On the Average Number of States of Partial Derivative Automata [⋆]

Sabine Broda[1], António Machiavelo[2], Nelma Moreira[1], Rogério Reis[1]
sbb@ncc.up.pt,ajmachia@fc.up.pt,{nam,rvr}@ncc.up.pt

[1] DCC-FC & LIACC, Universidade do Porto
Rua do Campo Alegre 1021/1055, 4169-007 Porto, Portugal
[2] CMUP, Universidade do Porto
Rua do Campo Alegre 687, 4169-007 Porto, Portugal

Abstract. The partial derivative automaton ($\mathcal{A}_{\mathrm{pd}}$) is usually smaller than other non-deterministic finite automata constructed from a regular expression, and it can be seen as a quotient of the Glushkov automaton ($\mathcal{A}_{\mathrm{pos}}$). By estimating the number of regular expressions that have ε as a partial derivative, we compute a lower bound of the average number of mergings of states in $\mathcal{A}_{\mathrm{pos}}$ and describe its asymptotic behaviour. This depends on the alphabet size, k, and its limit, as k goes to infinity, is $\frac{1}{2}$. The lower bound corresponds exactly to consider the $\mathcal{A}_{\mathrm{pd}}$ automaton for the marked version of the regular expression, i.e. where all its letters are made different. Experimental results suggest that the average number of states of this automaton, and of the $\mathcal{A}_{\mathrm{pd}}$ automaton for the unmarked regular expression, are very close to each other.

1 Introduction

There are several well-known constructions to obtain non-deterministic finite automata from regular expressions. The worst case analysis of both the complexity of the conversion algorithms, and the size of the resulting automata, are well studied. However, for practical purposes, the average case analysis can provide a much more useful information. Recently, Nicaud [Nic09] presented an average case study of the size of the Glushkov automata, proving that, on average, the number of transitions is linear in the size of the expression. This analysis was carried out using the framework of analytic combinatorics.

Following the same approach, in this paper we focus on the partial derivative automaton ($\mathcal{A}_{\mathrm{pd}}$), which was introduced by Antimirov [Ant96], and is a non-deterministic version of the Brzozowski automaton [Brz64]. In order to have an inductive definition of the set of states of $\mathcal{A}_{\mathrm{pd}}$, we consider Mirkin's formulation of prebases. The equivalence of the two constructions, Mirkin's prebases, and sets of partial derivatives, was pointed out by Champarnaud and Ziadi [CZ01]. We briefly revisit Mirkin's algorithm, due to an inaccuracy in that presentation.

[⋆] This work was partially funded by Fundação para a Ciência e Tecnologia (FCT) and Program POSI, and by projects RESCUE (PTDC/EIA/65862/2006) and CANTE (PTDC/EIA-CCO/101904/2008).

In 2002, Champarnaud and Ziadi [CZ02] showed that the partial derivative automaton is a quotient of the Glushkov automaton. As such, the $\mathcal{A}_{\text{pd}}$ automaton can be obtained from the $\mathcal{A}_{\text{pos}}$ automaton by merging states. The number of states in $\mathcal{A}_{\text{pd}}$, which never exceeds the number of states in $\mathcal{A}_{\text{pos}}$, appears to be significantly smaller in practice. In this work we are particularly interested in measuring this difference. By estimating the number of regular expressions that have ε as a partial derivative, we compute a lower bound for the average number of mergings of states in $\mathcal{A}_{\text{pos}}$, and study its asymptotic behaviour. This behaviour depends on the alphabet size, k, and its limit, as k goes to infinity, is half the number of states in $\mathcal{A}_{\text{pos}}$. Our experimental results suggest that this lower bound is very close to the actual value.

2 Regular Expressions and Automata

In this section we briefly review some basic definitions about regular expressions and finite automata. For more details, we refer the reader to Kozen [Koz97] or Sakarovitch [Sak09].

Let $\Sigma = \{\sigma_1, \dots, \sigma_k\}$ be an *alphabet* (set of *letters*) of size k. A *word* w over Σ is any finite sequence of letters. The *empty word* is denoted by ε. Let $\Sigma^\star$ be the set of all words over Σ. A *language* over Σ is a subset of $\Sigma^\star$. The set R of *regular expressions* over Σ is defined by:

$$\alpha := \emptyset \mid \varepsilon \mid \sigma_1 \mid \cdots \mid \sigma_k \mid (\alpha + \alpha) \mid (\alpha \cdot \alpha) \mid \alpha^\star \tag{1}$$

where the operator $\cdot$ (concatenation) is often omitted. The language $\mathcal{L}(\alpha)$ associated to α is inductively defined as follows: $\mathcal{L}(\emptyset) = \emptyset$, $\mathcal{L}(\varepsilon) = \{\varepsilon\}$, $\mathcal{L}(\sigma) = \{\sigma\}$ for $\sigma \in \Sigma$, $\mathcal{L}((\alpha + \beta)) = \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$, $\mathcal{L}((\alpha \cdot \beta)) = \mathcal{L}(\alpha) \cdot \mathcal{L}(\beta)$, and $\mathcal{L}(\alpha^\star) = \mathcal{L}(\alpha)^\star$. The *size* $|\alpha|$ of $\alpha \in \mathsf{R}$ is the number of symbols in α (parentheses not counted); the *alphabetic size* $|\alpha|_\Sigma$ is its number of letters. We define $\varepsilon(\alpha)$ by $\varepsilon(\alpha) = \varepsilon$ if $\varepsilon \in \mathcal{L}(\alpha)$, and $\varepsilon(\alpha) = \emptyset$ otherwise. If two regular expressions α and β are syntactically identical we write $\alpha \equiv \beta$. Two regular expressions α and β are *equivalent* if $\mathcal{L}(\alpha) = \mathcal{L}(\beta)$, and we write $\alpha = \beta$. With this interpretation, the algebraic structure $(\mathsf{R}, +, \cdot, \emptyset, \varepsilon)$ constitutes an idempotent semiring, and with the unary operator $\star$, a Kleene algebra.

