Partial Derivative Automaton for Regular Expressions with Shuffle

Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis[⋆]

CMUP & DCC, Faculdade de Ciências da Universidade do Porto, Portugal
Rua do Campo Alegre, 4169-007 Porto, Portugal
sbb@dcc.fc.up.pt,ajmachia@fc.up.pt,{nam,rvr}@dcc.fc.up.pt

Abstract. We generalize the partial derivative automaton to regular expressions with shuffle and study its size in the worst and in the average case. The number of states of the partial derivative automata is in the worst case at most 2^m, where m is the number of letters in the expression, while asymptotically and on average it is no more than $(\frac{4}{3})^m$.

1 Introduction

The class of regular languages is closed under shuffle (or interleaving operation), and extended regular expressions with shuffle can be much more succinct than the equivalent ones with disjunction, concatenation, and star operators. For the shuffle operation, Mayer and Stockmeyer [14] studied the computational complexity of membership and inequivalence problems. Inequivalence is exponential-time-complete, and membership is NP-complete for some classes of regular languages. In particular, they showed that for regular expressions (REs) with shuffle, of size n, an equivalent nondeterministic finite automaton (NFA) needs at most 2^n states, and presented a family of REs with shuffle, of size $\mathcal{O}(n)$, for which the corresponding NFAs have at least 2^n states. Gelade [10], and Gruber and Holzer [12,11] showed that there exists a double exponential trade-off in the translation from REs with shuffle to stantard REs. Gelade also gave a tight double exponential upper bound for the translation of REs with shuffle to DFAs. Recently, conversions of shuffle expressions to finite automata were presented by Estrade *et al.* [7] and Kumar and Verma [13]. In the latter paper the authors give an algorithm for the construction of an ε-free NFA based on the classic Glushkov construction, and the authors claim that the size of the resulting automaton is at most 2^{m+1}, where m is the number of letters that occur in the RE with shuffle.

In this paper we present a conversion of REs with shuffle to ε-free NFAs, by generalizing the partial derivative construction for standard REs [1,15]. For standard REs, the partial derivative automaton ($\mathcal{A}_{pd}$) is a quotient of the Glushkov automaton ($\mathcal{A}_{pos}$), and Broda *et al.* [2,3] showed that, asymptotically and on average, the size of $\mathcal{A}_{pd}$ is half the size of $\mathcal{A}_{pos}$. In the case of REs with shuffle

[⋆] Authors partially funded by the European Regional Development Fund through the programme COMPETE and by the Portuguese Government through the FCT under projects UID/MAT/00144/2013 and FCOMP-01-0124-FEDER-020486.

we show that the number of states of the partial derivative automaton is, in the worst-case, 2^m (with m as before) and an upper bound for the average size is, asymptotically, $(\frac{4}{3})^m$.

This paper is organized as follows. In the next section we review the shuffle operation and regular expressions with shuffle. In Section 3 we consider equation systems, for languages and expressions, associated with nondeterministic finite automata and define a solution for a system of equations for a shuffle expression. An alternative and equivalent construction, denoted by $\mathcal{A}_{pd}$, is given in Section 4 using the notion of partial derivative. An upper bound for the average number of states of $\mathcal{A}_{pd}$ using the framework of analytic combinatorics is given in Section 5. We conclude in Section 6 with some considerations about how to improve the presented upper bound and related future work.

2 Regular Expressions with Shuffle

Given an alphabet Σ, the shuffle of two words in $\Sigma^\star$ is a finite set of words defined inductively as follows, for $x, y \in \Sigma^\star$ and $a, b \in \Sigma$

$$\begin{aligned} x \sqcup\!\sqcup\, \varepsilon = \varepsilon \sqcup\!\sqcup\, x = \{x\} \\ ax \sqcup\!\sqcup\, by = \{\, az \mid z \in x \sqcup\!\sqcup\, by \,\} \cup \{\, bz \mid z \in ax \sqcup\!\sqcup\, y \,\}. \end{aligned}$$

This definition is extended to sets of words, i.e., languages, in the natural way:

$$L_1 \sqcup\!\sqcup\, L_2 = \{\, x \sqcup\!\sqcup\, y \mid x \in L_1, y \in L_2 \,\}.$$

It is well known that if two languages $L_1, L_2 \subseteq \Sigma^\star$ are regular then $L_1 \sqcup\!\sqcup\, L_2$ is regular. One can extend regular expressions to include the $\sqcup\!\sqcup$ operator. Given an alphabet Σ, we let $\mathsf{T}_{\sqcup\!\sqcup}$ denote the set containing $\emptyset$ plus all terms finitely generated from $\Sigma \cup \{\varepsilon\}$ and operators $+, \cdot, \sqcup\!\sqcup, ^*$, that is, the expressions τ generated by the grammar

$$\tau \rightarrow \emptyset \mid \alpha \tag{1}$$

$$\alpha \rightarrow \varepsilon \mid a \mid \alpha + \alpha \mid \alpha \cdot \alpha \mid \alpha \sqcup\!\sqcup\, \alpha \mid \alpha^\star \quad (a \in \Sigma). \tag{2}$$

As usual, the (regular) language $\mathcal{L}(\tau)$ represented by an expression $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$ is inductively defined as follows: $\mathcal{L}(\emptyset) = \emptyset$, $\mathcal{L}(\varepsilon) = \{\varepsilon\}$, $\mathcal{L}(a) = \{a\}$ for $a \in \Sigma$, $\mathcal{L}(\alpha^\star) = \mathcal{L}(\alpha)^\star$, $\mathcal{L}(\alpha + \beta) = \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$, $\mathcal{L}(\alpha\beta) = \mathcal{L}(\alpha)\mathcal{L}(\beta)$, and $\mathcal{L}(\alpha \sqcup\!\sqcup\, \beta) = \mathcal{L}(\alpha) \sqcup\!\sqcup\, \mathcal{L}(\beta)$. We say that two expressions $\tau_1, \tau_2 \in \mathsf{T}_{\sqcup\!\sqcup}$ are equivalent, and write $\tau_1 = \tau_2$, if $\mathcal{L}(\tau_1) = \mathcal{L}(\tau_2)$.

