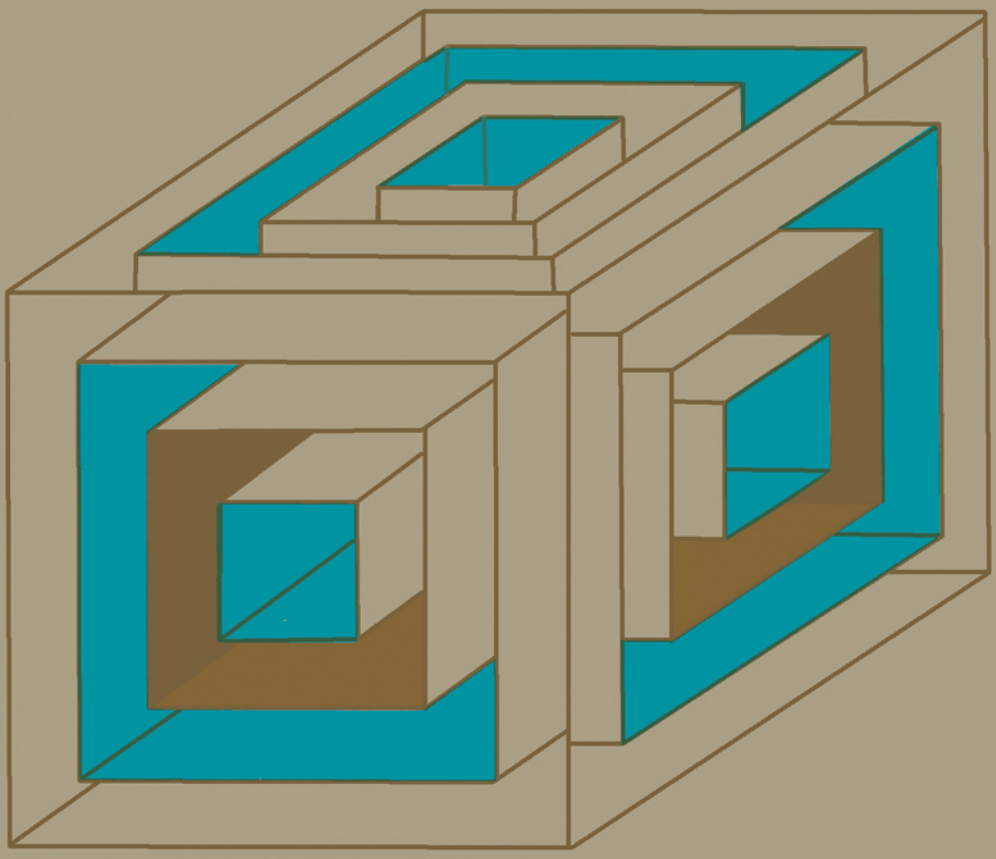
RESOLUTION of EQUATIONS in ALGEBRAIC STRUCTURES

2 REWRITING TECHNIQUES



EDITED BY
HASSAN AÏT-KACI
MAURICE NIVAT

Resolution of Equations in Algebraic Structures

Volume 2

Rewriting Techniques

This page intentionally left blank

Resolution of Equations in Algebraic Structures

Volume 2
Rewriting Techniques

Edited by

Hassan Aït-Kaci

ACA Systems Technology Laboratory
Programming Languages Group
Microelectronics and Computer Technology Corporation
Austin, Texas

Maurice Nivat

LITP
Université Paris VII
Paris, France

ACADEMIC PRESS, INC.
Harcourt Brace Jovanovich, Publishers
Boston San Diego New York
Berkeley London Sydney
Tokyo Toronto

ACADEMIC PRESS, INC.
1250 Sixth Avenue, San Diego, CA 92101

United Kingdom Edition published by
ACADEMIC PRESS INC. (LONDON) LTD.
24–28 Oval Road, London NW1 7DX

Library of Congress Cataloging-in-Publication Data

Resolution of equations in algebraic structures / edited by Hassan Aït-Kaci, Maurice Nivat.
p. cm.
Includes bibliographies and index.
Contents: v. 1. Algebraic techniques – v. 2. Rewriting techniques.
ISBN 0-12-046370-9 (v. 1). – ISBN 0-12-046371-7 (v. 2)
1. Algebra, Abstract. 2. Equations–Numerical solutions. I. Aït-Kaci, Hassan, Date- II. Nivat, M.
QA6162.R47 1989
512'.02–dc 19 88-21727
CIP

89 90 91 92 9 8 7 6 5 4 3 2 1
Printed in the United States of America

Contents

Contents of Volume 1: Algebraic Techniques

Contributors

Numbers in parentheses indicate the pages on which the authors' contributions begin.

