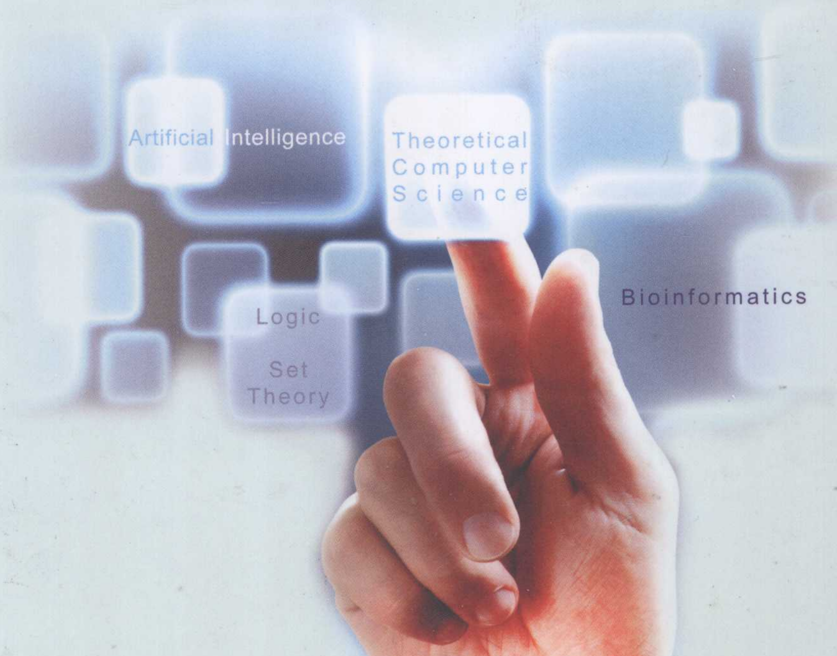
Mathematics, Computing, Language, and Life:
Frontiers in Mathematical Linguistics and Language Theory

Vol. 2

Edited by

Carlos Martín-Vide

Scientific Applications of Language Methods



Imperial College Press

Mathematics, Computing, Language, and Life:
Frontiers in Mathematical Linguistics and Language Theory

Vol. 2

Scientific Applications of Language Methods

Edited by

Carlos Martín-Vide

Universitat Rovira i Virgili, Spain

ICP

Imperial College Press

Published by

Imperial College Press
57 Shelton Street
Covent Garden
London WC2H 9HE

Distributed by

World Scientific Publishing Co. Pte. Ltd.
5 Toh Tuck Link, Singapore 596224
USA office: 27 Warren Street, Suite 401-402, Hackensack, NJ 07601
UK office: 57 Shelton Street, Covent Garden, London WC2H 9HE

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory — Vol. 2
SCIENTIFIC APPLICATIONS OF LANGUAGE METHODS

ISBN-13 978-1-84816-544-1
ISBN-10 1-84816-544-7

Typeset by Stallion Press
Email: enquiries@stallionpress.com

Printed in Singapore by B & Jo Enterprise Pte Ltd

Scientific Applications of Language Methods

Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory

Series Editor: Carlos Martín-Vide
Rovira I Virgili University, Tarragona, Spain

Vol. 1 Parsing Schemata for Practical Text Analysis
by Carlos Gómez-Rodríguez

Vol. 2 Scientific Applications of Language Methods
by Carlos Martín-Vide

Preface

Language theory, as originated from Chomsky's seminal work in the fifties last century and in parallel to Turing-inspired automata theory, was first applied to natural language syntax within the context of the first unsuccessful attempts to achieve reliable machine translation prototypes. After this, the theory proved to be very valuable in the study of programming languages and the theory of computing.

In the last 15–20 years, language and automata theory has experienced quick theoretical developments as a consequence of the emergence of new interdisciplinary domains and also as the result of demands for application to a number of disciplines.

Language methods (i.e. formal language methods) have been applied to a variety of fields, which can be roughly classified as:

- Computability and complexity,
- Natural language processing,
- Artificial intelligence, cognitive science, and programming,
- Bio-inspired computing and natural computing,
- Bioinformatics.

The connections of this broad interdisciplinary domain with other areas include: computational linguistics, knowledge engineering, theoretical computer science, software science, molecular biology, etc.