Example 1. Consider $\alpha_n = a_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup\, a_n$, where $n \geq 1$, $a_i \neq a_j$ for $1 \leq i \neq j \leq n$. Then,

$$\mathcal{L}(\alpha_n) = \{\, a_{i_1} \cdots a_{i_n} \mid i_1, \ldots, i_n \text{ is a permutation of } 1, \ldots, n\}.$$

We recall that standard regular expressions constitute a Kleene algebra and the shuffle operator $\sqcup\!\sqcup$ is commutative, associative, and distributes over $+$. It also follows that for all $a, b \in \Sigma$ and $\tau_1, \tau_2 \in \mathsf{T}_{\sqcup\!\sqcup}$,

$$a\tau_1 \sqcup\!\sqcup b\tau_2 = a(\tau_1 \sqcup\!\sqcup b\tau_2) + b(a\tau_1 \sqcup\!\sqcup \tau_2).$$

Given a language L, we define $\varepsilon(\tau) = \varepsilon(\mathcal{L}(\tau))$, where, $\varepsilon(L) = \varepsilon$ if $\varepsilon \in L$ and $\varepsilon(L) = \emptyset$ otherwise. A recursive definition of $\varepsilon : \mathsf{T}_{\sqcup\!\sqcup} \longrightarrow \{\emptyset, \varepsilon\}$ is given by the following: $\varepsilon(a) = \varepsilon(\emptyset) = \emptyset$, $\varepsilon(\varepsilon) = \varepsilon(\alpha^\star) = \varepsilon$, $\varepsilon(\alpha + \beta) = \varepsilon(\alpha) + \varepsilon(\beta)$, $\varepsilon(\alpha\beta) = \varepsilon(\alpha)\varepsilon(\beta)$, and $\varepsilon(\alpha \sqcup\!\sqcup \beta) = \varepsilon(\alpha)\varepsilon(\beta)$.

3 Automata and Systems of Equations

We first recall the definition of an NFA as a tuple $\mathcal{A} = \langle S, \Sigma, S_0, \delta, F\rangle$, where S is a finite set of states, Σ is a finite alphabet, $S_0 \subseteq S$ a set of initial states, $\delta : S \times \Sigma \longrightarrow \mathcal{P}(S)$ the transition function, and $F \subseteq S$ a set of final states. The extension of δ to sets of states and words is defined by $\delta(X, \varepsilon) = X$ and $\delta(X, ax) = \delta(\cup_{s \in X}\delta(s, a), x)$. A word $x \in \Sigma^\star$ is accepted by $\mathcal{A}$ if and only if $\delta(S_0, x) \cap F \neq \emptyset$. The *language of* $\mathcal{A}$ is the set of words accepted by $\mathcal{A}$ and denoted by $\mathcal{L}(\mathcal{A})$. The *right language of a state* s, denoted by $\mathcal{L}_s$, is the language accepted by $\mathcal{A}$ if we take $S_0 = \{s\}$. The class of languages accepted by all the NFAs is precisely the set of regular languages.

It is well known that, for each n-state NFA $\mathcal{A}$ over $\Sigma = \{a_1, \dots, a_k\}$, where $S = [1, n]$, having right languages $\mathcal{L}_1, \dots, \mathcal{L}_n$, it is possible to associate a system of linear language equations

$$\mathcal{L}_i = a_1\mathcal{L}_{1i} \cup \cdots \cup a_k\mathcal{L}_{ik} \cup \varepsilon(\mathcal{L}_i), \quad i \in [1, n]$$

where each $\mathcal{L}_{ij}$ is a (possibly empty) union of elements in $\{\mathcal{L}_1, \dots, \mathcal{L}_n\}$, and $\mathcal{L}(\mathcal{A}) = \bigcup_{i \in S_0} \mathcal{L}_i$.

In the same way, it is possible to associate with each regular expression a system of equations on expressions. Here, we extend this notion to regular expressions with shuffle.

Definition 2. *Consider* $\Sigma = \{a_1, \dots, a_k\}$ *and* $\alpha_0 \in \mathsf{T}_{\sqcup\!\sqcup}$. *A* support *of* α_0 *is a set* $\{\alpha_1, \dots, \alpha_n\}$ *that satisfies a system of equations*

$$\alpha_i = a_1\alpha_{1i} + \cdots + a_k\alpha_{ki} + \varepsilon(\alpha_i), \quad i \in [0, n] \tag{3}$$

for some $\alpha_{1i}, \dots, \alpha_{ki}$, *each one a (possibly empty) sum of elements in* $\{\alpha_1, \dots, \alpha_n\}$. *In this case* $\{\alpha_0, \alpha_1, \dots, \alpha_n\}$ *is called a* prebase *of* α_0.

It is clear from what was just said above, that the existence of a support of α implies the existence of an NFA that accepts the language determined by α.

Note that the system of equations (3) can be written in matrix form $\mathsf{A}_\alpha = \mathsf{C} \cdot \mathsf{M}_\alpha + \mathsf{E}_\alpha$, where M_α is the $k \times (n+1)$ matrix with entries α_{ij}, and A_α, C and E_α denote respectively the following three matrices,

$$\mathsf{A}_\alpha = \begin{bmatrix}\alpha_0 \cdots \alpha_n\end{bmatrix}, \quad \mathsf{C} = \begin{bmatrix}a_1 \cdots a_k\end{bmatrix}, \quad \text{and} \quad \mathsf{E}_\alpha = \begin{bmatrix}\varepsilon(\alpha_0) \cdots \varepsilon(\alpha_n)\end{bmatrix},$$

where, $\mathsf{C}\cdot\mathsf{M}_\alpha$ denotes the matrix obtained from C and M_α applying the standard rules of matrix multiplication, but replacing the multiplication by concatenation. This notation will be used below.

A support for an expression $\alpha \in \mathsf{T}_⧢$ can be computed using the function $\pi : \mathsf{T}_⧢ \longrightarrow \mathcal{P}(\mathsf{T}_⧢)$ recursively given by the following.