A *non-deterministic automaton* (NFA) $\mathcal{A}$ is a quintuple $(Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is the alphabet, $\delta \subseteq Q \times \Sigma \times Q$ the transition relation, q_0 the initial state, and $F \subseteq Q$ the set of final states. The *size* of a NFA is $|Q| + |\delta|$. For $q \in Q$ and $\sigma \in \Sigma$, we denote the set $\{p \mid (q, \sigma, p) \in \delta\}$ by $\delta(q, \sigma)$, and we can extend this notation to $w \in \Sigma^\star$, and to $R \subseteq Q$. The *language* accepted by $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}) = \{w \in \Sigma^\star \mid \delta(q_0, w) \cap F \neq \emptyset\}$.

3 The Partial Derivative Automaton

Let $S \cup \{\beta\}$ be a set of regular expressions. Then $S \odot \beta = \{\alpha\beta \mid \alpha \in S\}$ if $\beta \notin \{\emptyset, \varepsilon\}$, $S \odot \emptyset = \emptyset$, and $S \odot \varepsilon = S$. Analogously, one defines $\beta \odot S$.

For a regular expression α and a letter $\sigma \in \Sigma$, the set $\partial_\sigma(\alpha)$ of *partial derivatives* of α w.r.t. σ is defined inductively as follows:

$$\begin{aligned} \partial_\sigma(\emptyset) &= \partial_\sigma(\varepsilon) = \emptyset \\ \partial_\sigma(\sigma') &= \begin{cases} \{\varepsilon\} & \text{if } \sigma' \equiv \sigma \\ \emptyset & \text{otherwise} \end{cases} \\ \partial_\sigma(\alpha^\star) &= \partial_\sigma(\alpha) \odot \alpha^\star \end{aligned} \qquad \begin{aligned} \partial_\sigma(\alpha+\beta) &= \partial_\sigma(\alpha) \cup \partial_\sigma(\beta) \\ \partial_\sigma(\alpha\beta) &= \begin{cases} \partial_\sigma(\alpha) \odot \beta \cup \partial_\sigma(\beta) & \text{if } \varepsilon(\alpha) = \varepsilon \\ \partial_\sigma(\alpha) \odot \beta & \text{otherwise.} \end{cases} \end{aligned}$$

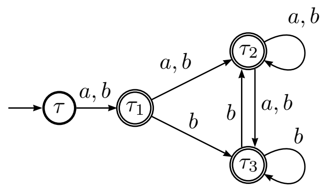
This definition can be extended to sets of regular expressions, to words, and to languages by: given $\alpha \in \mathsf{R}$ and $\sigma \in \Sigma$, $\partial_\sigma(S) = \cup_{\beta \in S} \partial_\sigma(\beta)$ for $S \subseteq \mathsf{R}$; $\partial_\varepsilon(\alpha) = \{\alpha\}$, $\partial_{w\sigma}(\alpha) = \partial_\sigma(\partial_w(\alpha))$ for $w \in \Sigma^\star$; and $\partial_L(\alpha) = \cup_{w \in L} \partial_w(\alpha)$ for $L \subseteq \Sigma^\star$. The *set of partial derivatives* of α, $\{\partial_w(\alpha) \mid w \in \Sigma^\star\}$, is denoted by $\mathrm{PD}(\alpha)$. The partial derivative automaton $\mathcal{A}_{\mathrm{pd}}(\alpha)$, introduced by Antimirov, is defined by

$$\mathcal{A}_{\mathrm{pd}}(\alpha) = (\mathrm{PD}(\alpha), \Sigma, \delta_{pd}, \alpha, \{q \in \mathrm{PD}(\alpha) \mid \varepsilon(q) = \varepsilon\}),$$

where $\delta_{pd}(q, \sigma) = \partial_\sigma(q)$, for all $q \in \mathrm{PD}(\alpha)$ and $\sigma \in \Sigma$.

Proposition 1 (Antimirov). $\mathcal{L}(\mathcal{A}_{\mathrm{pd}}(\alpha)) = \mathcal{L}(\alpha)$.

Example 1. Throughout the paper we will use the regular expression $\tau = (a + b)(a^\star + ba^\star + b^\star)^\star$, given by Ilie and Yu [IY03]. This example illustrates perfectly the purpose of our constructions in Section 5.1. For τ one has, $\mathrm{PD}(\tau) = \{\tau, \tau_1, \tau_2, \tau_3\}$, where $\tau_1 = (a^\star + ba^\star + b^\star)^\star$, $\tau_2 = a^\star \tau_1$ and $\tau_3 = b^\star \tau_1$. The corresponding automaton $\mathcal{A}_{\mathrm{pd}}(\tau)$ is the following:



3.1 Mirkin's Formulation

Champarnaud and Ziadi [CZ01] showed that partial derivatives and Mirkin's prebases [Mir66] lead to identical constructions of non-deterministic automata. In order to do this, they proposed a recursive algorithm for computing the Mirkin's prebases. However, that algorithm has an inaccuracy for the concatenation rule. Here, we give the corrected version of the algorithm.

