Automata for Regular Expressions with Shuffle

Sabine Broda, António Machiavelo, Nelma Moreira, Rogério Reis

CMUP & DCC, Faculdade de Ciências da Universidade do Porto
Rua do Campo Alegre, 4169-007 Porto, Portugal

Abstract

We generalize the partial derivative automaton and the position automaton to regular expressions with shuffle, and study their state complexity in the worst, as well as in the average case. The number of states of the partial derivative automaton ($\mathcal{A}_{pd}$) is, in the worst case, at most 2^m, where m is the number of letters in the expression. The asymptotic average is bounded by $(\frac{4}{3})^m$. We define a position automaton ($\mathcal{A}_{pos}$) that is homogeneous, but in which several states can correspond to a same position, and we show that $\mathcal{A}_{pd}$ is a quotient of $\mathcal{A}_{pos}$. The number of states of the position automaton is at most $1+m(2^m-1)$, while the asymptotic average is no more than $m(\frac{4}{3})^m$.

Keywords: regular expressions, shuffle operation, partial derivatives, finite automata, position automata, average case, analytic combinatorics

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 [16] studied the computational complexity of membership and nonequivalence problems. Nonequivalence 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 [12], and Gruber and Holzer [14, 13] 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

[☆]This work was partially funded by the European Regional Development Fund through the programme COMPETE and by the Portuguese Government through the FCT under project UID/MAT/00144/2013.

Email addresses: sbb@dcc.fc.up.pt (Sabine Broda), ajmachia@fc.up.pt (António Machiavelo), nam@dcc.fc.up.pt (Nelma Moreira), rvr@dcc.fc.up.pt (Rogério Reis)

shuffle to DFAs. Recently, conversions of shuffle expressions to finite automata were presented by Estrade *et al.* [9], and Kumar and Verma [15]. In the former an expression is transformed first into a parallel finite automaton and then to an ε-NFA of size 2^{2r-3c}, where r is the size of the expression and c the number of occurrences of the concatenation operator. In the latter the authors give an algorithm for the construction of an ε-free NFA based on the classic Glushkov/-position construction, which the authors claim to have at most 2^{m+1} states, where m is the number of letters that occur in the RE with shuffle. Each state corresponds to a set of positions of letters in RE, and in opposition to what happens in the position automaton for standard REs, the automaton is not homogeneous, i.e. the incoming transitions of a state do not share necessarily the same label.

In this paper we present a conversion method of REs with shuffle to ε-free NFAs, by generalizing the partial derivative construction for standard REs [1, 17]. For standard REs, the partial derivative automaton ($\mathcal{A}_{pd}$) is a quotient of the Glushkov/position automaton ($\mathcal{A}_{pos}$), and Broda *et al.* [3, 4] 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 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, $(\frac{4}{3})^m$. We also present a construction of a position automaton $\mathcal{A}_{pos}$ from a RE with shuffle which is homogeneous, and for which the partial derivative automaton is a quotient. The number of states of $\mathcal{A}_{pos}$ is, in the worst case, $1 + m(2^m - 1)$ (with m as before), and an upper bound for the average size is, asymptotically, $m(\frac{4}{3})^m$.

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. In Section 5, we give the construction of an automaton based on the notion of positions, denoted by $\mathcal{A}_{pos}$, and show that $\mathcal{A}_{pd}$ is a quotient of $\mathcal{A}_{pos}$. In Section 6, we study the average state complexity of both $\mathcal{A}_{pd}$ and $\mathcal{A}_{pos}$ using the framework of analytic combinatorics. We conclude in Section 7 with some considerations about the upper bounds obtained in this paper, and point out some possible directions for some 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:

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

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 $+, \cdot, \sqcup\!\sqcup, ^\star$, 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 $\tau_1 \doteq \tau_2$, if $\mathcal{L}(\tau_1) = \mathcal{L}(\tau_2)$. The set of alphabet symbols occurring in an expression τ is denoted by Σ_τ.

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 $+$. One also has that for all $a, b \in \Sigma$ and $\tau_1, \tau_2 \in \mathsf{T}_{\sqcup\!\sqcup}$,

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

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. Using the identity elements of $\cdot$ and $+$, and the absorbing property of $\emptyset$, 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 $\varepsilon(\alpha \sqcup\!\sqcup \beta) = \varepsilon(\alpha)\varepsilon(\beta)$. Moreover, in what follows we will always consider expressions reduced according to the following equations $\alpha \sqcup\!\sqcup \varepsilon \doteq \varepsilon \sqcup\!\sqcup \alpha \doteq \alpha$ and $\alpha\varepsilon \doteq \varepsilon\alpha \doteq \alpha$. These are natural simplifications that do not affect the complexity upper bounds obtained.

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, $S_0 \subseteq S$ the set of initial states, $\delta : S \times \Sigma \longrightarrow \mathcal{P}(S)$ the transition function, and $F \subseteq S$ the 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 $S_0 = \{s\}$. If two automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are isomorphic we say that $\mathcal{A}_1 \simeq \mathcal{A}_2$. An equivalence relation $\equiv$ on S is *right invariant* w.r.t. $\mathcal{A}$ if and only if for all $s, t \in S$, $s \equiv t$ implies that

- $s \in F$ if and only if $t \in F$;
- for all $s' \in \delta(s, a)$, $a \in \Sigma$, there exists $t' \in \delta(t, a)$, such that $s' \equiv t'$.

The quotient automaton $\mathcal{A}_{/\equiv}$ is equivalent to $\mathcal{A}$.

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

$$\mathcal{L}_i \quad = \quad a_1\mathcal{L}_{i1} \cup \cdots \cup a_k\mathcal{L}_{ik} \cup \varepsilon(\mathcal{L}_i), \quad i \in [1, n]$$

where each $\mathcal{L}_{ij} = \bigcup_{l \in \delta(i, a_j)} \mathcal{L}_l$, i.e (possibly empty) union of elements in $\{\mathcal{L}_1, \ldots, \mathcal{L}_n\}$, 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 1. *Consider* $\Sigma = \{a_1, \ldots, a_k\}$ *and* $\alpha_0 \in \mathsf{T}_{\sqcup\!\sqcup}$. *A* support *of* α_0 *is a set* $\{\alpha_1, \ldots, \alpha_n\}$ *that satisfies a system of equations*

$$\alpha_i \quad \doteq \quad a_1\alpha_{i1} + \cdots + a_k\alpha_{ik} + \varepsilon(\alpha_i), \quad i \in [0, n] \tag{3}$$

for some $\alpha_{i1}, \ldots, \alpha_{ik}$*, each one a (possibly empty) sum of elements in* $\{\alpha_1, \ldots, \alpha_n\}$*. In this case* $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$ *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. Namely, $\mathcal{A} = \langle \{\alpha_0, \ldots, \alpha_n\}, \Sigma, \{\alpha_0\}, \delta, F \rangle$, with $F = \{\alpha_i \mid \varepsilon(\alpha_i) = \varepsilon\}$ and $\delta(\alpha_i, a_j) = \chi(\alpha_{ij})$ (where $\chi(\alpha_{i_1} + \cdots + \alpha_{i_l}) = \{\alpha_{i_1}, \ldots, \alpha_{i_l}\}$).