Definition 3. *Given $\tau \in \mathsf{T}_⧢$, the set $\pi(\tau)$ is inductively defined by,*

$$\begin{aligned} \pi(\emptyset) = \pi(\varepsilon) &= \emptyset & \pi(\alpha+\beta) &= \pi(\alpha)\cup\pi(\beta)\\ \pi(a) &= \{\varepsilon\} \quad (a\in\Sigma) & \pi(\alpha\beta) &= \pi(\alpha)\beta\cup\pi(\beta)\\ \pi(\alpha^\star) &= \pi(\alpha)\alpha^\star & \pi(\alpha ⧢ \beta) &= \pi(\alpha) ⧢ \pi(\beta)\\ && &\cup\, \pi(\alpha) ⧢ \{\beta\} \cup \{\alpha\} ⧢ \pi(\beta), \end{aligned}$$

where, given $S,T \subseteq \mathsf{T}_⧢$ and $\beta \in \mathsf{T}_⧢ \setminus \{\emptyset,\varepsilon\}$, $S\beta = \{\ \alpha\beta \mid \alpha \in S\ \}$ and $S ⧢ T = \{\ \alpha ⧢ \beta \mid \alpha \in S, \beta \in T\ \}$, $S\varepsilon = \{\varepsilon\} ⧢ S = S ⧢ \{\varepsilon\} = S$, and $S\emptyset = \emptyset S = \emptyset$.

The following lemma follows directly from the definitions and will be used in the proof of Proposition 5.

Lemma 4. *If $\alpha,\beta \in \mathsf{T}_⧢$, then $\varepsilon(\beta)\cdot\mathcal{L}(\alpha) \subseteq \mathcal{L}(\alpha ⧢ \beta)$.*

Proposition 5. *If $\alpha \in \mathsf{T}_⧢$, then the set $\pi(\alpha)$ is a support of α.*

Proof. We proceed by induction on the structure of α. Excluding the case where α is $\alpha_0 ⧢ \beta_0$, the proof can be found in [15,6]. We now describe how to obtain a system of equations corresponding to an expression $\alpha_0 ⧢ \beta_0$ from systems for α_0 and β_0. Suppose that $\pi(\alpha_0) = \{\alpha_1,\dots,\alpha_n\}$ is a support of α_0 and $\pi(\beta_0) = \{\beta_1,\dots,\beta_m\}$ is a support of β_0. For α_0 and β_0 consider C, A_{α_0}, M_{α_0}, E_{α_0} and A_{β_0}, M_{β_0}, E_{β_0} as above. We wish to show that

$$\begin{aligned}\pi(\alpha_0 ⧢ \beta_0) = \{&\alpha_1 ⧢ \beta_1,\dots,\alpha_1 ⧢ \beta_m,\dots,\alpha_n ⧢ \beta_1,\dots,\alpha_n ⧢ \beta_m\}\\ &\cup \{\alpha_1 ⧢ \beta_0,\dots,\alpha_n ⧢ \beta_0\} \cup \{\alpha_0 ⧢ \beta_1,\dots,\alpha_0 ⧢ \beta_m\}\end{aligned}$$

is a support of $\alpha_0 ⧢ \beta_0$. Let $\mathsf{A}_{\alpha_0 ⧢ \beta_0}$ be the $(n+1)(m+1)$-entry row-matrix whose entires are

$$\begin{bmatrix}\alpha_0 ⧢ \beta_0 & \alpha_1 ⧢ \beta_1 \cdots \alpha_n ⧢ \beta_m & \alpha_1 ⧢ \beta_0 \cdots \alpha_n ⧢ \beta_0 & \alpha_0 ⧢ \beta_1 \cdots \alpha_0 ⧢ \beta_m\end{bmatrix}.$$

Then, $\mathsf{E}_{\alpha_0 ⧢ \beta_0}$ is defined as usual, i.e. containing the values of $\varepsilon(\alpha)$ for all entries α in $\mathsf{A}_{\alpha_0 ⧢ \beta_0}$.

Finally, let $\mathsf{M}_{\alpha_0 ⧢ \beta_0}$ be the $k \times (n+1)(m+1)$ matrix whose entries $\gamma_{l,(i,j)}$, for $l \in [1,k]$ and $(i,j) \in [0,n]\times[0,m]$, are defined by

$$\gamma_{l,(i,j)} = \alpha_{li} ⧢ \beta_j + \alpha_i ⧢ \beta_{lj}.$$

Note that, since by the induction hypothesis each α_{li} is a sum of elements in $\pi(\alpha)$ and each β_{lj} is a sum of elements in $\pi(\beta)$, after applying distributivity of $⧢$ over $+$ each element of $\mathsf{M}_{\alpha_0 ⧢ \beta_0}$ is in fact a sum of elements in $\pi(\alpha_0 ⧢ \beta_0)$.

We will show that $\mathsf{A}_{\alpha_0 ⧢ \beta_0} = \mathsf{C} \cdot \mathsf{M}_{\alpha_0 ⧢ \beta_0} + \mathsf{E}_{\alpha_0 ⧢ \beta_0}$. For this, consider $\alpha_i ⧢ \beta_j$ for some $(i,j) \in [0,n] \times [0,m]$. We have $\alpha_i = a_1\alpha_{1i} + \cdots + a_k\alpha_{ki} + \varepsilon(\alpha_i)$ and $\beta_j = a_1\beta_{1j} + \cdots + a_k\beta_{kj} + \varepsilon(\beta_j)$. Consequently, using properties of ⧢, namely distributivity over $+$, as well as Lemma 4,

$$\begin{aligned}
\alpha_i ⧢ \beta_j &= (a_1\alpha_{1i} + \cdots + a_k\alpha_{ki} + \varepsilon(\alpha_i)) ⧢ (a_1\beta_{1j} + \cdots + a_k\beta_{kj} + \varepsilon(\beta_j)) \\
&= a_1 \left(\alpha_{1i} ⧢ \beta_j + \alpha_i ⧢ \beta_{1j} + \varepsilon(\beta_j)\alpha_{1i} + \varepsilon(\alpha_i)\beta_{1j}\right) \ + \ \cdots \ + \\
&\quad\ a_k \left(\alpha_{ki} ⧢ \beta_j + \alpha_i ⧢ \beta_{kj} + \varepsilon(\beta_j)\alpha_{ki} + \varepsilon(\alpha_i)\beta_{kj}\right) + \varepsilon(\alpha_i ⧢ \beta_j) \\
&= a_1 \left(\alpha_{1i} ⧢ \beta_j + \alpha_i ⧢ \beta_{1j}\right) \ + \ \cdots \ + \\
&\quad\ a_k \left(\alpha_{ki} ⧢ \beta_j + \alpha_i ⧢ \beta_{kj}\right) + \varepsilon(\alpha_i ⧢ \beta_j) \\
&= a_1\gamma_{1,(i,j)} + \cdots + a_k\gamma_{k,(i,j)} + \varepsilon(\alpha_i ⧢ \beta_j).
\end{aligned}$$

□

It is clear from its definition that $\pi(\alpha)$ is finite. In the following proposition, an upper bound for the size of $\pi(\alpha)$ is given. Example 7 is a witness that this upper bound is tight.