This volume gives just a few examples of the sort of research involved in this framework, with the intention to reflect the spirit of the whole book series.

Carlos Martín-Vide

Contents

Chapter 1

Descriptional Complexity — An Introductory Survey

Markus Holzer and Martin Kutrib
Institut für Informatik, Universität Giessen,
Arndtstr. 2, 35392 Giessen, Germany,
E-mail: {holzer,kutrib}@informatik.uni-giessen.de

The purpose of the paper is to give an introductory survey of the main aspects and results regarding the relative succinctness of different representations of languages, such as finite automata, regular expressions, pushdown automata and variants thereof, context-free grammars, and descriptional systems from a more abstract perspective. Basic properties of these descriptional systems and their size measures are addressed. The trade-offs between different representations are either bounded by some recursive function, or reveal the phenomenon that the gain in economy of description can be arbitrary. In the latter case there is no recursive function serving as upper bound. We discuss developments relevant to the descriptional complexity of formal systems. The results presented are not proved but we merely draw attention to the big picture and some of the main ideas involved.

1.1 Introduction

In the field of theoretical computer science the term *descriptional complexity* has a well known meaning as it stands. Since the beginning of computer science descriptional complexity aspects of systems (automata, grammars, rewriting systems, etc.) have been a subject of intensive research [111]—since more than a decade the Workshop on "Descriptional Complexity of Formal Systems" (DCFS), formerly known as the Workshop on "Descriptional Complexity of Automata, Grammar, and Related Structures," has contributed substantially to the development of this field of research. The broad field of descriptional complexity of formal systems includes, but is not limited to, various measures of complexity of automata, grammars, languages and of related descriptional systems, succinctness of descriptional systems, trade-offs between complexity and mode of operation, etc., to mention a few.

The time has come to give an introductory survey of the main aspects and results regarding the relative succinctness of different representations of languages by finite automata, pushdown automata and variants thereof, context-free grammars, and descriptional systems from a more abstract perspective. Our tour mostly focuses on results that were found at the advent of descriptional complexity, for example, [52, 53, 59, 60, 98, 109, 112]. To this end, we have to unify the treatment of different research directions from the past. See also [38] for a recent survey of some of these results. Our write up obviously lacks completeness and it reflects our personal view of what constitute the most interesting relations of the aforementioned devices from a descriptional complexity point of view. In truth there is much more to the subject in question, than one can summarize here. For instance, the following current active research directions were not addressed in this summary: we skipped almost all results from the descriptional complexity of the operation problem which was revitalized in [137] after the dawn in the late 1970's. Moreover we will discuss anything on the subject of magic numbers a research field initiated in [73], and on the related investigations of determinization of nondeterministic finite automata accepting subregular languages done in [14] and others, and finally we left out the interesting field of research on the transition complexity of nondeterministic finite automata which has received a lot of attention during the last years [26, 46, 69, 70, 97].

In the next section, basic notions are given, and the basic properties of descriptional systems and their complexity measures are discussed and presented in a unified manner. A natural and important measure of descriptional complexity is the size of a representation of a language, that is, the length of its description. Section 1.3 is devoted to several aspects and results with respect to complexity measures that are recursively related to the sizes. A comprehensive overview of results is given concerning the question: how succinctly can a regular or a context-free language be represented by a descriptor of one descriptional system compared with the representation by an equivalent descriptor of the other descriptional system? Section 1.4 generalizes this point of view. Roughly speaking some, say, structural resource is fixed and its descriptional power is studied by measuring other resources. So, the complexity measures are not necessarily recursively related to the sizes of the descriptors. Here we stick with context-free grammars and subclasses as descriptional systems. Finally, Section 1.5 deals with the phenomenon of non-recursive trade-offs, that is, the trade-offs between representations of languages in different descriptional systems are not bounded by any recursive function. With other words, the gain in economy of description can be arbitrary. It turned out that most of the proofs appearing in the literature are basically relying on one of two fundamental schemes. These proof schemes are presented in a unified manner. Some important results are collected in a compilation of non-recursive trade-offs.