Leo Bachmair (1), *Department of Computer Science, State University of New York at Stony Brook, Stony Brook, New York 11794*

Nachum Dershowitz (1, 31), *Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801*

M. Dincbas (87), *European Computer-Industry Research Centre, Arabellastrasse 17, D-8000 München 81, Federal Republic of Germany*

Laurent Fribourg (117), *Laboratoire d'Information, Ecole Normale Supérieure, 45 rue d'Ulm, F-75230 Paris Cédex 05, France*

Joseph A. Goguen (297), *SRI International, 333 Ravenswood Avenue, Menlo Park, California 94205*

Stephane Kaplan (141), *Department of Computer Science, Hebrew University of Jerusalem, Givat Ram, Jerusalem, Israel*

Claude Kirchner (171), *Centre de Recherche en Informatique de Nancy, B. P. 239, F-54506 Vandœuvre-les-Nancy Cédex, France*

Wolfgang Küchlin (211), *Department of Computer and Information Sciences, Ohio State University, Columbus, Ohio 43210*

A. Martelli (245), *Università di Torino, Dipartimento di Informatica, C. so Svizzera 185, I-10149 Torino, Italy*

José Meseguer (297), *SRI International, 333 Ravenswood Avenue, Menlo Park, California 94205*

C. Moiso (245), *CSELT, via Reiss Romoli 274, I-10148 Torino, Italy*

Werner Nutt (297), *FB Informatik, Postfach 3049, Universität Kaiserslautern, D-6750 Kaiserslautern, Federal Republic of Germany*

Dominique Perrin (275), *LITP, Université de Paris VII, 2 place Jussieu, F-75251 Paris Cédex 05, France*

David A. Plaisted (1), *Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina 27514*

Jean-Luc Rémy (141), *Centre de Recherche en Informatique de Nancy, and Unité INRIA-Lourraine, B. P. 239, F-54506 Vandœuvre Cédex, France*

G. F. Rossi (245), *Università de Torino, Dipartimento di Informatica, C. so Svizzera 185, I-10149 Torino, Italy*

H. Simonis (87), *European Computer-Industry Research Centre, Arabellastrasse 17, D-8000 München 81, Federal Republic of Germany*

Gert Smolka (297), *FB Informatik, Postfach 3049, Universität Kaiserslautern, D-6750 Kaiserslautern, Federal Republic of Germany*

P. Van Hentenryck (87), *European Computer-Industry Research Centre, Arabellastrasse 17, D-8000 München 81, Federal Republic of Germany*

Foreword

Equations are pervasive in computer science. They appear in a wide variety of algebraic structures, and in a rich diversity of applications. Therefore, formal methods for establishing the existence of solutions to equations or for effectively finding such solutions constitute a body of scientific knowledge of fundamental importance. For as young a discipline as computer science, growing rapidly from several shoots into a multitude of branches, the danger of losing sight of the generality of equational approaches is real. Thus, it is important to recognize that results obtained in one area of computer science could be applied in another. The example of fixed-point equations is blatant, as these manifest themselves virtually everywhere from programming language semantics and domain theory to deductive databases. Surveying the state of the art in equation resolution reflects more than just a desire to avoid duplication of effort: it is bound to shed light on those general structures and methods that may be abstracted from each idiosyncratic application. This is not only bound to foster cross-fertilization and systematic understanding of the many equational structures and concepts found to be valuable in separate branches of computer science, but it is also a vital and necessary step in the concretization of the discipline that will enable it to endure as a science.

