Average Analysis of Glushkov Automata under a BST-Like Model

Cyril Nicaud[1], Carine Pivoteau[1], and Benoît Razet[2]

1 **LIGM, Univ. Paris-Est, CNRS UMR 8049, France**
{cyril.nicaud,carine.pivoteau}@univ-mlv.fr
2 **Tata Institute of Fundamental Research, Mumbai, India**
benoit.razet@gmail.com

Abstract

We study the average number of transitions in Glushkov automata built from random regular expressions. This statistic highly depends on the probabilistic distribution set on the expressions. A recent work shows that, under the uniform distribution, regular expressions lead to automata with a linear number of transitions. However, uniform regular expressions are not necessarily a satisfying model. Therefore, we rather focus on an other model, inspired from random binary search trees (BST), which is widely used, in particular for testing. We establish that, in this case, the average number of transitions becomes quadratic according to the size of the regular expression.

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2010.388

1 Introduction

Finite state automata are an essential data structure in computer science, which have been extensively studied since the fifties. Kleene, in his seminal paper [14], introduced regular expressions to describe the behavior of automata and showed a fundamental result: automata and regular expressions define the same objects, regular languages. Regular expressions are widely used in string searching programs and scripting languages such as *grep*, *sed*, *awk*, *Perl*, *Python* and *Ruby*. And most often, programs involving regular expressions are more efficient when the expressions are compiled into automata instead of interpreting them on the fly. The study of algorithms compiling (or one can say translating) regular expressions into automata is therefore a prolific area, which is still very active, in particular because of the large variety of automata and compilation techniques. A classical method to compare the resulting algorithms, besides time and space complexities, is to study the size of the output automaton, defined either as its number of states or of transitions.

In this article we study the average number of transitions of the automaton computed by a famous algorithm proposed by Glushkov [12] and independently by McNaughton and Yamada [16]. The automaton produced is now called Glushkov automaton or position automaton. The position automaton refers to the work of Berry and Sethi [3], who have provided a fast algorithm for compilation that associates to each *position* symbol of an expression, a state in the resulting automaton. This algorithm is also described in the standard textbook on compilers by Aho, Sethi and Ullman [1]. The worst-case complexity analysis on Berry-Sethi's algorithm shows that it produces an automaton with at most a quadratic number of transitions with respect to the size of the input regular expression (its number of symbols). But one may wonder what is the behavior of the algorithm in practice, which naturally leads to consider its average complexity.

IARCS Annual International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2010).
Editors: Kamal Lodaya, Meena Mahajan; pp. 388–399

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

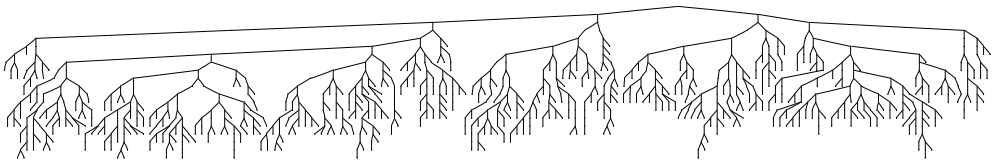


Figure 1 Random unary-binary tree (1000 nodes) according to the BST-like distribution.

A recent work [17] has shown that, considering the uniform distribution on regular expressions, the average number of transitions of Glushkov automaton is in fact linear, when the worst case is quadratic. Since a distribution which is uniform seems to be *a priori* natural, one may conclude that, in practice, the average number of transitions should be linear and not quadratic. Nevertheless, one can argue that this model may not be that relevant. For instance, the number of nested stars in a typical random expression is in $\Theta(\sqrt{n})$, which is much larger than expected. For testing purposes, an other natural distribution appears, inspired from random binary search trees (for short, we call it the *BST-like distribution*). In particular, this distribution has been used to generate random formulas of *Linear Temporal Logic* in order to validate algorithms in [6, 19]. To highlight the difference with the uniform model, Figures 1 and 2 display two random trees according to the two distributions: one shall be struck by their contrasting profiles, in particular looking at their height.

The main result of this paper is that the average number of transitions of Glushkov automaton is quadratic with respect to the size of the regular expression under a BST-like distribution. In this process, we also analyze in details the probability that a random regular expression recognizes the empty word.

This article is organized as follows. In Section 2, we define the BST-like distribution on regular expressions and recall some basic facts on Glushkov automata. The main theorem is given in Section 3. Intermediate results and their proofs are presented in Section 4. Finally, in the concluding section, we give experimental data to illustrate these results. Due to a lack of space, most of the proofs are sketched or omitted in this extended abstract.