Note that the system of equations (3) can be written in matrix form $\mathsf{A}_\alpha \doteq \mathsf{C} \cdot \mathsf{M}_\alpha + \mathsf{E}_\alpha$, where M_α is the $k \times (n+1)$ matrix with entries α_{ji} ($j \in [1, k]$, $i \in [0, n]$) and A_α, C and E_α denote respectively the following three matrices,

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

and $\mathsf{C} \cdot \mathsf{M}_\alpha$ 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 $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$ can be computed using the function $\pi : \mathsf{T}_{\sqcup\!\sqcup} \longrightarrow \mathcal{P}(\mathsf{T}_{\sqcup\!\sqcup})$ recursively given in following definition.

Definition 2. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, the set $\pi(\tau)$ is inductively defined by,*

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

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

The following lemma follows directly from the definitions and will be used in the proof of Proposition 2.

Lemma 1. *If $\alpha, \beta \in \mathsf{T}_{\sqcup\!\sqcup}$, then $\varepsilon(\beta) \cdot \mathcal{L}(\alpha) \subseteq \mathcal{L}(\alpha \sqcup\!\sqcup \beta)$.*

Proposition 2. *If $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, then the set $\pi(\tau)$ is a support of τ.*

Proof. For $\emptyset$ is obvious. We proceed by induction on the structure of α. Excluding the case where α is $\alpha_0 \sqcup\!\sqcup \beta_0$, the proof can be found in [17, 6]. We now describe how to obtain a system of equations corresponding to an expression $\alpha_0 \sqcup\!\sqcup \beta_0$ from systems for α_0 and β_0. Suppose that $\pi(\alpha_0) = \{\alpha_1, \ldots, \alpha_n\}$ is a support of α_0 and $\pi(\beta_0) = \{\beta_1, \ldots, \beta_m\}$ 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

$$\begin{array}{rcl}
\pi(\alpha_0 \sqcup\!\sqcup \beta_0) & = & \{\alpha_1 \sqcup\!\sqcup \beta_1, \ldots, \alpha_1 \sqcup\!\sqcup \beta_m, \ldots, \alpha_n \sqcup\!\sqcup \beta_1, \ldots, \alpha_n \sqcup\!\sqcup \beta_m\} \cup \\
 & & \cup\, \{\alpha_1 \sqcup\!\sqcup \beta_0, \ldots, \alpha_n \sqcup\!\sqcup \beta_0\} \cup \{\alpha_0 \sqcup\!\sqcup \beta_1, \ldots, \alpha_0 \sqcup\!\sqcup \beta_m\}
\end{array}$$

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

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

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

$$\gamma_{l,(i,j)} = \alpha_{il} \sqcup\!\sqcup \beta_j + \alpha_i \sqcup\!\sqcup \beta_{jl}.$$

Note that, since by the induction hypothesis each α_{il} is equivalent to a sum of elements in $\pi(\alpha)$ and each β_{jl} is equivalent to a sum of elements in $\pi(\beta)$, due to the distributivity of $\sqcup\!\sqcup$ over $+$, each element of $\mathsf{M}_{\alpha_0 \sqcup\!\sqcup \beta_0}$ is in fact equivalent to a sum of elements in $\pi(\alpha_0 \sqcup\!\sqcup \beta_0)$. We will show that $\mathsf{A}_{\alpha_0 \sqcup\!\sqcup \beta_0} \doteq \mathsf{C} \cdot \mathsf{M}_{\alpha_0 \sqcup\!\sqcup \beta_0} + \mathsf{E}_{\alpha_0 \sqcup\!\sqcup \beta_0}$. For this, consider $\alpha_i \sqcup\!\sqcup \beta_j$ for some $(i,j) \in [0,n] \times [0,m]$. We have $\alpha_i \doteq a_1\alpha_{i1} + \cdots + a_k\alpha_{ik} + \varepsilon(\alpha_i)$ and $\beta_j \doteq a_1\beta_{j1} + \cdots + a_k\beta_{jk} + \varepsilon(\beta_j)$. Consequently, using properties of $\sqcup\!\sqcup$, namely distributivity over $+$, as well as Lemma 1,

$$\begin{aligned}
\alpha_i \shuffle \beta_j &\doteq (a_1\alpha_{i1} + \cdots + a_k\alpha_{ik} + \varepsilon(\alpha_i)) \shuffle (a_1\beta_{j1} + \cdots + a_k\beta_{jk} + \varepsilon(\beta_j)) \\
&\doteq a_1\,(\alpha_{i1} \shuffle \beta_j + \alpha_i \shuffle \beta_{j1} + \varepsilon(\beta_j)\alpha_{i1} + \varepsilon(\alpha_i)\beta_{j1}) \ + \ \cdots \ + \\
&\ a_k\,(\alpha_{ik} \shuffle \beta_j + \alpha_i \shuffle \beta_{jk} + \varepsilon(\beta_j)\alpha_{ik} + \varepsilon(\alpha_i)\beta_{jk}) + \varepsilon(\alpha_i \shuffle \beta_j) \\
&\doteq a_1\,(\alpha_{i1} \shuffle \beta_j + \alpha_i \shuffle \beta_{j1}) \ + \ \cdots \ + \\
&\ a_k\,(\alpha_{ik} \shuffle \beta_j + \alpha_i \shuffle \beta_{jk}) + \varepsilon(\alpha_i \shuffle \beta_j) \\
&\doteq a_1\gamma_{1,(i,j)} + \cdots + a_k\gamma_{k,(i,j)} + \varepsilon(\alpha_i \shuffle \beta_j).
\end{aligned}$$

□

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 2 is a witness that this upper bound is tight.

