On the Average Size of Glushkov and Equation Automata for KAT Expressions[⋆]

Sabine Broda, António Machiavelo, Nelma Moreira, Rogério Reis
sbb@dcc.fc.up.pt,ajmachia@fc.up.pt,{nam,rvr}@dcc.fc.up.pt

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

Abstract. Kleene algebra with tests (KAT) is an equational system that extends Kleene algebra, the algebra of regular expressions, and that is specially suited to capture and verify properties of simple imperative programs. In this paper we study two constructions of automata from KAT expressions: the Glushkov automaton ($\mathcal{A}_{\mathsf{pos}}$), and a new construction based on the notion of prebase (equation automata, $\mathcal{A}_{\mathsf{eq}}$). Contrary to other automata constructions from KAT expressions, these two constructions enjoy the same descriptional complexity behaviour as their counterparts for regular expressions, both in the worst-case as well as in the average-case. In particular, our main result is to show that, asymptotically and on average the number of transitions of the $\mathcal{A}_{\mathsf{pos}}$ is linear in the size of the KAT expression.

1 Introduction

Kleene algebra with tests (KAT) [11] is an equational system for propositional program verification that combines Boolean algebra (BA) with Kleene algebra (KA), the algebra of the regular expressions. The equational theory of KAT is PSPACE-complete and can be reduced to the equational theory of KA, with an exponential cost [7, 14]. Several automata constructions from KAT expressions have been proposed in order to obtain feasible decision procedures for KAT expressions equivalence [12, 18, 13, 17, 1]. Regular sets of guarded strings [10] are the standard models for KAT (as regular languages are for KA) [12]. A coalgebraic approach based on the notion of (partial) derivatives and automata on guarded strings were developed by Kozen [13], and implemented, with slightly modifications, by Almeida *et al.* [1]. Silva [17] presented yet another automata construction, extending for KAT the Glushkov construction, well known for the conversion of regular expressions to nondeterministic finite automata [9]. All the constructions of automata on guarded strings, with the exception of Silva's, induce an exponential blow-up on the number of states/transitions of the automata. This is due to the use of all valuations of the boolean expressions that

⋆ This work was funded by the European Regional Development Fund through the programme COMPETE and by the FCT under projects PEst-C/MAT/UI0144/2011, CANTE-PTDC/EIA-CCO/101904/2008, and FCOMP-01-0124-FEDER-020486.

occur in a KAT expression, and also induces an extra exponential factor when testing the equivalence of two KAT expressions. In this paper, we present a new construction to obtain an automaton from a KAT expression, adapting the Mirkin construction of an *equation automaton* [15]. For regular expressions, this construction coincides with Antimirov's partial derivative automaton [5], that it is known to be a quotient of the Glushkov automaton [6]. The number of states of the Glushkov automaton equals to the number of occurrences of alphabetic symbols in the regular expression, and its number of transitions is, in the worst-case, quadratic in that number. Herein, we also observe that, in the worst-case, the number of transitions of the the Glushkov automaton is quadratic in the size of the KAT expression. Nicaud [16] and Broda *et al.* [2, 4] studied the average size of these two automata for regular languages, using the framework of analytic combinatorics. Asymptotically, the average size of the Glushkov automaton is linear in the size of the regular expression, and the size of the equation (partial derivative) automaton is half that size. We show that similar results hold for their analogue constructions for KAT expressions. The main outcome is the asymptotical linearity of the average number of transitions of the Glushkov automaton, i.e. for KAT expressions of size n it is $\Theta(n)$. This also provides an upper bound for the number of transitions of the equation automaton. These results come as a surprise, due to the bad behaviour of other automata constructions from KAT expressions, and can lead to more efficient decision procedures for KAT expressions equivalence. Note that the equation automaton can be more suitable to use in decision procedures based on coalgebraic methods, due to the fact that states correspond to combinations of subexpressions of the initial expressions.

2 KAT expressions, Automata, and Guarded Strings

Let $\mathsf{P} = \{p_1, \ldots, p_k\}$ be a non-empty set of *program* symbols and $\mathsf{T} = \{t_1, \ldots, t_l\}$ be a non-empty set of *test* symbols. The set of boolean expressions over T together with negation, disjunction and conjunction, is denoted by BExp, and the set of KAT expressions with disjunction, concatenation, and Kleene star, by Exp. The abstract syntax of KAT expressions, over an alphabet $\mathsf{P} \cup \mathsf{T}$, is given by the following grammar, where $p \in \mathsf{P}$ and $t \in \mathsf{T}$,

$$\mathsf{BExp}: \quad b \to 0 \mid 1 \mid t \mid \neg b \mid b + b \mid b \cdot b \tag{1}$$

$$\mathsf{Exp}: \quad e \to p \mid b \mid e + e \mid e \cdot e \mid e^\star. \tag{2}$$

As usual, we omit the operator $\cdot$ whenever it does not give rise to any ambiguity. The size of a KAT expression e, denoted by $|e|$, is the number of symbols in the syntactic tree of e. The set At, of *atoms* over T, is the set of all boolean assignments to all elements of T, $\mathsf{At} = \{x_1 \cdots x_l \mid x_i \in \{t_i, \overline{t}_i\},\ t_i \in \mathsf{T}\}$.

Now, 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 [12]. For $x = \alpha_1 p_1 \cdots p_{m-1} \alpha_m, y = \beta_1 q_1 \cdots q_{n-1} \beta_n \in \mathsf{GS}$, where $m, n \geq 1$, $\alpha_i, \beta_j \in \mathsf{At}$ and $p_i, q_j \in \mathsf{P}$, we define the *fusion product* $x \diamond y = \alpha_1 p_1 \cdots p_{m-1} \alpha_m q_1 \cdots q_{n-1} \beta_n$, if

$\alpha_m = \beta_1$ and undefined, otherwise. For sets $X, Y \subseteq \mathsf{GS}$, $X \diamond Y$ is the set of all $x \diamond y$ such that $x \in X$ and $y \in Y$. Let $X^0 = \mathsf{At}$ and $X^{n+1} = X \diamond X^n$, for $n \geq 0$.