Proposition 6. *Given $\alpha \in \mathsf{T}_{⧢}$, one has $|\pi(\alpha)| \leq 2^{|\alpha|_\Sigma} - 1$, where $|\alpha|_\Sigma$ denotes the number of alphabet symbols in α.*

Proof. The proof proceeds by induction on the structure of α. It is clear that the result holds for $\alpha = \emptyset$, $\alpha = \varepsilon$ and for $\alpha = a \in \Sigma$. Now, suppose the claim is true for α and β. There are four induction cases to consider. We will make use of the fact that, for $m, n \geq 0$ one has $2^m + 2^n - 2 \leq 2^{m+n} - 1$. For $\alpha^\star$, one has $|\pi(\alpha^\star)| = |\pi(\alpha)\alpha^\star| = |\pi(\alpha)| \leq 2^{|\alpha|_\Sigma} - 1 = 2^{|\alpha^\star|_\Sigma} - 1$. For $\alpha + \beta$, one has $|\pi(\alpha+\beta)| = |\pi(\alpha) \cup \pi(\beta)| \leq 2^{|\alpha|_\Sigma} - 1 + 2^{|\beta|_\Sigma} - 1 \leq 2^{|\alpha|_\Sigma + |\beta|_\Sigma} - 1 = 2^{|\alpha+\beta|_\Sigma} - 1$. For $\alpha\beta$, one has $|\pi(\alpha\beta)| = |\pi(\alpha)\beta \cup \pi(\beta)| \leq 2^{|\alpha|_\Sigma} - 1 + 2^{|\beta|_\Sigma} - 1 \leq 2^{|\alpha\beta|_\Sigma} - 1$. Finally, for $\alpha ⧢ \beta$, one has $|\pi(\alpha ⧢ \beta)| = |\pi(\alpha) ⧢ \pi(\beta) \cup \pi(\alpha) ⧢ \{\beta\} \cup \{\alpha\} ⧢ \pi(\beta)| \leq (2^{|\alpha|_\Sigma} - 1)(2^{|\beta|_\Sigma} - 1) + 2^{|\alpha|_\Sigma} - 1 + 2^{|\beta|_\Sigma} - 1 = 2^{|\alpha|_\Sigma + |\beta|_\Sigma} - 1 = 2^{|\alpha ⧢ \beta|_\Sigma} - 1$. □

Example 7. Considering $\alpha_n = a_1 ⧢ \cdots ⧢ a_n$, where $n \geq 1$, $a_i \neq a_j$ for $1 \leq i \neq j \leq n$ again, one has

$$|\pi(\alpha_n)| = |\{ \underset{i \in I}{⧢} a_i \mid I \subsetneq \{1, \ldots, n\} \}| = 2^n - 1,$$

where by convention $\underset{i \in \emptyset}{⧢} a_i = \varepsilon$.

The proof of Proposition 5 gives a way to construct a system of equations for an expression $\tau \in \mathsf{T}_{⧢}$, corresponding to an NFA that accepts the language represented by τ. This is done by recursively computing $\pi(\tau)$ and the matrices A_τ and E_τ, obtaining the whole NFA in the final step.

In the next section we will show how to build the same NFA in a more efficient way using the notion of partial derivative.

4 Partial Derivatives

Recall that the *left quotient* of a language L w.r.t. a symbol $a \in \Sigma$ is

$$a^{-1}L = \{\ x \mid ax \in L\ \}.$$

The left quotient of L w.r.t. a word $x \in \Sigma^\star$ is then inductively defined by $\varepsilon^{-1}L = L$ and $(xa)^{-1}L = a^{-1}(x^{-1}L)$. Note that for $L_1, L_2 \subseteq \Sigma^\star$ and $a, b \in \Sigma$ the shuffle operation satisfies $a^{-1}(L_1 ⧢ L_2) = (a^{-1}L_1) ⧢ L_2\ \cup\ L_1 ⧢ (a^{-1}L_2)$.

Definition 8. *The set of partial derivatives of a term $\tau \in \mathsf{T}_{⧢}$ w.r.t. a letter $a \in \Sigma$, denoted by $\partial_a(\tau)$, is inductively defined by*

$$\partial_a(\emptyset) = \partial_a(\varepsilon) = \emptyset \qquad \partial_a(b) = \begin{cases} \{\varepsilon\} & \text{if } b = a \\ \emptyset & \text{otherwise} \end{cases}$$

$$\begin{aligned} \partial_a(\alpha^*) &= \partial_a(\alpha)\alpha^* \\ \partial_a(\alpha + \beta) &= \partial_a(\alpha)\ \cup\ \partial_a(\beta) \\ \partial_a(\alpha\beta) &= \partial_a(\alpha)\beta\ \cup\ \varepsilon(\alpha)\partial_a(\beta) \\ \partial_a(\alpha ⧢ \beta) &= \partial_a(\alpha) ⧢ \{\beta\} \cup \{\alpha\} ⧢ \partial_a(\beta). \end{aligned}$$

The set of partial derivatives of $\tau \in \mathsf{T}_{⧢}$ w.r.t. a word $x \in \Sigma^*$ is inductively defined by $\partial_\varepsilon(\tau) = \{\tau\}$ and $\partial_{xa}(\tau) = \partial_a(\partial_x(\tau))$, where, given a set $S \subseteq \mathsf{T}_{⧢}$, $\partial_a(S) = \bigcup_{\tau \in S} \partial_a(\tau)$.

We let $\partial(\tau)$ denote the set of all partial derivatives of an expression τ, i.e. $\partial(\tau) = \bigcup_{x \in \Sigma^*} \partial_x(\tau)$, and by $\partial^+(\tau)$ the set of partial derivatives excluding the trivial derivative by ε, i.e. $\partial^+(\tau) = \bigcup_{x \in \Sigma^+} \partial_x(\tau)$. Given a set $S \subseteq \mathsf{T}_{⧢}$, we define $\mathcal{L}(S) = \bigcup_{\tau \in S} \mathcal{L}(\tau)$. The following result has a straightforward proof.

