Theoretical Computer Science 610 (2016) 78–90

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Prefix-free languages: Left and right quotient and reversal [☆]

Jozef Jirásek [a,1], Galina Jirásková [b,*,2], Monika Krausová [a], Peter Mlynárčik [b,2], Juraj Šebej [a,1]

[a] *Institute of Computer Science, P.J. Šafárik University, Jesenná 5, 040 01 Košice, Slovakia*
[b] *Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia*

ARTICLE INFO

Article history:
Received 2 November 2014
Received in revised form 10 August 2015
Accepted 24 August 2015
Available online 9 September 2015

Keywords:
Finite automata
Prefix-free languages
Left and right quotient
Reversal
State complexity

ABSTRACT

We investigate the left and right quotient, and the reversal operation on the class of prefix-free regular languages. We get the tight upper bounds 2^{n-1}, $n-1$, and $2^{n-2}+1$ on the state complexity of these three operations, respectively. To prove the tightness of these bounds, we use an $(n-1)$-letter alphabet for left quotient, a binary alphabet for right quotient, and a ternary alphabet for reversal. We also prove that these bounds cannot be met using languages defined over any smaller alphabet. For left quotient, we prove that the tight bound for an $(n-2)$-letter alphabet is $2^{n-1}-1$, and we provide exponential lower bounds for every smaller alphabet, except for the unary case. For the reversal operation on binary prefix-free languages, we get $2^{n-2}-7$ lower bound in the case of $n \bmod 3 \neq 2$, and $2^{n-2}-15$ lower bound in the remaining cases. We conjecture that our lower bounds on the state complexity of reversal on binary prefix-free languages are tight if $n \geq 12$. Our experimental results support this conjecture.

1. Introduction

A language is prefix-free if it does not contain two distinct strings one of which is a prefix of the other. Prefix-free languages are used in coding theory. In prefix codes, like variable-length Huffman codes or country calling codes, there is no codeword that is a proper prefix of any other codeword. With such a code, a receiver can identify each codeword without any special marker between words.

Motivated by prefix codes, the class of prefix-free regular languages has been recently investigated. It is known that every minimal deterministic automaton recognizing a prefix-free regular language must have exactly one final state, from which all transitions go to a dead state. Using this property, tight upper bounds on the state complexity of basic operations such as union, intersection, concatenation, star, and reversal have been obtained in [7] and strengthened in [11,13]; recall that the state complexity of an operation on regular languages is the maximal state complexity of the language resulting from the operation as a function of the state complexities of the arguments [2,5,6].

☆ This work was presented at the DCFS 2014 workshop held in Turku, Finland on August 5–8, 2014, and its extended abstract appeared in the workshop proceedings: H. Jürgensen, J. Karhumäki, and A. Okhotin (Eds.), *Descriptional Complexity of Formal Systems*, LNCS 8614, pp. 210—221.

* Corresponding author.
E-mail addresses: jozef.jirasek@upjs.sk (J. Jirásek), jiraskov@saske.sk (G. Jirásková), mon.krausova@gmail.com (M. Krausová), mlynarcik1972@gmail.com (P. Mlynárčik), juraj.sebej@gmail.com (J. Šebej).

1 Research supported by grant VEGA 1/0142/15.
2 Research supported by grant VEGA 2/0084/15.

http://dx.doi.org/10.1016/j.tcs.2015.08.031

The nondeterministic state complexity of basic regular operations has been investigated in [8,11], while the complexity of combined operations on prefix-free regular languages has been studied in [9].

In [10] it has been shown that the tight bound on the state complexity of cyclic shift on prefix-free languages is given by the function $(2n-3)^{n-2}$. To prove the tightness of this bound, the authors used a quaternary alphabet, and they proved that this bound cannot be met by any ternary language. On the other hand, they showed that the lower bounds in the binary and ternary cases are still exponential.

In this paper, we investigate the left and right quotient, and the reversal operations on the class of prefix-free languages. In the case of the left quotient operation, defined to be $K\backslash L = \{x \mid wx \in L \text{ and } w \in K\}$, we get an upper bound 2^{n-1} on its state complexity, and we prove that it is tight for an alphabet with at least $n-1$ symbols. We also show that this bound cannot be met using any smaller alphabet. Then we prove that the tight upper bound in the case of an $(n-2)$-letter alphabet is smaller just by one. Finally, we provide exponential lower bounds for every smaller alphabet, except for the unary case, where the tight upper bound is 1 if $n < m$, and it is $n - m + 2$ otherwise.

Then we study the right quotient operation, defined to be $L/K = \{x \mid xw \in L, w \in K\}$. We get an upper bound $n-1$ on its state complexity on prefix-free languages, and we prove that it is tight for an alphabet with at least two symbols. Recall that in the general case of regular languages, the tight bound is n [22].