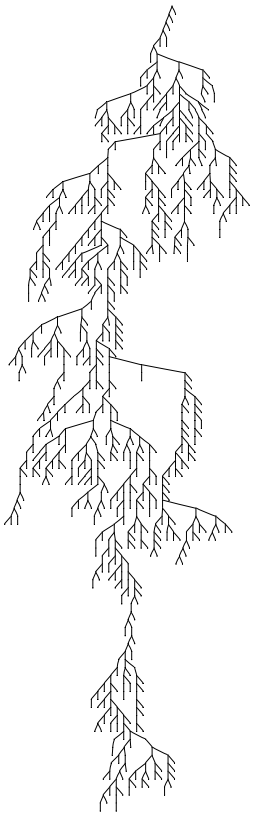


Figure 2 Uniform random unary-binary tree (1021 nodes).

2 Definitions

2.1 The BST-like distribution

We devote this section to the presentation of the trees corresponding to regular expressions, focusing on the fact that a BST-like distribution on such trees is not uniform. Recall that the uniform distribution on a finite set S is achieved by giving the same probability $1/|S|$ to all the elements of S.

2.1.0.1 Unary-binary plane trees

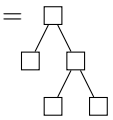
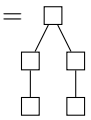
We first consider the classical model of unary-binary trees that are defined inductively as either single nodes (leaves) or nodes having exactly one child or two ordered children that are themselves unary-binary trees. The regular expressions considered in the sequel are a specialization of these trees. The number of nodes of a tree T is called its *size*, denoted by $|T|$. Following the recursive definition, one has a quite natural and simple algorithm `UB(`n`)` to produce a unary-binary tree of size n:

```
UB(n) ------------------------------------------------------------
     if n=1 then return a node (denoted by □)
     if n=2 then return □
                        |
                        □
     else, choose if the root is unary or binary
            if unary then return □
                                |
                                UB(n−1)
            else choose k between 1 and n − 2 and return □
                                                       /\
                                                      /  \
                                                UB(k)    UB(n−k−1)
-----------------------------------------------------------------
```

To transform this procedure into a random sampler, the (unspecified) choices are randomized in order to obey probabilistic laws that dictate the random distribution on the whole set of trees of size n. In this study, we consider the case where a unary node is chosen with probability $q \in]0,1[$ and a binary node with probability $1-q$. The size of the left child of a binary node is drawn uniformly at random. All the choices are independent, thus, in this model, the probability p of a tree is defined inductively by:

$$\begin{cases} p\left(\square\right) & = 1, \\ p\left(\begin{array}{c}\square\\|\\T\end{array}\right) & = q \cdot p(T), \\ p\left(\begin{array}{c}\square\\/\backslash\\T_1\ T_2\end{array}\right) & = \frac{1-q}{n-2} p(T_1)p(T_2), \quad \text{if } |T_1|+|T_2|+1 = n. \end{cases} \tag{1}$$

For any $n \geq 1$, p is a discrete probability measure on the set of unary-binary trees of size n, but one can notice that, for any value of q in $]0,1[$, the probability distribution induced by this definition is not uniform. This is readily checked, observing that the probabilities of the two following unary-binary trees of size 5 cannot match: the equation $p(T_1) = p(T_2)$ has no solution for q in $]0,1[$ when $p(T_1) = (1-q)^2/3$ and $p(T_2) = (1-q)/3$.

$T_1 =$  $T_2 =$ 

Actually, this probabilistic model is a natural extension of what is obtained for binary trees by choosing recursively the sizes of the two subtrees of a node uniformly at random; this corresponds exactly to the common random distribution on binary search trees (see [15]): to build a random BST of size n, nodes are inserted one at a time (using the standard insertion procedure for BST [5]), according to a uniform random permutation of $\{1,\ldots,n\}$. Therefore, from now on, we call this model the *BST-like distribution.*

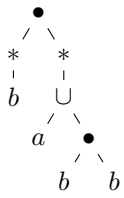


Figure 3 Unary-binary tree of size 9 representing the regular expression $b^* \bullet (a \cup b \bullet b)^*$.

2.1.0.2 Height of a random tree

To emphasize this dissimilarity, one can remark that the asymptotic profiles of the trees according to these two distributions highly differ. This is observable on different parameters, one of the most commonly studied being the height. According to [10], the average height of a large uniform unary-binary tree with n nodes is in $\Theta(\sqrt{n})$, when we show here that it is in $\Theta(\log n)$, according to our distribution. The later result was expectable, since this corresponds to the average height of binary search trees [18, 7, 8]. This explains the difference between the shapes of the two unary-binary trees of Figures 1 and 2, even though they are about the same size.