Proposition 9. *Given $x \in \Sigma^\star$ and $\tau \in \mathsf{T}_{⧢}$, one has $\mathcal{L}(\partial_x(\tau)) = x^{-1}\mathcal{L}(\tau)$.*

The following properties of $\partial^+(\tau)$ will be used in the proof of Proposition 11.

Lemma 10. *For $\tau \in \mathsf{T}_{⧢}$, the following hold.*

1. *If $\partial^+(\tau) \neq \emptyset$, then there is $\alpha_0 \in \partial^+(\tau)$ with $\varepsilon(\alpha_0) = \varepsilon$.*
2. *If $\partial^+(\tau) = \emptyset$ and $\tau \neq \emptyset$, then $\mathcal{L}(\tau) = \{\varepsilon\}$ and $\varepsilon(\tau) = \varepsilon$.*

Proof. 1. From the grammar rule (2) it follows that $\emptyset$ cannot appear as a subexpression of a larger term. Suppose that there is some $\gamma \in \partial^+(\tau)$. We conclude, from Definition 8 and from the previous remark, that there is some word $x \in \Sigma^+$ such that $x \in \mathcal{L}(\gamma)$. This is equivalent to $\varepsilon \in \mathcal{L}(\partial_x(\gamma))$, which means that there is some $\alpha_0 \in \partial_x(\gamma) \subseteq \partial^+(\tau)$ such that $\varepsilon(\alpha_0) = \varepsilon$.

2. $\partial^+(\tau) = \emptyset$ implies that $\partial_x(\tau) = \emptyset$ for all $x \in \Sigma^+$. Thus, $\mathcal{L}(\partial_x(\tau)) = \{\ y \mid xy \in \mathcal{L}(\tau)\ \} = \emptyset$, and consequently there is no word $z \in \Sigma^+$ in $\mathcal{L}(\tau)$. On the other hand, since $\emptyset$ does not appear in τ, it follows that $\mathcal{L}(\tau) \neq \emptyset$. Thus, $\mathcal{L}(\tau) = \{\varepsilon\}$. □

Proposition 11. *∂^+ satisfies the following:*

$$\begin{aligned} \partial^+(\emptyset) &= \partial^+(\varepsilon) = \emptyset & \partial^+(\alpha+\beta) &= \partial^+(\alpha) \cup \partial^+(\beta)\\ \partial^+(a) &= \{\varepsilon\} \quad (a \in \Sigma) & \partial^+(\alpha\beta) &= \partial^+(\alpha)\beta \cup \partial^+(\beta)\\ \partial^+(\alpha^*) &= \partial^+(\alpha)\alpha^* & \partial^+(\alpha \shuffle \beta) &= \partial^+(\alpha) \shuffle \partial^+(\beta)\\ & & &\cup \partial^+(\alpha) \shuffle \{\beta\} \cup \{\alpha\} \shuffle \partial^+(\beta). \end{aligned}$$

Proof. The proof proceeds by induction on the structure of α. It is clear that $\partial^+(\emptyset) = \emptyset$, $\partial^+(\varepsilon) = \emptyset$ and, for $a \in \Sigma$, $\partial^+(a) = \{\varepsilon\}$.

In the remaining cases, to prove that an inclusion $\partial^+(\gamma) \subseteq E$ holds for some expression E, we show by induction on the length of x that for every $x \in \Sigma^+$ one has $\partial_x(\gamma) \subseteq E$. We will therefore just indicate the corresponding computations for $\partial_a(\gamma)$ and $\partial_{xa}(\gamma)$, for $a \in \Sigma$. We also make use of the fact that, for any expression γ and letter $a \in \Sigma$, the set $\partial^+(\gamma)$ is closed for taking derivatives w.r.t. a, i.e., $\partial_a(\partial^+(\gamma)) \subseteq \partial^+(\gamma)$.

Now, suppose the claim is true for α and β. There are four induction cases to consider.

- For $\alpha+\beta$, we have $\partial_a(\alpha+\beta) = \partial_a(\alpha) + \partial_a(\beta) \ \subseteq \partial^+(\alpha) \cup \partial^+(\beta)$, as well as $\partial_{xa}(\alpha+\beta) = \partial_a(\partial_x(\alpha+\beta)) \subseteq \partial_a(\partial^+(\alpha)\cup\partial^+(\beta)) \subseteq \partial_a(\partial^+(\alpha))\cup\partial_a(\partial^+(\beta)) \subseteq \partial^+(\alpha) \cup \partial^+(\beta)$. Similarly, one proves that $\partial_x(\alpha) \in \partial^+(\alpha+\beta)$ and $\partial_x(\beta) \in \partial^+(\alpha+\beta)$, for all $x \in \Sigma^+$.
- For $\alpha^\star$, we have $\partial_a(\alpha^*) = \partial_a(\alpha)\alpha^* \subseteq \partial^+(\alpha)\alpha^*$, as well as

  $$\begin{aligned} \partial_{xa}(\alpha^*) = \partial_a(\partial_x(\alpha^*)) &\subseteq \partial_a(\partial^+(\alpha)\alpha^*) \subseteq \partial_a(\partial^+(\alpha))\alpha^* \cup \partial_a(\alpha^*)\\ &\subseteq \partial^+(\alpha)\alpha^* \cup \partial_a(\alpha)\alpha^* \subseteq \partial^+(\alpha)\alpha^*. \end{aligned}$$

  Furthermore, $\partial_a(\alpha)\alpha^\star = \partial_a(\alpha^\star) \subseteq \partial^+(\alpha^\star)$ and $\partial_{xa}(\alpha)\alpha^\star = \partial_a(\partial_x(\alpha))\alpha^\star \subseteq \partial_a(\partial_x(\alpha)\alpha^\star) \subseteq \partial_a(\partial^+(\alpha^\star)) \subseteq \partial^+(\alpha^\star)$.
- For $\alpha\beta$, we have $\partial_a(\alpha\beta) = \partial_a(\alpha)\beta \cup \varepsilon(\alpha)\partial_a(\beta) \subseteq \partial^+(\alpha)\beta \cup \partial^+(\beta)$ and

  $$\begin{aligned} \partial_{xa}(\alpha\beta) = \partial_a(\partial_x(\alpha\beta)) &\subseteq \partial_a(\partial^+(\alpha)\beta \cup \partial^+(\beta)) = \partial_a(\partial^+(\alpha)\beta) \cup \partial_a(\partial^+(\beta))\\ &\subseteq \partial_a(\partial^+(\alpha))\beta \cup \partial_a(\beta) \cup \partial_a(\partial^+(\beta)) \subseteq \partial^+(\alpha)\beta \cup \partial^+(\beta). \end{aligned}$$