Let α_0 be a regular expression. A set $\pi(\alpha_0) = \{\alpha_1, \ldots, \alpha_n\}$, where $\alpha_1, \ldots, \alpha_n$ are non-empty regular expressions, is called a *support* of α_0 if, for $i = 0, \ldots, n$, there are $\alpha_{il} \in \mathsf{R}$ ($l = 1, \ldots, k$), linear combinations of the elements in $\pi(\alpha_0)$, such that $\alpha_i = \sigma_1 \cdot \alpha_{i1} + \ldots + \sigma_k \cdot \alpha_{ik} + \varepsilon(\alpha_i)$, where, as above, $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ is the considered alphabet. If $\pi(\alpha)$ is a support of α, then the set $\pi(\alpha) \cup \{\alpha\}$ is called a *prebase* of α.

Proposition 2 (Mirkin/Champarnaud&Ziadi[1]). *Let α be a regular expression. Then the set $\pi(\alpha)$, inductively defined by*

$$\begin{aligned} \pi(\emptyset) &= \emptyset & \pi(\alpha+\beta) &= \pi(\alpha) \cup \pi(\beta) \\ \pi(\varepsilon) &= \emptyset & \pi(\alpha\beta) &= \pi(\alpha) \odot \beta \cup \pi(\beta) \\ \pi(\sigma) &= \{\varepsilon\} & \pi(\alpha^\star) &= \pi(\alpha) \odot \alpha^\star, \end{aligned}$$

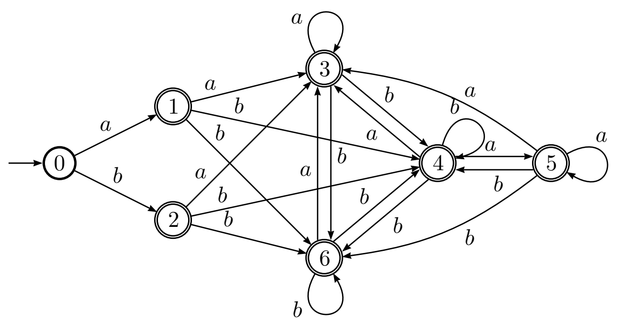
is a support of α.

In his original paper Mirkin showed that $|\pi(\alpha)| \leq |\alpha|_\Sigma$. Furthermore, Champarnaud and Ziadi established that $\mathrm{PD}(\alpha) = \pi(\alpha) \cup \{\alpha\}$. Thus $|\mathrm{PD}(\alpha)| \leq |\alpha|_\Sigma + 1$.

3.2 The Glushkov Automaton

To review the definition of the Glushkov automaton, let $\mathrm{Pos}(\alpha) = \{1, 2, \ldots, |\alpha|_\Sigma\}$ be the set of positions for $\alpha \in \mathsf{R}$, and let $\mathrm{Pos}_0(\alpha) = \mathrm{Pos}(\alpha) \cup \{0\}$. We consider the expression $\overline{\alpha}$ obtained by marking each letter with its position in α, i.e. $\mathcal{L}(\overline{\alpha}) \in \overline{\Sigma}^\star$ where $\overline{\Sigma} = \{\sigma^i \mid \sigma \in \Sigma, 1 \leq i \leq |\alpha|_\Sigma\}$. The same notation is used to remove the markings, i.e., $\overline{\overline{\alpha}} = \alpha$. For $\alpha \in \mathsf{R}$ and $i \in \mathrm{Pos}(\alpha)$, let $\mathsf{first}(\alpha) = \{i \mid \exists w \in \overline{\Sigma}^\star, \sigma^i w \in \mathcal{L}(\overline{\alpha})\}$, $\mathsf{last}(\alpha) = \{i \mid \exists w \in \overline{\Sigma}^\star, w\sigma^i \in \mathcal{L}(\overline{\alpha})\}$ and $\mathsf{follow}(\alpha, i) = \{j \mid \exists u, v \in \overline{\Sigma}^\star, u\sigma^i\sigma^j v \in \mathcal{L}(\overline{\alpha})\}$. The *Glushkov automaton* for α is $\mathcal{A}_{\mathrm{pos}}(\alpha) = (\mathrm{Pos}_0(\alpha), \Sigma, \delta_{\mathrm{pos}}, 0, F)$, with $\delta_{\mathrm{pos}} = \{(0, \overline{\sigma^j}, j) \mid j \in \mathsf{first}(\alpha)\} \cup \{(i, \overline{\sigma^j}, j) \mid j \in \mathsf{follow}(\alpha, i)\}$ and $F = \mathsf{last}(\alpha) \cup \{0\}$ if $\varepsilon(\alpha) = \varepsilon$, and $F = \mathsf{last}(\alpha)$, otherwise. We note that the number of states of $\mathcal{A}_{\mathrm{pos}}(\alpha)$ is exactly $|\alpha|_\Sigma + 1$. Champarnaud and Ziadi [CZ02] showed that the partial derivative automaton is a quotient of the Glushkov automaton. The right-invariant equivalence relation used in showing that the $\mathcal{A}_{\mathrm{pd}}$ is a quotient of $\mathcal{A}_{\mathrm{pos}}$ relates the sets first and last with (multi-)sets of partial derivatives w.r.t a letter.