1.2 Descriptional Systems and Complexity Measures

We denote the set of nonnegative integers by $\mathbb{N}$, and the powerset of a set S by 2^S. In connection with formal languages, strings are called *words*. Let Σ^* denote the set of all words over a finite alphabet Σ. The *empty word* is denoted by λ, and we set $\Sigma^+ = \Sigma^* - \{\lambda\}$. For the *reversal of a word* w we write w^R and for its *length* we write $|w|$. A *formal language* L is a subset of Σ^*. In order to avoid technical overloading in writing, two languages L and L' are considered to be equal, if they differ at most by the empty word, that is, $L - \{\lambda\} = L' - \{\lambda\}$. Throughout the article two automata or grammars are said to be *equivalent* if and only if they accept or generate the same language. We use $\subseteq$ for *inclusions* and $\subset$ for *strict inclusions*.

We first establish some notation for descriptional complexity. In order to be general, we formalize the intuitive notion of a representation or description of a family of languages. A *descriptional system* is a collection of encodings of items where each item *represents* or *describes* a formal language. In the following, we call the items *descriptors*, and identify the encodings of some language representation with the representation itself. A formal definition is:

Definition 1.1. A *descriptional system* $\mathcal{S}$ is a set of finite descriptors, such that each descriptor $D \in \mathcal{S}$ describes a formal language $L(D)$, and the underlying alphabet $\mathrm{alph}(D)$ over which D represents a language can be read off from D. The *family of languages represented (or described)* by $\mathcal{S}$ is $\mathscr{L}(\mathcal{S}) = \{ L(D) \mid D \in \mathcal{S} \}$. For every language L, the set $\mathcal{S}(L) = \{ D \in \mathcal{S} \mid L(D) = L \}$ is the set of its descriptors in $\mathcal{S}$.

Example 1.2. Pushdown automata (PDA) can be encoded over some fixed alphabet such that their input alphabets can be extracted from the encodings. The set of these encodings is a descriptional system $\mathcal{S}$, and $\mathscr{L}(\mathcal{S})$ is the family of context-free languages (CFL). □

Now we turn to measure the descriptors. Basically, we are interested in defining a complexity measure as general as possible to cover a wide range of approaches, and in defining it as precise as necessary to allow a unified framework for proofs. So, we consider a *complexity measure* for a descriptional system $\mathcal{S}$ to be a total, recursive mapping $c : \mathcal{S} \to \mathbb{N}$. The properties total and recursive are straightforward.

Example 1.3. The family of context-free grammars is a descriptional system. Examples for complexity measures are the number productions appearing in a grammar, or the number of nonterminals, or the total number of symbols, that is, the length of the encoding. □

Common notions as the *relative succinctness of descriptional systems* and our intuitive understanding of descriptional complexity suggest to consider the *size of descriptors*. From the viewpoint that a descriptional system is a collection of encoding strings, the length of the strings is a natural measure for the size. We denote it by length. In fact, we will use it to obtain a rough classification of different complexity measures. We distinguish between measures that (with respect to the underlying alphabets) are recursively related with length and measures that are not.

Definition 1.4. Let $\mathcal{S}$ be a descriptional system with complexity measure c. If there is a total, recursive function $g : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that $\mathsf{length}(D) \leq g(c(D), |\text{alph}(D)|)$, for all $D \in \mathcal{S}$, then c is said to be an *s-measure*.

Example 1.5. Let us consider a widely accepted measure of complexity for finite automata, that is, their number of states, which is denoted by state. The formal definition of a finite automaton is given in the next section. Is state an s-measure? What makes a difference between the number of states (say, for deterministic finite automata (DFA)) and the lengths of encoding strings? The answer is obvious, encoding strings are over some fixed alphabet whereas the input alphabet of DFAs is not fixed *a priori.* The number of transitions depends on the input alphabet while the number of states does not. But states and transitions both determine the lengths of encoding strings. Nevertheless, when finite automata are addressed then, actually, a fixed given input alphabet is assumed tacitly. Since we regarded this aspect in the definition of s-measures, the answer to the first question is yes, the number of states of finite automata is an s-measure. To this end, given a deterministic finite automaton A, we may choose $g(\mathsf{state}(A), \text{alph}(A)) = k \cdot \mathsf{state}(A) \cdot \text{alph}(A)$, where $\mathsf{state}(A) \cdot \text{alph}(A)$ is the number of transition rules, and k is a mapping that gives the length of a rule dependent on the actual encoding alphabet, the number of states and the number of input symbols.

