automata, *Russian Mathematical Surveys*, 16, 1-53, 1961.

[17] U. Hebisch and H. J. Weinert. Semirings: algebraic theory and applications in computer science, *World Scientific*, Singapore, 1993.

[18] L. Ilie and S. Yu. Follow automata, *Information and computation*, 186, 140-162, 2003.

[19] W. Kuich and A. Salomaa. Semirings, automata, languages. *EATCS Monographs on Theoretical Computer Science*, Volume 5. Springer-Verlag, Berlin, 1986. Princeton U. Press.

[20] S. Lombardy and J. Sakarovitch. Derivations of Rational Expressions with Multiplicity. MFCS'2002, K. Diks and W. Ritter, eds., *Lect. Notes in Comp. Sci.*, Springer, 2420: 471-482, 2002.

[21] S. Lombardy and J. Sakarovitch. Derivatives of Rational Expressions with Multiplicity. *Theoret. Comput. Sci.*, 332: 141-177, 2005.

[22] R. F. McNaughton and H. Yamada, Regular expressions and state graphs for automata, *IEEE Trans. Electronic Comput.* 9: 39-47, 1960.

[23] B. G. Mirkin. An algorithm for constructing a base in a language of regular expressions. *Engineering Cybernetics*, 5:110-116, 1966.

[24] J. Sakarovitch. Éléments de la théorie des automates. *Les classiques de l'informatique.*, Vuibert Paris, 2003.

[25] M. P. Schützenberger. On the definition of a family of automata. *Information and control*, 6:245-270, (1961).

[26] K. Thompson. Regular expression search algorithm, *Comm. ACM*, 11, 6, 419-422, 1968.

[27] D. Ziadi, J.-L. Ponty, and J.-M. Champarnaud. Passage d'une expression rationnelle à un automate fini non-déterministe. *Bull. Belg. Math. Soc.*, 4:177–203, 1997.

[28] D. Ziadi, Quelques aspects théoriques et algorithmiques des automates. *Thèse d'habilitation à diriger des recherches*, Université de Rouen, 2002.