The Formal Language Theory Column

by

Giovanni Pighizzini

Dipartimento di Informatica
Università degli Studi di Milano
20135 Milano, Italy
`pighizzini@di.unimi.it`

Average Size of Automata Constructions from Regular Expressions[*]

Sabine Broda[†]
sbb@dcc.fc.up.pt

António Machiavelo[†]
ajmachia@fc.up.pt

Nelma Moreira[†]
nam@dcc.fc.up.pt

Rogério Reis[†]
rvr@dcc.fc.up.pt

Abstract

Because of their succinctness and clear syntax, regular expressions are the common choice to represent regular languages. Deterministic finite automata are an excellent representation for testing equivalence, containment or membership, as these problems are easily solved for this model. However, minimal deterministic finite automata can be exponentially larger than the associated regular expression, while the corresponding nondeterministic finite automata can be linearly larger. The worst case of both the complexity of the conversion algorithms, and of the size of the resulting automata, are well studied. However, for practical purposes, estimates for the average case can provide much more useful information. In this paper we review recent results on the average size of automata resulting from several constructions and suggest several directions of research. Most results were obtained within the framework of analytic combinatorics.

1 Introduction

The methods to convert regular expressions (REs) into equivalent automata can be divided in three major classes, depending on whether the resulting automaton is deterministic (DFA), nondeterministic without ε-transitions (NFA) or nondeterministic with ε-transitions (ε-NFA). Paradigmatic methods of each class are Brzozowski's [13], Glushkov's [21] and Thompson's [47] constructions, respectively.

[*]This work was partially supported by CMUP (UID/MAT/00144/2013), which is funded by FCT (Portugal) with national (MEC) and European structural funds through the programs FEDER, under the partnership agreement PT2020.

[†]CMUP, DCC & DM, Faculdade de Ciências da Universidade do Porto, Portugal.

Brzozowski's method introduces the notion of derivative of a regular expression with respect to a symbol, a syntactic equivalent of the notion of left quotient for languages. It is well-known that regular languages have a finite number of left-quotients. To obtain a finite number of derivatives, it is necessary to consider regular expressions modulo some equational axioms, namely the associativity, commutativity and idempotence of union (ACI). A nondeterministic version of derivatives was introduced by Mirkin [35] and Antimirov [2]. Instead of a derivative being a regular expression, a set of regular expressions (partial derivatives) is considered. This avoids the necessity of using derivatives modulo equational axioms, but the associated construction (partial derivative automata) yields NFAs instead of DFAs.

Glushkov construction uses the positions of the letters occurring in a regular expression. The partial derivative automaton is a quotient of the Glushkov (or position) automaton, and very often is its smallest quotient [15, 22, 31]. If ε-transitions are eliminated from the Thompson automaton, the Glushkov automaton is obtained. Finally, the determination of the Glushkov automaton (by subset construction) produces the McNaughton-Yamada DFA [34].

The worst case of both the complexity of the conversion algorithms, and of the size of the resulting automata, are well studied [12, 16, 15] (see also [25] for a survey). However, for practical purposes, an estimate for the average case constitute a much more useful information.

In this paper we review recent results on the average size of automata resulting from several constructions, and suggest several directions of research. Most results were obtained within the framework of analytic combinatorics, a powerful tool for asymptotic average analysis, by relating the enumeration of combinatorial objects to the algebraic and complex analytic properties of generating functions. An introduction to this method, and a derivation of the asymptotic average size of several conversions between regular expressions and ε-NFAs, can be found in Broda *et al* [9]. Another approach to average complexity is to consider uniform random generators and to perform statistically significant experiments. The drawback of this approach is that it only gives results for a small range of object sizes and, due to their combinatorial nature, only modest ranges can usually be considered. However, whenever we refer to experimental results we mean results obtained in this limited context. Both in experimental and analytic results we consider the average with respect to the uniform distribution.

In the conversions from regular expressions to NFAs without ε-transitions, although position based methods can provide recursive definitions that endow analytic analysis, we will emphasise the role of derivatives when considering extended regular expressions, or other algebraic structures such as Kleene algebras with tests.

We briefly review some basic definitions about regular expressions and finite automata. For more details, we refer the reader to Kozen [27] or Sakarovitch [44]. The set R of *regular expressions* over a finite alphabet Σ is the smallest set containing $\emptyset$ and all the expressions generated by the following grammar:

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

where $\sigma_i \in \Sigma$ are *letters* and the operator $\cdot$ (concatenation) is often omitted. The language $\mathcal{L}(\alpha) \subseteq \Sigma^\star$ associated to α is inductively defined as $\mathcal{L}(\varepsilon) = \{\varepsilon\}$, $\mathcal{L}(\sigma) = \{\sigma\}$ 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 $\mathcal{L}(\alpha^\star) = \mathcal{L}(\alpha)^\star$. Also, $\mathcal{L}(\emptyset) = \emptyset$. The *size* $|\alpha|$ of $\alpha \in \mathsf{R}$ is the number of symbols in α, where parentheses are not counted; the *alphabetic size* $|\alpha|_\Sigma$ is its number of letter occurrences. We define $\varepsilon(\alpha)$ as $\varepsilon(\alpha) = \varepsilon$ if $\varepsilon \in \mathcal{L}(\alpha)$, and $\varepsilon(\alpha) = \emptyset$ otherwise. 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 *, a Kleene algebra (KA). Given a language $L \subseteq \Sigma^\star$ and a word $w \in \Sigma^\star$, the *left-quotient* of L w.r.t. w is the language $w^{-1}L = \{\ x \mid wx \in L\ \}$. A *nondeterministic finite automaton* (NFA) is a tuple $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ where Q is a finite set of states, Σ is the alphabet, $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ is the transition relation, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final states. When $I = \{q_0\}$, we just write q_0 instead of $\{q_0\}$. The *size* of an NFA, $\mathcal{A}$, is $|\mathcal{A}| = |Q| + |\delta|$, the number of states $|\mathcal{A}|_Q = |Q|$, and the number of transitions $|\mathcal{A}|_\delta = |\delta|$. An NFA that has transitions labelled with ε is an ε-NFA. An NFA is *deterministic* (DFA) if $|I| = 1$ and for each pair $(q, \sigma) \in Q \times \Sigma$ there exists at most one q' such that $(q, \sigma, q') \in \delta$. The *language* accepted by an automaton $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}) = \{\ w \in \Sigma^* \mid \delta(I, w) \cap F \neq \emptyset\ \}$. An equivalence relation E over Q is *right invariant* (or a *bisimulation*) if $E \subseteq (Q \setminus F)^2 \cup F^2$, and for any $p, q \in Q$, $\sigma \in \Sigma$ if $p\ E\ q$, then[1] $\delta(p,\sigma)/_E = \delta(q,\sigma)/_E$. The quotient automaton $\mathcal{A}/_E = (Q/_E, \Sigma, \delta_E, [q_0]_E, F/_E)$, where $\delta_E = \{\ ([p]_E, \sigma, [q]_E) \mid (p, \sigma, q) \in \delta\ \}$, satisfies $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}/_E)$. The largest bisimulation, *i.e.* the union of all bisimulation relations on Q, is called *bisimilarity* ($\equiv_b$).

2 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 $C(z) = \sum_{c \in \mathsf{C}} z^{|c|} = \sum_{n=0}^{\infty} c_n z^n$. We let $[z^n]C(z)$ denote the coefficient of z^n, c_n. The symbolic method [19] is a framework that allows to express a combinatorial class C in terms of simpler ones, $\mathsf{B}_1, \ldots, \mathsf{B}_n$, by means of specific operations, yielding

[1] Denoting by $S/_E$ the set $\{\ [s]_E \mid s \in S\ \}$.

the generating function $C(z)$ as 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.

Multivariate generating functions are used in order to obtain estimates about the asymptotic behaviour of parameters associated to combinatorial classes. Considering t weighting functions, $p_i : C \to \mathbb{C}$, for $1 \leq i \leq t$, let $c_{k_1,\ldots,k_t,n}$ be the number of objects c of size n with $p_1(c) = k_1, \ldots, p_t(c) = k_t$, one has the following multivariate weighting generating function

$$C(u_1, \ldots, u_t, z) = \sum_{n,k_1,\ldots,k_t \geq 0} c_{k_1,\ldots,k_t,n}\, u_1^{k_1} \cdots u_t^{k_t} z^n.$$

The functions $(p_i)_i$ give the respective weights of t features under consideration. Note that since C is a combinatorial class, the number of objects with a given size is finite, and therefore, for a fixed n, there is only a finite number of $c_{k_1,\ldots,k_t,n}$ which are different from 0. Also,

$$\left.\frac{\partial C(u_1, \ldots, u_t, z)}{\partial u_i}\right|_{u_i=1} = \sum_{\substack{n,k_j \geq 0\\ j \neq i}} \left(\sum_{k_i \geq 0} k_i c_{k_1,\ldots,k_t,n} \right) u_1^{k_1} \cdots u_{i-1}^{k_{i-1}} u_{i+1}^{k_{i+1}} \cdots u_t^{k_t} z^n,$$

where $\sum_{k_i \geq 0} k_i c_{k_1,\ldots,k_t,n}$ accounts for the cumulative presence of weight p_i in the objects of size n.