Proposition 3. *Given $\tau \in \mathsf{T}_{\shuffle}$, one has $|\pi(\tau)| \leq 2^{|\tau|_\Sigma} - 1$, where $|\tau|_\Sigma$ denotes the number of alphabet symbols in τ.*

Proof. The proof proceeds by induction on the structure of τ. It is clear that the result holds for $\emptyset$, ε and $a \in \Sigma$. 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 $\alpha \shuffle \beta$, one has $|\pi(\alpha \shuffle \beta)| = |\pi(\alpha) \shuffle \pi(\beta) \cup \pi(\alpha) \shuffle \{\beta\} \cup \{\alpha\} \shuffle \pi(\beta)| \leq (2^{|\alpha|_\Sigma} - 1)(2^{|\beta|_\Sigma} - 1) + 2^{|\alpha|_\Sigma} - 1 + 2^{|\beta|_\Sigma} - 1 = 2^{|\alpha|_\Sigma + |\beta|_\Sigma} - 1 = 2^{|\alpha \shuffle \beta|_\Sigma} - 1$. □

Example 2. *Considering $\alpha_n = a_1 \shuffle \cdots \shuffle a_n$, where $n \geq 1$, $a_i \neq a_j$ for $1 \leq i \neq j \leq n$ again, one has*

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

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

The proof of Proposition 2 gives a way to construct a system of equations for an expression $\tau \in \mathsf{T}_{\shuffle}$, 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 Derivative Automaton

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 $a^{-1}(L_1 \sqcup\!\sqcup L_2) = (a^{-1}L_1) \sqcup\!\sqcup L_2 \;\cup\; L_1 \sqcup\!\sqcup (a^{-1}L_2)$.

Definition 3. *The set of partial derivatives of a term $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$ w.r.t. a letter $a \in \Sigma$, denoted by $\partial_a(\tau)$, is inductively defined by*

$$\begin{aligned}
\partial_a(\emptyset) &= \partial_a(\varepsilon) = \emptyset & \partial_a(\alpha^\star) &= \partial_a(\alpha)\alpha^\star \\
\partial_a(b) &= \begin{cases} \{\varepsilon\} & \text{if } b = a \\ \emptyset & \text{otherwise} \end{cases} & \partial_a(\alpha + \beta) &= \partial_a(\alpha) \cup \partial_a(\beta) \\
& & \partial_a(\alpha\beta) &= \partial_a(\alpha)\beta \;\cup\; \varepsilon(\alpha)\partial_a(\beta) \\
& & \partial_a(\alpha \sqcup\!\sqcup \beta) &= \partial_a(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \partial_a(\beta),
\end{aligned}$$

where the expressions involving sets of expressions are as specified in Definition 2.

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

We let $\partial(\tau)$ denote the set of all partial derivatives of an expression τ, i.e. $\partial(\tau) = \bigcup_{x \in \Sigma^\star} \partial_x(\tau)$, 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 $S \subseteq \mathsf{T}_{\sqcup\!\sqcup}$, we define $\mathcal{L}(S) = \bigcup_{\tau \in S} \mathcal{L}(\tau)$. The following result has a straightforward proof.

Proposition 4. *Given $x \in \Sigma^\star$ and $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, 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 6.

Lemma 5. *For $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, 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 3 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, $\mathcal{L}(\tau) = \{\varepsilon\}$. $\square$

Proposition 6. *∂^+ satisfies the following:*

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

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 $\partial_a(\alpha+\beta) = \partial_a(\alpha) \cup \partial_a(\beta) \subseteq \partial^+(\alpha) \cup \partial^+(\beta)$, 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 $\partial_x(\alpha) \subseteq \partial^+(\alpha+\beta)$ and $\partial_x(\beta) \subseteq \partial^+(\alpha+\beta)$, for all $x \in \Sigma^+$.

- For $\alpha^\star$, we have $\partial_a(\alpha^\star) = \partial_a(\alpha)\alpha^\star \subseteq \partial^+(\alpha)\alpha^\star$, as well as

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

  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 5 that $\partial^+(\alpha) \neq \emptyset$, and that there is some $\alpha_0 \in \partial^+(\alpha)$ with $\varepsilon(\alpha_0) = \varepsilon$. 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 $\partial_x(\alpha) \sqcup\!\sqcup \partial_y(\beta) \subseteq \partial_y(\partial_x(\alpha) \sqcup\!\sqcup \{\beta\}) \subseteq \partial_y(\partial_x(\alpha \sqcup\!\sqcup \beta)) = \partial_{xy}(\alpha \sqcup\!\sqcup \beta) \subseteq \partial^+(\alpha \sqcup\!\sqcup \beta)$. $\square$

Corollary 7. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, one has $\partial^+(\tau) = \pi(\tau)$.*

We conclude that $\partial(\tau)$ corresponds to the set $\{\tau\} \cup \pi(\tau)$, 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 4. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, we define the partial derivative automaton associated with τ by*

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

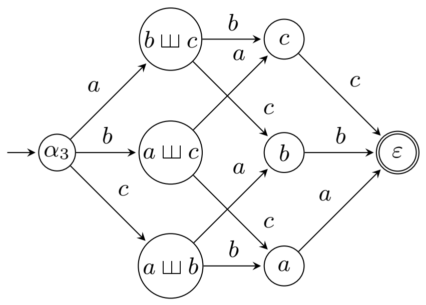
where $F_\tau = \{\ \gamma \in \partial(\tau) \mid \varepsilon(\gamma) = \varepsilon\ \}$ and $\delta_\tau(\gamma, a) = \partial_a(\gamma)$.

It is easy to see that the following holds.

Proposition 8. *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 2, $\mathcal{A}_{pd}(\alpha_n)$ has 2^n states which is exactly the bound presented by Mayer and Stockmeyer [16]. The case for $n = 3$ is presented in the following example.