Similarly, we can argue for other types of finite automata as nondeterministic or alternating ones either with one-way or two-way head motion, etc. If the number of transition rules depends on the number of states and the number of input symbols (and, of course, on the type of the automaton in question), and the length of the rules is bounded dependent on the type of the automaton, then state is an s-measure. □

Whenever we consider the relative succinctness of two descriptional systems $\mathcal{S}_1$ and $\mathcal{S}_2$, we assume the intersection $\mathscr{L}(\mathcal{S}_1) \cap \mathscr{L}(\mathcal{S}_2)$ to be non-empty.

Definition 1.6. Let $\mathcal{S}_1$ be a descriptional system with complexity measure c_1, and $\mathcal{S}_2$ be a descriptional system with complexity measure c_2. A total function $f : \mathbb{N} \to \mathbb{N}$, is said to be an *upper bound* for the increase in complexity when changing from a descriptor in $\mathcal{S}_1$ to an equivalent descriptor in $\mathcal{S}_2$, if for all $D_1 \in \mathcal{S}_1$ with $L(D_1) \in \mathscr{L}(\mathcal{S}_2)$ there exists a $D_2 \in \mathcal{S}_2(L(D_1))$ such that $c_2(D_2) \leq f(c_1(D_1))$.

If there is no recursive function serving as upper bound, the *trade-off is said to be non-recursive.* That is, whenever the trade-off from one descriptional system to another is non-recursive, one can choose an arbitrarily large recursive function f but the gain in economy of description eventually exceeds f when changing from the former system to the latter.

Definition 1.7. Let $\mathcal{S}_1$ be a descriptional system with complexity measure c_1, and $\mathcal{S}_2$ be a descriptional system with complexity measure c_2. A total function $f : \mathbb{N} \to \mathbb{N}$, is said to be a *lower bound* for the increase in complexity when changing from a descriptor in $\mathcal{S}_1$ to an equivalent descriptor in $\mathcal{S}_2$, if for infinitely many $D_1 \in \mathcal{S}_1$ with $L(D_1) \in \mathscr{L}(\mathcal{S}_2)$ there exists a *minimal* $D_2 \in \mathcal{S}_2(L(D_1))$ such that $c_2(D_2) \geq f(c_1(D_1))$.

1.3 Measuring Sizes

This section is devoted to several aspects of measuring descriptors with s-measures. A main field of investigation deals with the question: how succinctly can a language be represented by a descriptor of one descriptional system compared with the representation by an equivalent descriptor of the other descriptional system? An upper bound for the trade-off gives the maximal gain in economy of description, and conversely, the maximal blow-up (in terms of descriptional complexity) for simulations between the descriptional systems. A maximal lower bound for the trade-off terms the costs which are necessary in the worst cases.

1.3.1 *Descriptional Systems for Regular Languages*

Regular languages are represented by a large number of descriptional systems. So, it is natural to investigate the succinctness of their representations with respect to s-measures in order to optimize the space requirements. In this connection, many results have been obtained. On the other hand, the descriptional complexity of regular languages still offers challenging open problems. In the remainder of this subsection we collect and discuss some of these results and open problems.

1.3.1.1 *Finite Automata*

Here we measure the costs of representations by several types of finite automata in terms of the number of states, which is an s-measure by Example 1.5. Probably the most famous result of this nature is the simulation of nondeterministic finite automata by DFAs. Since several results come up with tight bounds in the exact number of states, it is advantageous to recall briefly the definitions of finite automata on which the results rely.

Definition 1.8. A *nondeterministic finite automaton* (NFA) 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 $\delta : Q \times \Sigma \to 2^Q$ is the *transition function*.

A finite automaton is *deterministic* (DFA) if and only if $|\delta(q, a)| = 1$, for all states $q \in Q$ and letters $a \in \Sigma$. In this case we simply write $\delta(q, a) = p$ instead of $\delta(q, a) = \{p\}$ assuming that the transition function is a mapping $\delta : Q \times \Sigma \to Q$.