



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

Vol. 2

Edited by

Carlos Martín-Vide

Scientific Applications of Language Methods



Artificial Intelligence

Theoretical
Computer
Science

Logic

Set
Theory

Bioinformatics

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



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

Copyright © 2011 by Imperial College Press

All rights reserved. This book, or parts thereof, may not be reproduced in any form or by any means, electronic or mechanical, including photocopying, recording or any information storage and retrieval system now known or to be invented, without written permission from the Publisher.

For photocopying of material in this volume, please pay a copying fee through the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, USA. In this case permission to photocopy is not required from the publisher.

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

1.	Descriptional Complexity — An Introductory Survey	1
	<i>M. Holzer and M. Kutrib</i>	
1.1	Introduction	1
1.2	Descriptional Systems and Complexity Measures	3
1.3	Measuring Sizes	6
1.4	Measuring Resources	33
1.5	Non-Recursive Trade-Offs	39
	References	51
2.	Classifying All Avoidable Sets of Partial Words of Size Two	59
	<i>F. Blanchet-Sadri, B. Blakeley, J. Gunter, S. Simmons and E. Weissenstein</i>	
2.1	Introduction	60
2.2	Preliminaries	62
2.3	Unavoidable Sets of Partial Words of Size Two	64
2.4	Canonical Forms	70
2.5	The Answer to the Conjectures	74
2.6	The Classification	95
2.7	Conclusion	99
	References	100
3.	On Glushkov \mathbb{K} -graphs	103
	<i>P. Caron and M. Flouret</i>	
3.1	Introduction	103
3.2	Definitions	105

3.3	Acyclic Glushkov WFA Properties	114
3.4	Glushkov \mathbb{K} -graph with Orbits	124
3.5	Algorithm for Orbit Reduction	127
3.6	Conclusion	131
	References	131
4.	Natural Language Dictionaries Implemented as Finite Automata	133
	<i>J. Daciuk, J. Piskorski and S. Ristov</i>	
4.1	Dictionaries as Finite Automata	133
4.2	Automata as Mappings	149
4.3	Construction Methods	165
4.4	Internal Structure and Compression	187
	References	200
5.	Tree-Language Based Querying of Hierarchically Structured and Semi-Structured Data	205
	<i>A. Berlea</i>	
5.1	Introduction	205
5.2	Preliminaries	207
5.3	Regular Forest Languages	217
5.4	Grammar Queries	230
5.5	Practical Application: A Pattern Language for XML Querying	259
5.6	Online Querying	275
5.7	Summary and Outlook	294
5.8	Proofs of Theorems	295
	References	306
6.	Quotient Monoids and Concurrent Behaviours	313
	<i>R. Janicki, J. Kleijn and M. Koutny</i>	
6.1	Introduction	314
6.2	Preliminaries	318
6.3	Partial Orders and Order Structures	326
6.4	Mazurkiewicz Traces	334
6.5	Comtraces	337
6.6	Generalised Comtraces	354
6.7	Elementary Net Systems	360

6.8	EN-systems with Inhibitor and Mutex Arcs	372
6.9	Concluding Remarks	382
	References	383
7.	Correction Queries in Active Learning	387
	<i>C. Tîrnăuică</i>	
7.1	Introduction	387
7.2	Preliminaries	389
7.3	Learning Models	391
7.4	Learning with Correction Queries	393
7.5	Polynomial Time Learning with Correction Queries	403
	References	416
8.	Applications of Grammatical Inference in Software Engineering: Domain Specific Language Development	421
	<i>M. Mernik, D. Hrnčič, B. R. Bryant and F. Javed</i>	
8.1	Introduction	422
8.2	Analysis Phase of DSL Development	424
8.3	Design Phase of DSL Development	426
8.4	Grammatical Inference and Language Design	429
8.5	Case Study	445
8.6	Related Work	448
8.7	Conclusion	452
	References	453
9.	Small Size Insertion and Deletion Systems	459
	<i>A. Alhazov, A. Krassovitskiy, Y. Rogozhin and S. Verlan</i>	
9.1	Introduction	459
9.2	Definitions	461
9.3	Basic Simulation Principles	466
9.4	Insertion-Deletion Systems with Rules of Small Size	470
9.5	Context-Free Insertion-Deletion Systems	472
9.6	One-Sided Contextual Insertion-Deletion Systems	481
9.7	Pure Insertion Systems	497
9.8	Graph-Controlled Insertion-Deletion Systems	503
9.9	Graph-Controlled Insertion-Deletion Systems with Priorities	514