Example 3. *The partial derivative automaton for $\alpha_3 = a \sqcup\!\sqcup b \sqcup\!\sqcup c$ is the following.*



4.1. Recursive Construction of $\mathcal{A}_{pd}$

The set of partial derivatives of a term $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$ w.r.t all $a \in \Sigma$ can be efficiently calculated using the function F recursively defined as follows:

$$\begin{array}{rclrcl}
\mathsf{F}(\emptyset) & = & \mathsf{F}(\varepsilon) = \emptyset & \mathsf{F}(\alpha+\beta) & = & \mathsf{F}(\alpha) \cup \mathsf{F}(\beta) \\
\mathsf{F}(a) & = & \{(a,\varepsilon)\} & \mathsf{F}(\alpha\beta) & = & \mathsf{F}(\alpha)\beta \cup \varepsilon(\alpha)\mathsf{F}(\beta) \\
\mathsf{F}(\alpha^\star) & = & \mathsf{F}(\alpha)\alpha^\star & \mathsf{F}(\alpha \sqcup\!\sqcup \beta) & = & \mathsf{F}(\alpha) \sqcup\!\sqcup \beta \cup \alpha \sqcup\!\sqcup \mathsf{F}(\beta),
\end{array}$$

where for $S, T \subseteq \Sigma \times (\mathsf{T}_{\sqcup\!\sqcup} \setminus \{\emptyset\})$ and $\alpha \in \mathsf{T}_{\sqcup\!\sqcup} \setminus \{\emptyset\}$, $S \circ \alpha = \{\ (a, \beta \circ \alpha) \mid (a,\beta) \in S\ \}$, $\alpha \circ S = \{\ (a, \beta\alpha \circ \beta) \mid (a,\beta) \in S\ \}$, for $\circ \in \{\cdot, \sqcup\!\sqcup\}$, and $\emptyset S = S\emptyset = \emptyset$. Clearly, for each $a \in \Sigma$, $\partial_a(\tau) = \{\alpha \mid (a,\alpha) \in \mathsf{F}(\tau)\}$. Based on the system of equations presented in the Section 3 and as also pointed out in the end of that section, the set of transitions of $\mathcal{A}_{pd}$ can be recursively defined using the following function.

$$\begin{aligned}
\mathsf{T}(\emptyset) &= \mathsf{T}(\varepsilon) = \mathsf{T}(a) = \emptyset,\ a \in \Sigma \\
\mathsf{T}(\alpha+\beta) &= \mathsf{T}(\alpha) \cup \mathsf{T}(\beta) \\
\mathsf{T}(\alpha\cdot\beta) &= \mathsf{T}(\alpha)\beta \cup \mathsf{T}(\beta) \cup \mathsf{L}(\alpha)\beta \times \mathsf{F}(\beta) \\
\mathsf{T}(\alpha^\star) &= \mathsf{T}(\alpha)\alpha^\star \cup (\mathsf{L}(\alpha) \times \mathsf{F}(\alpha))\alpha^\star \\
\mathsf{T}(\alpha \sqcup\!\sqcup \beta) &= \mathsf{T}(\alpha) \sqcup\!\sqcup \mathsf{T}(\beta) \cup \mathsf{T}(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \mathsf{T}(\beta),
\end{aligned}$$

where $\mathsf{L}(\alpha) = \{\ \alpha' \mid \alpha' \in \pi(\alpha),\ \varepsilon(\alpha') = \varepsilon\ \}$[1] and the result of the $\times$ operation is seen as a set of triples. The concatenation of a transition (α, a, β) with a expression γ is defined by $(\alpha,a,\beta)\gamma = (\alpha\gamma, a, \beta\gamma)$ and $(\alpha,a,\beta) \sqcup\!\sqcup (\alpha', a', \beta') = \{(\alpha \sqcup\!\sqcup \alpha', a, \beta \sqcup\!\sqcup \alpha'), (\alpha \sqcup\!\sqcup \alpha', a', \alpha \sqcup\!\sqcup \beta')\}$. Then for $S, T \subseteq (\mathsf{T}_{\sqcup\!\sqcup} \setminus \{\emptyset\}) \times \Sigma \times (\mathsf{T}_{\sqcup\!\sqcup} \setminus \{\emptyset\})$ and $\gamma \in \mathsf{T}_{\sqcup\!\sqcup} \setminus \{\emptyset\}$, $S\gamma = \{s\gamma \mid s \in S\}$, $S \sqcup\!\sqcup T = \bigcup_{s\in S, t \in T} s \sqcup\!\sqcup t$. Finally, $\delta_\tau = (\{\tau\} \times \mathsf{F}(\tau)) \cup \mathsf{T}(\tau)$.

It is straightforward to see that $|\mathsf{F}(\alpha)| \le |\alpha|_\Sigma$, and $|\mathsf{L}(\alpha)| \le |\partial(\alpha)| \le 2^{|\alpha|_\Sigma}$. However, it seems hard to estimate a tight upper bound for $|\mathsf{T}(\alpha)|$. Of course, $|\delta_\tau| \le |\mathsf{F}(\alpha)| + |\mathsf{T}(\alpha)| \le |\Sigma|\, 2^{2|\alpha|_\Sigma}$. For the expressions α_n presented in Example 1, $|\mathsf{F}(\alpha_n)| = |\alpha_n|_\Sigma = n$ and $|\delta_{\alpha_n}| = n2^{n-1}$.

[1]Which can also be defined recursively.

5. A Position Automaton

In this section we present another automaton construction, this time also based on the positions of alphabet symbols (letters) in regular expressions. For standard regular expressions, the position/Glushkov automaton has as many states as positions plus one, it is homogeneous, i.e. for every state all incoming transitions are labelled by the same alphabet symbol, and the partial derivative automaton is one of its quotients. Our construction will produce an homogeneous automaton as well. Also, every state except for the initial one will correspond to precisely one position in the regular expression, but the reverse is no longer true. In fact, a construction with such property cannot exist, when expressions with shuffle are considered, as shown by the following example.

Example 4. *Consider the expression $\tau = a \sqcup\!\sqcup b$ with $\mathcal{L}(\tau) = \{ab, ba\}$. In every homogeneous automaton for τ, there must exist a state s with incoming transitions labelled with a and right language $\mathcal{L}_s = \{b\}$, as well as another state t with incoming transitions labelled with a and right language $\mathcal{L}_t = \{\varepsilon\}$.*

Consequently, states of automata obtained by the construction defined in this section are labelled by pairs (γ, i), where i is a position of a letter in the original expression and $\gamma \in \mathsf{T}_{\sqcup\!\sqcup}$ describes the right language of the state.