2.1 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 an asymptotic estimate for their coefficients. We refer the reader to Flajolet and Sedgewick for an extensive study on this topic. Here we only state the results relevant for this paper. For $\rho \in \mathbb{C}$, $R > 1$ and $0 < \phi < \pi/2$, consider the domain $\Delta(\rho, \phi, R) = \{\ z \in \mathbb{C} \mid |z| < R,\ z \neq \rho, \text{ and } |Arg(z - \rho)| > \phi\ \}$, where $Arg(z)$ denotes the argument of $z \in \mathbb{C}$. A region is a Δ-*domain* at ρ if it is a $\Delta(\rho, \phi, R)$, 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, used by Nicaud [37].

Proposition 1. *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*

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

2.2 Generating Functions for Regular Expressions

For the regular expressions given in (1), an average case analysis of different descriptional measures, including the number of letters, has been presented by Nicaud [36, 37]. Here we introduce some of those results, deriving them in a slight different way and using a parametrised weighted generating function based on Broda et al. [9].

Using the recursive definition of R given by (1), the associated generating function $R_k(z)$ satisfies $R_k(z) = (k+1)z + zR_k(z) + 2zR_k(z)^2$. Solving this equation for $R_k(z)$, and considering that $R_k(0) = a_0 = 0$, one obtains $R_k(z) = \frac{1-z-\sqrt{\Delta_k(z)}}{4z}$, where $\Delta_k(z) = 1 - 2z - (7+8k)z^2$. The zeros of $\Delta_k(z)$ are $\rho_k = \frac{1}{1+\sqrt{8k+8}}$ and $\bar{\rho}_k = \frac{1}{1-\sqrt{8k+8}}$. The coefficients of the series of $\tilde{R}_k(z) = 4zR_k(z) + z = 1 - \sqrt{\Delta_k(z)}$, have the same asymptotical behaviour of the ones of $R_k(z)$. Now $\Delta_k(z) = (7 + 8k)(z - \overline{\rho}_k)\rho_k(1 - z/\rho_k)$, and since $(7+8k)(\rho_k - \overline{\rho}_k) = 4\sqrt{2k+2}$, one has

$$\tilde{R}_k(z) = 1 - \sqrt{\Delta_k(z)} = 1 - 2\sqrt[4]{2k+2}\sqrt{\rho_k}\sqrt{1 - z/\rho_k} + o\left(\sqrt{1 - z/\rho_k}\right).$$

By Proposition 1, one obtains

$$[z^n](4zR_k(z) + z) \sim \frac{\sqrt[4]{2k+2}\sqrt{\rho_k}}{\sqrt{\pi}}\rho_k^{-n}n^{-3/2},$$

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

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

To obtain estimates for the average value relative to some other measures on regular expressions, one can consider parameters $(c_\varepsilon, c_\sigma, c_+, c_\bullet, c_\star)$, where $c_\varkappa$ is the contribution of the operation $\varkappa$ expressed in the measure under consideration. This allows to consider a parametrized weighted generating function and an asymptotic estimation of its coefficients. For each set of parameters, one obtains estimations for the associated measure, such as number of letters, number of operators of a given type (concatenation, star, etc) and, as we will see in Section 3.1, the size of some automata constructions.

The general bivariate generating function, corresponding to those parameters, satisfies the following equation

$$C_k(u,z) = (u^{c_\varepsilon} + ku^{c_\sigma})z + (u^{c_+} + u^{c_\bullet})zC_k(u,z)^2 + u^{c_\star}zC_k(u,z).$$

Solving this equation for $C_k(u,z)$, and choosing the root that has positive coefficients, one sees that

$$C_k(u,z) = \frac{1 - u^{c_\star}z - \sqrt{(1-u^{c_\star}z)^2 - 4(ku^{c_\sigma} + u^{c_\varepsilon})(u^{c_+} + u^{c_\bullet})z^2}}{2(u^{c_+} + u^{c_\bullet})z}.$$

Deriving in order to u, and taking $u = 1$, the cumulative generating function obtained is

$$C_k(z) = \frac{a_k(z)\sqrt{\Delta_k(z)} + b_k(z)}{8z\sqrt{\Delta_k(z)}}, \tag{3}$$

where $\Delta_k(z)$ is as above, and

$$\begin{aligned} a_k(z) =\ & (c_+ + c_\bullet - 2c_\star)\,z - (c_+ + c_\bullet) \\ b_k(z) =\ & (4\,(2c_\sigma - c_+ - c_\bullet)\,k + 8c_\varepsilon - 3(c_+ + c_\bullet) - 2c_\star)\,z^2 + \\ & + 2\,(c_\star - c_+ - c_\bullet)\,z + c_+ + c_\bullet. \end{aligned}$$

Considering $G_k(z) = 8z\,C_k(z) - a_k(z) = \frac{b_k(z)}{\sqrt{\Delta_k(z)}}$, proceeding in a similar way to what was done to $R_k(z)$, and applying Proposition 1, one obtains

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

One concludes that for $n \geq 2$,

$$\begin{aligned} [z^n]C_k(z) &\sim \frac{b_k(\rho_k)}{16\sqrt[4]{2k+2}\,\sqrt{\rho_k}\,\sqrt{\pi}}\,\rho_k^{-(n+1)}(n+1)^{-\frac{1}{2}} && (5) \\ &= \frac{c_\star\sqrt{2k+2} + (c_+ + 2c_\sigma + c_\bullet)k + (c_+ + 2c_\varepsilon + c_\bullet)}{4\sqrt{\pi}\sqrt[4]{2k+2}}\rho_k^{\frac{1}{2}-n}(n+1)^{-\frac{1}{2}}. && (6) \end{aligned}$$

The following expression gives, for $n \geq 2$, the parametrised asymptotic estimate for the average size, for a given measure, for regular expressions of size n.

$$\begin{aligned} \frac{[z^n]C_k(z)}{[z^n]R_k(z)} &\sim \frac{b_k(\rho_k)}{4\rho_k\sqrt{2k+2}}\,(n+1) = \\ &= \rho_k\left(c_\star + (c_+ + c_\bullet)\sqrt{\frac{k+1}{2}} + (c_\sigma k + c_\varepsilon)\sqrt{\frac{2}{k+1}}\right)(n+1). \end{aligned}$$

Thus, for the considered measure, the average of its value per character of the original regular expression is, asymptotically,

$$\lim_{n\to\infty} \frac{[z^n]C_k(z)}{n[z^n]R_k(z)} = \rho_k \left(c_\star + (c_+ + c_\bullet)\sqrt{\frac{k+1}{2}} + (c_\sigma k + c_\varepsilon)\sqrt{\frac{2}{k+1}} \right). \tag{7}$$

The generating function for the number of letters occurring in a regular expression, $Let_k(z)$, can be obtained from (3) by making $c_\sigma = 1$ and null all the other parameters, giving

$$Let_k(z) = \frac{kz}{\sqrt{\Delta_k(z)}}, \tag{8}$$

and (6) yields

$$[z^n]Let_k(z) \sim \frac{k\sqrt{\rho_k}}{2\sqrt{\pi}\sqrt[4]{2k+2}}\rho_k^{-n}(n+1)^{-\frac{1}{2}}, \tag{9}$$

from which it follows that

$$\frac{[z^n]Let_k(z)}{n[z^n]R_k(z)} \sim \frac{2k\rho_k}{\sqrt{2k+2}} \xrightarrow[k\to\infty]{} \frac{1}{2}. \tag{10}$$

In the same manner one can easily obtain approximate values for the number of concatenations, disjunctions and stars.

3 Regular Expressions to NFAs

Because of their succinctness and clear syntax, regular expressions are the common choice to represent regular languages. DFAs are an excellent representation for testing equivalence, containment or membership, as these problems are easily solved for this model. For instance, recognition of a word w is $O(|w|)$ for DFAs, while it is $O(|w| \cdot |Q|^2)$ for NFAs. However, minimal DFAs can be exponentially larger than the associated REs, while the corresponding NFAs can be linearly larger. Since NFA minimisation is PSPACE-complete, the aim is to obtain directly from the REs small NFAs usable for practical purposes. In this section we summarise the known results on the average size of different NFA constructions from REs.

3.1 Average Size of ε-NFAs

We consider here three constructions, introduced respectively by Thompson in 1968 [47] ($\mathcal{A}_T$), by Sippu and Soisalon-Soininen in 1990 [46] ($\mathcal{A}_{SSS}$), and by Ilie and Yu in 2003 [26] ($\mathcal{A}_{\varepsilon\text{-}fol}$), that transform a regular expression α into an equivalent ε-NFA. Generically, denoting the result by $\mathcal{N}_\alpha$, all three algorithms associate with the (atomic) regular expressions ε and σ the same ε-NFAs, as given in Figure 1.

