Monographs in Computer Science

Editors

David Gries
Fred B. Schneider

Springer Science+Business Media, LLC

Domenico Cantone
Eugenio Omodeo
Alberto Policriti

Set Theory for Computing

From Decision Procedures to Declarative Programming with Sets

Foreword by Jacob T. Schwartz

With 67 Figures

Springer

Domenico Cantone
Dipartimento di Matematica e Informatica
Università di Catania
I-95125 Catania
Italy
cantone@dmi.unict.it

Eugenio Omodeo
Dipartimento di Matematica Pura
e Applicata
Università di L'Aquila
I-67100 L'Aquila
Italy
omodeo@univaq.it

Alberto Policriti
Dipartimento di Matematica e Informatica
Università degli Studi di Udine
I-33100 Udine
Italy
policrit@dimi.uniud.it

Series Editors:
David Gries
Department of Computer Science
The University of Georgia
415 Boyd Graduate Studies
Research Center
Athens, GA 30602-7404
USA

Fred B. Schneider
Department of Computer Science
Cornell University
4115C Upson Hall
Ithaca, NY 14853-7501
USA

Cover illustration: The seventeenth-century engraving reproduced on the front cover of this book (courtesy of Anna Omodeo) shows a butterfly-winged genius getting from Mercury, the god of creative thinking, various tools to be put at the service of Nature. In spite of our good will, we have not been able to find the exact source of the image.

Library of Congress Cataloging-in-Publication Data
Cantone, Domenico.
Set theory for computing: from decision procedures to declarative programming with sets/Domenico Cantone, Eugenio Omodeo, Alberto Policriti.
p. cm. — (Monographs in computer science)
Includes bibliographical references and index.

1. Set theory. 2. Computable functions. I. Omodeo, Eugenio. II. Policriti, Alberto. III. Title. IV. Series.
QA248 .C1945 2001
511.3'22—dc21 2001020446

Printed on acid-free paper.

Production managed by Francine McNeill; manufacturing supervised by Jacqui Ashri.
Photocomposed copy prepared from the authors' LaTeX 2e files using a revised version of Springer's svsing6.cls macro.

ISBN 978-1-4419-2905-1 ISBN 978-1-4757-3452-2 (eBook)
DOI 10.1007/978-1-4757-3452-2

To Maria Pia and Riccardo

To Paola, Pietro, and Sara

To Daniela and

to the memory of my parents

Foreword

If one sets out to construct programming languages of resolutely high level, while taking a deliberately relaxed view of the efficiency concerns that dominate the design of traditional lower-level programming languages such as C, the key question to be faced is this: What objects and operations best represent the most pervasive, "bread-and-butter" operations that the language's intended users will most commonly face? Since programming languages must deal with an endlessly expanding range of applications, this question can never have any single or final answer. For example, the coming of the desktop computer, with its display screen, has lent great interest to the design of objects (commonly called "widgets") and operations for the mouse-driven interactive interface, and to the design of languages focused on notions of "keyboardless programming." Also, niche sublanguages, for example, the language of regular expressions for the important area of string manipulation, will always remain important elements of the general picture. Nevertheless, we can identify one area of preeminent commonality that no general-purpose high-level language can afford to ignore: the manipulation of ordered and unordered aggregates (sets and lists) and of interelement associations (maps). Recognizing this, most recent major programming languages have incorporated such objects in one or another form: the "container classes" of Java, "arrays" of Javascript, "hashes" of Perl, and so forth. The older SETL language, emphasized in the present work, takes a particularly direct and comprehensive approach to objects of this class by wholesale adoption of the common mathematical syntax and semantics of hereditarily finite set theory.

As examples given in the present work show, this works well for the expression of many algorithms. If more justification is needed, we may note that the set-theoretic formalisms developed in the nineteenth and early twentieth centuries made it possible to reexpress the whole intuitive content of geometry, arithmetic, and calculus ("analysis") in set-theoretic terms. We may say in consequence of these developments that all the key concepts of standard mathematics can be digested without forcing any adjustment of the set-theoretic foundation constructed for arithmetic, analysis, and geometry. In particular, this foundation also proves to support all the more abstract mathematical constructions elaborated in such twentieth-century fields as topology, abstract algebra, and category theory. Indeed, these were expressed set-theoretically from their inception. So (if we ignore a few ongoing explorations whose significance remains to be determined) set theory currently stands as a comfortable and universal basis for the whole of mathematics.