Given a KAT expression e we define $\mathsf{GS}(e) \subseteq \mathsf{GS}$ inductively as follows:

$$\begin{aligned} \mathsf{GS}(p) &= \{\, \alpha p \beta \mid \alpha, \beta \in \mathsf{At} \,\} \\ \mathsf{GS}(b) &= \{\, \alpha \mid \alpha \in \mathsf{At} \wedge \alpha \leq b \,\} \end{aligned} \qquad \begin{aligned} \mathsf{GS}(e_1 + e_2) &= \mathsf{GS}(e_1) \cup \mathsf{GS}(e_2) \\ \mathsf{GS}(e_1 \cdot e_2) &= \mathsf{GS}(e_1) \diamond \mathsf{GS}(e_2) \\ \mathsf{GS}(e_1^\star) &= \cup_{n \geq 0} \mathsf{GS}(e_1)^n, \end{aligned}$$

where $\alpha \leq b$ if $\alpha \rightarrow b$ is a propositional tautology.

Example 1. Consider $e = t_1 + (\neg t_1)(t_2 p)^*$, where $\mathsf{P} = \{p\}$ and $\mathsf{T} = \{t_1, t_2\}$. Then, $\mathsf{At} = \{t_1 t_2, t_1 \overline{t_2}, \overline{t_1} t_2, \overline{t_1} \overline{t_2}\}$ and

$$\begin{aligned} \mathsf{GS}(e) &= \mathsf{GS}(t_1) \cup \mathsf{GS}(\neg t_1) \diamond \mathsf{GS}((t_2 p)^*) \\ &= \{t_1 t_2, t_1 \overline{t_2}\} \cup \{\overline{t_1} t_2, \overline{t_1} \overline{t_2}\} \cup \{\, \overline{t_1} (t_2 p)^n \alpha \mid n \geq 1, \alpha \in \mathsf{At} \,\} \end{aligned}$$

Given two KAT expressions e_1 and e_2, we say that they are *equivalent*, and write $e_1 = e_2$, if $\mathsf{GS}(e_1) = \mathsf{GS}(e_2)$.

A (non-deterministic) automaton over the alphabets P and T is a tuple $\mathcal{A} = \langle S, s_0, o, \delta \rangle$, where S is a finite set of states, $s_0 \in S$ is the initial state, $o : S \rightarrow \mathsf{BExp}$ is the output function, and $\delta \subseteq 2^{S \times (\mathsf{BExp} \times \mathsf{P}) \times S}$ is the transition relation. A guarded string $\alpha_1 p_1 \ldots p_{n-1} \alpha_n$, with $n \geq 1$, is accepted by the automaton $\mathcal{A}$ if and only if there is a sequence of states $s_0, s_1, \ldots, s_{n-1} \in S$, where $s_0 = s$, and, for $i = 1, \ldots, n-1$, one has $\alpha_i \leq b_i$ for some $(s_{i-1}, (b_i, p_i), s_i) \in \delta$, and $\alpha_n \leq o(s_{n-1})$. The set of all guarded strings accepted by $\mathcal{A}$ is denoted by $\mathsf{GS}(\mathcal{A})$.

Formally, we define $L : S \longrightarrow \mathsf{GS} \longrightarrow \{0, 1\}$, by structural induction on $x \in \mathsf{GS}$ as follows.

$$L(s)(\alpha) = \begin{cases} 1 \text{ if } \alpha \leq o(s), \\ 0 \text{ otherwise.} \end{cases} \quad L(s)(\alpha p x) = \begin{cases} 1 \text{ if } (s, (b, p), s') \in \delta \text{ for } s' \in S, \\ \quad b \text{ s.t. } \alpha \leq b, \text{ and } L(s')(x) = 1, \\ 0 \text{ otherwise.} \end{cases}$$

Given $s \in S$, let $\mathsf{GS}(s) = \{\, x \in \mathsf{GS} \mid L(s)(x) = 1 \,\}$. Then, $\mathsf{GS}(\mathcal{A}) = \mathsf{GS}(s_0)$. We say that a KAT expression $e \in \mathsf{Exp}$ is *equivalent* to an automaton $\mathcal{A}$, and write $e = \mathcal{A}$, if $\mathsf{GS}(\mathcal{A}) = \mathsf{GS}(e)$.

In the next two sections we present two different constructions of automata that are equivalent to a given KAT expression.

3 The Glushkov Automaton

The definition of the Glushkov automaton for KAT expressions follows closely the one given by Silva [17]. Let $\tilde{e}$ denote the KAT expression obtained by considering the number of elements of P occurring in e, marking each one with its appearance number, that is called its *position*. The same notation is used to denote the removal of the markings, *i.e.*, $\tilde{\tilde{e}} = e$. The set of positions in an expression e is denoted by $\mathsf{pos}(e)$. Note that this marking does not apply to test symbols, which always remain unchanged.

Example 2. Consider $e = t_1 p(pq^\star t_2 + t_3 q)^\star$, where $\mathsf{P} = \{p, q\}$ and $\mathsf{T} = \{t_1, t_2, t_3\}$. Then, $\tilde{e} = t_1 p_1 (p_2 q_3^\star t_2 + t_3 q_4)^\star$.