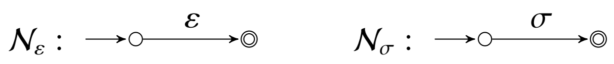

Figure 1: ε-NFAs for atomic expressions

Thus, for all three constructions we have

$$\begin{aligned}
|\mathcal{N}_\varepsilon| &= |\mathcal{N}_\varepsilon|_Q + |\mathcal{N}_\varepsilon|_\delta &&= 2+1 &&= 3\\
|\mathcal{N}_\sigma| &= |\mathcal{N}_\sigma|_Q + |\mathcal{N}_\sigma|_\delta &&= 2+1 &&= 3.
\end{aligned}$$

The ε-NFA's for compound regular expressions are constructed inductively from the automata corresponding to their subexpressions. In Thompson's construction, the automaton $\mathcal{N}_{\beta_1+\beta_2}$ (in Figure 2) is built from $\mathcal{N}_{\beta_1}$ and $\mathcal{N}_{\beta_2}$ introducing a new initial state with ε-transitions to the initial states of both $\mathcal{N}_{\beta_1}$ and $\mathcal{N}_{\beta_2}$, as well as a new final state and ε-transitions from the final states of $\mathcal{N}_{\beta_1}$ and $\mathcal{N}_{\beta_2}$. It follows that this construction introduces exactly 2 new states and 4 new transitions for each + operator. Thus, we have

$$\begin{aligned}
|\mathcal{N}_{\beta_1+\beta_2}|_Q &= |\mathcal{N}_{\beta_1}|_Q + |\mathcal{N}_{\beta_2}|_Q + 2\\
|\mathcal{N}_{\beta_1+\beta_2}|_\delta &= |\mathcal{N}_{\beta_1}|_\delta + |\mathcal{N}_{\beta_2}|_\delta + 4,
\end{aligned} \tag{11}$$

and consequently,

$$|\mathcal{N}_{\beta_1+\beta_2}| = |\mathcal{N}_{\beta_1}| + |\mathcal{N}_{\beta_2}| + 6. \tag{12}$$

The remaining construction cases are presented in Figure 2.

To obtain the weighted generating function for the size of the resulting automata, according to a given measure, we consider the parameters $(c_\varepsilon, c_\sigma, c_+, c_\bullet, c_\star)$ represented in Table 1.

ε-NFAs	States	Transitions	Combined Size
$\mathcal{A}_T$	$(2,2,2,0,2)$	$(1,1,4,1,4)$	$(3,3,6,1,6)$
$\mathcal{A}_{SSS}$	$(2,2,0,-1,2)$	$(1,1,2,0,3)$	$(3,3,2,-1,5)$
$\mathcal{A}_{\varepsilon-fol}$	$(2,2,-2,-1,1)$	$(1,1,0,0,2)$	$(3,3,-2,-1,3)$

Table 1: Parameters for the 3 constructions

Now, note that for all the constructions here considered, the worst-case complexity is reached for expressions with only one letter and $n-1$ stars. For such an expression (which has size n), the size of the corresponding $\mathcal{A}_T$, $\mathcal{A}_{SSS}$ and $\mathcal{A}_{\varepsilon-fol}$ automaton is, respectively, $6n-3$, $5n-2$ and $3n$. In Table 2, we illustrate

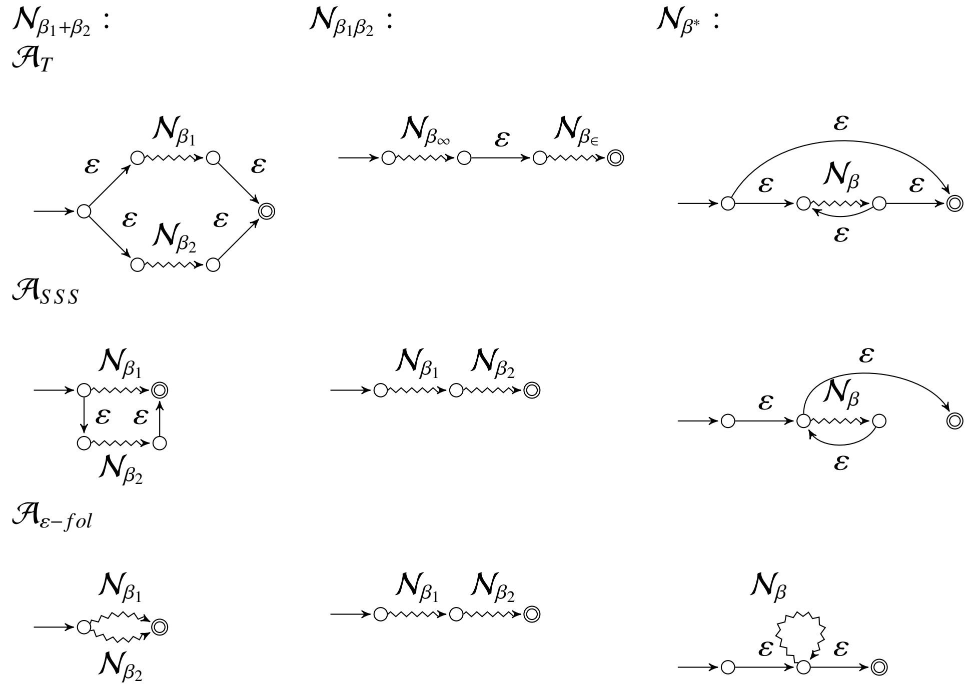
$\mathcal{N}_{\beta_1+\beta_2}$:
$\mathcal{N}_{\beta_1\beta_2}$:
$\mathcal{N}_{\beta^*}$:
$\mathcal{A}_T$
$\mathcal{A}_{SSS}$
$\mathcal{A}_{\varepsilon-fol}$
$\mathcal{N}_{\beta_1}$
$\mathcal{N}_{\beta_2}$
$\mathcal{N}_{\beta_\infty}$
$\mathcal{N}_{\beta_\epsilon}$
$\mathcal{N}_{\beta}$
ε

Figure 2: $\mathcal{A}_T$, $\mathcal{A}_{SSS}$ and $\mathcal{A}_{\varepsilon-fol}$ constructions

the discrepancy between the average and the worst case, for the combined size, by presenting the values of the expression in Equation (7) for different values of k, the limit as k goes to infinity, and the size of the worst case, which does not depend on k. As the alphabet grows, the size of the obtained ε-NFA's is much smaller, asymptotically and on average, than in the worst case. For instance, in the case of the $\mathcal{A}_{\varepsilon-fol}$, the ratio between these values is 0.25. This construction also exhibits the best behaviour of the three.

k	2	10	50	100	∞	worst-case
$\mathcal{A}_T$	3.72	3.51	3.38	3.34	3.25	6
$\mathcal{A}_{SSS}$	2.30	2.06	1.90	1.86	1.75	5
$\mathcal{A}_{\varepsilon-fol}$	1.13	0.97	0.86	0.83	0.75	3

Table 2: Average vs. worst-case combined size

3.2 Average Size of the Glushkov Automata

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 $\overline{\Sigma} = \{ \sigma^i \mid \sigma \in \Sigma, 1 \leq i \leq |\alpha|_\Sigma \}$. For $\alpha \in \mathsf{R}$ and $i \in \mathrm{Pos}(\alpha)$, let the sets *first*, *last* and *follow* be $\mathsf{ft}(\alpha) = \{ i \mid \exists w \in \overline{\Sigma}^\star, \sigma^i w \in \mathcal{L}(\overline{\alpha}) \}$, $\mathsf{lt}(\alpha) = \{ i \mid \exists w \in \overline{\Sigma}^\star, w\sigma^i \in \mathcal{L}(\overline{\alpha}) \}$ and $\mathsf{fw}(\alpha, i) = \{ j \mid \exists u, v \in \overline{\Sigma}^\star, u\sigma^i\sigma^j v \in \mathcal{L}(\overline{\alpha}) \}$, respectively. These sets can be inductively defined as follows:

$$\begin{array}{lcl} \mathsf{ft}(\emptyset) &=& \mathsf{ft}(\varepsilon) = \emptyset \\ \mathsf{ft}(\sigma_i) &=& \{i\} \\ \mathsf{ft}(\alpha^\star) &=& \mathsf{ft}(\alpha) \end{array} \quad \begin{array}{lcl} \mathsf{ft}(\alpha+\beta) &=& \mathsf{ft}(\alpha) \cup \mathsf{ft}(\beta) \\ \mathsf{ft}(\alpha\beta) &=& \begin{cases} \mathsf{ft}(\alpha) \cup \mathsf{ft}(\beta) & \text{if } \varepsilon(\alpha) = \varepsilon \\ \mathsf{ft}(\alpha) & \text{otherwise,} \end{cases} \end{array} \tag{13}$$