Having come to this realization, we decided, on the occasion of a visit of Maurice Nivat to MCC in 1986, to do something about it. We agreed to organize a colloquium to which we would invite researchers, working with equations, from diverse areas in computer science. The idea was to obtain from the best researchers the solid and stable results of the equational approaches they had devised or mastered in particular applications. Although the aim was to bring together experts from several areas of computer science, we

realized that for the size of the meeting to stay within reason, as well as for exchanges to be feasible and bear fruit, it would be wise to limit ourselves to a restricted notion of equations and algebraic structures—namely, those relevant to symbolic computation and the foundation of programming. In May of 1987, under the generous sponsorship of MCC in Austin, Texas, and some partial contribution from INRIA in Paris, France, the Colloquium on Resolution of Equations in Algebraic Structures (CREAS) took place in Lakeway, Texas, with thirty outstanding participants from Australia, France, Germany, Italy, and the United States. For three overloaded days, participants exposed and exchanged a wealth of results, ideas, and prospects.

The meeting was unanimously judged to be a success. Therefore, it became our duty as organizers to edit a book containing the highlights of the contributions to CREAS, asking each willing participant to write a special original piece. Gathering twenty-two contributions from some of the best researchers working in equation solving, the book would be meant as a reference, a compendium of results, methods, algorithms, and state of the art in resolution of equations in algebraic structures. It would be destined to go on the shelves of computer science libraries, university departments, and research laboratories, as well as to serve as a fine textbook for graduate students coming to this field of research.

The resulting collection of papers came to a size too large to fit in one volume. Therefore, it was decided to divide it into two volumes—one of which you are holding in your hands. The split was made easy, as a conceptual line appeared naturally that created a balanced partition between algebraic techniques and, more specifically, rewriting techniques. Thus, although both volumes share the common title *Resolution of Equations in Algebraic Structures*, Volume 1 is subtitled *Algebraic Techniques* and Volume 2 is subtitled *Rewriting Techniques*. More than just a convenient coincidence, the separation of the specific area of rewriting techniques for equational problems translates the historical fact that a great deal has happened lately in this field, which has made it one of the most active fields of research in symbolic computation. Thus it is only natural to devote a well-delineated volume to these among all equation-resolution techniques.

Each volume's contents are organized in alphabetical order by first authors. In each, an introduction reviews each contribution in order

of appearance, giving an informal summary of the work and results reported. The purpose of these reviews is to serve as a quick first-pass reference to the reader, putting the contribution in context, stressing the significance of the work.

Finally, we would like to express some thoughts of acknowledgement. We are, of course, indebted to the Advanced Computer Architecture program of MCC for their generous financial help for organizing CREAS, as well as for providing their efficient logistic support before, during, and after the colloquium. In particular, CREAS would not have been without the encouragement and support of Woody Bledsoe and Bob Boyer. Many thanks also to INRIA for agreeing to pay for the travel of their participating researchers. We owe Denise White (the charming and ever-smiling MCC coordinator of CREAS) a great deal for taking care of every detail and person without ever pulling out a shotgun. We were lucky to find at Lakeway Inn in Lakeway, Texas, two extremely qualified and helpful professionals, Jo Ann Freeman and Lisa Parker, who provided a most propitious environment for CREAS. Naturally, we are especially grateful to the contributors of this volume for gracefully taking the (short) time to write and polish their articles, and for bending to our editorial whims. Finally, we acknowledge the kind help and assistance of Sari Kalin and Alice Peters, of Academic Press Boston. In addition, the first editor would like to thank Carlo Zaniolo, director of the Languages Group of MCC's ACA Systems Technology, and the members of the LIFE project for their understanding and patience during the interminable time he spent editing these volumes. Most of all, he needs to express his guilty gratitude to his loving and patient family for having foolishly stuck his foot into an unexpected vortex of never-ending obligations.

We wish you, the reader, a pleasant time sharing the excitement felt by all who attended CREAS, and hope that we have contributed in giving you the means and tools to pursue your own research, solving equations in algebraic structures.