9.10	Bibliographical Remarks	520
	References	521
10.	Accepting Networks of Evolutionary Word and Picture Processors: A Survey	525
	<i>F. Manea, C. Martín-Vide and V. Mitrana</i>	
10.1	Introduction	525
10.2	Basic Definitions	527
10.3	Computational Power	536
10.4	Universal ANEPs and ANEPFCs	541
10.5	A Direct Simulation	544
10.6	Accepting Networks of Splicing Processors	545
10.7	Problem Solving with ANEPs/ANEPFCs	548
10.8	Accepting Networks of Picture Processors	549
	References	558
11.	Quantum Automata and Periodic Events	563
	<i>C. Mereghetti and B. Palano</i>	
11.1	Introduction	563
11.2	Preliminaries	566
11.3	Testing Periodicity on Unary 1qfa's	570
11.4	Synthesis of 1qfa's Inducing Periodic Events	572
11.5	Application to Periodic Languages	580
	References	582
12.	Soliton Circuits and Network-Based Automata: Review and Perspectives	585
	<i>M. Bartha and M. Krész</i>	
12.1	Introduction	586
12.2	Basic Concepts	588
12.3	Soliton Graphs and Automata	593
12.4	Elementary Decomposition of Soliton Graphs and Automata	597
12.5	Characterizing Soliton Automata	600
12.6	Complete Systems of Soliton Automata	610
12.7	Algorithms for Soliton Automata	615
12.8	Extensions of the Model and Further Research	623

12.9 Summary	627
References	628
13. Inferring Leadership Structure from Data on a Syntax Change in English	633
<i>W. Garrett Mitchener</i>	
13.1 Introduction	634
13.2 The Available Data	639
13.3 Formulation of the Variable Influence Model	640
13.4 Fitting the <i>Do</i> -support Data	644
13.5 Results and Discussion	650
13.6 Future Directions	652
13.7 Conclusion	656
References	658
14. Weighted Automata Modeling for High Density Linkage Disequilibrium Mapping	663
<i>T. Trang</i>	
14.1 Introduction	664
14.2 Biological Problem and Formulation	667
14.3 Fundamentals of Discrete Structures	676
14.4 Variable Length Finite Automata (VLFA)	682
14.5 Experimental Results on SNP Data	706
14.6 Conclusion	719
References	720
<i>Author Index</i>	723
<i>Subject Index</i>	725

Chapter 1

Descriptive 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, push-down automata and variants thereof, context-free grammars, and descriptive systems from a more abstract perspective. Basic properties of these descriptive 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 descriptive 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 *descriptive complexity* has a well known meaning as it stands. Since the beginning of computer science descriptive 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 “Descriptive Complexity of Formal Systems” (DCFS), formerly known as the Workshop on “Descrip-

tional Complexity of Automata, Grammar, and Related Structures," has contributed substantially to the development of this field of research. The broad field of descriptonal complexity of formal systems includes, but is not limited to, various measures of complexity of automata, grammars, languages and of related descriptonal systems, succinctness of descriptonal 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 descriptonal systems from a more abstract perspective. Our tour mostly focuses on results that were found at the advent of descriptonal 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 descriptonal 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 descriptonal 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 descriptonal systems and their complexity measures are discussed and presented in a unified manner. A natural and important measure of descriptonal 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 descriptonal system compared with the representation by an equivalent descriptor of the other descriptonal sys-

tem? Section 1.4 generalizes this point of view. Roughly speaking some, say, structural resource is fixed and its descriptive 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 descriptive 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 descriptive 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 Descriptive 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 descriptive complexity. In order to be general, we formalize the intuitive notion of a representation or description of a family of languages. A *descriptive 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 *descriptive 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 $\text{alph}(D)$ over which D represents a language can be read off from D . The *family of languages represented (or described)*

by \mathcal{S} is $\mathcal{L}(\mathcal{S}) = \{L(D) \mid D \in \mathcal{S}\}$. For every language L , the set $S(L) = \{D \in \mathcal{S} \mid L(D) = L\}$ 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 descriptonal system \mathcal{S} , and $\mathcal{L}(\mathcal{S})$ is the family of context-free languages (CFL). \square

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 descriptonal system \mathcal{S} to be a total, recursive mapping $c : \mathcal{S} \rightarrow \mathbb{N}$. The properties total and recursive are straightforward.

Example 1.3. The family of context-free grammars is a descriptonal 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. \square

Common notions as the *relative succinctness of descriptonal systems* and our intuitive understanding of descriptonal complexity suggest to consider the *size of descriptors*. From the viewpoint that a descriptonal 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 descriptonal system with complexity measure c . If there is a total, recursive function $g : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ such that $\text{length}(D) \leq g(c(D), |\text{alph}(D)|)$, 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 $g(\text{state}(A), \text{alph}(A)) = k \cdot \text{state}(A) \cdot \text{alph}(A)$, where $\text{state}(A) \cdot \text{alph}(A)$ 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. \square

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

Definition 1.6. Let \mathcal{S}_1 be a descriptive system with complexity measure c_1 , and \mathcal{S}_2 be a descriptive system with complexity measure c_2 . A total function $f : \mathbb{N} \rightarrow \mathbb{N}$, 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 $L(D_1) \in \mathcal{L}(\mathcal{S}_2)$ 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 descriptive 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 descriptive system with complexity measure c_1 , and \mathcal{S}_2 be a descriptive system with complexity measure c_2 . A total function $f : \mathbb{N} \rightarrow \mathbb{N}$, 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 $L(D_1) \in \mathcal{L}(\mathcal{S}_2)$ 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 $\delta : Q \times \Sigma \rightarrow 2^Q$ 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 $\delta : Q \times \Sigma \rightarrow Q$.