$$\begin{aligned} \mathsf{fw}(\emptyset) &= \mathsf{fw}(\varepsilon) = \mathsf{fw}(\sigma_i) = \emptyset \\ \mathsf{fw}(\alpha\beta) &= \mathsf{fw}(\alpha) \cup \mathsf{fw}(\beta) \\ \mathsf{fw}(\alpha\beta) &= \mathsf{fw}(\alpha) \cup \mathsf{fw}(\beta) \cup \mathsf{lt}(\alpha) \times \mathsf{ft}(\beta) \\ \mathsf{fw}(\alpha^\star) &= \mathsf{fw}^\star(\alpha) \\ \mathsf{fw}^\star(\emptyset) &= \mathsf{fw}^\star(\varepsilon) = \emptyset \\ \mathsf{fw}^\star(\sigma_i) &= \{(i,i)\} \\ \mathsf{fw}^\star(\alpha+\beta) &= \mathsf{fw}^\star(\alpha) \cup \mathsf{fw}^\star(\beta) \cup \mathsf{cs}(\alpha,\beta) \\ \mathsf{fw}^\star(\alpha\beta) &= \begin{cases} \mathsf{fw}^\star(\alpha) \cup \mathsf{fw}^\star(\beta) \cup \mathsf{cs}(\alpha,\beta) & \text{if } \varepsilon(\alpha) = \varepsilon(\beta) = \varepsilon \\ \mathsf{fw}^\star(\alpha) \cup \mathsf{fw}(\beta) \cup \mathsf{cs}(\alpha,\beta) & \text{if } \varepsilon(\beta) = \varepsilon \\ \mathsf{fw}(\alpha) \cup \mathsf{fw}^\star(\beta) \cup \mathsf{cs}(\alpha,\beta) & \text{if } \varepsilon(\alpha) = \varepsilon \\ \mathsf{fw}(\alpha) \cup \mathsf{fw}(\beta) \cup \mathsf{cs}(\alpha,\beta) & \text{otherwise} \end{cases} \\ \mathsf{fw}^\star(\alpha^\star) &= \mathsf{fw}^\star(\alpha), \end{aligned} \tag{14}$$

with $\mathsf{cs}(\alpha,\beta) = \mathsf{lt}(\alpha) \times \mathsf{ft}(\beta) \cup \mathsf{lt}(\beta) \times \mathsf{ft}(\alpha)$. The *Glushkov automaton* for α is $\mathcal{A}_{\mathrm{pos}}(\alpha) = (\mathrm{Pos}_0(\alpha), \Sigma, \delta_{\mathrm{pos}}, 0, F)$, with $\delta_{\mathrm{pos}} = \{ (0, \overline{\sigma}^j, j) \mid j \in \mathsf{ft}(\alpha) \} \cup \{ (i, \overline{\sigma}^j, j) \mid j \in \mathsf{fw}(\alpha, i) \}$ and $F = \mathsf{lt}(\alpha) \cup \{0\}$ if $\varepsilon(\alpha) = \varepsilon$, and $F = \mathsf{lt}(\alpha)$, otherwise.

The definition of lt is almost identical, differing only in the case of concatenation, where the branches are swapped. The definition of the fw function here presented deviates from the usual one in the case of the $\star$ operator. This version ensures that the unions are all disjoint. The same result can be obtained if the regular expression is first transformed into star normal form, *i.e.* such that for all subexpressions β^* one has $\varepsilon \notin \mathcal{L}(\beta)$ [12].

The number of states of $\mathcal{A}_{pos}(\alpha)$ is exactly $n + 1$ where $n = |\alpha|_\Sigma$. Thus, the average number of its states coincides with the average number of letters determined in Equation (10), *i.e.* asymptotically and on average the number of states of $\mathcal{A}_{pos}(\alpha)$ is half the size of α. On the other hand, the number of transitions in $\mathcal{A}_{pos}(\alpha)$ is, in the worst case, $n^2 + n$. Nicaud's main result in [37] is that, on average, the number of transitions is $O(|\alpha|)$. However, his computation of the number of transitions was not exact because the definition used for the fw function did not take into account the possible non-disjoint unions of its results. The version of the fw function presented above allows for an exact counting.

The generating function for the number of transitions is $T_k(z) = F_k(z) + E_k(z)$, where $F_k(z)$ and $E_k(z)$ are the generating functions associated with ft and fw, respectively. By symmetry, the generating function $L_k(z)$ for lt is the same as $F_k(z)$, which was computed by Nicaud: $L_k(z) = F_k(z) = \frac{kz}{1-z-3zR_k(z)-zR_{k,\varepsilon}(z)}$, where $R_{k,\varepsilon}(z)$ denotes the generating function for regular expressions whose languages contain ε and is given by $R_{k,\varepsilon}(z) = \frac{z+zR_k(z)}{1-2zR_k(z)}$. An estimate for the number of transitions of $\mathcal{A}_{pos}$ was given in [7] as

$$[z^n]T_k(z) \sim \frac{(1+\rho_k)(2+16\rho_k+10\rho_k^2-12\rho_k^3)}{8\rho_k\sqrt{\pi}(1-5\rho_k^2)\sqrt{2-2\rho_k}}\rho_k^{-n}n^{-\frac{1}{2}}. \tag{15}$$

Hence, an estimate for the average number of transitions *per* state is

$$\frac{[z^n]T_k(z)}{[z^n]Let_k(z)} \sim \frac{(1+\rho_k)(2+16\rho_k+10\rho_k^2-12\rho_k^3)}{(1-2\rho_k-7\rho_k^2)(1-5\rho_k^2)}. \tag{16}$$

An estimate for the average number of transitions *per* regular expression is:

$$\frac{[z^n]T_k(z)}{[z^n]R_k(z)} \sim \frac{(1+\rho_k)(1+8\rho_k+5\rho_k^2-6\rho_k^3)}{(1-\rho_k)(1-5\rho_k^2)}n. \tag{17}$$

Since ρ_k tends to 0 as k goes to ∞, it follows that for large values of k, the average number of transitions *per* state is approximately 2, while the average number of transitions *per* automaton is approximately the size of the original regular expression.

3.3 Average Size of the Partial Derivative Automata

The partial derivative automaton of a regular expression was introduced, independently and through two distinct approaches, by Mirkin [35] and Antimirov [2]. Champarnaud and Ziadi [18] proved that the two formulations are equivalent. For a RE α and a symbol $\sigma \in \Sigma$, the set of partial derivatives of α w.r.t. σ is defined inductively as follows:

$$\begin{array}{lcl@{\qquad}lcl}
\partial_\sigma(\emptyset) & = & \partial_\sigma(\varepsilon) = \emptyset & \partial_\sigma(\alpha+\beta) & = & \partial_\sigma(\alpha) \cup \partial_\sigma(\beta) \\
\partial_\sigma(\sigma') & = & \begin{cases} \{\varepsilon\}, & \text{if } \sigma' = \sigma \\ \emptyset, & \text{otherwise} \end{cases} & \partial_\sigma(\alpha\beta) & = & \partial_\sigma(\alpha)\beta \cup \varepsilon(\alpha)\partial_\sigma(\beta) \\
 & & & \partial_\sigma(\alpha^\star) & = & \partial_\sigma(\alpha)\alpha^\star,
\end{array} \tag{18}$$

where, for any $S \subseteq \mathsf{R}$, $\beta \in \mathsf{R}$, $S\emptyset = \emptyset S = \emptyset$, $S\varepsilon = \varepsilon S = S$, and $S\beta = \{\ \alpha\beta \mid \alpha \in S\ \}$ if $\beta \neq \emptyset$, and $\beta \neq \varepsilon$. The definition of partial derivative can be naturally extended to sets of regular expressions, words, and languages. One has $\bigcup_{\tau\in\partial_w(\alpha)}\mathcal{L}(\tau) = w^{-1}\mathcal{L}(\alpha)$, and the set of all partial derivatives of α w.r.t. words is denoted by $\mathrm{PD}(\alpha) = \bigcup_{w\in\Sigma^\star}\partial_w(\alpha)$. Note that the set $\mathrm{PD}(\alpha)$ is always finite [2], as opposed to the set of Brzozowski's derivatives, which is only finite modulo ACI.

The partial derivative automaton is defined by $\mathcal{A}_{pd}(\alpha) = (\mathrm{PD}(\alpha), \Sigma, \delta_{pd}, \alpha, F_{pd})$, where $\delta_{pd} = \{\ (\tau, \sigma, \tau') \mid \tau \in \mathrm{PD}(\alpha) \wedge \tau' \in \partial_\sigma(\tau)\ \}$, and $F_{pd} = \{\ \tau \in \mathrm{PD}(\alpha) \mid \varepsilon(\tau) = \varepsilon\ \}$. Mirkin's construction of the $\mathcal{A}_{pd}(\alpha)$ is based on solving a system of equations of the form $\alpha_i = \sigma_1\alpha_{i1} + \ldots + \sigma_k\alpha_{ik}$ if $\varepsilon(\alpha_i) = \emptyset$, and $\alpha_i = \sigma_1\alpha_{i1} + \ldots + \sigma_k\alpha_{ik} + \varepsilon$ otherwise, with $\alpha_0 \equiv \alpha$ and α_{ij}, $1 \le j \le k$, linear combinations the α_i, $0 \le i \le n$, $n \ge 0$. A solution (called the *support* of α) $\pi(\alpha) = \{\alpha_1, \ldots, \alpha_n\}$ can be obtained recursively on the structure of α as follows:

$$\begin{array}{rcl@{\qquad}rcl}
\pi(\emptyset) & = & \emptyset & \pi(\alpha\cup\beta) & = & \pi(\alpha)\cup\pi(\beta)\\
\pi(\varepsilon) & = & \emptyset & \pi(\alpha\beta) & = & \pi(\alpha)\beta\cup\pi(\beta)\\
\pi(\sigma) & = & \{\varepsilon\} & \pi(\alpha^\star) & = & \pi(\alpha)\alpha^\star.
\end{array} \tag{19}$$