For $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, let $\overline{\tau}$ denote the expression obtained by marking each letter with its position in τ. The same notation is used to remove the markings, i.e., $\overline{\overline{\tau}} = \tau$. Now, let $\mathsf{Pos}(\tau) = \{1, 2, \ldots, |\tau|_\Sigma\}$ be the set of positions for $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, and let $\mathsf{Pos}_0(\tau) = \mathsf{Pos}(\tau) \cup \{0\}$. An expression where all occurrences of alphabet symbols are marked will be called a *marked expression.*

Example 5. *For $\tau = a \sqcup\!\sqcup a \sqcup\!\sqcup b$, over the alphabet $\Sigma = \{a, b\}$, one has $\overline{\tau} = a_1 \sqcup\!\sqcup a_2 \sqcup\!\sqcup b_3$ and $\mathsf{Pos}(\tau) = \{1, 2, 3\}$.*

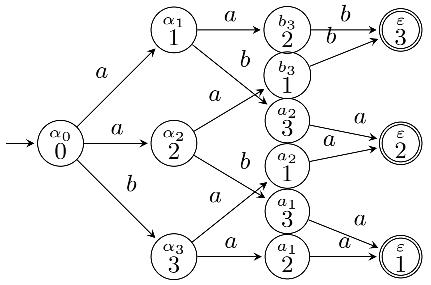
Definition 5. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, we define the* position automaton *associated with τ by*

$$\mathcal{A}_{pos}(\tau) = \langle S^0_{\text{pos}}(\tau), \Sigma, \{(\overline{\tau}, 0)\}, \delta_{\text{pos}}, F_{\text{pos}} \rangle,$$

where $S^0_{\text{pos}}(\tau) = S_{\text{pos}}(\overline{\tau}) \cup \{(\overline{\tau}, 0)\}$, $S_{\text{pos}}(\overline{\tau}) = \{\ (\gamma, i) \mid \gamma \in \partial_{a_i}(\partial(\overline{\tau})),\ a_i \in \Sigma_{\overline{\tau}}\ \}$, $F_{\text{pos}} = \{\ (\gamma, i) \in S^0_{\text{pos}}(\tau) \mid \varepsilon(\gamma) = \varepsilon\ \}$ and

$$\delta_{\text{pos}}((\gamma, i), a) = \{\ (\beta, j) \mid \beta \in \partial_{a_j}(\gamma), a = \overline{a_j}\ \}.$$

Example 6. *For τ from Example 5 $\mathcal{A}_{pos}(\tau)$ has twelve states $(\alpha_0, 0)$, $(\alpha_1, 1)$, $(\alpha_2, 2)$, $(\alpha_3, 3)$, $(b_3, 2)$, $(b_3, 1)$, $(a_2, 3)$, $(a_2, 1)$, $(a_1, 3)$, $(a_1, 2)$, $(\varepsilon, 3)$, $(\varepsilon, 2)$, and $(\varepsilon, 1)$, where $\alpha_0 = a_1 \sqcup\!\sqcup a_2 \sqcup\!\sqcup b_3$, $\alpha_1 = a_2 \sqcup\!\sqcup b_3$, $\alpha_2 = a_1 \sqcup\!\sqcup b_3$, and $\alpha_3 = a_1 \sqcup\!\sqcup a_2$. The automaton is the one depicted below.*



Note that our construction corresponds to the construction of a c-continuation automaton for standard regular expressions [7], which is known to be isomorphic to the position automaton. As in the case for standard expressions, with the c-continuation automaton we can, given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, show that $\mathcal{A}_{pd}(\tau)$ is a quotient of $\mathcal{A}_{pos}(\tau)$. For this purpose, we first consider the automaton $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ obtained from the partial derivative automaton $\mathcal{A}_{pd}(\overline{\tau})$ by unmarking labels of transitions. We will show that $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ is a quotient of $\mathcal{A}_{pos}(\tau)$ by the right-invariant equivalence relation $\equiv_1$, defined on the set of states in $\mathcal{A}_{pos}(\tau)$, by $(\alpha, i) \equiv_1 (\beta, j)$ iff $\alpha = \beta$.

Proposition 9. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, we have $\mathcal{A}_{pos}(\tau)/_{\equiv_1} \simeq \overline{\mathcal{A}_{pd}(\overline{\tau})}$.*

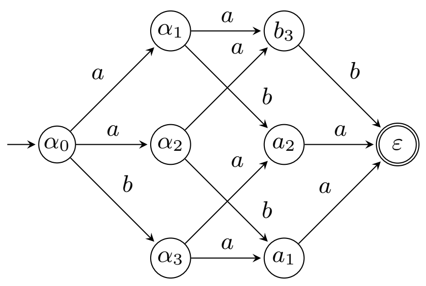
Proof. In order to show that $\equiv_1$ is right invariant w.r.t. $\mathcal{A}_{pos}(\tau)$, consider two equivalent states $(\alpha, i), (\alpha, j) \in S^0_{\text{pos}}(\tau)$ and $(\beta, l) \in \delta_{\text{pos}}((\alpha, i), a)$, $a \in \Sigma$. By Definition 5, we have $\beta \in \partial_{a_j}(\alpha)$ and $a = \overline{a_j}$. Thus, $(\beta, l) \in \delta_{\text{pos}}((\alpha, j), a)$. Furthermore, $\varepsilon((\alpha, i)) = \varepsilon((\alpha, j))$.

Now, recall that $\overline{\tau}$ is the initial state of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$, while $(\overline{\tau}, 0)$ labels the initial state in $\mathcal{A}_{pos}(\tau)$. It is also clear that $\gamma \in \partial(\overline{\tau})$ if and only if there is at least one state $(\gamma, i) \in S^0_{\text{pos}}(\tau)$. Furthermore, by Definition 5, $(\beta, j) \in \delta_{\text{pos}}((\gamma, i), a)$ if and only if $\beta \in \partial_{a_j}(\gamma)$ and $a = \overline{a_j}$. By the definition of $\mathcal{A}_{pd}$ we know that there exists a transition labelled by a_j from state γ to state β in $\mathcal{A}_{pd}(\overline{\tau})$ and thus there exists a transition labelled by a, from state γ to β in $\overline{\mathcal{A}_{pd}(\overline{\tau})}$. We conclude that the map ϕ, that associates to each equivalence class $[(\alpha, i)]$ the expression α, is an isomorphism between $\mathcal{A}_{pos}(\tau)/_{\equiv_1}$ and $\overline{\mathcal{A}_{pd}(\overline{\tau})}$. $\square$

Example 7. *For $\tau = a \sqcup\!\sqcup a \sqcup\!\sqcup b$, relation $\equiv_1$ induces the partition below on the set of states of $\mathcal{A}_{pos}(\tau)$ in Example 6.*

$$\{\ \{(\alpha_0, 0)\}, \{(\alpha_1, 1)\}, \{(\alpha_2, 2)\}, \{(\alpha_3, 3)\}, \{(b_3, 2), (b_3, 1)\},$$
$$\{(a_2, 3), (a_2, 1)\}, \{(a_1, 3), (a_1, 2)\}, \{(\varepsilon, 3), (\varepsilon, 2), (\varepsilon, 1)\}\ \}.$$

The corresponding automaton $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ is represented in the following diagram:



In the following we will show that $\mathcal{A}_{pd}(\tau)$ is a quotient of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ and, consequently, also of $\mathcal{A}_{pos}(\tau)$. We need the following results, which are easily proved by structural induction.