▶ **Proposition 1.** The average height of a unary-binary tree of size n according to the BST-like distribution is in $\Theta(\log n)$.

Proof. (Sketch) This proof is adapted from the second edition of Introduction to Algorithms [5, p. 265–268]. Let X_n be the random variable associated to the height of the unary-binary trees of size n and let $Y_n = \lambda^{X_n}$, with $\lambda = \frac{3}{2+q}$. Using the fact that the height of a tree that is not reduced to a single node is bounded from above by the sum of the heights of its children plus one, we get that for all positive integer n, $\mathbb{E}[Y_n] \leq y_n$, where $(y_n)_{n\in\mathbb{N}^*}$ is defined by: $y_1 = 1$, $y_2 = \lambda$, and for all $n \geq 3$,

$$y_n = \lambda q y_{n-1} + \frac{2\lambda(1-q)}{n-2} \sum_{\ell=1}^{n-2} y_\ell.$$

Since $\lambda = \frac{3}{2+q}$, one can prove by direct induction that $y_n \leq \binom{n+1}{2}$, for any positive integer n. We conclude by Jensen's inequality, since $\lambda^{\mathbb{E}[X_n]} \leq \mathbb{E}[\lambda^{X_n}] \leq y_n$. ◀

2.2 Random regular expressions

We consider non-empty regular expressions on an alphabet A, represented as unary-binary plane trees. The internal nodes are either the unary star operator $*$ or one of the two binary operators: union $\cup$ and concatenation $\bullet$. The leaves (external nodes) are either letters of A or the empty word ε (see example of Figure 3). Let $\mathcal{T}_n$ denote the set of all regular expressions with n nodes (both internal and external), and $\mathcal{T} = \cup_{n\in\mathbb{N}}\mathcal{T}_n$ be the set of all regular expressions. The size of an expression $T \in \mathcal{T}$ corresponds to the number of nodes of the tree, that is the number of symbols in the expression (excluding parentheses). A language defined on A is *denoted* by a regular expression when it is exactly the set of words obtained by interpreting each symbol $*$, $\bullet$ or $\cup$ as the associated regular operation on sets of words. Let $L(T)$ be the language denoted by $T \in \mathcal{T}$.

We extend the probabilistic model of the unary-binary trees defined in Section 2.1.0.1 to regular expressions as follows. Let $p_\varepsilon \in]0, 1[$ be the probability associated to ε and p

be a mapping from A to $]0,1[$ such that $\sum_{a\in A} p(a) = 1 - p_\varepsilon$. The mapping p is extended inductively to regular expressions by:

$$\begin{cases} p\left(\overset{*}{|}_{T}\right) &= p(T) & \text{if } |T| = 1, \\ p\left(\overset{*}{|}_{T}\right) &= q \cdot p(T) & \text{if } |T| \geq 2, \\ p\left(\overset{\cup}{/\!\backslash}_{T_1\ T_2}\right) &= \frac{1-q}{2(n-2)} p(T_1)p(T_2) & \text{if } |T_1| + |T_2| + 1 = n, \\ p\left(\overset{\bullet}{/\!\backslash}_{T_1\ T_2}\right) &= \frac{1-q}{2(n-2)} p(T_1)p(T_2) & \text{if } |T_1| + |T_2| + 1 = n, \end{cases} \tag{2}$$

where q in $]0,1[$ is the probability for an internal node to be the star operator. One can check by induction on $n \geq 1$ that p is a discrete probability measure on $\mathcal{T}_n$, i.e.,

$$\sum_{T\in\mathcal{T}_n} p(T) = 1. \tag{3}$$

Note that the $\cup$-nodes and the $\bullet$-nodes have the same probability to be generated in this distribution (mostly to keep the following computations trackable).

According to this model, the algorithm `UB(`n`)` producing unary-binary trees transforms into a random sampler `RE(`n`)` for regular expressions. This sampler has been used to generate the random tree displayed by Figure 1, forgetting the labels, with $q = 1/3$. As for the uniform tree of Figure 2, it has been produced by a Boltzmann sampler [9].

```
RE(n) ------------------------------------------------------------
    if n=1 then return ε with proba p_ε or a letter ℓ with proba p(ℓ)
    if n=2 then return (RE(1))*
    else, choose "unary" with proba q or "binary" with proba 1 − q
           if "unary" then return (RE(n − 1))*
           else choose k uniformly at random between 1 and n − 2
                  return RE(k) ∪ RE(n − k − 1) with proba 1/2
                  or return RE(k) • RE(n − k − 1) with proba 1/2
-------------------------------------------------------------------
```

