Information and Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

On the computational complexity of problems related to distinguishability sets ☆

Markus Holzer *, Sebastian Jakobi

Institut für Informatik, Universität Giessen, Arndtstr. 2, 35392 Giessen, Germany

ARTICLE INFO

Article history:
Received 1 December 2015
Available online xxxx

Keywords:
Distinguishability sets
Synchronization
Computational complexity
State complexity
Deterministic finite automata

ABSTRACT

We study the computational complexity of problems related to distinguishability sets for regular languages. Roughly speaking, the distinguishability set $\mathsf{D}(L)$ for a (not necessarily regular) language L consists of all those words w for which there exists x and y such that word xw is in L if and only if word yx is *not* in L; hence the word w distinguishes the two prefixes x and y. One can view this mapping from L to its distinguishability set as an operator $\mathsf{D}\colon 2^{\Sigma^*} \to 2^{\Sigma^*}$ with $L \mapsto \mathsf{D}(L)$. In particular, we investigate the complexity of the representation problem, i.e., deciding for two given automata A and B, whether B accepts the distinguishability set of $L(A)$. It is shown that this problem and some of its variants are highly intractable, namely PSPACE-complete. In fact, determining the size of an automaton for $\mathsf{D}(L(A))$ is already PSPACE-complete. On the other hand, questions related to the hierarchy induced by iterated application of the D-operator turn out to be much easier. For instance, the question whether for a given automaton A, the accepted language is equal to its own distinguishability set, i.e., whether $L(A) = \mathsf{D}(L(A))$ holds, is shown to be NL-complete. As a byproduct of our investigations, we found a nice characterization of synchronizing automata, namely that a (minimal) automaton A is synchronizing if and only if $\mathsf{D}(L(A)) = \mathsf{D}^2(L(A))$.

1. Introduction

There is a vast literature documenting the importance of the notion of finite automata and problems thereof as an enormously valuable concept in theoretical computer science and applications. Although the history of finite automata dates back around 60 years, even nowadays this is a vivid area of research. For instance, recently, the language $\mathsf{D}(L(A))$ that distinguishes between all non-equivalent states or quotients of a given deterministic finite automaton A was considered in more detail in [1]. There a systematic study of general properties of $\mathsf{D}(L(A))$ is carried out from a descriptional and a formal language theoretical point of view. Observe that the idea of distinguishability is not new and has a long and fruitful history, see, e.g., Moore's seminal paper on gedankenexperiments [2]—a brief summary on some developments is given in [1], too. The motivation to study this language and its properties stems from electronic circuit testing. There a property is tested by applying several inputs to the circuit, and checking the produced output. Since circuits and finite automata are closely related by simulating each other, it is natural to ask for the minimality of these models. For finite automata the

☆ This is a revised and expanded version of a paper presented at the 17th International Workshop on Descriptional Complexity of Formal Systems (DCFS) held in Waterloo, ON, Canada, June 25–27, 2015.

* Corresponding author.
E-mail addresses: holzer@informatik.uni-giessen.de (M. Holzer), sebastian.jakobi@informatik.uni-giessen.de (S. Jakobi).

http://dx.doi.org/10.1016/j.ic.2017.09.003
0890-5401/

Table 1
The problems related to D-sets studied in this paper with problem name, problem description, and its computation complexity. In the problem description "I:" refers to the input and "Q:" to the question that has to be answered. Moreover, sc refers to the deterministic state complexity of the language under consideration.

Problem name	Problem description		Complexity
DFA-D-Membership	I: Q:	DFA A and word w. Is $w \in \mathsf{D}(A)$?	L-complete
Lang.-D-Membership	I: Q:	DFA A and word w. Is $w \in \mathsf{D}(L(A))$?	NL-complete
D-Set-Size	I: Q:	DFA A and integer k. Is $\mathrm{sc}(\mathsf{D}(L(A))) \leq k$?	PSPACE-compl.
L-Versus-D	I: Q:	DFAs A and B. Is $L(B) = \mathsf{D}(L(A))$?	PSPACE-compl.
–	I: Q:	DFAs A and B. Is $L(B) \supseteq \mathsf{D}(L(A))$?	NL-complete
	I: Q:	DFAs A and B. Is $L(B) \subseteq \mathsf{D}(L(A))$?	PSPACE-compl.
L-Equals-D	I: Q:	DFA A. Is $L(A) = \mathsf{D}(L(A))$?	NL-complete
D-Equals-DSquare	I: Q:	DFA A. Is $\mathsf{D}(L(A)) = \mathsf{D}^2(L(A))$?	NL-complete
L-Equals-DSquare	I: Q:	DFA A. Is $L(A) = \mathsf{D}^2(L(A))$?	NL-complete