Definition 1. *We recursively define the functions* first*,* follow*,* last*, and* out *according to grammar* (2)

$$\mathsf{first} : \mathsf{Exp} \longrightarrow 2^{\mathsf{BExp} \times \mathsf{P}}$$

$$\mathsf{first}(p) = \{(1, p)\} \qquad \mathsf{first}(e_1 + e_2) = \mathsf{first}(e_1) \cup \mathsf{first}(e_2)$$

$$\mathsf{first}(b) = \emptyset \qquad \mathsf{first}(e_1 \cdot e_2) = \begin{cases} \mathsf{first}(e_1) & \textit{if } \mathsf{out}(e_1) = 0 \\ \mathsf{first}(e_1) \cup \mathsf{out}(e_1) \cdot_1 \mathsf{first}(e_2) & \textit{otherwise.} \end{cases}$$

$$\mathsf{first}(e^\star) = \mathsf{first}(e),$$

$$\mathsf{last} : \mathsf{Exp} \longrightarrow 2^{\mathsf{P} \times \mathsf{BExp}}$$

$$\mathsf{last}(p) = \{(p, 1)\} \qquad \mathsf{last}(e_1 + e_2) = \mathsf{last}(e_1) \cup \mathsf{last}(e_2)$$

$$\mathsf{last}(b) = \emptyset \qquad \mathsf{last}(e_1 \cdot e_2) = \begin{cases} \mathsf{last}(e_2) & \textit{if } \mathsf{out}(e_2) = 0 \\ \mathsf{last}(e_2) \cup \mathsf{last}(e_1) \cdot_2 \mathsf{out}(e_2) & \textit{otherwise.} \end{cases}$$

$$\mathsf{last}(e^\star) = \mathsf{last}(e),$$

$$\mathsf{follow} : \mathsf{Exp} \longrightarrow 2^{\mathsf{P} \times \mathsf{BExp} \times \mathsf{P}} \qquad \mathsf{out} : \mathsf{Exp} \longrightarrow \mathsf{BExp}$$

$$\mathsf{follow}(p) = \emptyset \qquad \mathsf{out}(p) = 0$$

$$\mathsf{follow}(b) = \emptyset \qquad \mathsf{out}(b) = b$$

$$\mathsf{follow}(e_1 + e_2) = \mathsf{follow}(e_1) \cup \mathsf{follow}(e_2) \qquad \mathsf{out}(e_1 + e_2) = \mathsf{out}(e_1) + \mathsf{out}(e_2)$$

$$\mathsf{follow}(e_1 \cdot e_2) = \mathsf{follow}(e_1) \cup \mathsf{follow}(e_2) \cup \mathsf{last}(e_1) \otimes \mathsf{first}(e_2) \qquad \mathsf{out}(e_1 \cdot e_2) = \mathsf{out}(e_1) \cdot \mathsf{out}(e_2)$$

$$\mathsf{out}(e^\star) = 1.$$

$$\mathsf{follow}(e^\star) = \mathsf{follow}(e) \cup \mathsf{last}(e) \otimes \mathsf{first}(e).$$

where, for $X \subseteq \mathsf{BExp} \times \mathsf{Exp}$, $Y \subseteq \mathsf{Exp} \times \mathsf{BExp}$ *and* $b \in \mathsf{BExp}$*, we have* $b \cdot_1 X = \{ (bb', p) \mid (b', p) \in X \}$, $Y \cdot_2 b = \{ (b'b, p) \mid (p, b') \in Y \}$ *and* $Y \otimes X = \{ (p, bb', p') \mid (p, b) \in Y, (b', p') \in X \}$*, with the caveat that* $0 \cdot b = b \cdot 0 = 0$ *and* $1 + b = b + 1 = 1$.

Informally, given $e \in \mathsf{Exp}$, the elements of $\mathsf{first}(e)$ are pairs (b, p) such that $\alpha p x \in \mathsf{GS}(e)$ and $\alpha \leq b$; the elements of $\mathsf{last}(e)$ are pairs (p, b) such that $x p \alpha \in \mathsf{GS}(e)$ and $\alpha \leq b$; the elements of $\mathsf{follow}(e)$ are triplets (p, b, q) such that $x p \alpha q y \in \mathsf{GS}(e)$ and $\alpha \leq b$; and $\mathsf{out}(e) \in \mathsf{BExp}$ corresponds to the values $\alpha \in \mathsf{At}$ such that $\alpha \leq e$.

Example 3. Consider the expression $\tilde{e}$ of Example 2. One has,

$$\begin{aligned}
\mathsf{first}(\tilde{e}) &= \{(t_1, p_1)\}\} \\
\mathsf{last}(\tilde{e}) &= \{(p_1, 1), (p_2, t_2), (q_3, t_2), (q_4, 1)\} \\
\mathsf{follow}(\tilde{e}) &= \{(p_1, 1, p_2), (p_1, t_3, q_4), (p_2, 1, q_3), (p_2, t_2, p_2), (p_2, t_2 t_3, q_4) \\
&\quad (q_3, 1, q_3), (q_3, t_2, p_2), (q_3, t_2 t_3, q_4), (q_4, 1, p_2), (q_4, t_3, q_4)\} \\
\mathsf{out}(\tilde{e}) &= \mathsf{out}(t_1) \cdot \mathsf{out}(p_1) \cdot \mathsf{out}((p_2 q_3^\star t_2 + t_3 q_4)^\star) = t_1 \cdot 0 \cdot 1 = 0
\end{aligned}$$

