Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On Boolean combinations forming piecewise testable languages

Tomáš Masopust [a,b,*,1], Michaël Thomazo [c,2]

[a] Institute of Theoretical Computer Science and Center of Advancing Electronics Dresden (cfaed), TU Dresden, Germany
[b] Institute of Mathematics, Czech Academy of Sciences, Czechia
[c] Inria, France

ARTICLE INFO

Article history:
Received 28 July 2016
Received in revised form 1 December 2016
Accepted 13 January 2017
Available online xxxx

Keywords:
Automata
Languages
k-piecewise testability
Complexity

ABSTRACT

A regular language is k-piecewise testable (k-PT) if it is a Boolean combination of languages of the form $L_{a_1a_2\dots a_n} = \Sigma^* a_1 \Sigma^* a_2 \Sigma^* \cdots \Sigma^* a_n \Sigma^*$, where $a_i \in \Sigma$ and $0 \leq n \leq k$. Given a finite automaton $\mathscr{A}$, if the language $L(\mathscr{A})$ is piecewise testable, we want to express it as a Boolean combination of languages of the above form. The idea is as follows. If the language is k-PT, then there exists a congruence $\sim_k$ of finite index such that $L(\mathscr{A})$ is a finite union of $\sim_k$-classes. Every such class is characterized by an intersection of languages of the from L_u, for $|u| \leq k$, and their complements. To represent the $\sim_k$-classes, we make use of the $\sim_k$-canonical DFA. We identify the states of the $\sim_k$-canonical DFA whose union forms the language $L(\mathscr{A})$ and use them to construct the required Boolean combination. We study the computational and descriptional complexity of related problems.

1. Introduction

A regular language L over an alphabet Σ is *piecewise testable* (PT) if it is a finite Boolean combination of languages of the form $L_{a_1a_2\dots a_n} = \Sigma^* a_1 \Sigma^* a_2 \Sigma^* \cdots \Sigma^* a_n \Sigma^*$, where $a_i \in \Sigma$ and $n \geq 0$. If the language is piecewise testable, then it is a finite Boolean combination of languages of the form L_u, where the length of $u \in \Sigma^*$ is at most k. In this case, the language is called *k-piecewise testable* (k-PT).

In this paper, we study the problem of translating an automaton representing a piecewise testable language into a Boolean combination of languages of the form L_u. The motivation comes from the simplification of XML Schema, since such expressions resemble XPath-like expressions used in the BonXai schema language. The reader is referred to Martens et al. [21] for more details. Since every piecewise testable language is k-PT for some $k \geq 0$, and a k-PT language is also $(k+1)$-PT, we focus on the Boolean combination of languages L_u, where the length of u is bounded by the minimal k for which the language is k-PT. From this point of view, we are interested in translating an automaton to the form of a generalized regular expression (a regular expression allowing the operation of complement). Generalized regular expressions can be non-elementary more succinct than classical regular expressions [6,29,9] and not much is known about these transformations [7]. There are many different Boolean combinations describing the same language, and it is not clear which of them is

* Corresponding author at: Institute of Theoretical Computer Science and Center of Advancing Electronics Dresden (cfaed), TU Dresden, Germany.
E-mail addresses: tomas.masopust@tu-dresden.de (T. Masopust), michael.thomazo@inria.fr (M. Thomazo).

[1] Research supported by the German Research Foundation (DFG) in Emmy Noether grant KR 4381/1-1 (DIAMOND).
[2] Research supported by the Alexander von Humboldt Foundation.

http://dx.doi.org/10.1016/j.tcs.2017.01.017

Please cite this article in press as: T. Masopust, M. Thomazo, On Boolean combinations forming piecewise testable languages, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.01.017

the best representation. The choice significantly depends on applications. We are interested in those Boolean combinations that resemble the disjunctive normal form of logical formulas rather than in the most concise representation.

The basic idea to perform this translation can be outlined as follows. Let L be a language over Σ (represented by its minimal DFA) and let the equivalence relation $\sim_k$ on Σ^* be defined by $u \sim_k v$ if u and v have the same sets of (scattered) subwords up to length k, denoted by $sub_k(u) = sub_k(v)$. Then L is piecewise testable if and only if there exists a nonnegative integer k such that $\sim_k \subseteq \sim_L$, where $\sim_L$ is the Myhill congruence [24], that is, every k-PT language is a finite union of $\sim_k$-classes. As shown, e.g., by Klíma [17], the $\sim_k$-classes can be described by languages of the form $[w]_{\sim_k} = \bigcap_{u \in sub_k(w)} L_u \cap \bigcap_{u \notin sub_k(w), |u| \leq k} \overline{L_u}$, where $\overline{L_u}$ denotes the complement of L_u. The high-level approach is thus:

1. Check whether the regular language L is piecewise testable.
2. If so, compute the minimal $k \geq 0$ for which L is k-piecewise testable.
3. Compute the finite number of representatives of the equivalence classes that form the union of the language L, express them as above and form their union.