Example 2. The Glushkov automaton for τ, $\mathcal{A}_{\mathrm{pos}}(\tau)$, is the following:



[1] The rule for concatenation in [CZ01] is $\pi(\alpha\beta) = \pi(\alpha) \odot \beta \cup \varepsilon(\alpha) \odot \pi(\beta)$, which, e.g., produces $\pi(ab) = \{b\}$. But, $\mathrm{PD}(ab) = \{ab, b, \varepsilon\}$, thus $\pi(ab) \supseteq \{b, \varepsilon\}$.

4 Generating Functions and Analytic Methods

A *combinatorial class* C is a set of objects on which a non-negative integer function (size) $|\cdot|$ is defined, and such that for each $n \geq 0$, the number of objects of size n, c_n, is finite. The *generating function* $C(z)$ of C is the formal power series

$$C(z) = \sum_{c \in \mathsf{C}} z^{|c|} = \sum_{n=0}^{\infty} c_n z^n.$$

The symbolic method (Flajolet and Sedgewick [FS08]) is a framework that allows the construction of a combinatorial class C in terms of simpler ones, $\mathsf{B}_1, \ldots, \mathsf{B}_n$, by means of specific operations, and such that the generating function $C(z)$ of C is a function of the generating functions $B_i(z)$ of B_i, for $1 \leq i \leq n$. For example, given two disjoint combinatorial classes A and B, with generating functions $A(z)$ and $B(z)$, respectively, the union $A \cup B$ is a combinatorial class whose generating function is $A(z) + B(z)$. Other usual admissible operations are the cartesian product and the Kleene closure.

Usually multivariate generating functions are used in order to obtain estimates about the asymptotic behaviour of various parameters associated to combinatorial classes. Here, however, we consider *cost generating functions*, as Nicaud did. Given $f : \mathsf{C} \to \mathbb{N}$, the *cost generating function* $F(z)$ of C associated to f is $F(z) = \sum_{c \in \mathsf{C}} f(c) z^{|c|} = \sum_{n \geq 0} f_n z^n$, with $f_n = \sum_{c \in \mathsf{C}, |c| = n} f(c)$. With $[z^n]F(z)$ denoting the coefficient of z^n, the average value of f for the uniform distribution on the elements of size n of C is, obviously,

$$\mu_n(\mathsf{C}, f) = \frac{[z^n]F(z)}{[z^n]C(z)}.$$

For the regular expressions given in (1), but without $\emptyset$, an average case analysis of different descriptional measures, including the number of letters or the size of its Glushkov automaton, has been presented by Nicaud. In particular, it was shown that, for the generating function for regular expressions, $R_k(z)$, which satisfies

$$R_k(z) = \frac{1 - z - \sqrt{\Delta_k(z)}}{4z}, \text{ where } \Delta_k(z) = 1 - 2z - (7 + 8k)z^2, \tag{2}$$

one has

$$[z^n]R_k(z) \sim \frac{\sqrt{2(1 - \rho_k)}}{8\rho_k \sqrt{\pi}} \rho_k^n n^{-3/2}, \text{ where } \rho_k = \frac{1}{1 + \sqrt{8k + 8}}. \tag{3}$$

Here $[z^n]R_k(z)$ is the number of regular expressions α with $|\alpha| = n$.

Nicaud also showed that the cost generating function for the number of letters in an element $\alpha \in \mathsf{R}$ is

$$L_k(z) = \frac{kz}{\sqrt{\Delta_k(z)}}, \tag{4}$$

and satisfies

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

From this he deduced that

$$\frac{[z^n]L_k(z)}{[z^n]R_k(z)} \sim \frac{4k\rho_k^2}{1-\rho_k}n. \tag{6}$$

For $k = 2$ this results in approximately $0.277n$ (and not $0.408n$, as stated by Nicaud), and it is easy to see that

$$\lim_{k\to\infty} \frac{4k\rho_k^2}{1-\rho_k} \nearrow \frac{1}{2}. \tag{7}$$

This means that the average number of letters in a regular expression grows to about half its size, for large alphabets. In particular, for $k = 10, 100, 1000$ we have $\frac{4k\rho_k^2}{1-\rho_k} = 0.467, 0.485, 0.494$ respectively.

4.1 Analytic Asymptotics

Generating functions can be seen as complex analytic functions, and the study of their behaviour around their dominant singularities gives us access to the asymptotic form of their coefficients. We refer the reader to Flajolet and Sedgewick for an extensive study on this topic. Here we only state the propositions and lemmas used in this paper. Let $R > 1$ and $0 < \phi < \pi/2$ be two real numbers, the *domain* $\Delta(\phi, R)$ at $z = \xi$ is $\Delta(\phi, R) = \{z \in \mathbb{C} \mid |z| < R,\ z \neq \xi, \text{ and } |Arg(z - \xi)| > \phi\}$, where $Arg(z)$ denotes the argument of $z \in \mathbb{C}$. A *domain* is a Δ-*domain* at ξ if it is a $\Delta(\phi, R)$ at ξ for some R and ϕ. The generating functions we consider have always a unique dominant singularity, and satisfy one of the two conditions of the following proposition, given by Nicaud.

Proposition 3. *Let $f(z)$ be a function that is analytic in some Δ-domain at $\rho \in \mathbb{R}^+$. If at the intersection of a neighborhood of ρ and its Δ-domain,*

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

The following straightforward lemma was used though out our analytic computations.