Note that to choose a random element in a set $S = \{s_1, \ldots, s_n\}$, each with probability $p(s_i)$, one simply needs to pick a random value in the interval $I = [0,1[$ and return the element corresponding to the subinterval of I where it belongs, when I is divided according to p:

$$I = [0, p(s_1)[\cup [p(s_1), p(s_1) + p(s_2)[\cup \cdots \cup [1 - p(s_n), 1[.$$

2.3 Glushkov automaton

We give here the formal construction to compute the Glushkov automaton [12, 16, 3] of any regular expression and introduce the notations used in the sequel.

Let m be the number of letter symbols in T, for $T \in \mathcal{T}$. We consider the expression $\widetilde{T}$ obtained from T by distinguishing the letters with subscripts in $\{1, \ldots, m\}$, marking them from left to right on its string representation, or equivalently using depth-first order on its tree representation. For instance $T_0 = b^* \bullet (a \cup b \bullet b)^*$ is changed into $\widetilde{T_0} = b_1^* \bullet (a_2 \cup b_3 \bullet b_4)^*$. We denote by $\mathtt{Pos}(T)$ the set of subscripted letters in $\widetilde{T}$: $\mathtt{Pos}(T_0) = \{b_1, a_2, b_3, b_4\}$ in the example. We also denote by ν the function from $\mathtt{Pos}(T)$ to A that removes the subscripts; for instance, $\nu(a_2) = a$.

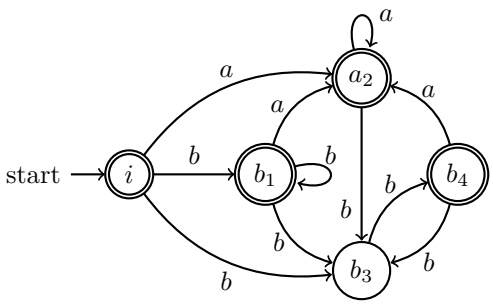


Figure 4 Glushkov automaton for the expression $\widetilde{T_0} = b_1^* \bullet (a_2 \cup b_3 \bullet b_4)^*$.

The automaton construction we study relies on the relative positions of letters in the words recognized by this automaton; thus we introduce the three following sets of distinguished letters that are used to describe these positions. For any regular expression T, let $\texttt{First}(T)$ and $\texttt{Last}(T)$ be the sets defined by

$$\texttt{First}(T) = \{\alpha \in \texttt{Pos}(T) \mid \exists u \in L(\widetilde{T}),\ u \text{ starts with the letter } \alpha\},$$
$$\text{and} \quad \texttt{Last}(T) = \{\alpha \in \texttt{Pos}(T) \mid \exists u \in L(\widetilde{T}),\ u \text{ ends with the letter } \alpha\}.$$

For instance, $\texttt{First}(T_0) = \{b_1, a_2, b_3\}$ and $\texttt{Last}(T_0) = \{b_1, a_2, b_4\}$. And for any letter α in $\texttt{Pos}(T)$, the set $\texttt{Follow}(T, \alpha)$ is defined by

$$\texttt{Follow}(T, \alpha) = \{\beta \in \texttt{Pos}(T) \mid \exists u \in L(\widetilde{T}),\ \alpha \text{ and } \beta \text{ are consecutive letters in } u\}.$$

The *Glushkov automaton* of T, also called the *position automaton*, is the automaton $\mathcal{A}_T$ defined by $\mathcal{A}_T = (A, Q, R, \{i\}, F)$ with $Q = \texttt{Pos}(T) \cup \{i\}$, $F = \texttt{Last}(T) \cup \{i\}$ if $\varepsilon \in L(T)$ and $F = \texttt{Last}(T)$ otherwise, and

$$R = \{i \xrightarrow{\nu(\alpha)} \alpha \mid \alpha \in \texttt{First}(T)\} \cup \{\alpha \xrightarrow{\nu(\beta)} \beta \mid \beta \in \texttt{Follow}(T, \alpha)\}.$$

This classical construction provides an automaton that recognizes $L(T)$. As an example, the Glushkov automaton of T_0 is depicted by Figure 4.

3 Main result

▶ **Theorem 2.** *In the BST-like model, the average number of transitions in the Glushkov automaton of a size n regular expression is quadratic, i.e., in $\Theta(n^2)$.*

Proof. First, assume that the expected size f_n of `First` (its cardinality) is linear with respect to the size n of the regular expression, i.e., that f_n satisfies the asymptotic equivalent $f_n \sim Kn$, for some positive real K that only depends on p_ε and q. The proof of this result (Theorem 7), which is technical, is given in the next section.