Lemma 10. *Consider a marked expression α, and $\beta \in \partial_{a_i}(\alpha)$ for some marked alphabet symbol a_i and $a = \overline{a_i}$. Then, $\overline{\beta} \in \partial_a(\overline{\alpha})$.*

Lemma 11. *Consider a marked expression α and an unmarked expression $\beta \in \partial_a(\overline{\alpha})$. Then, there is some symbol a_i in α with $a = \overline{a_i}$ and a marked expression $\beta' \in \partial_{a_i}(\alpha)$, such that $\overline{\beta'} = \beta$.*

Using these results, we will show that $\mathcal{A}_{pd}(\tau)$ is a quotient of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$, by the right-invariant equivalence relation $\equiv_2$, defined on the states of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ by $\alpha \equiv_2 \beta$ if and only if $\overline{\alpha} = \overline{\beta}$.

Proposition 12. *Given $\tau \in \mathsf{T}_{\sqcup\!\sqcup}$, we have $\overline{\mathcal{A}_{pd}(\overline{\tau})}/_{\equiv_2} \simeq \mathcal{A}_{pd}(\tau)$.*

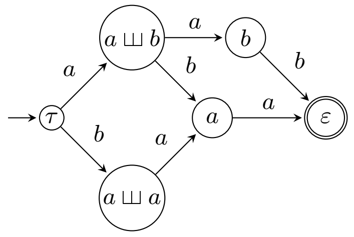
Proof. First we show that $\equiv_2$ is right invariant. Consider (marked) states α, β, and α' in $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ with $\overline{\alpha} = \overline{\beta}$, and such that there is a transition from α to α' labelled by a. This means that there is a symbol a_i in α such that $\overline{a_i} = a$ and $\alpha' \in \partial_{a_i}(\alpha)$. It follows from Lemma 10 that $\overline{\alpha'} \in \partial_a(\overline{\alpha}) = \partial_a(\overline{\beta})$. Furthermore, by Lemma 11, there is $\beta' \in \partial_{a_i}(\beta)$ such that $\alpha' \equiv_2 \beta'$. Also, $\varepsilon(\alpha) = \varepsilon(\overline{\alpha})$.

Now consider the map ϕ that associates to each equivalence class $[\alpha]$ of $\equiv_2$ the unmarked expression $\overline{\alpha}$. It is easy to see that ϕ is an isomorphism between $\overline{\mathcal{A}_{pd}(\overline{\tau})}/_{\equiv_2}$ and $\mathcal{A}_{pd}(\tau)$. Indeed, the initial states of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ and $\mathcal{A}_{pd}(\tau)$ are, by definition, respectively $\overline{\tau}$ and τ. It follows from Lemma 10 that for every marked state γ in $\overline{\mathcal{A}_{pd}(\overline{\tau})}$, there is exactly one state labelled with $\overline{\gamma}$ in $\mathcal{A}_{pd}(\tau)$. On the other hand, by Lemma 11, for every (unmarked) state β in $\mathcal{A}_{pd}(\tau)$ there exists β' in $\overline{\mathcal{A}_{pd}(\overline{\tau})}$, such that $\overline{\beta'} = \beta$. We conclude that the set of states of $\mathcal{A}_{pd}(\tau)$ corresponds to the set of equivalent classes of $\equiv_2$ on $\overline{\mathcal{A}_{pd}(\overline{\tau})}$. By the right-invariance of $\equiv_2$, we also conclude that if there is a transition from $[\alpha]$ to $[\alpha']$ labelled by $a \in \Sigma$ in $\overline{\mathcal{A}_{pd}(\overline{\tau})}/_{\equiv_2}$, then there is a transition by a from $\overline{\alpha}$ to $\overline{\alpha'}$ in $\mathcal{A}_{pd}(\tau)$. $\square$

Example 8. *For $\tau = a \sqcup\!\sqcup\, a \sqcup\!\sqcup\, b$, relation $\equiv_2$ induces on the set of states of $\overline{\mathcal{A}_{pd}(\overline{\tau})}$ the partition*

$$\{\{\alpha_0\}, \{\alpha_1, \alpha_2\}, \{\alpha_3\}, \{b_3\}, \{a_1, a_2\}, \{\varepsilon\}\}.$$

This corresponds precisely to the automaton $\mathcal{A}_{pd}(\tau)$ below.



Corollary 13. *Consider $\tau \in \mathsf{T}_{\sqcup}$, then $\mathcal{L}(\mathcal{A}_{pos}(\tau)) = \mathcal{L}(\tau)$.*

We now give a recursive definition of S_{pos}.

Proposition 14. *Given $\tau \in \mathsf{T}_{\sqcup}$, the set $S_{\text{pos}}(\overline{\tau})$ is inductively defined by,*

$$\begin{aligned} S_{\text{pos}}(\emptyset) &= S_{\text{pos}}(\varepsilon) = \emptyset & S_{\text{pos}}(\alpha+\beta) &= S_{\text{pos}}(\alpha) \cup S_{\text{pos}}(\beta) \\ S_{\text{pos}}(a_i) &= \{(\varepsilon, i)\} & S_{\text{pos}}(\alpha\beta) &= S_{\text{pos}}(\alpha)\beta \cup S_{\text{pos}}(\beta) \\ S_{\text{pos}}(\alpha^\star) &= S_{\text{pos}}(\alpha)\alpha^\star & S_{\text{pos}}(\alpha \sqcup \beta) &= S_{\text{pos}}(\alpha) \sqcup_p S_{\text{pos}}(\beta) \cup \\ & & &\cup S_{\text{pos}}(\alpha) \sqcup \{\beta\} \cup \{\alpha\} \sqcup S_{\text{pos}}(\beta), \end{aligned}$$

where, given $S, T \subseteq \mathsf{T}_{\sqcup} \times \mathsf{Pos}$ and $\beta \in \mathsf{T}_{\sqcup} \setminus \{\emptyset, \varepsilon\}$, $S\beta = \{\ (\alpha\beta, i) \mid (\alpha, i) \in S\ \}$, $S \sqcup \beta = \{\ (\alpha \sqcup \beta, i) \mid (\alpha, i) \in S\ \}$, $\beta \sqcup S = \{\ (\beta \sqcup \alpha, i) \mid (\alpha, i) \in S\ \}$, $S \sqcup_p T = \bigcup_{(\alpha,i)\in S, (\beta,j)\in T} \{\ (\alpha \sqcup \beta, i), (\alpha \sqcup \beta, j)\ \}$, $\varepsilon \sqcup S = S \sqcup \varepsilon = S\varepsilon = S$, and $S\emptyset = \emptyset S = \emptyset$.