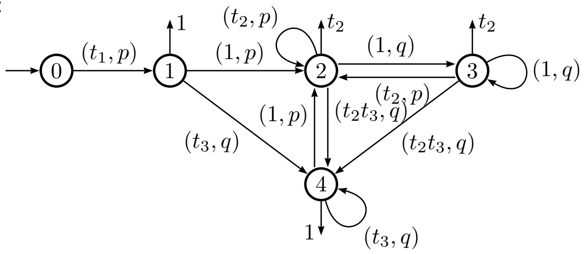
Definition 2 (Glushkov Automaton). *For* $e \in \mathsf{Exp}$*, we define the Glushkov automaton* $\mathcal{A}_{\mathsf{pos}}(e) = \langle \mathsf{pos}(e) \cup \{0\}, 0, o, \delta_{pos} \rangle$*, where* $o(0) = \mathsf{out}(\tilde{e})$*,* $o(i) = b$ *if*

$i > 0$ and $(p_i, b) \in \mathsf{last}(\tilde{e})$, and $o(i) = 0$, otherwise; and

$$\delta_{pos} = \{\ (0, (b, p), j) \mid (b, p_j) \in \mathsf{first}(\tilde{e}), p = \tilde{p_j}\ \} \bigcup$$
$$\{\ (i, (b, p), j) \mid (p_i, b, p_j) \in \mathsf{follow}(\tilde{e}), p = \tilde{p_j}\ \}\,.$$

Analogously to what happens for regular expressions, given an expression e the Glushkov automaton $\mathcal{A}_{\mathsf{pos}}(e)$ has exactly $|e|_{\mathsf{P}} + 1$ states, where $|e|_{\mathsf{P}}$ denotes the number of occurrences of program symbols (elements of P) in the expression e. This means that the boolean parts of an expression do not affect the number of states in its corresponding Glushkov automaton, contrary to what happens in other constructions, cf. [12, 13]. Furthermore, the number of transitions of $\mathcal{A}_{\mathsf{pos}}(e)$ is in the worst-case $\mathcal{O}(|e|^2_{\mathsf{P}})$. This results from the fact that, for every marked expression $\tilde{e}$ and for every marked program symbols p_j and p_i, there is at most one pair $(b, p_j) \in \mathsf{first}(\tilde{e})$ and at most one pair $(p_j, b') \in \mathsf{last}(\tilde{e})$, for $b, b' \in \mathsf{BExp}$; and there is at most one tuple $(p_i, b, p_j) \in \mathsf{follow}(\tilde{e})$, for $b \in \mathsf{BExp}$.

Example 4. Consider again the expression e of Example 2 and the functions computed in Example 3. In this case, one has $\mathsf{pos}(e) = \{1, 2, 3, 4\}$ and the $\mathcal{A}_{\mathsf{pos}}(e)$, is the following:



Proposition 1. *[17, Th. 3.2.7] For every KAT expression, $e \in \mathsf{Exp}$, one has $\mathsf{GS}(\mathcal{A}_{\mathsf{pos}}(e)) = \mathsf{GS}(e)$.*

4 The Equation Automaton

In this section, we give a definition of the equation automaton for a KAT expression, extending the notion of prebase of a regular expression due to Mirkin [15]. Here we do not consider the equivalence of this construction to the partial derivative automata [13, 1], since KAT derivatives are considered with respect to αp, ($\alpha \in \mathsf{At}$) and we want to avoid the possible exponential blow-up associated to the set of atoms.

Definition 3. *Given $e \in \mathsf{Exp}$, a set of non-null expressions $\mathsf{E} = \{e_1, \ldots, e_n\} \subseteq \mathsf{Exp}$ is called a* support *of e, if the following system of equations holds:*

$$\begin{aligned}
e \equiv e_0 &= P_{01}e_1 + \cdots + P_{0n}e_n + \mathsf{out}(e_0)\\
e_1 &= P_{11}e_1 + \cdots + P_{1n}e_n + \mathsf{out}(e_1)\\
&\vdots\\
e_n &= P_{n1}e_1 + \cdots + P_{nn}e_n + \mathsf{out}(e_n),
\end{aligned} \tag{3}$$

where $P_{ij} = \sum_{r=1}^{k} b_{ijr}p_r$, for $0 \leq i, j \leq n$. For the components bp of a sum P_{ij}, we write $bp \prec P_{ij}$. If E is a support of e, then the set $\mathsf{E}_0 = \mathsf{E} \cup \{e\}$ is called a prebase *of e.*

Note that, if E is a support of e, we may have $e \in \mathsf{E}$. The system of equations (3) can be written in matrix form $\mathsf{E}_0 = \mathsf{PE}+\mathsf{O}(\mathsf{E}_0)$, where P is the $(n+1)\times n$ matrix with entries P_{ij}, and E, E_0 and $\mathsf{O}(\mathsf{E}_0)$ are, respectively, the column matrices

$$\mathsf{E} = \begin{bmatrix} e_1 \\ \vdots \\ e_n \end{bmatrix}, \quad \mathsf{E}_0 = \begin{bmatrix} e_0 \\ \vdots \\ e_n \end{bmatrix} \quad \text{and} \quad \mathsf{O}(\mathsf{E}_0) = \begin{bmatrix} \mathsf{out}(e_0) \\ \vdots \\ \mathsf{out}(e_n) \end{bmatrix}.$$

In the following, we will arbitrarily interchange the matrix and the set notation for the above values. Considering the notion of KAT expression equivalence, we have the following lemma.

Lemma 1. *Given $e \in \mathsf{Exp}$ and $\mathsf{E}_0 = \{e = e_0, e_1, \ldots, e_n\}$ a prebase of e, then*

a) $\alpha \in \mathsf{GS}(e_i)$ iff $\alpha \leq \mathsf{out}(e_i)$;
b) $\alpha px \in \mathsf{GS}(e_i)$ iff there are $j \in \{0, \ldots, n\}$ and $b \in \mathsf{BExp}$, such that $bp \prec P_{ij}$, $\alpha \leq b$, and $x \in \mathsf{GS}(e_j)$.