We study the computational and descriptional complexity of this approach, provide an overview of related results, and formulate several open problems.

The complexity of the first step has been studied in the literature. Simon [26] proved that PT languages are exactly those regular languages whose syntactic monoid is $\mathscr{J}$-trivial, which gives decidability. Stern [28] showed that the problem is decidable in polynomial time for languages represented by DFAs and Cho and Huynh [5] proved NL-completeness for DFAs. Later, Trahtman [31] showed that the problem is solvable in time quadratic with respect to the number of states of the DFA and linear with respect to the size of the alphabet, and Klíma and Polák [19] gave an algorithm that is quadratic in the size of the input alphabet and linear in the number of states of the DFA. For languages represented by NFAs, the problem is PSPACE-complete [11].

The second step gives rise to the *k-piecewise testability* problem formulated as follows:

INPUT: an automaton (DFA or NFA) $\mathscr{A}$
OUTPUT: YES if and only if $L(\mathscr{A})$ is k-piecewise testable

The problem is trivially decidable for any k because there are only finitely many k-PT languages over the alphabet of $\mathscr{A}$. We investigate and overview the computational complexity of this problem. The upper bound complexity for DFAs has been independently solved in [10,18,23]. The co-NP upper bound on the k-piecewise testability problem for DFAs first appeared in [10] without proof, formulated in terms of separability.[3] In this paper, we recall (without proof) the result of [18] showing that the problem is co-NP-complete for DFAs if $k \geq 4$. We then focus on the complexity of the problem for $k < 4$. In particular, for the input given as the minimal DFA, the problem is trivial for $k = 0$, belongs to AC^0 for $k = 1$ (Theorem 6), and is NL-complete for $k = 2, 3$ (Theorems 13 and 18). For NFAs, we show that the problem is PSPACE-complete for any $k \geq 0$ (Theorem 20).

There is an interesting observation by Klíma and Polák [19] that if the depth of a minimal DFA recognizing a PT language is k, then the language is k-PT. (Bounds for finite languages and upward and downward closures have recently been investigated by Karandikar and Schnoebelen [16].) The observation reduces Step 2 to solving a finite number of k-piecewise testability problems, since the upper bound on k is given by the depth of the minimal DFA equivalent to $\mathscr{A}$. The opposite implication does not hold, therefore we investigate the relationship between the depth of an NFA and k-piecewise testability of its language. We show that, for every $k \geq 0$, there exists a k-PT language with an NFA of depth $k - 1$ and with the minimal DFA of depth $2^k - 1$ (Theorem 27). Although it is well known that DFAs can be exponentially larger than NFAs, a by-product of our result is that all the exponential number of states of the DFA form a simple path, which is, in our opinion, a result of interest by its own. In addition, the reverse of the NFAs constructed in the proof is deterministic, partially ordered and locally confluent. Therefore, our result also provides a further insight into the complexity of the reverse of piecewise testable languages previously studied in [4,14].

The last step of the approach requires to compute those $\sim_k$-classes, whose union forms the language L, and to express them as the intersection of languages of the form L_u or its complements. To identify these equivalence classes, we make use of the $\sim_k$-canonical DFA, whose states correspond to $\sim_k$-classes. We construct the $\sim_k$-canonical DFA and compute its accepting states by intersection with the input automaton. The accepting states then represent the $\sim_k$-classes forming the language L. The $\sim_k$-canonical DFA can be effectively constructed. Moreover, although the precise size of the $\sim_k$-canonical DFA is not known, see the estimations in [15], we show that the tight upper bound on its depth is $\binom{k+n}{k} - 1$, where n is the cardinality of the alphabet (Theorem 31).

This paper is an extended version of paper [23] presented at the DLT 2015 conference, containing full proofs and updated with the latest results and open problems. After introducing the necessary notions (Section 2), we introduce the approach on an example (Section 3), before studying the complexity of the k-piecewise testability problem for DFAs (Section 4) and NFAs (Section 5). We finish by investigating the depth of minimal DFAs (Section 6).

[3] The result is a consequence of a proof that is omitted in the conference version.

Please cite this article in press as: T. Masopust, M. Thomazo, On Boolean combinations forming piecewise testable languages, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.01.017