minimization problem dates back to the early beginnings of automata theory. Minimization asks whether two states are equivalent or not. Thus, minimization for finite automata can be described by considering only words from $\mathsf{D}(L(A))$, i.e., words that distinguish between non-equivalent states of a given deterministic finite automaton A.

What is missing in the investigation in [1] is the computational complexity of problems related to the language $\mathsf{D}(L(A))$, for a deterministic finite automaton A. For instance, from the motivation given in [1] the complexity of the following problem that is related to minimization is relevant: how hard is it to decide for a given word w and a deterministic finite automaton A, whether w belongs to $\mathsf{D}(L(A))$? Other questions on the representation size of the $\mathsf{D}(L(A))$ language by automata or on the iterated application of the D-operation, when viewed as an operator $\mathsf{D}\colon 2^{\Sigma^*} \to 2^{\Sigma^*}$ with $L \mapsto \mathsf{D}(L)$, can be asked. It turns out that the computational complexity of these problems varies from L- to PSPACE-completeness, which is an enormous span in complexity. Our precise complexity results are depicted in Table 1. Moreover, the PSPACE-completeness results are very interesting, since only deterministic finite automata are involved, and normally, standard problems that deal with deterministic devices turn out to be of lower complexity, see, e.g., [3]. In fact, the problems related to representability of distinguishability sets turn out to be highly intractable, namely PSPACE-complete. Even determining the size of an automaton for $\mathsf{D}(L(A))$ is already PSPACE-complete. On the other hand, questions related to the hierarchy induced by iterated application of the distinguishability operator turn out to be much easier, namely NL-complete. This significant decrease in complexity goes hand in hand with a very interesting structure of this hierarchy, namely, it collapses to its third level, i.e., $\mathsf{D}^2(L) = \mathsf{D}^3(L)$, for *every* language L [1]. As a spin-off of our investigations, we found a nice characterization of synchronizing automata, namely that a (minimal) automaton A is synchronizing if and only if $\mathsf{D}(L(A)) = \mathsf{D}^2(L(A))$. During the last decade, synchronizing automata and Černý's Conjecture were a very active research area, see, e.g., [4] for a survey on these topics. Our investigation on the computational complexity of the distinguishability operator D can be seen as a first step towards a better understanding of other distinguishability operators as, e.g., described in [1].

The paper is organized as follows: the next section provides basic definitions concerning finite automata and the distinguishability operator D. Moreover, also some important properties of the D-operation are listed. After that, we first discuss the complexity of deciding whether a word belongs to the distinguishability set of a given language. We will see that the complexity of this problem depends on a subtle detail in the problem definition. Then we investigate the complexity of problems related to the representability of distinguishability sets. We close our studies with a summary of the obtained results and give hints for further research.

2. Preliminaries

We recall some definitions on finite automata as contained in [5]. A *deterministic finite automaton* (DFA) is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$, where Q is the finite set of *states*, Σ is the finite set of *input symbols*, $q_0 \in Q$ is the *initial state*, $F \subseteq Q$ is the set of *accepting states*, and $\delta\colon Q \times \Sigma \to Q$ is the *transition function*. The *language accepted* by the DFA A is defined as

$$L(A) = \{ w \in \Sigma^* \mid \delta(q_0, w) \in F \},$$

where the transition function is recursively extended to $\delta\colon Q \times \Sigma^* \to Q$.

Please cite this article in press as: M. Holzer, S. Jakobi, On the computational complexity of problems related to distinguishability sets, Inf. Comput. (2017), http://dx.doi.org/10.1016/j.ic.2017.09.003