Given a prebase $\mathsf{E}_0 = \{e = e_0, e_1, \ldots, e_n\}$, an NFA can be defined by

$$\mathcal{A}_{\mathsf{E}_0}(e) = \langle \{e = e_0, e_1, \ldots, e_n\}, e, \mathsf{out}, \delta \rangle, \tag{4}$$

where

$$\delta = \{\ (e_i, (b_{ijr}, p_r), e_j) \mid 0 \leq i, j \leq n,\ 1 \leq r \leq k\ \}. \tag{5}$$

Using Lemma 1, it is easy to see that $\mathsf{GS}(\mathcal{A}_{\mathsf{E}_0}(e)) = \mathsf{GS}(e)$.

Definition 4 (Equation Automaton). *For $e \in \mathsf{Exp}$, we define the equation automaton $\mathcal{A}_{\mathsf{eq}}(e) = \langle \{e\} \cup \pi(e), e, \mathsf{out}, \delta_{\mathsf{eq}} \rangle$ with $\delta_{\mathsf{eq}} = \{\ (e_i, (b, p), e_j) \mid bp \prec P_{ij} \in \mathsf{P}(e)\ \}$ and where $\pi : \mathsf{Exp} \to 2^{\mathsf{Exp}}$ is defined by induction on the structure of e as follows:*

$$\begin{array}{ll} \pi(p) = \{1\} & \pi(e+f) = \pi(e) \cup \pi(f) \\ & \pi(e \cdot f) = \pi(e)f \cup \pi(f) \\ \pi(b) = \emptyset & \pi(e^\star) = \pi(e)e^\star, \end{array} \tag{6}$$

and $\mathsf{P}(e)$ is a $(n+1) \times n$ matrix with entries $P_{ij} \in \mathsf{Exp}$, $n = |\pi(e)|$, and which is inductively defined on the structure of e by

$$\mathsf{P}(p) = \begin{bmatrix} 1p \\ 0 \end{bmatrix} \qquad \mathsf{P}(b) = 0 \qquad \mathsf{P}(e+f) = \left[\begin{array}{c:c} \mathsf{P}(e) & \begin{array}{c} \mathsf{P}(f)\big|_0 \\ 0 \end{array} \\ \hdashline 0 & \mathsf{P}(f)\big|_{1..m} \end{array}\right]$$

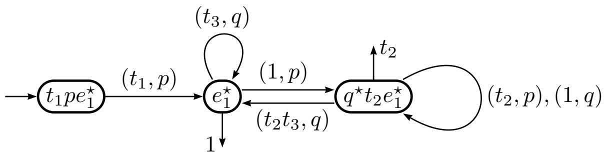
$$\mathsf{P}(ef) = \left[\begin{array}{c:c} \mathsf{P}(e) & \mathsf{O}(\pi(e)) \odot P(f)\big|_0 \\ \hdashline 0 & \mathsf{P}(f)\big|_{1..m} \end{array}\right] \quad \mathsf{P}(e^\star) = P(e) + \begin{bmatrix} 0 \\ \mathsf{O}(\pi(e)) \odot P(e)\big|_0 \end{bmatrix}, \tag{7}$$

where $\mathsf{P}(f)$ *is an* $(m+1)\times m$ *matrix, for some* $m>0$*;* $P(f)\big|_0$ *denotes the first row of matrix* $\mathsf{P}(f)$*;* $P(f)\big|_{1..m}$ *denotes the matrix* $\mathsf{P}(f)$ *without the first row; and the* $\odot$ *operator is defined as follows*

$$\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \odot \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} x_1y_1 & \dots & x_1y_m \\ \vdots & & \vdots \\ x_ny_1 & \dots & x_ny_m \end{bmatrix}.$$

Note, that the above definition of π follows closely the one for regular expressions by Mirkin. Analogously, it can be easily shown that for a KAT expression e, one has $|\pi(e)| \le |e|_{\mathsf{P}}$. Consequently, the number of states in the equation automaton of an expression e is $|\pi(e)\cup\{e\}| \le |e|_{\mathsf{P}}+1$, thus smaller or equal than the number of states of the Glushkov automaton.

Example 5. For the expression $e = t_1pe_1^\star$ of Example 2, where $e_1 = pq^\star t_2 + t_3q$. One has $\pi(e) = \{q^\star t_2e_1^\star, e_1^\star\}$ and the $\mathcal{A}_{\mathsf{eq}}(e)$ is the following:



One has $|e|_{\mathsf{P}}+1 = 5$, while the number of states of $\mathcal{A}_{\mathsf{eq}}(e)$ is $|\pi(e)\cup\{e\}| = 3$.

In order to show that the equation automaton $\mathcal{A}_{\mathsf{eq}}(e)$ is equivalent to e it is enough to show that the function π is a support of e.

Proposition 2. *Given* $e \in \mathsf{Exp}$*, one has* $\mathsf{GS}(\mathcal{A}_{\mathsf{eq}}(e)) = \mathsf{GS}(e)$.

Proof (Sketch). By Lemma 1, it is enough to show that $\mathcal{A}_{\mathsf{eq}}(e) = \mathcal{A}_{\mathsf{E}_0}(e)$, where E_0 is the prebase $\{e\}\cup\pi(e)$. The proof that $\pi(e)$ is a support proceeds by induction on the structure of e and follows the lines of Mirkin's proof. The definition of the matrix P ensures that the system of equations (3) is satisfied by $\{e\}\cup\pi(e)$.