  Also, $\partial_a(\alpha)\beta \subseteq \partial_a(\alpha\beta) \subseteq \partial^+(\alpha\beta)$ and

  $$\partial_{xa}(\alpha)\beta = \partial_a(\partial_x(\alpha))\beta \subseteq \partial_a(\partial_x(\alpha)\beta) \subseteq \partial_a(\partial^+(\alpha\beta)) \subseteq \partial^+(\alpha\beta).$$

  Finally, if $\varepsilon(\alpha) = \varepsilon$, then $\partial_a(\beta) \subseteq \partial_a(\alpha\beta)$ and $\partial_{xa}(\beta) = \partial_a(\partial_x(\beta)) \subseteq \partial_a(\partial_x(\alpha\beta)) = \partial_{xa}(\alpha\beta)$. We conclude that $\partial_x(\beta) \subseteq \partial_x(\alpha\beta)$ for all $x \in \Sigma^+$, and therefore $\partial^+(\beta) \subseteq \partial^+(\alpha\beta)$. Otherwise, $\varepsilon(\alpha) = \emptyset$, and it follows from Lemma 10 that $\partial^+(\alpha) \neq \emptyset$, and that there is some $\alpha_0 \in \partial^+(\alpha)$ with $\varepsilon(\alpha_0) = \emptyset$. As above, this implies that $\partial_x(\beta) \subseteq \partial_x(\alpha_0\beta)$ for all $x \in \Sigma^+$. On the other hand, have already shown that $\partial^+(\alpha)\beta \subseteq \partial^+(\alpha\beta)$. In particular, $\alpha_0\beta \in \partial^+(\alpha\beta)$. From these two facts, we conclude that $\partial_x(\beta) \subseteq \partial_x(\alpha_0\beta) \subseteq \partial_x(\partial^+(\alpha\beta)) \subseteq \partial^+(\alpha\beta)$, which finishes the proof for the case of concatenation.

– For $\alpha \sqcup\!\sqcup \beta$, we have

$$\begin{aligned}\partial_a(\alpha \sqcup\!\sqcup \beta) &= \partial_a(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \partial_a(\beta)\\ &\subseteq \partial^+(\alpha) \sqcup\!\sqcup \partial^+(\beta) \cup \partial^+(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \partial^+(\beta)\end{aligned}$$

and

$$\begin{aligned}\partial_{xa}(\alpha \sqcup\!\sqcup \beta) &\subseteq \partial_a(\partial^+(\alpha) \sqcup\!\sqcup \partial^+(\beta) \cup \partial^+(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \partial^+(\beta))\\ &= \partial_a(\partial^+(\alpha) \sqcup\!\sqcup \partial^+(\beta)) \cup \partial_a(\partial^+(\alpha) \sqcup\!\sqcup \{\beta\}) \cup \partial_a(\{\alpha\} \sqcup\!\sqcup \partial^+(\beta))\\ &= \partial_a(\partial^+(\alpha)) \sqcup\!\sqcup \partial^+(\beta) \cup \partial^+(\alpha) \sqcup\!\sqcup \partial_a(\partial^+(\beta)) \cup \partial_a(\partial^+(\alpha)) \sqcup\!\sqcup \{\beta\}\\ &\quad \cup \partial^+(\alpha) \sqcup\!\sqcup \partial_a(\beta) \cup \partial_a(\alpha) \sqcup\!\sqcup \partial^+(\beta) \cup \{\alpha\} \sqcup\!\sqcup \partial_a(\partial^+(\beta))\\ &\subseteq \partial^+(\alpha) \sqcup\!\sqcup \partial^+(\beta) \cup \partial^+(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \partial^+(\beta).\end{aligned}$$

Now we prove that for all $x \in \Sigma^+$, one has $\partial_x(\alpha) \sqcup\!\sqcup \{\beta\} \subseteq \partial_x(\alpha \sqcup\!\sqcup \beta)$, which implies $\partial^+(\alpha) \sqcup\!\sqcup \{\beta\} \subseteq \partial^+(\alpha \sqcup\!\sqcup \beta)$. In fact, we have $\partial_a(\alpha) \sqcup\!\sqcup \{\beta\} \subseteq \partial_a(\alpha \sqcup\!\sqcup \beta)$ and

$$\begin{aligned}\partial_{xa}(\alpha) \sqcup\!\sqcup \{\beta\} &\subseteq \partial_a(\partial_x(\alpha)) \sqcup\!\sqcup \{\beta\}\\ &\subseteq \partial_a(\partial_x(\alpha) \sqcup\!\sqcup \{\beta\}) \subseteq \partial_a(\partial_x(\alpha \sqcup\!\sqcup \beta)) = \partial_{xa}(\alpha \sqcup\!\sqcup \beta).\end{aligned}$$

Showing that $\{\alpha\} \sqcup\!\sqcup \partial_x(\beta) \subseteq \partial_x(\alpha \sqcup\!\sqcup \beta)$ is analogous. Finally, for $x, y \in \Sigma^+$ we have $\partial_x(\alpha) \sqcup\!\sqcup \partial_y(\beta) \subseteq \partial_y(\partial_x(\alpha) \sqcup\!\sqcup \{\beta\}) \subseteq \partial_y(\partial_x(\alpha \sqcup\!\sqcup \beta)) = \partial_{xy}(\alpha \sqcup\!\sqcup \beta) \subseteq \partial^+(\alpha \sqcup\!\sqcup \beta)$. □

Corollary 12. *Given $\alpha \in \mathsf{T}_{\sqcup\!\sqcup}$, one has $\partial^+(\alpha) = \pi(\alpha)$.*

We conclude that $\partial(\alpha)$ corresponds to the set $\{\alpha\} \cup \pi(\alpha)$, as is the case for standard regular expressions. It is well known that the set of partial derivatives of a regular expression gives rise to an equivalent NFA, called the Antimirov automaton or partial derivative automaton, that accepts the language determined by that expression. This remains valid in our extension of the partial derivatives to regular expressions with shuffle.