Mathematics, however, goes well beyond any mere scheme for computation, no matter how effective and general. Ambitious to deal with universals, it must inevitably become a mechanism for sentence manipulation, which regards the formulas of any system with which it deals as objects themselves subject to rules of manipulation and possessing (or failing to possess) semantic properties like models and a range of validity, perhaps universal. However, since it has been known since Gödel that these key metamathematical properties of formulas can never be calculated algorithmically for any sufficiently rich class of formulas (certainly not for all the formulas of set theory), this range of mathematical investigations must be approached with a cautious respect for the bottomless pit of undecidability that lies near. Nevertheless, determination to explore this challenging area motivates the present book.

The requisite caution is expressed in the careful choice of formalisms for investigation. Ideally, these should be (a) highly expressive, in regard to some class of formulas of clear interest; and (b) manageable, in that they form either (b.i) a decidable class of formulas, or at least (b.ii) a semidecidable class of formulas, or, if even this fails, (b.iii) a class of formulas allowing important and interesting manipulations. Work like that reported in the present book is fundamentally shaped by the need to deal with trade-offs between expressivity and manageability.

The present book selects a variety of such formalisms, some classical, some less so, for close investigation. These include (i) the propositional calculus, which is both decidable and basic to much else; (ii) various predicate systems, with quantifiers restricted to make them manageable; (iii) a formal theory of lists; (iv) subformalisms of set theory, again restricted to make them manageable; and (v) map calculi.

Concerning these last, which may stand for the whole class of "less standard" logics (also including modal and temporal logics) that have been developed and investigated in this century, we may remark that their special interest (relative to "vanilla" set theory) lies in their potential ability to pinpoint areas possessing special semantic and formal advantage. These should allow classes of statements, broad enough to satisfy the requirements of some significant application domain, but constrained enough to be more decidable, or at least more effectively manipulable, than the set-theoretic translations that they are certain to possess. Map theory is a candidate of the kind that the present work studies in some detail.

Finally, the present work reviews some of the basic techniques, including formula normalization, resolution, unification, and the "tableau" technique elaborated in studies by its authors, for decision, and in some cases efficient decision, of the classes of logical formulas considered. The many issues touched on make the book both an introduction to its subject area and a monograph informing the reader of quite recent developments, some of whose most interesting parts are the work of its authors.

New York University Jacob T. Schwartz

February 2001

Preface

This introductory-level text on computable set theory provides a thorough, up-to-date, and comprehensive account of set-oriented symbolic manipulation methods suitable for automated reasoning. Its main objectives are

- to provide a flexible formalization for a variety of set languages, which are too often tackled in unduly naive terms; and
- to clarify the semantics of set constructs firmly established in modern specification languages and in the programming practice.

Set notions are, in fact, so common in computer science today (as they are in mathematics) that virtually everybody in the field has a rather reliable intuitive grasp of them. However, even with familiar notions, frequent recourse to intuition may be tiresome; accordingly, the control imposed by systematic techniques promises increased reliability. Moreover, we believe that a formalized logical characterization is the safest guide for a computationally-oriented use of the structured datatypes one may need: a thread which can help in both the conception and use of software tools—which, in the case of sets, will have a broad spectrum of applications.

In response to diverse needs and applications, research in the field has produced a wealth of approaches and semantics related to sets. The ability of mastering today's variety of systems by means of crisp formal tools is a prerequisite for a high and fine-tunable degree of control over sets and aggregates in general, of the same kind one achieves over numbers thanks to algebra. To enable us to do that, many manageable and widespanning algorithmic methods and deductive techniques are presented in this book. Topics in the book include semantic unification, decision algorithms, modal logics, declarative programming, tableau-based proof techniques, and theory-based theorem proving.

Intended Readers

This book is particularly addressed to graduate and postgraduate students, scholars, and researchers in computer science, artificial intelligence, logic, and computational mathematics who feel a need to complement their intuitive understanding of set concepts with the ability to master them by symbolic and logically based algorithmic methods and deductive techniques.