5 Average Size of Glushkov and Equation Automata

In this section, we estimate the asymptotic average size of the Glushkov and equation automata. This is done by the use of the standard methods of analytic combinatorics as expounded by Flajolet and Sedgewick [8]. These apply to generating functions $A(z) = \sum_n a_nz^n$ for a combinatorial class $\mathcal{A}$ with a_n objects of size n, denoted by $[z^n]A(z)$, and also bivariate functions $C(u,z) = \sum_\alpha u^{c(\alpha)}z^{|\alpha|}$, where $c(\alpha)$ is some measure of the object $\alpha \in \mathcal{A}$.

In order to apply this method, it is necessary to have an unambiguous description of the objects of the combinatorial class. One can see that the grammar (2) for KAT expressions is ambiguous. But, we can use the following non-ambiguous grammar for KAT expressions, where expressions AExp correspond to KAT expressions with at least one program symbol $p \in \mathsf{P}$. For simplicity, we exclude expressions that contain subexpressions of the form $b^\star$, as their semantics correspond to At and thus are equivalent to 1.

$$\begin{aligned} \mathsf{BExp}: &\quad b \to 0 \mid 1 \mid t \mid \neg b \mid b+b \mid b \cdot b && (8)\\ \mathsf{AExp}: &\quad a \to p \mid a+a \mid a+b \mid b+a \mid a\cdot a \mid a \cdot b \mid b \cdot a \mid a^\star && (9)\\ \mathsf{Exp}: &\quad e \to b \mid a. && (10)\end{aligned}$$

From the definitions above, one can compute the generating functions $B_l(z)$, $A_{k,l}(z)$, and $E_{k,l}(z)$, for the number of boolean expressions BExp, expressions AExp, and KAT expressions Exp, respectively. However, it is easy to see that $B_l(z)$ and $E_{k,l}(z)$ coincide with the generating function of standard regular expressions. Considering the following grammar for regular expressions with an alphabet Σ of size m,

$$r \to 0 \mid 1 \mid \sigma \in \Sigma \mid r+r \mid r \cdot r \mid r^\star, \tag{11}$$

its generating function is given by

$$R_m(z) = \frac{1 - z - \sqrt{\Delta_m(z)}}{4z}, \text{ where } \Delta_m(z) = 1 - 2z - (15+8m)z^2. \tag{12}$$

We have $B_l(z) = R_l(z)$ and $E_{k,l}(z) = R_{k+l}(z)$. The first equality is due to the similarity of the grammars (8) and (11), where one has the negation operator and the other, the star operator. This fact and the exclusion of expressions of the form $b^\star$ leads to the second equality. As a consequence, we can easily adapt most of the results obtained for regular expressions [16, 2, 4] to KAT expressions.

Using the technique presented in Section 5 of Broda *et al.* [3] applied to (12), the asymptotic estimates for the number of regular expressions of size n is

$$[z^n]R_m(z) \sim \frac{\sqrt{\rho_m}\,\sqrt[4]{2m+4}}{4\sqrt{\pi}}\,\rho_m^{-(n+1)}(n+1)^{-\frac{3}{2}}. \tag{13}$$

where $\rho_m = \frac{-1+2\sqrt{2m+4}}{15+8m}$ is the radius of convergence of $R_m(z)$. This estimate differs from the one presented by Nicaud, and exhibits a slightly faster convergence in experimental tests.

Average State Complexity of the Glushkov Automaton. The generating function for the number of alphabetic symbols in regular expressions is $L_m(z) = \frac{mz}{\sqrt{\Delta_m(z)}}$ and $[z^n]L_m(z) \sim \frac{m}{2\sqrt{\rho_m \pi}\,\sqrt[4]{2m+4}}\rho_m^{-(n-1)}n^{-\frac{1}{2}}$. For regular expressions, this implies that, as the alphabet size grows, the average number of alphabetic symbols in a regular expression is $\frac{1}{2}$ its size. In KAT expressions we can estimate both the number of test symbols in boolean expressions, as well as, the number of program symbols in KAT expressions. Let $T_l(z)$ and $P_{k,l}(z)$ be their respective generating functions. We have, $T_l(z) = L_l(z)$ and $P_{k,l} = \frac{k}{k+l}L_{k+l}(z)$. Therefore, the probability, for a uniform distribution, that a symbol in a boolean expression of size n is a test symbol is

$$\frac{[z^n]T_l(z)}{n\,[z^n]B_l(z)} \sim \frac{\left(4\,l+8-\sqrt{2\,l+4}\right)l}{(15+8\,l)\,(l+2)}\left(1+\frac{1}{n}\right)^{3/2} \tag{14}$$

and the probability that a symbol in a KAT expression of size n is a program symbol is

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

The average number of test symbols in a boolean expression grows to about half of their size, as l tends to ∞. The average number of program symbols for a growing value of $k+l$ tends to $\frac{1}{2(1+c)}$, 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.

Average State Complexity of the Equation Automaton. Considering Definition 4, one notes that whenever $1 \in \pi(e) \cap \pi(f)$, two states are merged into a single one in $\pi(e+f)$. Analogously, for $\pi(ef)$, when $1 \in \pi(e)$ and $f \in \pi(f)$. These facts allowed to estimate an upper bound for the reduction of the number of states in the equation automaton, when compared with the number of states in the Glushkov automaton [2]. Actually, the presence of boolean expressions does not affect the computations, so we have exactly the same results for KAT expressions. Letting $I_{k,l}(z)$ be the cumulative generating function of the mergings, and using the results in [2], one has

$$\frac{[z^n]I_{k,l}(z)}{[z^n]E_{k,l}(z)} \sim \lambda_{k,l}\,n, \tag{16}$$

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

$$\frac{[z^n]I_{k,l}(z)}{[z^n]P_{k,l}(z)} \sim \frac{\lambda_{k,l}}{\eta_{k,l,n}}. \tag{17}$$

One can see that, for a fixed value of l this ratio approaches $\frac{1}{2}$, as k grows.