Lemma 1. *If $f(z)$ is an entire function with $\lim_{z\to\rho} f(z) = a$ and $r \in \mathbb{R}$, then*

$$f(z)(1 - z/\rho)^r = a(1 - z/\rho)^r + o((1 - z/\rho)^r).$$

5 The Average Number of State Mergings

5.1 Regular Expressions with ε as a Partial Derivative

Since $\mathcal{A}_{\mathrm{pd}}(\alpha)$ is a quotient of the Glushkov automaton, we know that it has at most $|\alpha|_{\Sigma}+1$ states. But this upper bound is reached if and only if, in every step, during the computation of $\pi(\alpha)$, all unions are disjoint. There are however two cases in which this clearly does not happen. Whenever $\varepsilon \in \pi(\beta) \cap \pi(\gamma)$,

$$|\pi(\beta+\gamma)| = |\pi(\beta) \cup \pi(\gamma)| \leq |\pi(\beta)| + |\pi(\gamma)| - 1, \tag{8}$$

and also

$$\begin{aligned} |\pi(\beta\gamma^{\star})| = |\pi(\beta) \odot \gamma^{\star} \cup \pi(\gamma^{\star})| &= |\pi(\beta) \odot \gamma^{\star} \cup \pi(\gamma) \odot \gamma^{\star}| \\ &\leq |\pi(\beta)| + |\pi(\gamma)| - 1. \end{aligned} \tag{9}$$

In this section we will estimate the number of non-disjoint unions formed during the computation of $\pi(\alpha)$, that are due to either one of the above cases. This corresponds to the merging of states in the Glushkov automaton. Notice that there might be additional mergings resulting from other identical elements in the support of the regular expressions. Therefore our estimation is only a lower bound of the actual number of state mergings, that turns out to be surprisingly tight as shown in Section 6.

Example 3. In order to illustrate the effect of ε being a partial derivative of a subexpression, we consider the marked version of τ, $\overline{\tau} = (a_1+b_2)(a_3^{\star}+b_4a_5^{\star}+b_6^{\star})^{\star}$. In $\mathcal{A}_{\mathrm{pos}}(\tau)$ each position corresponds to a state. Now, note that, for instance, one has

$$\pi(a_1+a_2) = \pi(a_1) \cup \pi(a_2) = \{\varepsilon\} \cup \{\varepsilon\} = \{\varepsilon\},$$

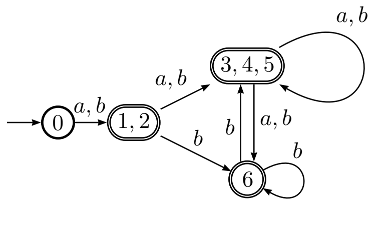
and

$$\pi(b_4a_5^{\star}) = \pi(b_4) \odot a_5^{\star} \cup \pi(a_5) \odot a_5^{\star} = \{\varepsilon\} \odot a_5^{\star} \cup \{\varepsilon\} \odot a_5^{\star}.$$

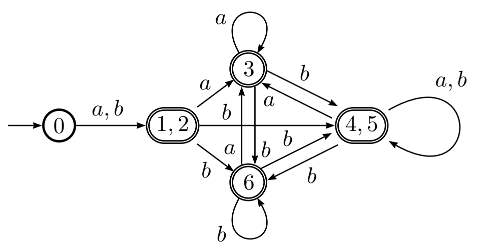
These two cases originate the mergings of states 1 and 2, as well as 4 and 5 of $\mathcal{A}_{\mathrm{pos}}(\tau)$. There is another state merging that is due to neither of the above cases. In fact,

$$\pi(a_3^{\star}+b_4a_5^{\star}) = \pi(a_3) \odot a_3^{\star} \cup \pi(b_4) \odot a_5^{\star} = \{\varepsilon\} \odot a_3^{\star} \cup \{\varepsilon\} \odot a_5^{\star}.$$

Since the $\mathcal{A}_{\mathrm{pd}}(\tau)$ is computed for the unmarked τ, there is also the merging of states 3 and 4. This is not the case for $\mathcal{A}_{\mathrm{pd}}(\overline{\tau})$, as can be seen in the following diagrams:



$\mathcal{A}_{\mathrm{pd}}(\tau)$



$\mathcal{A}_{\mathrm{pd}}(\overline{\tau})$

As this example suggests, the lower bound for the number of mergings of states that is computed in this paper is precisely the number of mergings that arise when obtaining $\mathcal{A}_{\mathrm{pd}}(\overline{\alpha})$ from $\mathcal{A}_{\mathrm{pos}}(\alpha)$.