Champarnaud and Ziadi [18] proved that $\mathrm{PD}(\alpha) = \pi(\alpha)\cup\{\alpha\}$ and that the two constructions lead to the same automaton. As noted by Broda et al. [7], Mirkin's algorithm to compute $\pi(\alpha)$ also provides an recursive definition of the set of transitions of $\mathcal{A}_{pd}(\alpha)$. Let $\varphi(\alpha) = \{\ (\sigma,\gamma) \mid \gamma\in\partial_\sigma(\alpha), \sigma\in\Sigma\ \}$ and $\lambda(\alpha) = \{\ \alpha' \mid \alpha' \in \pi(\alpha),\ \varepsilon(\alpha') = \varepsilon\ \}$, where both sets can be recursively defined using (18) and (19). We have, $\delta_{pd} = \{\alpha\}\times\varphi(\alpha)\cup F(\alpha)$ where the result of the $\times$ operation is seen as a set of triples and the set F is defined inductively by:

$$\begin{array}{rcl}
F(\emptyset) & = & F(\varepsilon) = F(\sigma) = \emptyset,\ \sigma\in\Sigma\\
F(\alpha+\beta) & = & F(\alpha)\cup F(\beta)\\
F(\alpha\beta) & = & F(\alpha)\beta\cup F(\beta)\cup\lambda(\alpha)\beta\times\varphi(\beta)\\
F(\alpha^\star) & = & F(\alpha)\alpha^\star\cup(\lambda(\alpha)\times\varphi(\alpha))\alpha^\star.
\end{array} \tag{20}$$

For all $\alpha\in\mathsf{R}$, $\mathcal{A}_{pd}(\alpha) = (\pi(\alpha)\cup\{\alpha\}, \Sigma, \{\alpha\}\times\varphi(\alpha)\cup F(\alpha), \alpha, \lambda(\alpha)\cup\varepsilon(\alpha)\{\alpha\})$.

In his original paper, Mirkin showed that $|\pi(\alpha)| \le |\alpha|_\Sigma$. Since $\mathcal{A}_{pd}(\alpha)$ is a quotient of the Glushkov automaton [17], we know that it has at most $|\alpha|_\Sigma+1$ states. But this upper bound is reached if and only if, at 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 $\varepsilon\in\pi(\beta)\cap\pi(\gamma)$, $|\pi(\beta+\gamma)| = |\pi(\beta)\cup\pi(\gamma)| \le |\pi(\beta)| + |\pi(\gamma)|-1$ and also $|\pi(\beta\gamma^\star)| = |\pi(\beta)\gamma^\star\cup\pi(\gamma^\star)| = |\pi(\beta)\gamma^\star\cup\pi(\gamma)\gamma^\star| \le |\pi(\beta)|+|\pi(\gamma)|-1$. These observations lead to the computation of a lower bound of the number of state mergings [6]. The respective generating function is $I_k(z) = \frac{(z+z^2)R_{\pi,k}(z)^2}{\sqrt{\Delta_k(z)}}$, where $R_{\pi,k}(z)$ is the generating function of $\alpha\in\mathsf{RE}$ such that $\varepsilon\in\pi(\alpha)$. The asymptotic estimate of the (cumulative) number of mergings is

$$[z^n]I_k(z) \sim \frac{1+\rho_k}{64}\frac{\left(a_k(\rho_k)+b(\rho_k)^2-2b(\rho_k)\sqrt{a_k(\rho_k)}\right)}{\sqrt{\pi}\sqrt{2-2\rho_k}}\rho_k^{-(n+1)}n^{-1/2}, \tag{21}$$

where now $a_k(z) = 16z^4 - 24z^3 + (64k + 1)z^2 + 6z + 1$ and $b(z) = -4z^2 + 3z + 1$.

From (2) and (21) one easily gets the following asymptotic estimate for the average number of mergings

$$\frac{[z^n]I_k(z)}{[z^n]R_k(z)} \sim \lambda_k\, n, \tag{22}$$

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 $\lim\limits_{k\to\infty} \rho_k = 0$, and that $\lim\limits_{k\to\infty} a_k(\rho_k) = 9$, while $\lim\limits_{k\to\infty} b(\rho_k) = 1$, one gets that $\lim_{k\to\infty} \lambda_k = \frac{1}{4}$. This means that, for a RE 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 $\mathcal{A}_{pd}$ automaton, as compared to the ones of the $\mathcal{A}_{pos}$ automaton, it is enough to compare the number of mergings for an expression α with the number of letters in α. From (9) and (22) one gets

$$\frac{[z^n]I_k(z)}{[z^n]L_k(z)} \sim \frac{1-\rho_k}{4k\rho_k^2}\lambda_k. \tag{23}$$

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 $\mathcal{A}_{pd}$ automaton is about one half of the number of states of the $\mathcal{A}_{pos}$ automaton, and about one quarter of the size of the corresponding RE.

In [7] the same technique was used for the estimation of the number of transitions of $\mathcal{A}_{pd}$. Observing the equations (20), in this case one first estimates the number of mergings that occur in $\lambda(\alpha)$ and then the corresponding number of transition mergings. Letting It_z being the generating function for the number of transitions mergings, one has

$$[z^n]It_k(z) \sim \frac{(1+\rho_k)\left(a(\rho_k)\sqrt{b(\rho_k)} + c(\rho_k)\right)}{16\sqrt{\pi}\rho_k\sqrt{2-2\rho_k}\,(1-5\rho_k^2)d(\rho_k)}\rho_k^{-n}n^{-\frac{1}{2}} \tag{24}$$

where $a(z)$, $b(z)$, $c(z)$, and $d(z)$ are some fixed polynomials. Therefore, a lower bound for the average number of mergings per transition of the Glushkov automaton is given by

$$[z^n]\frac{It_k(z)}{T_k(z)} \sim \frac{a(\rho_k)\sqrt{b(\rho_k)} + c(\rho_k)}{4(1+8\rho_k+5\rho_k^2-6\rho_k^3)d(\rho_k)}. \tag{25}$$

Because $lim_{k\to\infty}[z^n]\frac{It_k(z)}{T_k(z)} = \frac{1}{2}$, asymptotically with respect to k, the number of transitions in $\mathcal{A}_{pd}$ is at most half the number of transitions in $\mathcal{A}_{pos}$.

3.4 Other NFAs and Open Problems

Another well-known quotient of the Glushkov automaton is the *follow automaton*, $\mathcal{A}_f$, which can also be obtained by eliminating the ε-transitions from the $\mathcal{A}_{\varepsilon-fol}$ [26]. It is known that if the regular expression is in star normal form, the $\mathcal{A}_{pd}$ is always smaller than or equal to $\mathcal{A}_f$. The conversion to star normal form is linear and experimental results suggest that, on average, regular expressions are in that form [22]. Although it is an open problem to theoretically show that this is the case, we believe that on average the $\mathcal{A}_f$ is not smaller than $\mathcal{A}_{pd}$. Experimental results also suggest that on average the $\mathcal{A}_{pd}$ almost coincides with the bisimilarity of $\mathcal{A}_{pos}$, $\mathcal{A}_{pos}/_{\equiv_b}$. Maia et al. [31] characterised, for finite languages, the graph properties of $\mathcal{A}_{pd}$, and determined under which conditions $\mathcal{A}_{pd} \simeq \mathcal{A}_{pos}/_{\equiv_b}$. It is an open problem to obtain an asymptotic average behaviour of these two constructions.

There are some related constructions of automata from regular expressions. Instead of left quotients one can consider right quotients. Given a language L, the *right-quotient* of L w.r.t. a word w is the language $Lw^{-1} = \{\ x \mid xw \in L\ \}$. It is not difficult to verify that $Lw^{-1} = (w^R)^{-1}L^R$, where $(\ )^R$ represents the reversal operation. Then sets of right-partial derivatives and the *right-partial derivative automaton* ($\overleftarrow{\mathcal{A}}_{pd}$) can be naturally defined. However, as $(\mathcal{A}_{pd}(\alpha^R))^R \simeq \overleftarrow{\mathcal{A}}_{pd}$, we can conclude that the number of states of $\overleftarrow{\mathcal{A}}_{pd}$ are, asymptoticaly and on average, half of the number of states of $\mathcal{A}_{pos}$. Yamamoto [48] introduced the *prefix automaton* of a RE, $\mathcal{A}_{pre}$, as a quotient of the Thompson automaton that corresponds also to the quotient of the Glushkov automaton by a right-invariant relation that identifies states of $\mathcal{A}_{pos}$ with the same left language. Maia et al. [32] characterised the $\mathcal{A}_{pre}$ automaton as a solution of a system of equations, and presented a recursive definition of $\mathcal{A}_{pre}$ akin to the one given for $\mathcal{A}_{pd}$ in equations (19) and (20). Using the framework of analytic combinatorics and techniques similar to the ones described in Section 3.3, it was shown that, as the size of alphabet grows, the average number of states of the $\mathcal{A}_{pre}$ automaton approaches the number of states of the $\mathcal{A}_{pos}$ automaton.