Proof. The proof is by induction on the structure of $\overline{\tau}$. Here we only present the case for $\sqcup$. The remaining cases follow directly from the definition of S_{pos} and Proposition 6. We have,

$$\begin{aligned} S_{\text{pos}}(\alpha \sqcup \beta) &= \{\ (\gamma, i) \mid \gamma \in \partial_{a_i}(\partial(\alpha \sqcup \beta))\ \} \\ &= \{\ (\gamma, i) \mid \gamma \in \partial_{a_i}(\alpha \sqcup \beta) \cup\ \partial_{a_i}(\partial^+(\alpha \sqcup \beta))\ \} \\ &= \{\ (\gamma, i) \mid \gamma \in \partial_{a_i}(\alpha) \sqcup \{\beta\} \cup\ \{\alpha\} \sqcup \partial_{a_i}(\beta) \cup \partial_{a_i}(\partial^+(\alpha)) \sqcup \partial^+(\beta) \\ &\quad \cup\ \partial^+(\alpha) \sqcup \partial_{a_i}(\partial^+(\beta)) \cup \partial_{a_i}(\partial^+(\alpha)) \sqcup \{\beta\} \cup \partial^+(\alpha) \sqcup \partial_{a_i}(\beta) \\ &\quad \cup\ \partial_{a_i}(\alpha) \sqcup \partial^+(\beta) \cup \{\alpha\} \sqcup \partial_{a_i}(\partial^+(\beta))\ \} \\ &= S_{\text{pos}}(\alpha) \sqcup \{\beta\} \cup \{\alpha\} \sqcup S_{\text{pos}}(\beta) \\ &\quad \cup \{\ (\gamma, i) \mid \gamma \in \partial_{a_i}(\partial(\alpha)) \sqcup \partial^+(\beta) \cup \partial^+(\alpha) \sqcup \partial_{a_i}(\partial(\beta))\ \} \\ &= S_{\text{pos}}(\alpha) \sqcup \{\beta\} \cup \{\alpha\} \sqcup S_{\text{pos}}(\beta) \cup S_{\text{pos}}(\alpha) \sqcup_p S_{\text{pos}}(\beta). \end{aligned}$$

The last equality holds because, on one hand $(\alpha' \sqcup \beta', i) \in S_{\text{pos}}(\alpha) \sqcup_p S_{\text{pos}}(\beta)$ if and only if $(\alpha', i) \in S_{\text{pos}}(\alpha)$ and $(\beta', j) \in S_{\text{pos}}(\beta)$ for some marking j, or $(\alpha', j) \in S_{\text{pos}}(\alpha)$ and $(\beta', i) \in S_{\text{pos}}(\beta)$ for some marking j. On the other hand, for $\tau \in \mathsf{T}_{\sqcup}$, $\bigcup_{a_j \in \Sigma_{\overline{\tau}}} \partial_{a_j}(\partial(\overline{\tau})) = \partial^+(\overline{\tau})$. ☐

Finally, we note that for a standard RE the set $\partial_{a_i}(\partial(\overline{\alpha}))$, $a_i \in \Sigma_{\overline{\alpha}}$, has at most one element [8, 2, 7], while for REs with shuffle it can have more. More precisely, $|\partial_{a_i}(\partial(\overline{\alpha}))|$ equals the number of states with label (γ, i) in $S^0_{\text{pos}}(\alpha)$.

Example 9. *One has $|\partial_{a_1}(\partial(a_1 \sqcup a_2 \sqcup b_3))| = |\{a_2 \sqcup b_3, a_2, b_3, \varepsilon\}| = 4$.*

6. Average State Complexity

In this section, we estimate the asymptotic average size of the number of states in partial derivative automata, which also will induce a estimate on the number of states of the position automata. This is done by the use of the standard methods of analytic combinatorics as expounded by Flajolet and Sedgewick [11], 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.* [5]. 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

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

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

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

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 2 it follows that, given an expression α, an upper bound, $p(\alpha)$, for the number of elements[2] in the set $\pi(\alpha)$ satisfies:

$$\begin{array}{rcl@{\qquad}rcl}
p(\varepsilon) & = & 0 & p(\alpha+\beta) & = & p(\alpha)+p(\beta) \\
p(a) & = & 1, \text{ for } a \in \Sigma & p(\alpha\beta) & = & p(\alpha)+p(\beta) \\
p(\alpha^\star) & = & p(\alpha) & p(\alpha \sqcup\!\sqcup \beta) & = & p(\alpha)p(\beta)+p(\alpha)+p(\beta).
\end{array} \quad (4)$$

[2]This upper bound corresponds to the case where all unions in $\pi(\alpha)$ are disjoint, and where the identifications $\alpha \cdot \varepsilon = \varepsilon \cdot \alpha = \alpha$ and $\alpha \sqcup\!\sqcup \varepsilon = \varepsilon \sqcup\!\sqcup \alpha = \alpha$ are not taken into account.

From this, we conclude that the generating function $P_k(z)$ satisfies

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

thus

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

where

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

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 $\rho'_k = \frac{-1+2\sqrt{3+4k}}{11+16k}$.

6.1. 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 [11, 18, 5]:

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

$$[z^n]f(z) \sim \frac{a}{\sqrt{\pi}}\,\rho^{-n}n^{-1/2}.$$

Using the standard analytic combinatorics methods, as in [5], one can show that the generating function for the number of $\mathsf{T}_{\sqcup}$-expressions of size n, $R_k(z)$, falls into case 1 above, and

$$[z^n]R_k(z) \quad \sim \quad \frac{(3+3k)^{\frac{1}{4}}}{6\sqrt{\pi}}\rho_k^{-n-\frac{1}{2}}n^{-\frac{3}{2}}. \tag{5}$$

Similarly, one can show that, for the number of alphabet symbols in all expression of size n, the corresponding generating function $L_k(z)$ falls into case 2., and

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

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

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

For large values of k, one concludes from this that the number of letters in an expression is roughly half its size.