Also, database and programming language designers and practitioners who are interested in the uses of formal reasoning in computer science will find in this book a clear view of the use of sets and aggregates in such critical issues as the specification of problems, algorithms, and abstract data-types, and algorithmic program verification.

The style of presentation, largely self-contained, is rigorous and accurate. Some familiarity with symbolic logic is helpful but is not a requirement.

Salient Features and Computational Aspects

The presentation will be axiomatic and will survey several variants of the Zermelo-Fraenkel (ZF) theory tailored to different application needs: for instance, unlike ZF, one such variant will be meant to deal exclusively with finite sets; some theories will place individuals at the bottom of the construction of sets, whereas others will found the entire construction solely on the null set; also, some theories allow the membership relation to form cycles, which some other theories forbid, etc.

Concrete, computable models will also be supplemented. These are models whose sets can be algorithmically construed and manipulated. To allow this, quite often the sets under study will be finite, or they will admit a canonical term representation. On occasion, they will be flat sets tractable in purely Boolean terms, but, in general, a set can belong to other sets.

Several chapters of the book are devoted to decidability-related issues and set unification. The axiomatic approach will make it possible to highlight precisely the assumptions that ensure the decidability of some syntactically delimited fragments of set theories. Some of the decision procedures will be presented in the form of efficient saturation strategies for tableau systems.

We feel that this book will meet the needs of many researchers because it is neither a text on axiomatic set theory nor a text on algorithmics, but rather an attempt to bridge a gap between the two. We hope that it will convince some readers of the usefulness of sets in specifications that, ideally, should be both executable and declarative. At any rate, we have the strong expectation that formalized sets will prove crucial in raising the standards of program-correctness technology.

Acknowledgments

We are much indebted to Jacob T. Schwartz (New York University), who gave the initial impulse to the research that originated this book and with whom we maintain stimulating exchanges of ideas on proof-engineering issues. We are grateful to Martin Davis (New York University and University of California at Berkeley), who gave big encouragement and inspiration, and to Alfredo Ferro (University of Catania) and Franco Parlamento (University of Udine), who actively took part since the beginning in much of the research reported in this book.

Among the many other people who took part in the development of the ideas presented in this book, we wish to thank in particular Gianfranco Rossi (University of Parma), Agostino Dovier (University of Udine), and Enrico Pontelli (University of New Mexico), whose contribution is apparent in Chapters 8 and 13; Piero Ursino, whose Ph.D. thesis is reflected partly in the subject matter of Chapter 9; and Andrea Formisano, whose Ph.D. thesis is related to parts of Chapters 3 and 7.

We are also indebted to Vincenzo Cutello (University of Catania) and Calogero G. Zarba, who collaborated in the developments reported in Section 10.3 and Chapter 14, respectively.

Our warm thanks go to Johan van Benthem (University of Amsterdam), and to Giovanna D'Agostino and Angelo Montanari (University of Udine), who contributed in the research which grounds the material presented in Chapter 12.

Ernst-Erich Doberkat (University of Dortmund) and Mohamed Hibti (INOVIA, France), with their positive criticisms, obliged us to revise various parts of this book, thus improving its quality; Patrick Cegielski gave us helpful suggestions.

Two of the authors were kindly hosted at Stanford University by Zohar Manna when Chapter 9 was being prepared.

Finally, we would like to thank Dorella Bellè, Gianluca Cincotti, Marianna Nicolosi-Asmundo, Carla Piazza, Franca Rinaldi, Rosa Ruggeri Cannata, and Matteo Slanina, who contributed in various ways to this project; Anna Omodeo, who kindly provided the image for the cover of this book, and Giuseppe Scollo, who "forced" us to use the BSCW system for collaborative authoring.