Finally, given the set of positions of a RE α, $\mathrm{Pos}(\alpha)$, instead of considering the fw function as in the $\mathcal{A}_{pos}$ automaton, one can consider the pr function, that gives the set of positions that can precede a given position *i.e.* $\mathsf{pr}(\alpha, j) = \{\ i \mid ua_ia_jv \in \mathcal{L}(\overline{\alpha})\ \}$, for $j \in \mathrm{Pos}(\alpha)$. The *previous automaton* $\mathcal{A}_{pre}$ is defined akin to $\mathcal{A}_{pos}$, but considering a unique distinct final state. It follows that $\mathcal{A}_{prev}(\alpha) \simeq (\mathcal{A}_{pos}(\alpha^R))^R$. Thus, the number of states coincides with the number of states of $\mathcal{A}_{pos}$. The interest in this construction relies in its connection to the recent *au point* DFA construction [4, 39] that we will mention in Section 5.

4 Extended Regular Expressions to NFAs

Although regular languages are trivially closed for boolean operations, the manipulation of intersection and complementation with regular expressions or nondeterministic finite automata is non-trivial and leads to an exponential blow up. However, there are several applications where extended regular expressions are used to represent information and it is important to study their conversion to automata. Caron *et al.* [14] extended the notion of partial derivatives and partial derivative automaton to regular expressions with intersection and complementation.

Broda et al. [11] extended the same notions to regular expressions with shuffle and studied the average number of states of the corresponding partial derivative automaton. The complexity of the shuffle (or interleaving) operation is well studied in the worst case. Mayer and Stockmeyer [33] showed that for REs with shuffle, of size n, an equivalent NFA needs at most 2^n states, and presented a family of REs with shuffle, of size $O(n)$, for which the corresponding NFAs have at least 2^n states. Gelade [20], and Gruber and Holzer [24] showed that there exists a double exponential trade-off in the translation from REs with shuffle to standard REs. Gelade also gave a tight double exponential upper bound for the translation of REs with shuffle to DFAs.

Broda et al. showed that the number of states of the partial derivative automata is in the worst case at most 2^m, where m is the number of letters in the expression, while asymptotically and on average it is no more than $(\frac{4}{3})^m$. Considering the grammar for regular expressions (1) with one more rule for shuffle $\alpha \sqcup\!\sqcup \alpha$, we can extend the definition of the support π in (19) by:

$$\pi(\alpha \sqcup\!\sqcup \beta) \quad = \quad \pi(\alpha) \sqcup\!\sqcup \pi(\beta) \cup \pi(\alpha) \sqcup\!\sqcup \{\beta\} \cup \{\alpha\} \sqcup\!\sqcup \pi(\beta), \tag{26}$$

where for $S, T \subseteq \mathsf{R}$, $S \sqcup\!\sqcup T = \{\ \alpha \sqcup\!\sqcup \beta \mid \alpha \in S, \beta \in T\ \}$, and $\{\varepsilon\} \sqcup\!\sqcup S = S \sqcup\!\sqcup \{\varepsilon\} = S$. With this expression it is easy to see that now $|\pi(\alpha)| \leq 2^{|\alpha_\Sigma|}$ and this bound is reachable for the family of regular expressions $\alpha_n = a_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup a_n$, where $n \geq 1$, $a_i \neq a_j$ for $1 \leq i \neq j \leq n$. Let $P_k(z)$ be the generating function for an upper bound for the number of elements in π. For expressions of size n, one has,

$$[z^n]P_k(z) \quad \sim \quad \frac{-(3+3k)^{\frac{1}{4}}\rho_k^{-n-\frac{1}{2}} + (3+4k)^{\frac{1}{4}}(\rho'_k)^{-n-\frac{1}{2}}}{2\sqrt{\pi}}(n+1)^{-\frac{3}{2}},$$

where now $\rho_k = \frac{-1+2\sqrt{3+3k}}{11+12k}$ and $\rho'_k = \frac{-1+2\sqrt{3+4k}}{11+16k}$.

For a regular expression α of size n, let avP and avL be the average size of π and the average alphabetic size, respectively. Taking into account the worst case upper bound, we have compared the values of $\log_2 avP$ and avL, obtaining

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

Therefore, one has the following significant improvement, when compared with the worst case, for the average-case upper bound: for large values of k and n an upper bound for the average number of states of $\mathcal{A}_{pd}$ is $(\frac{4}{3} + o(1))^{|\alpha|_\Sigma}$.

Regular expressions with intersection have a worst-case behaviour similar to shuffle [20, 24]. The definition of the support π in this case is just $\pi(\alpha \cap \beta) = \pi(\alpha) \cap \pi(\beta)$ (with the $\cap$ for sets of REs defined as above for $⧢$, only interchanging the operators). However, the analytic analysis of the correspondent generating function is much harder and an estimation of an upper bound of the average size is an on-going work by the authors of this paper. Despite that, experimental results suggest that the average number of states in $\mathcal{A}_{pd}$ automata is much smaller than $2^{|\alpha|_\Sigma}$.

We note that for both intersection and complementation is not clear how to extend the position based constructions.

5 Regular Expressions to DFAs

A word derivative w.r.t. a RE α, $w^{-1}\alpha$ is such that $\mathcal{L}(w^{-1}\alpha) = w^{-1}\mathcal{L}(\alpha)$. The set of word derivatives $D(\alpha)$ is finite modulo the ACI axioms. The Brzozowski derivative automaton can be defined by: $A_{\mathcal{B}}(\alpha) = (\mathcal{D}(\alpha), \Sigma, \delta, [\alpha], F)$, where $F = \{ [d] \in \mathcal{D}(\alpha) \mid \varepsilon(d) = \varepsilon \}$, and $\delta([q], \sigma) = [\sigma^{-1}q]$, for all $[q] \in \mathcal{D}(\alpha), \sigma \in \Sigma$. McNaughton and Yamada [34] presented a DFA construction from an RE that coincides with the determinization of the $\mathcal{A}_{pos}$ automaton. Recently, Asperti [4] introduced a DFA construction from an RE that uses *pointed* REs α' where some letters are annotated with a point and $\mathcal{L}(\alpha')$ is the set words that start at some *pointed* letter. All these constructions lead to DFAs that have a size that is, in the worst case, exponential in the size of the initial RE. For all these constructions, no average-case complexity results are known, as far as the authors are aware of. Some experimental results suggest that *au point* is on average smaller then the other constructions. To understand why this happens and to find estimates of the average size of each construction is thus an open problem.

6 KATs to NTAs

Kleene algebra with tests (KAT) [28] is a decidable equational system combining Kleene and Boolean algebras, and is specially suited to capture and verify properties of simple imperative programs. The equational theory of KAT is PSPACE-complete. The decidability, conciseness and expressiveness of KAT motivated its recent automatisation within several theorem provers [40, 41, 3] and functional languages [1, 42]. Most of those implementations use (variants of) the coalgebraic automaton on guarded strings developed by Kozen [30]. In that approach, derivatives are considered over symbols of the from $\mathsf{v}p$, where p is an alphabetic *program* symbol and v a valuation of boolean variables (the *guard*, normally called atom). This induces an exponential blow-up on the number of states or transitions of the automata. This exponential growth was avoided in Silva [45], and in Broda et al. [8, 10], by using for KAT standard finite automata, where transitions are labeled both with program symbols and boolean tests (instead of atoms).

The abstract syntax of KAT expressions, over an alphabet $\mathsf{P} = \{p_1, \dots, p_k\}$ of program symbols and $\mathsf{T} = \{t_0, \dots, t_{l-1}\}$ of boolean variables (tests), can be given by the following unambiguous grammar, suitable for applying the symbolic method.

$$\begin{aligned} \mathsf{BExp} : b &\rightarrow 0 \mid 1 \mid t \mid \neg b \mid (b + b) \mid (b \cdot b) && (27)\\ \mathsf{AExp} : a &\rightarrow p \mid (a + a) \mid (a + b) \mid (b + a) \mid (a \cdot a) \mid (a \cdot b) \mid (b \cdot a) \mid a^\star && (28)\\ \mathsf{Exp} : e &\rightarrow b \mid a. && (29) \end{aligned}$$

Here BExp, AExp, and Exp represent sets of boolean expressions, KAT expressions with at least one program symbol $p \in \mathsf{P}$, and KAT expressions, respectively. For simplicity, the grammar excludes subexpressions of the form $b^\star$, as their semantics correspond to the set of all boolean assignments and thus are equivalent to 1. For the negation of test symbols we use $\bar{t}$ instead of $\neg t$. The set At, of *atoms* over T, is the set of all boolean assignments to all elements of T, $\mathsf{At} = \{\ x_0 \cdots x_{l-1} \mid x_i \in \{t_i, \bar{t}_i\},\ t_i \in \mathsf{T}\ \}$. Elements of At are denoted by v, and we write $\mathsf{v} \leq b$, if $\mathsf{v} \rightarrow b$ is a boolean tautology.

The set of *guarded strings* over P and T is $\mathsf{GS} = (\mathsf{At} \cdot \mathsf{P})^\star \cdot \mathsf{At}$. Regular sets of guarded strings form the standard language-theoretic model for KAT [29]. A *(nondeterministic) automaton with tests* (NTA) over the alphabets P and T is a tuple $\mathcal{A} = \langle Q, q_0, o, \delta \rangle$, where Q is a finite set of states, $q_0 \in Q$ is the initial state, $o : Q \rightarrow \mathsf{BExp}$ is the output function, and $\delta \subseteq Q \times (\mathsf{BExp} \times \mathsf{P}) \times Q$ is the transition relation. A guarded string $\mathsf{v}_0\sigma_1 \dots \sigma_n\mathsf{v}_n$, with $n \geq 0$, is accepted by the automaton $\mathcal{A}$ if and only if there is a sequence of states $q_0, q_1, \dots, q_n \in Q$, where q_0 is the initial state, and, for $i = 0, \dots, n-1$, one has $\mathsf{v}_i \leq b_i$ for some $(q_i, (b_i, \sigma_{i+1}), q_{i+1}) \in \delta$, and $\mathsf{v}_n \leq o(q_n)$.