Recall that Markov inequality states that if X is a non-negative random variable with expectation $\mathbb{E}[X]$, then for any positive real number a,

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}[X]}{a}.$$

For $n \geq 1$, let $X_n : \mathcal{T}_n \to \mathbb{R}$ be the random variable that associates $n - |\texttt{First}(T)|$ to any $T \in \mathcal{T}_n$. This random variable is non-negative, since $|\texttt{First}(T)|$ is at most n for any

element of $\mathcal{T}_n$. Therefore, setting $a = \alpha n$ in Markov inequality, with $1 - K < \alpha < 1$, we obtain that

$$\mathbb{P}\left(X_n \geq \alpha n\right) \leq \frac{\mathbb{E}[X_n]}{\alpha n},$$

and thus

$$\mathbb{P}\left(|\texttt{First}\left(T\right)| \leq (1-\alpha)n\right) \leq \frac{n - f_n}{\alpha n}.$$

The right quantity is asymptotically equivalent to $\frac{1-K}{\alpha} < 1$, then there exists two real numbers β and γ in $]0,1[$, with $\beta < (1-\alpha)$ and $0 < \gamma < 1 - \frac{1-K}{\alpha}$, such that for n large enough,

$$\mathbb{P}\left(|\texttt{First}\left(T\right)| \geq \beta n\right) \geq \gamma.$$

By symmetry, this result also holds for $\texttt{Last}\left(T\right)$. Moreover, the probability that a regular expression T of size $n+2$ satisfies the following conditions:

$$T = \overset{\bullet}{\underset{T_1}{\wedge}\underset{T_2}{\vee}}, \qquad |T_1| \in \left[\lfloor \tfrac{n}{3} \rfloor, \lceil \tfrac{2n}{3} \rceil\right], \qquad |\texttt{Last}\left(T_1\right)| \geq \beta |T_1| \quad \text{and} \quad |\texttt{First}\left(T_2\right)| \geq \beta |T_2|,$$

is at least, for n large enough,

$$\underbrace{\frac{1-q}{2}}_{\text{root}=\bullet}\ \underbrace{\frac{1}{3}}_{|T_1|}\ \underbrace{\gamma}_{|\texttt{Last}(T_1)|}\ \underbrace{\gamma}_{|\texttt{First}(T_2)|} = \frac{(1-q)\gamma^2}{6} > 0.$$

Note that for any a in $\texttt{Last}\left(T_1\right)$ and any b in $\texttt{First}\left(T_2\right)$, the transition $a \xrightarrow{\nu(b)} b$ is in the automaton, since the letter b is in $\texttt{Follow}\left(T_1 \bullet T_2, a\right)$. Therefore, any tree satisfying the above conditions yields to an automaton with at least $|\texttt{Last}\left(T_1\right)| \cdot |\texttt{First}\left(T_2\right)| \geq \beta^2 n^2$ transitions. Therefore, the expected number of transitions is bounded below by

$$\frac{(1-q)\gamma^2}{6}\beta^2 n^2 = \Omega\left((n+2)^2\right),$$

in the Glushkov automaton of a size $n+2$ expression. The $\mathcal{O}(n^2)$ bound being obvious, the result follows. ◀

The next section is devoted to the exposition of some intermediate results to complete this proof. Among them, the key point is given by Theorem 7, which states that the average size of `First` (resp. `Last`) is linear with respect to the size of the regular expression; considering only the sub-expressions of the form $T_1 \bullet T_2$, one can observe that the number of new transitions they imply in the automata is $|\texttt{Last}\left(T_1\right)| \cdot |\texttt{First}\left(T_2\right)|$, which justify the quadratic number of transitions in the whole automata. The other point is that the size of `First` (resp. `Last`) is highly related to the probability of recognizing the empty word, given by Theorem 3 which states that a large expression recognizes ε with high probability.

4 Some properties of random expressions in the BST-like model

4.1 Analytic tools

In the sequel, the proofs mostly rely on techniques of analytic combinatorics. To study the asymptotic behavior of a sequence $(a_n)_{n\in\mathbb{N}}$, the idea is to consider its *generating function* $A(z)$, which is the formal power series defined by

$$A(z) = \sum_{n\in\mathbb{N}} a_n z^n.$$

From a recursive specification of $(a_n)_{n\in\mathbb{N}}$, one can often get a functional equation satisfied by $A(z)$. At this point, several theorems exist to compute asymptotic estimates of its Taylor coefficients, which are exactly the a_n's. These theorems mainly use the theory of complex analysis, seeing generating functions as analytic functions from $\mathbb{C}$ to $\mathbb{C}$. The main idea is that asymptotic equivalents for the coefficients of a generating function can be obtained by studying it around its dominant singularities (its singularities of smallest moduli).