5.1 Average Transition Complexity of the Glushkov Automaton

Now we compute an upper bound for the asymptotic average of the number of transitions in a Glushkov automaton with respect to the size of the corresponding KAT expression. As observed before, the number of transitions of $\mathcal{A}_{\mathsf{pos}}(e)$ is, in the worst-case, quadratic in $|e|_{\mathsf{P}}$. Below, we show that it is on average linear in

$|e|$. As the number of transitions must be at least equal to the number of states minus 1, on average, that number should be $\Omega(n)$ for KAT expressions of size n.

By Definition 2, the number of transitions is the sum of the sizes of the sets first and follow. In order to obtain a sufficiently accurate upper bound, we have to identify the KAT expressions e such that $\mathsf{out}(e) = 0$. We begin to define the grammars that generate, respectively, "guaranteed" tautologies b_1, "guaranteed" falsities b_0, and, based on these, KAT expressions e_0 such that $\mathsf{out}(e_0) = 0$. As usual, $e_{\overline{0}}$ denotes an KAT expression that is not generated by this grammar, etc.

$$\begin{aligned}
&b_1 \to 1 \mid \neg b_0 \mid b_1 + b \mid b_{\overline{1}} + b_1 \mid b_1 \cdot b_1\\
&b_0 \to 0 \mid \neg b_1 \mid b_0 + b_0 \mid b_0 \cdot b \mid b_{\overline{0}} \cdot b_0\\
&a_0 \to p \mid a_0 + a_0 \mid a_0 + b_0 \mid b_0 + a_0 \mid a_0 \cdot a \mid a_{\overline{0}} \cdot a_0 \mid a_0 \cdot b \mid a_{\overline{0}} b_0 \mid b_0 \cdot a \mid b_{\overline{0}} \cdot a_0\\
&e_0 \to b_0 \mid a_0
\end{aligned}$$

The corresponding generating functions $B_{1,l}(z)$, $B_{0,l}(z)$, $A_{0,k,l}(z)$, and $E_{0,k,l}(z)$ satisfy the following equations,

$$\begin{aligned}
B_{1,l}(z) &= z + zB_{0,l}(z) + 2zB_l(z)B_{1,l}(z)\\
B_{0,l}(z) &= z + zB_{1,l}(z) + 2zB_l(z)B_{0,l}(z)\\
A_{0,k,l}(z) &= kz + 2zA_{k,l}(z)B_{0,l}(z) + 2zA_{0,k,l}(z)B_{k,l}(z) + 2zA_{0,k,l}(z)A_{k,l}(z)\\
E_{0,k,l}(z) &= B_{0,l}(z) + A_{0,k,l}(z)
\end{aligned}$$

from which we obtain $B_{1,l}(z) = B_{0,l}(z) = \frac{B_l(z)}{(l+2)}$, $A_{0,k,l}(z) = \frac{kz+2zB_{0,l}(z)A_{k,l}(z)}{1-2zE_{k,l}(z)}$. Finally the generating function for a lower bound of the number of expressions e such that $\mathsf{out}(e) = 0$ is

$$E_{0,k,l}(z) = \frac{k(l+2)z + (1-2zB_l(z))B_l(z)}{(l+2)(1-2zE_{k,l}(z))}.$$

Now, we can compute the generating functions of $\mathsf{first}(e)$ and $\mathsf{last}(e)$, $F_{k,l}(z)$ and $S_{k,l}(z)$, respectively, which coincide with the ones for regular expressions [16], except that they depend on the function $E_{0,k,l}(z)$,

$$F_{k,l}(z) = S_{k,l}(z) = \frac{kz}{1 - z - 4zE_{k,l}(z) + zE_{0,k,l}(z)}.$$

In Definition 1, $\mathsf{follow}(e)$ is defined using non-disjoint unions for the case of $e^\star$, and that does not allow an exact counting. Broda *et al.* [4] presented a new recursive definition without non-disjoint unions which yielded an exact generating function for the number of transitions of the Glushkov automaton (for regular expressions). Since $E_{0,k,l}(z)$ corresponds to lower bounds for the number of expressions e s.t. $\mathsf{out}(e) = 0$, here we use a slightly simplified version and obtain an upper bound for the size of $\mathsf{follow}(e)$. Our approximation $\mathsf{Fol}(e)$ of the $\mathsf{follow}(e)$ set is given by the recursive definition below, where there is no need to distinguish between a and e expressions. We have,

$$\begin{aligned} \mathsf{Fol}(p) &= \mathsf{Fol}(b) = \emptyset \\ \mathsf{Fol}(e+f) &= \mathsf{Fol}(e) \cup \mathsf{Fol}(f) \\ \mathsf{Fol}(e \cdot f) &= \mathsf{Fol}(e) \cup \mathsf{Fol}(f) \cup \mathsf{last}(e) \otimes \mathsf{first}(f) \\ \mathsf{Fol}(e^\star) &= \mathsf{Fol}^\star(e) \end{aligned} \tag{18}$$

$$\begin{aligned} \mathsf{Fol}^\star(b) &= \emptyset \\ \mathsf{Fol}^\star(p) &= \{(p,1,p)\} \\ \mathsf{Fol}^\star(e+f) &= \mathsf{Fol}^\star(e) \cup \mathsf{Fol}^\star(f) \cup \mathsf{Cross}(e,f) \\ \mathsf{Fol}^\star(e \cdot f) &= \mathsf{Fol}^\star(e) \ \cup\ \mathsf{Fol}^\star(f) \ \cup\ \mathsf{Cross}(e,f) \\ \mathsf{Fol}^\star(e^\star) &= \mathsf{Fol}^\star(e), \end{aligned} \tag{19}$$

with $\mathsf{Cross}(e,f) = \mathsf{last}(e) \otimes \mathsf{first}(f) \cup \mathsf{last}(f) \otimes \mathsf{first}(e)$. The corresponding generating functions satisfy the following equations,

$$Fol_{k,l}(z) = 4zFol_{k,l}(z)E_{k,l}(z) + zF_{k,l}(z)^2 + zFol^\star_{k,l}(z)$$

$$Fol^\star_{k,l}(z) = kz + 4zFol^\star_{k,l}(z)E_{k,l}(z) + 4zF_{k,l}(z)^2 + zFol^\star_{k,l}(z).$$

