Information Sciences 370–371 (2016) 204–220

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A sufficient condition to polynomially compute a minimum separating *DFA*

Manuel Vázquez de Parga, Pedro García, Damián López*

Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Spain

ARTICLE INFO

Article history:
Received 25 July 2015
Revised 15 July 2016
Accepted 21 July 2016
Available online 22 July 2016

Keywords:
Minimal separating *DFA*
Minimal consistent *DFA*
Model checking
Minimization of incompletely specified automata

ABSTRACT

The computation of a minimal separating automaton (MSA) for regular languages has been studied from many different points of view, from synthesis of automata or Grammatical Inference to the minimization of incompletely specified machines or Compositional Verification. In the general case, the problem is NP-complete, but this drawback does not prevent the problem from having a real application in the above-mentioned fields. In this paper, we propose a sufficient condition that guarantees that the computation of the MSA can be carried out with polynomial time complexity.

1. Introduction

In this work, we study the problem of computing a minimal separating automaton (MSA) for regular languages. This problem has been studied from many different points of view. We note that, in the general case, the decision problem is NP-complete. This complexity result can be derived from the results on synthesis of automata by Trakhtenbrot and Barzdin [30], who also state a (strict) condition that guarantees polynomial computation. In the Grammatical Inference (GI) framework, Gold proves in [14] that the decision problem of obtaining a *DFA* with a given number of states that is compatible with a finite (positive and negative) sample is NP-complete. In the same GI framework, Angluin proves in [1] that even a small modification of the condition stated by Trakhtenbrot and Barzdin implies that the problem is NP-complete. Also, in [25], Pfleeger studies the complexity of the *minimization of incompletely specified finite state machines* obtaining the same complexity bound.

Briefly speaking, all of these problems can be enunciated as the problem of computing (given any two (regular) languages L_+ and L_-) an automaton with the smallest number of states that accepts the strings in L_+ and rejects all the strings in L_- (the behavior of the automaton with respect to the strings not in $L_+ \cup L_-$ is irrelevant). Despite the exponential time complexity in the worst case of the general problem, in our work we prove a sufficient condition that guarantees that the computation can be carried out with polynomial time complexity.

One of the first approaches to the problem was presented by Trakhtenbrot and Barzdin [30], where the authors study the problem of computing the minimum deterministic finite automaton (*DFA*) which is consistent with respect to a finite set of strings of a target language and its complement. In their work, the authors prove that the *DFA* can be obtained with

* Corresponding author. Fax: 34 963877359.
E-mail addresses: mvazquez@dsic.upv.es (M. Vázquez de Parga), pgarcia@dsic.upv.es (P. García), dlopez@dsic.upv.es (D. López).

http://dx.doi.org/10.1016/j.ins.2016.07.053
0020-0255/

polynomial complexity whenever a *uniformly-complete* sample is available (a sample that exclusively contains every string over the alphabet up to a given length).

Several authors study the computation of the *minimal cover-automaton* as a compact representation of a finite set of strings over an alphabet [6,7,18]. Taking into account the results by Trakhtenbrot and Barzdin, the computation of the minimal cover-automaton can be stated as the problem of obtaining the minimum *DFA* such that L_+ is finite and L_- is the language that contains the strings not in L_+ whose length is lower than or equal to an integer n that denotes the length of the longest string in L_+. This allows the finite set L_+ to be described by using the cover-automaton obtained together with n.

As mentioned above, another problem that is related to the computation of the MSA is the minimization of incompletely specified state machines, where incomplete means that either the transition function or the membership of some states to the set of accepting states is undefined. Thus, by taking into account the result of the analysis of any given string using an incompletely specified machine, it is possible to distinguish a set of strings that are accepted, a set of strings that are rejected, and a third set of strings that can either be accepted or rejected. Among the different approaches to the problem, in [22], the authors address the task by enumerating the possible reductions of the input machine and selecting the minimum one. Despite its complexity, the method has been used recently (i.e., in [8]). In the context of circuit design, in [27] propose exact methods (based on the computation of sets of *compatible* states) as well as heuristics. In [23], in a more general approach, Pena and Oliveira propose a method that takes into account previous GI methods. With the exception of the heuristics proposed, none of the enumerated works bypass the exponential complexity of the problem.