In this article, the recursive descriptions of sequences lead to ordinary differential equations for their generating functions. These equations can be solved using the well-known variation-of-constants method, and the solutions have similar properties: they have a unique dominant singularity at 1 and satisfy the required analytic conditions. Therefore, provided the expansion of $A(z)$ near 1 is of the form

$$A(z) = C(1-z)^\alpha + O\left((1-z)^\beta\right) \qquad \text{with} \qquad \alpha \in \mathbb{R}\setminus\mathbb{N}, \quad \alpha < \beta \quad \text{and} \quad C \neq 0,$$

Transfer Theorem [11] gives that $a_n \sim \frac{C}{\Gamma(-\alpha)} n^{-\alpha-1}$, where Γ is Euler's Gamma function, the analytic continuation of $s \mapsto \int_0^\infty t^{s-1}e^{-t}dt$.

For more information on analytic combinatorics techniques, the reader is referred to the comprehensive book by Flajolet and Sedgewick [11].

4.2 Recognizing the empty word

This section is devoted to the proof of Theorem 3 which gives the probability that a regular expression of a given size recognizes the empty word.

Let r_n denote the probability that a size n regular expression does not recognize ε, with the convention $r_0 = 0$:

$$r_n = \sum_{\substack{T\in\mathcal{T}_n \\ \varepsilon\notin L(T)}} p(T).$$

▶ **Theorem 3.** *A large random regular expression recognizes the empty word with high probability. More precisely, in the BST-like model, the probability that a size n regular expression does not recognize ε is asymptotically equivalent to*

$$r_n \sim \frac{C}{n^q} \qquad \textit{with} \qquad C = \frac{(1-p_\varepsilon)}{e^{1-q}\Gamma(1-q)}\left(1 - \int_0^1 \frac{e^{(1-q)t}(1-t)^{1-q}-1}{t^2}dt\right).$$

Using basic computations, one can establish the following lemma from Equation (2):

▶ **Lemma 4.** *The sequence $(r_n)_{n\in\mathbb{N}}$ satisfies $r_1 = 1-p_\varepsilon$, $r_2 = 0$ and for any $n \geq 1$,*

$$r_{n+2} = \frac{1-q}{n}\sum_{\ell=1}^{n} r_\ell.$$

Let $R(z) = \sum_{n\in\mathbb{N}} r_n z^n$, with $r_0 = 0$, denote the generating function associated to the sequence $(r_n)_{n\in\mathbb{N}}$. For all $n \in \mathbb{N}$, since it is as a probability, r_n is in $[0,1]$; then $R(z)$ is analytic at 0 and its radius of convergence is at least 1.

▶ **Lemma 5.** *The generating function $R(z)$ satisfies the following differential equation*

$$z\frac{d}{dz}R(z) - \frac{(1-q)z^2 - 2z + 2}{1-z}R(z) + (1-p_\varepsilon)z = 0.$$

Proof. (Sketch) Multiply the general formula of Lemma 4 by nz^{n+2} and sum for $n \geq 1$. Then identify the expressions of the power series $R(z)$ and $\frac{d}{dz}R(z)$. ◀

▶ **Proposition 6.** Let g be the function defined by

$$g(z) = \frac{e^{(1-q)z}(1-z)^{1-q} - 1}{z^2}.$$

The function $g(z)$ has a false pole at zero, that can be removed, and one has

$$R(z) = (1-p_\varepsilon)\left(1 - z\int_0^z g(t)dt\right) ze^{(q-1)z}(1-z)^{q-1}.$$

Proof. The formula is obtained by the variation-of-constants method. Once stated, one can also directly verify that it satisfies the differential equation of Lemma 5 with the same initial conditions as $R(z)$. ◀

of Theorem 3. The proof is an application of analytic combinatorics techniques, and more precisely of singularity analysis of generating functions (see [11, Ch. VI]).