Finally, we examine the reversal operation defined to be $L^R = \{w^R \mid w \in L\}$, where w^R stands for the string w written backwards. The operation preserves regularity as shown already by Rabin and Scott in 1959 [17]: A nondeterministic finite automaton for the reverse of a regular language can be obtained from an automaton recognizing the given language by swapping the role of initial and final states, and by reversing all the transitions. This gives the upper bound 2^n on the state complexity of reversal on regular languages. Its tightness in the ternary case has been pointed out already by Mirkin [16], who noticed that a ternary Lupanov's witness automaton for determinization [15] is a reverse of a deterministic automaton. The binary witness languages meeting the upper bound 2^n have been presented in [12,14,18,19].

In the case of prefix-free languages, the upper bound on the state complexity of reversal is $2^{n-2}+1$ [7], and in the first part of Section 5 we present a simple proof of its tightness in the ternary case. Then we show that this upper bound cannot be met by any binary language. In the case of binary prefix-free languages, we get the lower bound $2^{n-2}-7$ whenever $n \bmod 3 \neq 2$, and the lower bound $2^{n-2}-15$ otherwise. Thus our lower bounds on the state complexity of reversal on binary prefix-free languages are smaller just by a constant factor than the upper bound 2^{n-2}.

We also did some computations concerning the state complexity of the reversal operation on binary prefix-free languages. While for some small values of n our lower bounds can be exceeded, starting with $n = 12$, we were not able to find any binary prefix-free language exceeding our lower bound $2^{n-2}-7$ in the case of $n \bmod 3 \neq 2$, and our lower bound $2^{n-2}-15$ in the remaining cases. We strongly conjecture that these lower bounds are tight if $n \geq 12$.

2. Preliminaries

In this section, we recall some basic definitions and preliminary results. For further details and all unexplained notions, the reader may refer to [20,21].

Let Σ be a finite alphabet and Σ^* the set of all strings over the alphabet Σ including the empty string ε. A language is any subset of Σ^*. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (Q, \Sigma, \delta, I, F)$, where Q is a finite set of states, Σ is a finite alphabet, $\delta\colon Q \times \Sigma \to 2^Q$ is the transition function which is extended to the domain $2^Q \times \Sigma^*$ in the natural way, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final states. The *language accepted by* A is the set $L(A) = \{w \in \Sigma^* \mid \delta(I, w) \cap F \neq \emptyset\}$.

An NFA A is *deterministic* (DFA) (and complete) if $|I| = 1$ and $|\delta(q, a)| = 1$ for each q in Q and each a in Σ. In such a case, we write $q \cdot a = q'$ instead of $\delta(q, a) = \{q'\}$. A non-final state q is a *dead state* if $q \cdot a = q$ for each a in Σ.

The *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the number of states in the minimal DFA for L. It is well known that a DFA is minimal if all its states are reachable from its initial state, and no two of its states are equivalent.

The *state complexity of an operation* on regular languages is the maximal state complexity of the language resulting from the operation as a function of the state complexities of the arguments [2,5,6].

Formally, if f is a k-ary operation on regular languages over an alphabet Σ preserving regularity, then the state complexity of the operation f is given by a function sc_f from $\mathbb{N}^k$ to $\mathbb{N}$ defined as $\mathrm{sc}_f(n_1, n_2, \ldots, n_k) = \max\{\mathrm{sc}(f(L_1, L_2, \ldots, L_k)) \mid L_i \subseteq \Sigma^* \text{ and } \mathrm{sc}(L_i) = n_i \text{ for } i = 1, 2, \ldots, k\}$.

Every nondeterministic automaton $A = (Q, \Sigma, \delta, I, F)$ can be converted to an equivalent DFA $A' = (2^Q, \Sigma, \cdot, I, F')$, where $F' = \{R \in 2^Q \mid R \cap F \neq \emptyset\}$ and $R \cdot a = \delta(R, a)$ for each R in 2^Q and each a in Σ [17]. The DFA A' is called the *subset automaton* of the NFA A. The subset automaton need not be minimal since some of its states may be unreachable or equivalent. The following lemma shows that in some cases, we can guarantee the distinguishability of the states in a subset automaton.

Lemma 1. *For every state q of an NFA N, let there be a string w_q such that w_q is accepted by N from the state q, but it is rejected from any other state. Then all the states in the subset automaton of the NFA N are pairwise distinguishable.*

Proof. Two distinct subsets of the subset automaton of the NFA N differ in a state q, and the string w_q distinguishes the two subsets. $\square$