The authors acknowledge financial support from the CNR (National Research Council) of Italy, under Research Projects S.E.T.A. and log(SETA), and with "short-term mobility" grants for stays at the New York University of two of the authors. Support came also from the MURST 40% project *"Tecniche formali per la specifica, l'analisi, la verifica, la sintesi e la trasformazione di sistemi software."*

University of Catania — Domenico Cantone
University of L'Aquila — Eugenio Omodeo
University of Udine — Alberto Policriti

February 2001

Contents

Foreword **vii**

Preface **xi**

I INTRODUCTION **1**

1 What Is Computable Set Theory? **3**

1.1 Tentative Subdivision of Computational Logic 5
1.2 One or Several Theories on Sets? 6
1.3 Computable Models of Aggregate Theories 8
1.4 Decidability Issues in Set Theory 9
1.5 Further Means of Automation for Reasoning on Aggregates 11
1.6 Application Scenarios for Computable Set Theory 12

2 Logical Background **14**

2.1 Free Expressions over a Signature 16
2.2 Propositional Calculus 19
2.3 Predicate Calculus and Map Calculus 28
2.4 Theories, in Various Formalisms 49

II BASICS FOR SET-THEORETIC REASONING **59**

3 Axiomatic Views of Aggregates **61**
3.1 The Barest Rudiments of Set Theory 62
3.2 The Theory of Pure Finite Sets 66
3.3 Variant Axiomatizations of Aggregate Theories 77

4 Semantic Views of Aggregates **87**
4.1 Universes of Finite (Nested) Aggregates 88
4.2 Computable Set Models 97
4.3 Appendix: Tiny-SETL Programs 108

5 Sets for Problem Solving **112**
5.1 Conveniences Based on Notation 112
5.2 Set-Based Conceptual Tools 113
5.3 Abstract Characterization of Data Types 119
5.4 Problem Specifications 124
5.5 Specification of Algorithms and Problem Solution Techniques 128
5.6 Sets for Program Verification 130
5.7 Declarative Specification of a Problem Solution Technique 138
5.8 (*) Synthesis of Strongly Computable Set Models 142

6 Decision Problems and Some Solutions **148**
6.1 Satisfiability and Validity 149
6.2 Unification Under Given Axioms 168
6.3 Taxonomy and Normalization of Formulas 179
6.4 Review of Solvable Fragments of Set Theory 190

7 Inference Techniques and Methods **196**
7.1 Resolution-Like Inferential Systems 196
7.2 Semantic Tableaux 207
7.3 Logic Programming and Its Extensibility 223

III DECISION METHODS **245**

8 Set/Hyperset Unification Algorithms **247**
8.1 Blind-Search Unification in Pure Context 248
8.2 Goal-Directed Set-Unification in the Blended Context . . 256
8.3 (*) Ingredients of Goal-Driven Hyperset Unification . . . 259

9 (*) A Syllogistic Solitaire **265**
9.1 Deciding a Fragment of Set Theory by Simulating a Partition 266
9.2 The Trace Theorem 269
9.3 Mimicking a Syllogistic Game by One of Predictable Length 275
9.4 Syllogistic Games Applied to the Set-Satisfiability Problem 281

10 Stratified Syllogistics **284**
10.1 Two-Level Syllogistic 284
10.2 (*) Topological Two-Level Syllogistic 287
10.3 Three-Level Syllogistic with Singleton, Powerset, and Unionset 291

IV SET-SPECIFIC INFERENCE ENGINES **305**

11 Quantitative Set Reasoning **307**
11.1 Case Study: Multilevel Syllogistic with a Cardinality Operator 307
11.2 A Further Extension of Multilevel Syllogistic, with Maps 313
11.3 Reduction of Multilevel Syllogistic to Propositional Calculus 314

12 Set Theory for Nonclassic Logics **318**
12.1 Modal Logics 318
12.2 Translation of Propositional Modal Formulas into Set Terms 322
12.3 Eliciting Set Principles that Ensure Faithfulness of the Translation 326
12.4 Faithfulness of the Translation in Means of Proof 327
12.5 Set-Based Resolution for Nonclassic Logics 331
12.6 (*) Other Logics 335

13 Logic Programming with Sets **340**
13.1 Axiomatic Description of the Blended Universe of Sets 341
13.2 A Variant of SLD-Resolution Specialized for Set-Handling 342

14 Syllogistic Tableaux **350**
14.1 A Decidable Tableau Calculus for **MLSS** 351
14.2 Extending **MLSS** with Function Symbols and Restricted Quantification 361

References 374

Tables and Index of Symbols 393

Association and priority rules 393
Truth tables 393
Common abbreviations in first-order languages 394
Substitution-related notation 394
Map-related notation 395
Function-related notation 395
Basic laws on sets 396
Abbreviations in mathematical English 397

Index of Terms 399