The function $g(z) = z^{-2}(e^{(1-q)z}(1-z)^{1-q} - 1)$ has its unique dominant singularity at 1 where we have:

$$g(z) = -1 + e^{1-q}(1-z)^{1-q} + O((1-z)).$$

Hence, by Singular Integration Theorem [11, p. 420], the antiderivative of g satisfies near 1 the following development:

$$\begin{aligned}\int_0^z g(t)dt &= -1 + O(1-z) - \frac{e^{1-q}}{2-q}(1-z)^{2-q} + \int_0^1 (g(t)+1)dt + O((1-z)^2)\\ &= \int_0^1 g(t)dt + O(1-z).\end{aligned}$$

Hence, $R(z)$ has its unique dominant singularity at 1 too, and near 1 one has

$$R(z) = (1-p_\varepsilon)e^{q-1}\left(1 - \int_0^1 g(t)\right)(1-z)^{q-1} + O((1-z)^q).$$

Using Transfer Theorem [11, p. 393], we obtain

$$r_n \sim \frac{(1-p_\varepsilon)e^{q-1}\left(1-\int_0^1 g(t)\right)}{\Gamma(1-q)} n^{-q},$$

concluding the proof.

◀

4.3 The average size of `First` is linear

In this section, we establish the following theorem. Some of the proofs are omitted, since they are similar to those of Theorem 3.

▶ **Theorem 7.** *The average size of* `First` *for a size n regular expression, according to the BST-like model, is asymptotically equivalent to $K\,n$, for some real constant $K \in]0,1[$.*

Let f_n be the average cardinality of `First` for regular expressions of size n:

$$f_n = \sum_{T\in\mathcal{T}_n} |\,\texttt{First}\,(T)\,| \cdot p(T).$$

Note that, by symmetry, f_n is also the average size of `Last` for regular expressions in $\mathcal{T}_n$.

▶ **Lemma 8.** *The sequence* $(f_n)_{n\in\mathbb{N}}$ *satisfies* $f_1 = f_2 = 1 - p_\varepsilon$ *and for any* $n \geq 1$,

$$f_{n+2} = qf_{n+1} + \frac{2(1-q)}{n}\sum_{\ell=1}^{n} f_\ell - \frac{1-q}{2n}\sum_{\ell=1}^{n} r_\ell f_{n+1-\ell}.$$

Let $F(z) = \sum_{n\in\mathbb{N}} f_n z^n$, with $f_0 = 0$, be the generating function associated to the sequence $(f_n)_{n\in\mathbb{N}}$.

▶ **Lemma 9.** *The generating function* $F(z)$ *satisfies the following differential equation*

$$z(1-qz)\frac{d}{dz}F(z) - \left(2 - qz + \frac{2(1-q)z^2}{1-z} - \frac{1-q}{2}zR(z)\right)F(z) + (1-p_\varepsilon)z = 0.$$

Solving this equation, we obtain the following proposition.

▶ **Proposition 10.** Let h be the function defined by

$$h(z) = \frac{(1-z)^2}{z^2(1-qz)^{2/q}} - \frac{1}{z^2}.$$

The function $h(z)$ has a false pole at zero, which can be removed, and one has

$$F(z) = \frac{z(1-qz)^{2/q-1}}{(1-z)^2}(1-p_\varepsilon)\exp\left(-\frac{1-q}{2}\int_0^z \frac{R(t)}{1-qt}dt\right)\left(1 + (1-q)z - z\int_0^z h(t)dt\right).$$

The proof of Theorem 7 is an analysis of $F(z)$ near its unique dominant singularity 1, using our result on $R(z)$. We obtain that

$$F(z) = \frac{K}{(1-z)^2}\left(1 + O\left((1-z)^q\right)\right) \quad \text{and} \quad f_n \sim K\, n,$$

with

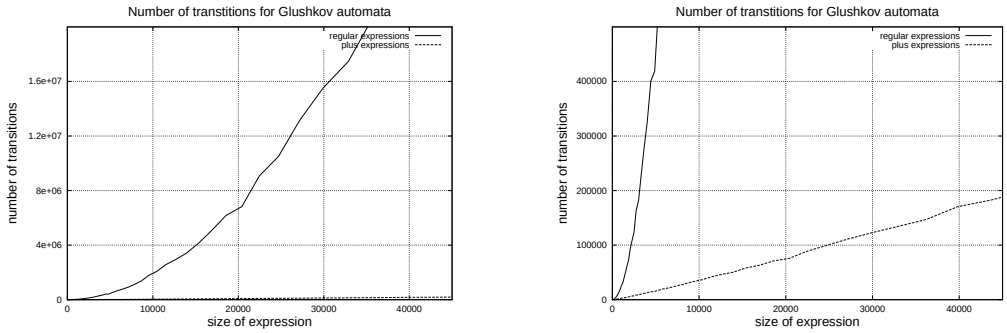
$$K = (1-p_\varepsilon)(1-q)^{2/q-1}\exp\left(-\frac{1-q}{2}\int_0^1 \frac{R(t)}{1-qt}dt\right)\left(2 - q - \int_0^1 h(t)dt\right).$$

5 Conclusion and perspectives

In this article, we analyzed the average size of Glushkov automata associated to random regular expressions, in the BST-like model. Using analytic combinatorics techniques, we proved that, unlike in the uniform case, the average number of transitions in an automaton is quadratic with respect to the size of the expression.