Definition 13. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, we define the partial derivative automaton associated with τ by*

$$\mathcal{A}_{pd}(\tau) = \langle \partial(\tau), \Sigma, \{\tau\}, \delta_\tau, F_\tau \rangle,$$

where $F_\tau = \{\ \gamma \in \partial(\tau) \mid \varepsilon(\gamma) = \varepsilon\ \}$ and $\delta_\tau(\gamma, a) = \partial_a(\gamma)$.

It is easy to see that the following holds.

Proposition 14. *For every state $\gamma \in \partial(\tau)$, the right language $\mathcal{L}_\gamma$ of γ in $\mathcal{A}(\tau)$ is equal to $\mathcal{L}(\gamma)$, the language represented by γ. In particular, the language accepted by $\mathcal{A}_{pd}(\tau)$ is exactly $\mathcal{L}(\tau)$.*

Note that for the REs α_n considered in examples 1 and 7, $\mathcal{A}_{pd}(\alpha_n)$ has 2^n states which is exactly the bound presented by Mayer and Stockmeyer [14].

5 Average State Complexity of the Partial Derivative Automaton

In this section, we estimate the asymptotic average size of the number of states in partial derivative automata. This is done by the use of the standard methods of analytic combinatorics as expounded by Flajolet and Sedgewick [9], which apply to generating functions $A(z) = \sum_n a_n z^n$ associated with combinatorial classes. Given some measure of the objects of a class $\mathcal{A}$, the coefficient a_n represents the sum of the values of this measure for all objects of size n. We will use the notation $[z^n]A(z)$ for a_n. For an introduction of this approach applied to formal languages, we refer to Broda *et al.* [4]. In order to apply this method, it is necessary to have an unambiguous description of the objects of the combinatorial class, as is the case for the specification of $\mathsf{T}_{\sqcup\!\sqcup}$-expressions without $\emptyset$ in (2). For the length or size of a $\mathsf{T}_{\sqcup\!\sqcup}$-expression α we will consider the number of symbols in α, not counting parentheses. Taking $k = |\Sigma|$, we compute from (2) the generating functions $R_k(z)$ and $L_k(z)$, for the number of $\mathsf{T}_{\sqcup\!\sqcup}$-expressions without $\emptyset$ and the number of alphabet symbols in $\mathsf{T}_{\sqcup\!\sqcup}$-expressions without $\emptyset$, respectively. Note that excluding one object, $\emptyset$, of size 1 has no influence on the asymptotic study.

According to the specification in (2) the generating function $R_k(z)$ for the number of $\mathsf{T}_{\sqcup\!\sqcup}$-expressions without $\emptyset$ satisfies

$$R_k(z) = z + kz + 3zR_k(z)^2 + zR_k(z),$$

thus,

$$R_k(z) = \frac{(1-z) - \sqrt{\Delta_k(z)}}{6z}, \text{ where } \Delta_k(z) = 1 - 2z - (11+12k)z^2.$$

The radius of convergence of $R_k(z)$ is $\rho_k = \frac{-1+2\sqrt{3+3k}}{11+12k}$. Now, note that the number of letters $l(\alpha)$ in an expression α satisfies: $l(\varepsilon) = 0$, in $l(a) = 1$, for $a \in \Sigma$, $l(\alpha + \beta) = l(\alpha) + l(\beta)$, etc. From this, we conclude that the generating function $L_k(z)$ satisfies

$$L_k(z) = kz + 3zL_k(z)R_k(z) + zL_k(z),$$

thus,

$$L_k(z) = \frac{(-kz)}{6zR_k(z) + z - 1} = \frac{kz}{\sqrt{\Delta_k(z)}}.$$

Now, let $P_k(z)$ denote the generating function for the size of $\pi(\alpha)$ for $\mathsf{T}_{\sqcup\!\sqcup}$-expressions without $\emptyset$. From Definition 3 it follows that, given an expression α, an upper bound, $p(\alpha)$, for the number of elements[1] in the set $\pi(\alpha)$ satisfies:

$$\begin{aligned} p(\varepsilon) &= 0 & p(\alpha+\beta) &= p(\alpha) + p(\beta) \\ p(a) &= 1, \quad \text{for } a \in \Sigma & p(\alpha\beta) &= p(\alpha) + p(\beta) \\ p(\alpha^\star) &= p(\alpha) & p(\alpha \sqcup\!\sqcup \beta) &= p(\alpha)p(\beta) + p(\alpha) + p(\beta). \end{aligned}$$

[1] This upper bound corresponds to the case where all unions in $\pi(\alpha)$ are disjoint.

From this, we conclude, using the symbolic method [9], that the generating function $P_k(z)$ satisfies

$$P_k(z) = kz + 6zP_k(z)R_k(z) + zP_k(z) + zP_k(z)^2,$$

thus

$$P_k(z) = Q_k(z) + S_k(z),$$

where

$$Q_k(z) = \frac{\sqrt{\Delta_k(z)}}{2z}, \qquad S_k(z) = -\frac{\sqrt{\Delta'_k(z)}}{2z},$$

and $\Delta'_k(z) = 1 - 2z - (11 + 16k)z^2$. The radii of convergence of $Q_k(z)$ and $S_k(z)$ are respectively ρ_k (defined above) and $\rho'_k = \frac{-1+2\sqrt{3+4k}}{11+16k}$.

5.1 Asymptotic analysis

A generating function f can be seen as a complex analytic function, and the study of its behaviour around its dominant singularity ρ (in case there is only one, as it happens with the functions considered here) gives us access to the asymptotic form of its coefficients. In particular, if $f(z)$ is analytic in some appropriate neighbourhood of ρ, then one has the following [9,16,4]:

1. if $f(z) = a - b\sqrt{1 - z/\rho} + o\left(\sqrt{1 - z/\rho}\right)$, with $a, b \in \mathbb{R}$, $b \neq 0$, then
$$[z^n]f(z) \sim \frac{b}{2\sqrt{\pi}}\,\rho^{-n}n^{-3/2};$$
2. if $f(z) = \frac{a}{\sqrt{1-z/\rho}} + o\left(\frac{1}{\sqrt{1-z/\rho}}\right)$, with $a \in \mathbb{R}$, and $a \neq 0$, then
$$[z^n]f(z) \sim \frac{a}{\sqrt{\pi}}\,\rho^{-n}n^{-1/2}.$$