For the generating function for the number of states in $\mathcal{A}_{pd}$, $P_k(z)$, one can show that

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

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

$$avP = \frac{[z^n]P_k(z)}{[z^n]R_k(z)} \sim 3\left(\left(\frac{3+4k}{3+3k}\right)^{\frac{1}{4}}\left(\frac{\rho_k}{\rho_k'}\right)^{n+\frac{1}{2}} - 1\right).$$

Taking into account Proposition 3, 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 $(\frac{4}{3} + o(1))^{|\alpha|_\Sigma}$.*

Since each non-initial state of $\mathcal{A}_{pos}(\alpha)$ is labelled by a pair (γ, i), with $\gamma \in \pi(\alpha)$ and $i \in \mathsf{Pos}(\alpha)$, one has $|S_{\text{pos}}(\alpha)| \leq |\alpha|_\Sigma\, |\pi(\alpha)|$. This and the previous result readily imply the following.

Proposition 16. *For large values of k and n an upper bound for the average number of states of $\mathcal{A}_{pos}$ is $|\alpha|_\Sigma\,(\frac{4}{3} + o(1))^{|\alpha|_\Sigma}$.*

7. Conclusion and Future Work

A construction of the position automaton, based on the standard position functions First, Last and Follow, would be interesting to obtain. As mentioned above, it is often necessary to create more than one state for each position in the expression. Thus, the Follow function has to take into account some context to distinguish states corresponding to the same position. The approach of Kumar and Verma [15] does this, but the automaton produced is in general not homogeneous. This means that the states in the constructed automaton correspond to more than one position in the expression. In fact, it seems that their construction leads to an automaton somewhere between $\mathcal{A}_{pd}$ and $\mathcal{A}_{pos}$.

We implemented the constructions of $\mathcal{A}_{pd}$ and $\mathcal{A}_{pos}$ for REs with shuffle in the FAdo system [10], and performed some experimental tests for small values of n and k. Those experiments over statistically significant samples of uniform random generated REs with shuffle suggest that the upper bounds obtained in the last section falls far short of their true value. This is not surprising as repeated elements can occur in the construction of the sets of states.

In a previous work [3], we identified classes of standard REs that capture a significant reduction on the size of $\pi(\alpha)$. In the case of REs with shuffle, that brings about only a marginal reduction in the number of states, but a drastic increase in the complexity of the associated generating function. Thus the expected gain does not seem to justify the subsequent difficult asymptotic study. So, both for $\pi(\alpha)$ and $S_{\text{pos}}(\alpha)$, improved techniques must be further investigated in order to obtain more accurate estimates. For NFAs another important complexity measure is the number of transitions. It would be interesting to extend, to the automata here defined, the work of Nicaud [18] and Broda et al. [4], although with the equations derived from the definitions in Section 4.1 the resulting task seems quite cumbersome.

Sulzmann and Thiemann [19] extended the notion of Brzozowski derivative for several variants of the shuffle operator. It would also be interesting to extend the notion of partial derivative to those shuffle variants, and to carry out a descriptional complexity study of those constructions.

References

[1] Antimirov, V. M., 1996. Partial derivatives of regular expressions and finite automaton constructions. Theor. Comput. Sci. 155 (2), 291–319.

[2] Berry, G., Sethi, R., 1986. From regular expressions to deterministic automata. Theoret. Comput. Sci. 48, 117–126.

[3] Broda, S., Machiavelo, A., Moreira, N., Reis, R., 2011. On the average state complexity of partial derivative automata. International Journal of Foundations of Computer Science 22 (7), 1593–1606.

[4] Broda, S., Machiavelo, A., Moreira, N., Reis, R., 2012. On the average size of Glushkov and partial derivative automata. International Journal of Foundations of Computer Science 23 (5), 969–984.

[5] Broda, S., Machiavelo, A., Moreira, N., Reis, R., 2014. A hitchhiker's guide to descriptional complexity through analytic combinatorics. Theor. Comput. Sci. 528, 85–100.

[6] Champarnaud, J. M., Ziadi, D., 2001. From Mirkin's prebases to Antimirov's word partial derivatives. Fundam. Inform. 45 (3), 195–205.

[7] Champarnaud, J. M., Ziadi, D., 2002. Canonical derivatives, partial derivatives and finite automaton constructions. Theor. Comput. Sci. 289, 137–163.

[8] Chen, H., Yu, S., 2012. Derivatives of regular expressions and an application. In: Dinneen, M. J., Khoussainov, B., Nies, A. (Eds.), Computation, Physics and Beyond- WTCS 2012, Revised Selected and Invited Papers. Vol. 7160 of LNCS. Springer, pp. 343–356.

[9] Estrade, B. D., Perkins, A. L., Harris, J. M., 2006. Explicitly parallel regular expressions. In: Ni, J., Dongarra, J. (Eds.), 1st IMSCCS. IEEE Computer Society, pp. 402–409.

[10] FAdo, P., Access date:01.10.2014. FAdo: tools for formal languages manipulation. `http://fado.dcc.fc.up.pt/`.

[11] Flajolet, P., Sedgewick, R., 2008. Analytic Combinatorics. CUP.

[12] Gelade, W., 2010. Succinctness of regular expressions with interleaving, intersection and counting. Theor. Comput. Sci. 411 (31-33), 2987–2998.

[13] Gruber, H., 2010. On the descriptional and algorithmic complexity of regular languages. Ph.D. thesis, Justus Liebig University Giessen.

[14] Gruber, H., Holzer, M., 2008. 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. Vol. 5126 of LNCS. Springer, pp. 39–50.

[15] Kumar, A., Verma, A. K., 2014. A novel algorithm for the conversion of parallel regular expressions to non-deterministic finite automata. Applied Mathematics & Information Sciences 8, 95–105.

[16] Mayer, A. J., Stockmeyer, L. J., 1994. Word problems-this time with interleaving. Inf. Comput. 115 (2), 293–311.

[17] Mirkin, B. G., 1966. An algorithm for constructing a base in a language of regular expressions. Engineering Cybernetics 5, 51—57.

[18] Nicaud, C., 2009. On the average size of Glushkov's automata. In: Dediu, A. H., Ionescu, A.-M., Martín-Vide, C. (Eds.), Proc. 3rd LATA. Vol. 5457 of LNCS. Springer, pp. 626–637.

[19] Sulzmann, M., Thiemann, P., 2015. Derivatives for regular shuffle expressions. In: Dediu, A. H., Formenti, E., Martín-Vide, C., Truthe, B. (Eds.), Proc. 9th LATA. Vol. 8977 of LNCS. Springer, pp. 275–286.