Silva [45] presented the Glushkov construction for KAT, and Broda et al. [8, 10] defined the partial derivative automaton for KAT. The asymptotic average size of both constructions were studied in [8]. It was shown that, contrary to other automata constructions for KAT expressions, they enjoy the same descriptional complexity behaviour as their counterparts for regular expressions.

Consider the generating function

$$R_m(z) = \frac{1 - z - \sqrt{\Delta_m(z)}}{4z}, \text{ where } \Delta_m(z) = 1 - 2z - (15 + 8m)z^2, \qquad (30)$$

for the number of regular expressions generated by the grammar in (1), including $\emptyset$, which is almost identical to the one in Subsection 2.2. It is easy to see that $B_l(z) = R_l(z)$ and $E_{k,l}(z) = R_{k+l}(z)$, where l and k are respectively the sizes of P and T, and $B_l(z)$ and $E_{k,l}(z)$ respectively the generating functions for BExp and Exp. Using the technique presented in Section 2 applied to (30), the asymptotic estimates for the number of regular expressions of size m is

$$[z^n]R_m(z) \sim \frac{\sqrt{\rho_m}\,\sqrt[4]{2m+4}}{4\sqrt{\pi}}\,\rho_m^{-(m+1)}(m+1)^{-\frac{3}{2}}, \tag{31}$$

where $\rho_m = \frac{-1+2\sqrt{2m+4}}{15+8m}$ is the radius of convergence of $R_m(z)$. Let $P_{k,l}(z)$ denote the generating function for the number of program symbols in KAT expressions. Then, we have $P_{k,l} = \frac{k}{k+l}Let_{k+l}(z)$, with $Let_m(z)$ as in (8). Therefore, the probability, for a uniform distribution, that a symbol in a KAT expression of size n is a program symbol is

$$\frac{[z^n]P_{k,l}(z)}{n\,[z^n]E_{k,l}(z)} \sim \frac{\left(4(k+l)+8-\sqrt{2(k+l)+4}\right)k}{(15+8\,(k+l))\,(k+l+2)}\left(1+\frac{1}{n}\right)^{3/2} = \eta_{k,l,n}. \tag{32}$$

The average number of program symbols, as $k+l$ increases, tends to $\frac{1}{2(c+1)}$, where $c = \frac{l}{k}$. For instance, if $l = k$, $l = 2k$, and $l = \frac{1}{2}k$, this limit is, respectively, $\frac{1}{4}$, $\frac{1}{6}$, and $\frac{1}{3}$. Furthermore, for any ratio c, the asymptotic average number of states in Glushkov automata is less than half the size of the corresponding expressions.

Since for KAT the recursive definition of the support π just differs by adding $\pi(b) = \emptyset$, which does not affect the computations, one can apply the method used in Subsection 3.3 in order to get an upper bound for the state complexity of the partial derivative automaton. One obtains,

$$\frac{[z^n]I_{k,l}(z)}{[z^n]E_{k,l}(z)} \sim \lambda_{k,l}\,n, \tag{33}$$

where $\lambda_{k,l} = \frac{1+\rho_{k+l}}{16(1-\rho_{k+l})}\left(a_k(\rho_{k+l}) + b(\rho_{k+l})^2 - 2b(\rho_{k+l})\sqrt{a_k(\rho_{k+l})}\right)$, with $a_k(z) = 16z^4 - 24z^3 + (64k+1)z^2 + 6z + 1$, and $b(z) = -4z^2 + 3z + 1$. Therefore

$$\frac{[z^n]I_{k,l}(z)}{[z^n]P_{k,l}(z)} \sim \frac{\lambda_{k,l}}{\eta_{k,l,n}}. \tag{34}$$

One can see that, for a fixed value of l this ratio approaches $\frac{1}{2}$, as k grows. This means that the number of states in the equation automaton is asymptotically, and on average, half the number of states in the Glushkov automaton.

It is more difficult to obtain a sufficiently accurate upper bound for the average number of transitions in the Glushkov NTAs. In particular, several grammars for expressions with different properties, such as for KAT expressions that have no atom $\mathsf{v} \in \mathsf{At}$ in their associated language, have to be considered. In this case the computations no longer mirror the ones for regular expressions, but nevertheless the same result is reached: asymptotically, and on average, the number of transitions of the Glushkov automaton is linear in the size of the KAT expression. To estimate the average number of transitions in the $\mathcal{A}_{pd}$ for KAT expressions is an open problem.

6.1 SKA and SKAT

Synchronous Kleene algebra (SKA), introduced by Prisacariu [43], combines KA with a synchrony model of concurrency. Synchronous here means that two concurrent processes execute a single action simultaneously at each time instant of a unique global clock.

A SKA over a finite set $\mathsf{A_B}$ is given by a structure $(\mathcal{A}, +, \cdot, \times, {}^*, 0, 1, \mathsf{A_B})$, where $\mathsf{A_B} \subseteq \mathcal{A}$, $(\mathcal{A}, +, \cdot, {}^*, 0, 1)$ is a Kleene algebra, and $\times$ is a binary operator that is associative, commutative, distributive over $+$, with absorvent element 0 and identity 1. Furthermore, it satisfies $a \times a = a$, $\forall a \in \mathsf{A_B}$, as well as the following equation for synchrony: $(\alpha^\times \cdot \alpha) \times (\beta^\times \cdot \beta) = (\alpha^\times \times \beta^\times) \cdot (\alpha \times \beta)$, where $\alpha^\times$ and $\beta^\times$ are of the form $a_1 \times \cdots \times a_n$, for $a_i \in \mathsf{A_B}$.

Broda et al. [5] defined the partial derivative automaton $\mathcal{A}_{pd}$ for SKA. It was shown that the worst-case upper bound for the number of states of this automaton coincides with the one for the $\mathcal{A}_{pd}$ for regular expressions with shuffle. This implies the same upper bound for the average number of states of the $\mathcal{A}_{pd}$ for SKA. Prisacariu also generalised Kleene algebra with tests to the synchronous setting (SKAT). Broda et al. extended NTA's for SKAT, as well as the derivative based methods already developed for SKA. Also in this case, experimental results suggest that on average the size of $\mathcal{A}_{pd}$ for both SKA and SKAT are much smaller than the worst-case upper bound. Thus, a more fine-grained study of the average-case complexity of these automata in the analytic combinatorics framework is worthwhile.

7 Conclusions

We presented recent results on the average size of automata obtained from regular expressions and some extended expressions. The framework of analytic combinatorics was the main tool for estimating the asymptotic number of states and of transitions for automata, as a function of the expressions' size. In general it is necessary to obtain generating functions associated with the measures under consideration and then to be able to estimate the asymptotic behaviour of their coefficients.

Both tasks may turn out to be very hard, and small differences on recursive definitions can lead to functions with completely different analytic behaviours.

We finish by pointing out some future directions of work. Concerning ε-NFAs, there are other conversions from REs to automata with better worst-case complexity. In particular, Gruber and Gulan's construction is optimal w.r.t. the alphabetic size of regular expressions [23, 25]. This construction corresponds to applying the ε-follow construction to the star normal form of the initial expression. Thus, if one is able to estimate the asymptotic number of expressions in star normal form of a given size, one can proceed as in Section 3.1 and obtain the average size of this construction.

Concerning DFAs, the main ingredient is to tackle the subset construction, i.e. determinization, within the analytic framework. Any progress in this direction will allow to obtain estimates for the average-case complexity of the RE to DFA conversions.

Nicaud et al. [38] studied the average number of transitions of Glushkov automata under the non-uniform distribution, inspired by random binary search trees (BST-like). With this distribution, the average number of transitions of the Glushkov automaton is quadratic with respect to the size of the regular expression. We believe that the same result is valid for the partial derivative automaton. However, we think that one needs a better understanding of the relevance of the different distributions for REs before this approach is applied to all the previous described constructions.

References

[1] Ricardo Almeida. Decision Algorithms for Kleene Algebra with Tests and Hoare Logic. Master's thesis, Faculdade de Ciências da Universidade do Porto, 2012.

[2] Valentin M. Antimirov. Partial Derivatives of Regular Expressions and Finite Automaton Constructions. *Theoretical Computer Science*, 155(2):291–319, 1996.

[3] Alasdair Armstrong, Georg Struth, and Tjark Weber. Program Analysis and Verification Based on Kleene Algebra in Isabelle/HOL. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *4th Inter. Conference ITP 2013, Rennes, France. Proceedings*, volume 7998 of *Lecture Notes on Computer Science*, pages 197–212. Springer, 2013.

[4] Andrea Asperti, Claudio Sacerdoti Coen, and Enrico Tassi. Regular Expressions, au point. *CoRR*, abs/1010.2604, 2010.