We implemented the procedure `RE(n)` given in Section 2.2 in order to confirm empirically our theoretical results. One of these experiments is displayed by Figure 5 (plain line). The x-axis represents the size of the regular expressions and the y-axis represents the number of transitions in the corresponding Glushkov automata. The dotted line corresponds to an other bench of experiments, involving a different kind of regular expressions, where the Kleene Star operator $*$ (reflexive and transitive closure of the concatenation) has been replaced by a $+$ operator (only transitive closure of the concatenation). Considering the classical regular expressions, the quadratic behavior clearly appears on Figure 5, whereas it seems to be linear for expressions using only the $+$ operator.

One can reasonably expect to prove the linear behavior observed in Figure 5b, using the techniques of the present paper combined with those of [17]. This seems to be confirmed by the calculations we have already performed. A natural extension of this work is therefore to complete this proof. In a different direction, the average analysis of other constructions related to Glushkov automata, could be considered. Among them are the Follow automaton by Ilie and Yu [13] and Antimirov automaton [2], which are both quotients of Glushkov automaton (see [4]).



(a) **(b)**

Figure 5 Number of transitions of Glushkov automata with respect to the size of expressions defined on the alphabet $A = \{a, b\}$, with parameters $q = \frac{1}{3}$, $p_\varepsilon = \frac{1}{100}$ and $p(a) = p(b)$. Note that (a) and (b) display the same data, but with different scales.

References

1 Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: principles, techniques, and tools*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1986.

2 Valentin Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *Theoretical Computer Science*, 155(2):291–319, 1996.

3 Gérard Berry and Ravi Sethi. From regular expressions to deterministic automata. *Theoretical Computer Science*, 48(1):117–126, 1986.

4 Jean-Marc Champarnaud and Djelloul Ziadi. Canonical derivatives, partial derivatives and finite automaton constructions. *Theoretical Computer Science*, 289(1):137 – 163, 2002.

5 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, second edition, 2001.

6 Marco Daniele, Fausto Giunchiglia, and Moshe Y. Vardi. Improved automata generation for Linear Temporal Logic. In Nicolas Halbwachs and Doron Peled, editors, *CAV*, volume 1633 of *Lecture Notes in Computer Science*, pages 249–260. Springer, 1999.

7 Luc Devroye. A note on the height of binary search trees. *Journal of the ACM*, 33(3):489–498, 1986.

8 Michael Drmota. An analytic approach to the height of binary search trees. *Journal of the ACM*, 50:89–119, 2001.

9 Philippe Duchon, Philippe Flajolet, Guy Louchard, and Gilles Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability and Computing*, 13(4–5):577–625, 2004.

10 Philippe Flajolet and Andrew Odlyzko. The average height of binary trees and other simple trees. *The Journal of Computer and System Sciences*, 25(2):171–213, 1982.

11 Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

12 Victor Glushkov. The abstract theory of automata. *Russian Mathematical Surveys*, 16:1–53, 1961.

13 Lucian Ilie and Sheng Yu. Follow automata. *Information and Computation*, 186(1):140–162, 2003.

14 Stephen Cole Kleene. Representation of events in nerve nets and finite automata. *Automata Studies, Annals of Mathematics Studies*, 36, 1956.

15 Conrado Martínez. *Statistics Under the BST Model*. PhD thesis, UPC, Spain, 1992.

16 Robert McNaughton and Hisao Yamada. Regular expressions and state graphs for automata. *IRE Transactions on Electronic Computers*, 9:39–47, 1960.

17 Cyril Nicaud. On the average size of Glushkov's automata. In Adrian Horia Dediu, Armand-Mihai Ionescu, and Carlos Martín-Vide, editors, *LATA*, volume 5457 of *Lecture Notes in Computer Science*, pages 626–637. Springer, 2009.

18 John M. Robson. The height of binary search trees. *Australian Computer Journal*, 11(4):151–153, 1979.

19 Heikki Tauriainen and Keijo Heljanko. Testing SPIN's LTL formula conversion into Büchi automata with randomly generated input. In Klaus Havelund, John Penix, and Willem Visser, editors, *SPIN*, volume 1885 of *Lecture Notes in Computer Science*, pages 54–72. Springer, 2000.