Hence, by 1. one has for the number of $\mathsf{T}_{\sqcup\!\sqcup}$-expressions of size n,

$$[z^n]R_k(z) \sim \frac{(3+3k)^{\frac{1}{4}}}{6\sqrt{\pi}}\rho_k^{-n-\frac{1}{2}}(n+1)^{-\frac{3}{2}} \tag{4}$$

and by 2. for the number of alphabet symbols in all expression of size n,

$$[z^n]L_k(z) \sim \frac{k}{2\sqrt{\pi}(3+3k)^{\frac{1}{4}}}\rho_k^{-n+\frac{1}{2}}n^{-\frac{1}{2}}. \tag{5}$$

Consequently, the average number of letters in an expression of size n, which we denote by avL, is asymptotically given by

$$avL = \frac{[z^n]L_k(z)}{[z^n]R_k(z)} \sim \frac{3k\rho_k}{\sqrt{3+3k}}\frac{(n+1)^{\frac{3}{2}}}{n^{\frac{1}{2}}}.$$

Finally, by 1., one has for the size of expressions of size n,

$$\begin{aligned}[z^n]P_k(z) &= [z^n]Q_k(z) + [z^n]S_k(z)\\ &\sim \frac{-(3+3k)^{\frac{1}{4}}\rho_k^{-n-\frac{1}{2}} + (3+4k)^{\frac{1}{4}}(\rho'_k)^{-n-\frac{1}{2}}}{2\sqrt{\pi}}(n+1)^{-\frac{3}{2}},\end{aligned}$$

and the average size of $\pi(\alpha)$ for an expression α of size n, denoted by avP, is asymptotically given by

$$avP = \frac{[z^n]P_k(z)}{[z^n]R_k(z)}.$$

Taking into account Proposition 6, we want to compare the values of $\log_2 avP$ and avL. In fact, one has

$$\lim_{n,k\to\infty} \frac{\log_2 avP}{avL} = \log_2 \frac{4}{3} \sim 0.415.$$

This means that,

$$\lim_{n,k\to\infty} avP^{1/avL} = \frac{4}{3}.$$

Therefore, one has the following significant improvement, when compared with the worst case, for the average case upper bound.

Proposition 15. *For large values of k and n an upper bound for the average number of states of $\mathcal{A}_{pd}$ is $(\frac{4}{3} + o(1))^{|\alpha|_\Sigma}$.*

6 Conclusion and Future Work

We implemented the construction of the $\mathcal{A}_{pd}$ for REs with shuffle in the FAdo system [8] and performed some experimental tests for small values of n and k. Those experiments over statistically significant samples of uniform random generated REs suggest that the upper bound obtained in the last section falls far short of its true value. This is not surprising as in the construction of $\pi(\alpha) \cup \{\alpha\}$ repeated elements can occur.

In previous work [2], we identified classes of standard REs that capture a significant reduction on the size of $\pi(\alpha)$. In the case of REs with shuffle, those classes enforce only a marginal reduction in the number of states, but a drastic increase in the complexity of the associated generating function. Thus the expected gains don't seem to justify its quite difficult asymptotic study.

Sulzmann and Thiemann [17] extended the notion of Brzozowski derivative for several variants of the shuffle operator. It will be interesting to carry out a descriptional complexity study of those constructions and to see if it is interesting to extend the notion of partial derivative to those shuffle variants.

An extension of the partial derivative construction for extended REs with intersection and negation was recently presented by Caron *et. al* [5]. It will be also interesting to study the average complexity of this construction.

References

1. Antimirov, V.M.: Partial derivatives of regular expressions and finite automaton constructions. Theoret. Comput. Sci. 155(2), 291–319 (1996)
2. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: On the average state complexity of partial derivative automata. International Journal of Foundations of Computer Science 22(7), 1593–1606 (2011)
3. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: On the average size of Glushkov and partial derivative automata. International Journal of Foundations of Computer Science 23(5), 969–984 (2012)
4. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: A hitchhiker's guide to descriptional complexity through analytic combinatorics. Theor. Comput. Sci. 528, 85–100 (2014)
5. Caron, P., Champarnaud, J., Mignot, L.: Partial derivatives of an extended regular expression. In: Dediu, A.H., Inenaga, S., Martín-Vide, C. (eds.) 5th LATA 2011. LNCS, vol. 6638, pp. 179–191. Springer (2011)
6. Champarnaud, J.M., Ziadi, D.: From Mirkin's prebases to Antimirov's word partial derivatives. Fundam. Inform. 45(3), 195–205 (2001)
7. Estrade, B.D., Perkins, A.L., Harris, J.M.: Explicitly parallel regular expressions. In: Ni, J., Dongarra, J. (eds.) 1st IMSCCS. pp. 402–409. IEEE Computer Society (2006)
8. FAdo, P.: FAdo: tools for formal languages manipulation. `http://fado.dcc.fc.up.pt/` (Access date:01102014)
9. Flajolet, P., Sedgewick, R.: Analytic Combinatorics. CUP (2008)
10. Gelade, W.: Succinctness of regular expressions with interleaving, intersection and counting. Theor. Comput. Sci. 411(31-33), 2987–2998 (2010)
11. Gruber, H.: On the descriptional and algorithmic complexity of regular languages. Ph.D. thesis, Justus Liebig University Giessen (2010)
12. Gruber, H., Holzer, M.: Finite automata, digraph connectivity, and regular expression size. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) 35th ICALP. LNCS, vol. 5126, pp. 39–50. Springer (2008)
13. Kumar, A., Verma, A.K.: A novel algorithm for the conversion of parallel regular expressions to non-deterministic finite automata. Applied Mathematics & Information Sciences 8, 95–105 (2014)
14. Mayer, A.J., Stockmeyer, L.J.: Word problems-this time with interleaving. Inf. Comput. 115(2), 293–311 (1994)
15. Mirkin, B.G.: An algorithm for constructing a base in a language of regular expressions. Engineering Cybernetics 5, 51—57 (1966)
16. Nicaud, C.: On the average size of Glushkov's automata. In: Dediu, A.H., Ionescu, A.M., Martín-Vide, C. (eds.) Proc. 3rd LATA. LNCS, vol. 5457, pp. 626–637. Springer (2009)
17. Sulzmann, M., Thiemann, P.: Derivatives for regular shuffle expressions. In: Dediu, A.H., Formenti, E., Martín-Vide, C., Truthe, B. (eds.) 9th LATA. LNCS, vol. 8977, pp. 275–286. Springer (2015), to appear