From now on, α will denote regular expressions given in (1), but without $\emptyset$, and its generating function, $R_k(z)$ is given by (2). As mentioned, the number of mergings for an expression α depends on the number of subexpressions with ε in its support. We will estimate this number first. The grammar

$$\alpha_\varepsilon := \sigma \in \Sigma \mid \alpha_\varepsilon + \alpha \mid \alpha_{\overline{\varepsilon}} + \alpha_\varepsilon \mid \alpha \cdot \alpha_\varepsilon \mid \alpha_\varepsilon \cdot \varepsilon$$

generates the set of regular expressions for which $\varepsilon \in \pi(\alpha_\varepsilon)$, that is denoted by R_ε. The remaining regular expressions, that are not generated by this grammar, are denoted by $\alpha_{\overline{\varepsilon}}$.
The generating function for R_ε, $R_{\varepsilon,k}(z)$, satisfies

$$\begin{aligned} R_{\varepsilon,k}(z) = kz + zR_{\varepsilon,k}(z)R_k(z) + z\left(R_k(z) - R_{\varepsilon,k}(z)\right)R_{\varepsilon,k}(z) \\ + zR_k(z)R_{\varepsilon,k}(z) + z^2R_{\varepsilon,k}(z), \end{aligned}$$

which is equivalent to

$$zR_{\varepsilon,k}(z)^2 - \left(z^2 + 3zR_k(z) - 1\right)R_{\varepsilon,k}(z) - kz = 0,$$

and from which one gets

$$R_{\varepsilon,k}(z) = \frac{\left(z^2 + 3zR_k(z) - 1\right) + \sqrt{\left(z^2 + 3zR_k(z) - 1\right)^2 + 4kz^2}}{2z}. \tag{10}$$

One has

$$8zR_{\varepsilon,k}(z) = -b(z) - 3\sqrt{\Delta_k(z)} + \sqrt{a_k(z) + 6b(z)\sqrt{\Delta_k(z)} + 9\Delta_k(z)}, \tag{11}$$

with

$$\begin{aligned} a_k(z) &= 16z^4 - 24z^3 + (64k+1)z^2 + 6z + 1 \\ b(z) &= -4z^2 + 3z + 1. \end{aligned} \tag{12}$$

and $\Delta_k(z)$ as in (2). Using the binomial theorem, Lemma 1 and Proposition 3, one gets

$$[z^n]R_{\varepsilon,k}(z) \sim \frac{3}{16\sqrt{\pi}}\left(1 - \frac{b(\rho_k)}{\sqrt{a_k(\rho_k)}}\right)\sqrt{2(1-\rho_k)}\,\rho_k^{-(n+1)}n^{-3/2}. \tag{13}$$

Therefore

$$\frac{[z^n]R_{\varepsilon,k}(z)}{[z^n]R_k(z)} \sim \frac{3}{2}\left(1 - \frac{b(\rho_k)}{\sqrt{a_k(\rho_k)}}\right). \tag{14}$$

Note that $\lim\limits_{k\to\infty}\rho_k = 0$, $\lim\limits_{k\to\infty}a_k(\rho_k) = 9$ and $\lim\limits_{k\to\infty}b(\rho_k) = 1$, and so the asymptotic ratio of regular expressions with ε on their derivatives approaches 1 as $k \to \infty$.

5.2 The Generating Function of Mergings

Let $i(\alpha)$ be the number of non-disjoint unions appearing, due to (8) or (9), during the computation of $\pi(\alpha)$, $\alpha \in \mathsf{R}$. These correspond to state mergings of Glushkov automata. Splitting the regular expressions into the disjoint classes α_ε and $\alpha_{\overline{\varepsilon}}$, $i(\alpha)$ verifies

$$\begin{aligned}
i(\varepsilon) &= 0\\
i(\sigma) &= 0\\
i(\alpha_\varepsilon + \alpha_\varepsilon) &= i(\alpha_\varepsilon) + i(\alpha_\varepsilon) + 1\\
i(\alpha_\varepsilon + \alpha_{\overline{\varepsilon}}) &= i(\alpha_\varepsilon) + i(\alpha_{\overline{\varepsilon}})\\
i(\alpha_{\overline{\varepsilon}} + \alpha) &= i(\alpha_{\overline{\varepsilon}}) + i(\alpha)\\
i(\alpha_\varepsilon \cdot \alpha_\varepsilon^\star) &= i(\alpha_\varepsilon) + i(\alpha_\varepsilon) + 1\\
i(\alpha_\varepsilon \cdot \overline{\alpha_\varepsilon^\star}) &= i(\alpha_\varepsilon) + i(\overline{\alpha_\varepsilon^\star})\\
i(\alpha_{\overline{\varepsilon}} \cdot \alpha) &= i(\alpha_{\overline{\varepsilon}}) + i(\alpha)\\
i(\alpha^\star) &= i(\alpha),
\end{aligned}$$

where $\overline{\alpha_\varepsilon^\star}$ denotes regular expressions that are not of the form $\alpha_\varepsilon^\star$. Clearly, the generating function for these expressions is $R_k(z) - zR_{\varepsilon,k}(z)$.

The cost generating function of the mergings, $I_k(z)$, can now be obtained from these equations by adding the contributions of each single one of them. These contributions can be computed as here exemplified for the contribution of the regular expressions of the form $(\alpha_\varepsilon + \alpha_\varepsilon)$:

$$\begin{aligned}
\sum_{(\alpha_\varepsilon+\alpha_\varepsilon)} i(\alpha_\varepsilon + \alpha_\varepsilon) z^{|(\alpha_\varepsilon+\alpha_\varepsilon)|} &= z \sum_{\alpha_\varepsilon} \sum_{\alpha_\varepsilon} (i(\alpha_\varepsilon) + i(\alpha_\varepsilon) + 1) z^{|\alpha_\varepsilon|} z^{|\alpha_\varepsilon|}\\
&= z \sum_{\alpha_\varepsilon} \sum_{\alpha_\varepsilon} (i(\alpha_\varepsilon) + i(\alpha_\varepsilon)) z^{|\alpha_\varepsilon|} z^{|\alpha_\varepsilon|}\\
&\qquad + z \sum_{\alpha_\varepsilon} \sum_{\alpha_\varepsilon} z^{|\alpha_\varepsilon|} z^{|\alpha_\varepsilon|}\\
&= 2zI_{\varepsilon,k}(z) R_{\varepsilon,k}(z) + zR_{\varepsilon,k}(z)^2,
\end{aligned}$$

where $I_{\varepsilon,k}(z)$ is the generating function for the mergings coming from α_ε.