In some circumstances, the use of state machines in the modeling of algorithms, sequential logic circuits and communication protocols allows the verification of the system to be reduced to the computation of a MSA. *Compositional verification* is one of these approaches and is considered to be a way to scale up Model Checking [9]. Once two components M_1 and M_2 and the property P (to hold) are characterized by means of regular languages, it is possible to check that the composition of M_1 and M_2 (the intersection of the component languages) fulfills the property (the intersection is a sublanguage of $L(P)$) if there is a contextual assumption A such that the following inference rule is satisfied:

$$\frac{L(M_1) \cap L(A) \subseteq L(P) \qquad L(M_2) \subseteq L(A)}{L(M_1) \cap L(M_2) \subseteq L(P)}$$

The main drawback of applying this *assume-guarantee* rule is the need for expert knowledge in order to obtain the contextual assumptions, while the minimality of the assumption model is important in terms of performance [8]. When regular models are considered, this approach to Model Checking can be reduced to the problem of finding the MSA for the languages $L(M_2)$ (which plays the role of L_+) and $L(M_1) - L(P)$ (which plays the role of L_-).

Among the results in this field, in [15], the authors address the task in the context of the design of logic circuits and propose a heuristic that iteratively constructs a *contradicting sequence* that is used to find incompatible states. In [8], the authors use a version of the L^* algorithm by Angluin [2] in order to obtain an incompletely specified state machine that is then reduced by using the algorithm proposed in [22]. In [21], Neider addresses the problem by representing the desired properties of the *DFA* in terms of a logical formula and using standard SAT or SMT solvers to find a solution.

As mentioned above, the goal of Grammatical Inference is to obtain the MSA in the special case of two finite languages L_+ and L_-. Despite the negative results related to the complexity of the problem [1,14], recent work in this field proves that it is possible to relax the uniformly-complete criterion in order to compute the minimal consistent *DFA* with polynomial complexity [31]. Related work in the same field of Grammatical Inference allows to propose a heuristic to compute a *small DFA* that is consistent with a finite input by inferring a team of automata using different order-criteria on the prefix tree acceptor for the sample and selecting the smallest *DFA* obtained [12].

In this paper, we study the conditions that allow the MSA with polynomial complexity to be obtained. We prove a sufficient condition over the whole set of strings that are involved in the problem that guarantees that the process can be carried out with polynomial time complexity. Because the condition we propose takes into account the strings in the union of the sets L_+ and L_-, for the sake of simplicity (and without lack of generality), we state this problem as the following task: given two regular languages L_+ and $L_\cup$, where $L_+ \subseteq L_\cup$, compute a minimal *DFA* that accepts the strings in L_+ and rejects the strings in $L_\cup - L_+$ with polynomial time complexity (i.e., the search of a *DFA* that separates L_+ and $L_\cup - L_+$). Any automaton that fulfills these conditions is considered to be *consistent with respect to* L_+ *and* $L_\cup - L_+$.

2. Notation and definitions

In this section, we summarize the main definitions used in the paper. We recommend [16] to the reader for further notions or definitions.

Let Σ be a finite alphabet and let Σ^* be the set of strings over Σ, where λ denotes the empty word and $|x|$ denotes the length of x (thus, $|\lambda| = 0$). A *language* L over Σ is any subset of Σ^*. Here we recall the definition of the *canonical order* over Σ^* as being the order that first classifies the shorter strings and considers the alphabetic order for those strings of the same length.

A *(non-deterministic) finite automaton* is a 5-tuple $A = (Q, \Sigma, \delta, I, F)$, where Q is a finite set of states, Σ is an alphabet, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states and δ: $Q \times \Sigma \to 2^Q$ is the transition function, which can also be seen as a subset of $Q \times \Sigma \times Q$. The transition function can be extended in a natural way to Σ^* as well as to 2^Q. Given