[5] Sabine Broda, Sílvia Cavadas, Miguel Ferreira, and Nelma Moreira. Derivative Based Methods for Deciding SKA and SKAT. Submitted.

[6] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. On the Average State Complexity of Partial Derivative Automata: an Analytic Combinatorics Approach. *International Journal of Foundations of Computer Science*, 22(7):1593–1606, 2011.

[7] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. On the Average Size of Glushkov and Partial Derivative Automata. *International Journal of Foundations of Computer Science*, 23(5):969–984, 2012.

[8] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. On the Average Size of Glushkov and Equation Automata for KAT Expressions. In *19th Inter. Symposium on Fundamentals of Computation Theory*, volume 8070 of *Lecture Notes on Computer Science*, pages 72–83. Springer, 2013.

[9] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. A Hitchhiker's Guide to Descriptional Complexity through Analytic Combinatorics. *Theoretical Computer Science*, 528:85–100, 2014.

[10] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. On the Equivalence of Automata for KAT-expressions. In Arnold Beckmann, Erzsébet Csuhaj-Varjú, and Klaus Meer, editors, *Language, Life, Limits - 10th Conference on Computability in Europe, CiE 2014, Budapest, Hungary, June 23-27, 2014. Proceedings*, volume 8493 of *Lecture Notes on Computer Science*, pages 73–83. Springer, 2014.

[11] Sabine Broda, António Machiavelo, Nelma Moreira, and Rogério Reis. Partial Derivative Automaton for Regular Expressions with Shuffle. In Jeffrey Shallit and Alexander Okhotin, editors, *Proceedings of the 17th Int. Workshop on Descriptional Complexity of Formal Systems (DCFS15)*. Springer, 2015.

[12] Anne Brüggemann-Klein. Regular Expressions into Finite Automata. *Theoretical Computer Science*, 48:197–213, 1993.

[13] John Brzozowski. Derivatives of Regular Expressions. *J. Association for Computer Machinery*, 11(4):481–494, 1964.

[14] Pascal Caron, Jean-Marc Champarnaud, and Ludovic Mignot. Partial Derivatives of an Extended Regular Expression. In Adrian Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications — 5th International Conference, LATA 2011, Tarragona, Spain, May 26-31, 2011. Proceedings*, volume 6638 of *Lecture Notes in Computer Science*, pages 179–191. Springer, 2011.

[15] Jean-Marc Champarnaud, Faissal Ouardi, and Djelloul Ziadi. Follow Automaton versus Equation Automaton. In Lucian Ilie and Detlef Wotschke, editors, *DCFS*, volume Report No. 619, pages 145–153. Department of Computer Science, The University of Western Ontario, Canada, 2004.

[16] Jean-Marc Champarnaud and Djelloul Ziadi. Computing the Equation Automaton of a Regular Expression in Space and Time. In Amihood Amir and Gad M. Landau,

editors, *CPM*, volume 2089 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 2001.

[17] Jean-Marc Champarnaud and Djelloul Ziadi. Canonical Derivatives, Partial Derivatives and Finite Automaton Constructions. *Theoretical Computer Science*, 289:137–163, 2002.

[18] Jean-Marc Champarnaud and Djelloull Ziadi. From Mirkin's Prebases to Antimirov's Word Partial Derivatives. *Fundamenta Informaticae*, 45(3):195–205, 2001.

[19] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[20] Wouter Gelade. Succinctness of Regular Expressions with Interleaving, Intersection and Counting. *Theoretical Computer Science*, 411(31-33):2987–2998, 2010.

[21] V. M. Glushkov. The Abstract Theory of Automata. *Russian Math. Surveys*, 16(5):1–53, 1961.

[22] Hugo Gouveia, Nelma Moreira, and Rogério Reis. Small NFAs from Regular Expressions: Some Experimental Results. In Fernando Ferreira, Hélia Guerra, Elvira Mayordomo, and João Rasga, editors, *Proceedings of 6th Conference on Computability in Europe (CIE 2010)*, pages 194–203, Ponta Delgada, Azores, Portugal, June/July 2010. CMATI.

[23] Hermann Gruber and Stefan Gulan. Simplifying Regular Expressions. In Adrian Horia Dediu, Henning Fernau, and Carlos Martín-Vide, editors, *4th International Conference on Language and Automata Theory and Applications, LATA 2010. Proceedings*, volume 6031 of *Lecture Notes on Computer Science*, pages 285–296, Trier, Germany, 05 2010. Springer.

[24] Hermann Gruber and Markus Holzer. Finite Automata, Digraph Connectivity, and Regular Expression Size. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *35th ICALP*, volume 5126 of *Lecture Notes on Computer Science*, pages 39–50. Springer, 2008.

[25] Hermann Gruber and Markus Holzer. From Finite Automata to Regular Expressions and Back-A Summary on Descriptional Complexity. In Zoltán Ésik and Zoltán Fülöp, editors, *Proceedings 14th International Conference on Automata and Formal Languages, AFL 2014, Szeged, Hungary, May 27-29, 2014.*, volume 151 of *EPTCS*, pages 25–48, 2014.

[26] Lucien Ilie and Sheng Yu. Follow Automata. *Information and Computation*, 186(1):140–162, 2003.

[27] Dexter Kozen. *Automata and Computability*. Springer, 1997.

[28] Dexter Kozen. Kleene Algebra with Tests. *Trans. on Prog. Lang. and Systems*, 19(3):427–443, 05 1997.

[29] Dexter Kozen. Automata on Guarded Strings and Applications. *Matématica Contemporânea*, 24:117–139, 2003.

[30] Dexter Kozen. On the Coalgebraic Theory of Kleene Algebra with Tests. Computing and Information Science Technical Reports `http://hdl.handle.net/1813/10173`, Cornell University, May 2008.

[31] Eva Maia, Nelma Moreira, and Rogério Reis. Partial Derivative and Position Bisimilarity Automata. In Markus Holzer and Martin Kutrib, editors, *Implementation and Application of Automata, 19th International Conference (CIAA 2014)*, volume 8587 of *Lecture Notes on Computer Science*, pages 264–277. Springer, 2014.

[32] Eva Maia, Nelma Moreira, and Rogério Reis. Prefix and Right-Partial Derivative Automata. In Mariya Soskova and Victor Mitrana, editors, *Computability in Europe (CiE 2015)*, number 9136 in Theoretical Computer Science and General Issues, pages 1–10. Springer, 2015.

[33] Alain J. Mayer and Larry J. Stockmeyer. Word Problems—This Time with Interleaving. *Information and Computation*, 115(2):293–311, 1994.

[34] R. McNaughton and H. Yamada. Regular Expressions and State Graphs for Automata. *IEEE Transactions on Electronic Computers*, 9:39–47, 1960.

[35] B. G. Mirkin. An Algorithm for Constructing a Base in a Language of Regular Expressions. *Engineering Cybernetics*, 5:51–57, 1966.

[36] Cyril Nicaud. *Étude du comportement en moyenne des automates finis et des langages rationnels*. PhD thesis, Université de Paris 7, 2000.

[37] Cyril Nicaud. On the Average Size of Glushkov's Automata. In Adrian Horia Dediu, Armand-Mihai Ionescu, and Carlos Martín-Vide, editors, *Proc. 3rd LATA*, volume 5457 of *Lecture Notes on Computer Science*, pages 626–637. Springer, 2009.

[38] Cyril Nicaud, Carine Pivoteau, and Benoît Razet. Average Analysis of Glushkov Automata under a BST-Like Model. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*, pages 388–399, 2010.

[39] Tobias Nipkow and Dmitriy Traytel. Unified Decision Procedures for Regular Expression Equivalence. *Archive of Formal Proofs*, 2014, 2014.

[40] David Pereira. *Towards Certified Program Logics for the Verification of Imperative Programs*. PhD thesis, University of Porto, 2013.

[41] Damien Pous. Kleene Algebra with Tests and Coq Tools for while Programs. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving — 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013. Proceedings*, volume 7998 of *Lecture Notes in Computer Science*, pages 180–196. Springer, 2013.

[42] Damien Pous. Symbolic Algorithms for Language Equivalence and Kleene Algebra with Tests. In Sriram K. Rajamani and David Walker, editors, *42nd POPL 2015*, pages 357–368. ACM, 2015.

[43] Cristian Prisacariu. Synchronous Kleene algebra. *J. Log. Algebr. Program.*, 79(7):608–635, 2010.

[44] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

[45] Alexandra Silva. Position Automata for Kleene Algebra with Tests. *Sci. Ann. Comp. Sci.*, 22(2):367–394, 2012.

[46] Seppo Sippu and Eljas Soisalon-Soininen. *Parsing Theory*, volume II: LR(k) and LL(k) Parsing of *EATCS Monographs on Theoretical Computer Science*. Springer, 1990.

[47] K. Thompson. Regular Expression Search Algorithm. *Communications of the ACM*, 11(6):410–422, 1968.

[48] Hiroaki Yamamoto. A New Finite Automaton Construction for Regular Expressions. In Suna Bensch, Rudolf Freund, and Friedrich Otto, editors, *Sixth Workshop on Non-Classical Models for Automata and Applications - NCMA 2014, Kassel, Germany, July 28-29, 2014. Proceedings*, volume 304 of *books@ocg.at*, pages 249–264. Österreichische Computer Gesellschaft, 2014.