Applying this technique to the remaining cases, we obtain

$$I_k(z) = \frac{(z+z^2)R_{\varepsilon,k}(z)^2}{\sqrt{\Delta_k(z)}}. \tag{15}$$

The asymptotic value of the coefficients of this generating function can now be computed using (11), and again Lemma 1 and Proposition 3, yielding

$$[z^n]I_k(z) \sim \frac{1+\rho_k}{64} \frac{\left(a_k(\rho_k) + b(\rho_k)^2 - 2b(\rho_k)\sqrt{a_k(\rho_k)}\right)}{\sqrt{\pi}\sqrt{2-2\rho_k}} \rho_k^{-(n+1)} n^{-1/2}. \tag{16}$$

$n =$	10	20	50	100	200	400
$k = 2$	1.34	1.14	1.05	1.03	1.01	1.01
$k = 3$	1.35	1.12	1.05	1.02	1.01	1.01
$k = 5$	1.38	1.12	1.04	1.02	1.01	1.01
$k = 10$	—	1.13	1.04	1.02	1.01	1.01
$k = 20$	—	—	1.04	1.02	1.01	1.01
$k = 50$	—	—	—	1.02	1.01	1.01
$k = 100$	—	—	—	—	1.01	1.04

Table 1. Accuracy of the approximation

Table 1 exhibits the ratio between the approximation given by this computation and the actual coefficients of the power series of $I_k(z)$, for several values of k and n.

From (3) and (16) one easily gets the following asymptotic estimate for the average number of mergings

$$\frac{[z^n]I_k(z)}{[z^n]R_k(z)} \sim \lambda_k\, n, \tag{17}$$

where $\lambda_k = \frac{(1+\rho_k)}{16(1-\rho_k)}\left(a_k(\rho_k) + b(\rho_k)^2 - 2b(\rho_k)\sqrt{a_k(\rho_k)}\right)$. Using again the fact that $\lim\limits_{k\to\infty} \rho_k = 0$, $\lim\limits_{k\to\infty} a_k(\rho_k) = 9$ and $\lim\limits_{k\to\infty} b(\rho_k) = 1$, one gets that

$$\lim_{k\to\infty} \lambda_k = \frac{1}{4}.$$

This means that, for a regular expression of size n, the average number of state mergings is, asymptotically, about $\frac{n}{4}$.

In order to obtain a lower bound for the reduction in the number of states of the $\mathcal{A}_{\text{pd}}$ automaton, as compared to the ones of the $\mathcal{A}_{\text{pos}}$ automaton, it is enough to compare the number of mergings for an expression α with the number of letters in α. From (5) and (17) one gets

$$\frac{[z^n]I_k(z)}{[z^n]L_k(z)} \sim \frac{1-\rho_k}{4k\rho_k^2}\lambda_k. \tag{18}$$

It is easy to see that

$$\lim_{k\to\infty} \frac{1-\rho_k}{4k\rho_k^2}\lambda_k = \frac{1}{2}.$$

In other words, asymptotically, the average number of states of the $\mathcal{A}_{\text{pd}}$ automaton is about one half of the number of states of the $\mathcal{A}_{\text{pos}}$ automaton, and about one quarter of the size of the corresponding regular expression, by (7). As shown in Table 2 the actual values are close to these limits already for small alphabets.

6 Comparison with Experimental Results

In order to compare our estimates with the actual number of states in a $\mathcal{A}_{\text{pd}}$ automaton we ran some experiments. We used the FAdo library [AAA$^+$09,fad09], that includes algorithms for computing the Glushkov automaton and the partial derivatives automaton corresponding to a given regular expression. For the results to be statistically significant, regular expressions must be uniformly random generated. The FAdo library implements the method described by Mairson [Mai94] for the uniform random generation of context-free languages. The random generator has as input a grammar and the size of the words to be generated. To obtain regular expressions uniformly generated in the size of the syntactic tree (i.e. parentheses not counted), a prefix notation version of the grammar (1) was used. For each size, n, samples of 1000 regular expressions were generated. Table 2 presents the average values obtained for $n \in \{1000, 2000\}$ and $k \in \{2, 3, 5, 10, 20, 30, 50\}$, and the two last columns give the asymptotic ratios obtained in (17) and (18) for the corresponding values of k.