Solving these, one gets

$$Fol_{k,l}(z) = \frac{z\left(kz + F^2_{k,l}(z)\,(1+3z-4zE_{k,l}(z))\right)}{1 - z - 8zE_{k,l}(z) + 4z^2E_{k,l}(z) + (4zE_{k,l}(z))^2} \tag{20}$$

By the definition of $\mathsf{first}(e)$ it is straightforward to see that the size of this set is at most $|e|_{\mathsf{P}}$. Consequently, we can ignore the contribution of $F_{k,l}(z)$ in the computation of the upper bound for the number of transitions. Concerning $Fol_{k,l}(z)$ it is possible to see, with the help of an algebraic symbolic manipulator, that this function has the form

$$Fol_{k,l}(z) = \frac{U_{k,l}(z)}{V_{k,l}(z)\Delta_{k,l}(z)},$$

where $U_{k,l}(z)$, $V_{k,l}(z)$ are defined in a neighbourhood of 0 with radius larger than ρ_{k+l}. This shows that $z = \rho_{k+l}$ is the singularity of $Fol_{k,l}(z)$ closest to the origin, and there is no other in the circumference $|z| = \rho_{k+l}$. Using the same technique as exposed in [2, 3], one gets

$$[z^n]Fol_{k,l}(z) \sim \frac{c_{k,l}}{\sqrt{\pi(2-2\rho_{k+l})}}\,\rho_{k+l}^{-n}\,n^{-1/2}, \tag{21}$$

where $c_{k,l}$ is a constant that depends on k and l, through a rather complicated expression. It turns out that $\lim\limits_{k,l\to\infty} \frac{c_{k,l}}{\sqrt{k+l}} = \frac{17}{8}\sqrt{2} \simeq 3.182$.

Using now (13), one obtains

$$\frac{[z^n]Fol_{k,l}(z)}{[z^n]E_{k,l}(z)} \sim \frac{4c_{k,l}}{\sqrt{2-2\rho_{k+l}}\,\sqrt[4]{2k+2l+4}} \left(1+\frac{1}{n}\right)^{3/2} n. \qquad (22)$$

This means that the average number of transitions per automaton is approximately the size of the original KAT expression.

References

1. Almeida, R., Broda, S., Moreira, N.: Deciding KAT and Hoare logic with derivatives. In: Faella, M., Murano, A. (eds.) Proc. 3rd GANDALF. EPTCS, vol. 96, pp. 127–140 (2012)
2. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: On the average state complexity of partial derivative automata. International Journal of Foundations of Computer Science 22(7), 1593–1606 (2011)
3. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: An introduction to descriptional complexity of regular languages through analytic combinatorics. Tech. Rep. DCC-2012-05, DCC - FC, Universidade do Porto (07 2012)
4. Broda, S., Machiavelo, A., Moreira, N., Reis, R.: On the average size of Glushkov and partial derivative automata. International Journal of Foundations of Computer Science 23(5), 969–984 (2012)
5. Champarnaud, J.M., Ziadi, D.: From Mirkin's prebases to Antimirov's word partial derivatives. Fundam. Inform. 45(3), 195–205 (2001)
6. Champarnaud, J.M., Ziadi, D.: Canonical derivatives, partial derivatives and finite automaton constructions. Theoret. Comput. Sci. 289, 137–163 (2002)
7. Cohen, E., Kozen, D., Smith, F.: The complexity of Kleene algebra with tests. Tech. Rep. TR96-1598, Computer Science Department, Cornell University (07 1996)
8. Flajolet, P., Sedgewick, R.: Analytic Combinatorics. CUP (2008)
9. Glushkov, V.M.: The abstract theory of automata. Russian Math. Surveys 16, 1–53 (1961)
10. Kaplan, D.M.: Regular expressions and the equivalence of programs. J. Comput. Syst. Sci. 3(4), 361–386 (1969)
11. Kozen, D.: Kleene algebra with tests. Trans. on Prog. Lang. and Systems 19(3), 427–443 (05 1997)
12. Kozen, D.: Automata on guarded strings and applications. Matématica Contemporânea 24, 117–139 (2003)
13. Kozen, D.: On the coalgebraic theory of Kleene algebra with tests. Tech. Rep. http://hdl.handle.net/1813/10173, Cornell University (05 2008)
14. Kozen, D., Smith, F.: Kleene algebra with tests: Completeness and decidability. In: van Dalen, D., Bezem, M. (eds.) Proc. 10th CSL. LNCS, vol. 1258, pp. 244–259. Springer (1996)
15. Mirkin, B.G.: An algorithm for constructing a base in a language of regular expressions. Engineering Cybernetics 5, 51—57 (1966)
16. Nicaud, C.: On the average size of Glushkov's automata. In: Dediu, A.H., Ionescu, A.M., Martín-Vide, C. (eds.) Proc. 3rd LATA. LNCS, vol. 5457, pp. 626–637. Springer (2009)
17. Silva, A.: Kleene Coalgebra. Ph.D. thesis, Radboud Universiteit Nijmegen (2010)
18. Worthington, J.: Feasibly Reducing KAT Equations to KA Equations. ArXiv e-prints (01 2008)