k	$\lvert\alpha\rvert$	$\lvert\alpha\rvert_\Sigma$	$\lvert\mathrm{Pos}_0\rvert$	$\lvert\delta_{\mathrm{pos}}\rvert$	$\lvert\mathrm{PD}\rvert$	$\lvert\delta_{\mathrm{pd}}\rvert$	$\lvert\overline{\mathrm{PD}}\rvert$	$\frac{\lvert\alpha\rvert_\Sigma-\lvert\mathrm{PD}\rvert}{\lvert\alpha\rvert_\Sigma}$	$\frac{\lvert\alpha\rvert_\Sigma}{\lvert\alpha\rvert}$	$\frac{[z^n]I(z)}{n\times[z^n]R(z)}$	$\frac{[z^n]I(z)}{[z^n]L(z)}$
2	1000	276	277	3345	187	1806	190	**0.323**	0.276	0.084	**0.304**
2	2000	553	554	7405	374	3951	380	**0.324**	0.277		
3	1000	318	319	2997	206	1564	208	**0.352**	0.318	0.107	**0.337**
3	2000	638	639	6561	410	3380	416	**0.357**	0.319		
5	1000	364	365	2663	223	1339	226	**0.387**	0.364	0.135	**0.372**
5	2000	728	729	5535	446	2768	451	**0.387**	0.364		
10	1000	405	406	2203	236	1079	238	**0.417**	0.405	0.168	**0.409**
10	2000	809	810	4616	471	2235	475	**0.418**	0.405		
20	1000	440	441	1842	245	875	246	**0.443**	0.44	0.192	**0.435**
20	2000	880	881	3735	489	1768	492	**0.444**	0.44		
30	1000	453	454	1676	247	796	248	**0.455**	0.453	0.203	**0.447**
30	2000	906	907	3380	496	1603	498	**0.453**	0.453		
50	1000	466	467	1516	250	718	251	**0.464**	0.466	0.214	**0.459**
50	2000	933	934	3065	499	1441	500	**0.465**	0.467		
100	—	—	—	—	—	—	—	—	—	0.225	**0.471**
1000	—	—	—	—	—	—	—	—	—	0.242	**0.491**

Table 2. Experimental results for uniform random generated regular expressions

As can be seen from the columns with bold entries, the asymptotic averages obtained with the analytic methods are very close to the values obtained experimentally. In general, even for small values of n, the ratio of the number of states of $\mathcal{A}_{\mathrm{pd}}$ to the number of states of $\mathcal{A}_{\mathrm{pos}}$ coincide (within an error of less than 3%) with our (asymptotic) estimates. These results indicate that occurrences of ε in the set of partial derivatives are the main reason for a smaller number of states in the $\mathcal{A}_{\mathrm{pd}}$ automaton, when compared with the one in the $\mathcal{A}_{\mathrm{pos}}$ automaton. This is confirmed by comparing the column containing the number of states of $\mathcal{A}_{\mathrm{pd}}$ ($\lvert\mathrm{PD}\rvert$) with the one containing those of its marked version ($\lvert\overline{\mathrm{PD}}\rvert$).

7 Final remarks

In this paper we studied, using analytic methods, the average number of states of partial derivative automata. We proved this number to be, on average, half the number of states when considering the Glushkov automata case. An approach similar to the one applied here can be used to estimate the average number of transitions of $\mathcal{A}_{\mathrm{pd}}$. According to Table 2, this number also seems to be half the number of transitions of $\mathcal{A}_{\mathrm{pos}}$. At first sight, one would expect that the use of alternative grammars for the generation of regular expressions, with less redundancy, such as the ones presented by Lee and Shallit [LS05], would lead to different results. However, experimental studies do not support this expectation, since they do not show significant differences from the results here presented.

References

[AAA$^+$09] A. Almeida, M. Almeida, J. Alves, N. Moreira, and R. Reis. FAdo and GUItar: tools for automata manipulation and visualization. In S. Maneth, editor, *14th CIAA 2009*, volume 5642 of *LNCS*, pages 65–74. Springer-Verlag, 2009.

[Ant96] V. M. Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *Theoret. Comput. Sci.*, 155(2):291–319, 1996.

[Brz64] J. A. Brzozowski. Derivatives of regular expressions. *JACM*, 11(4):481–494, October 1964.

[CZ01] J. M. Champarnaud and D. Ziadi. From Mirkin's prebases to Antimirov's word partial derivatives. *Fundam. Inform.*, 45(3):195–205, 2001.

[CZ02] J. M. Champarnaud and D. Ziadi. Canonical derivatives, partial derivatives and finite automaton constructions. *Theoret. Comput. Sci.*, 289:137–163, 2002.

[fad09] FAdo: tools for formal languages manipulation. `http://www.ncc.up.pt/FAdo`, Access date:1.12.2009.

[FS08] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. CUP, 2008.

[IY03] L. Ilie and S. Yu. Follow automata. *Inf. Comput.*, 186(1):140–162, 2003.

[Koz97] D. C. Kozen. *Automata and Computability*. Undergrad. Texts in Computer Science. Springer-Verlag, 1997.

[LS05] J. Lee and J. Shallit. Enumerating regular expressions and their languages. In M. Domaratzki, A. Okhotin, K. Salomaa, and S. Yu, editors, *9th CIAA 2004*, volume 3314 of *LNCS*, pages 2–22. Springer-Verlag, 2005.

[Mai94] H. G. Mairson. Generating words in a context-free language uniformly at random. *Information Processing Letters*, 49:95–99, 1994.

[Mir66] B. G. Mirkin. An algorithm for constructing a base in a language of regular expressions. *Engineering Cybernetics*, 5:51—57, 1966.

[Nic09] C. Nicaud. On the average size of Glushkov's automata. In A. Dediu, A.-M.i Ionescu, and C. M. Vide, editors, *3rd LATA 2009*, volume 5457 of *LNCS*, pages 626–637. Springer-Verlag, 2009.

[Sak09] J. Sakarovitch. *Elements of Automata Theory*. CUP, 2009.