Hassan Aït-Kaci
Maurice Nivat

This page intentionally left blank

A Preview of Volume 2: Rewriting Techniques

In *Completion Without Failure*, Bachmair, Dershowitz, and Plaisted present an elaboration of the Knuth-Bendix completion method. Whereas general completion attempts to construct a complete (i.e., Church-Rosser and terminating) set of reductions from a set of equational axioms, their *unfailing completion* purports to find a *ground* Church-Rosser system, in which every ground term has a unique normal form. Ground confluence is not as restricted as it sounds, as it is sufficient for most applications, including theorem proving. Building on Bachmair's dissertation work, they view completion as a *proof simplification* process, and derive a ground Church-Rosser system. They show in addition that it is refutationally complete for theories of Horn clauses with equality.

In *Completion and Its Applications*, Nachum Dershowitz gives a detailed tutorial of the Knuth-Bendix completion method which generates a set of confluent and finite terminating set of reduction rules, if such exists, from a finite set of equations. This method is perhaps the single most important result in equational theorem proving, as it allows for the testing of the validity of an equation by reducing each side to normal form and comparing them. Following work done with Bachmair, Dershowitz gives an abstract characterization of the method seen as a six-rule inference system. This is not only elegant and concise, but also makes it possible to regard it as a proof simplification system, as in transitional proof theory. Extensions of the method dealing with termination, unfailing completion, and associative-communicative completion are discussed. Most interestingly, Dershowitz surveys a wealth of applications of completion, touching congruence closure, meta-unification, program synthesis,

"inductionless" induction, and theorem proving. This contribution is a wonderful technical, albeit highly readable, introduction to the Knuth-Bendix completion method.

In *Extending Equation Solving and Constraint Handling in Logic Programming*, Dincbas, Simonis, and van Hentenryck discuss various practical techniques that may be used to extend Prolog as a constraint solver. Prolog, in practice, may be regarded as consisting of three computational features: relation composition, backtracking, and first-order term unification. The latter operation enforces uninterpreted equality constraints on terms. As it lately has become clear, the mechanisms of Prolog that work with first-order terms (Herbrand Universe) and unification may be adapted to richer structures and type on constraints as long as an effective and efficient procedure for solving these constraints is provided. This theme is that of *constraint logic programming*, a powerful paradigm that emerges as most promising for declarative programming and executable specifications. In this particular contribution, Dincbas, Simonis, and van Hentenryck focus on enhancing Prolog with techniques solving boolean equations, imposing inequality, disequality, and finitary domain constraints on variables, as well as look-ahead trimming of the search space. They give several examples to illustrate the methods.

In *Proofs by Combinatory Induction on Recursively Reducible Expressions*, Fribourg introduces a new principle of induction based on combinatory logic. This principle is tailored to detect recursively reducible expressions defined by equational axioms. This is achieved by using equality theories induced by the familiar $\mathbf{S}, \mathbf{K}, \mathbf{I}, \mathbf{C}, \mathbf{B}$ combinators and introducing two new ones: a *recursion* combinator $\mathbf{R}'$ and an *iteration* combinator $\mathbf{J}$. It is shown that combinatory induction is equivalent to "fertilizing" induction for equational reasoning, the process by which an instance of one of the sides of the induction hypothesis equation is recognized and replaced by the corresponding other side in the hypothesis, the latter being thereafter discarded. Many examples are treated in detail with a special treatment of linear lists. The combinatory-inductive proofs are in general shorter than proofs by conventional structural induction, and they are *modular* in that they are easily decomposable into independent lemmas.

In *Completion Algorithms for Conditional Rewriting Systems*, Kaplan and Rémy tackle the difficult problem of term rewriting with conditional guards. Such systems are extremely useful in that they

are more general than plain term rewriting systems, which cannot discriminate among candidates to be rewritten other than by first-order term matching. Allowing boolean preconditions as guards of some rewrite rules thus achieves greater expressiveness, albeit at the price of more difficult theory. Kaplan and Rémy introduce the necessary formal notions and survey in a uniform way key results they previously reported in disseminated publications. In particular, they give a necessary and sufficient condition for confluence of conditional rewriting, and a practical unification algorithm modulo conditional rewriting by introducing the notion of *conditional narrowing*. A Knuth-Bendix completion method also is described, which deals with conditions using a notion of *contextual rewriting*, a technique by which conditional rewriting carries preconditions down as a conjunctive context. They also define a *hierarchical system* as the pair of a nonconditional rewriting system and a conditional one. Intuitively, the former is used to evaluate the conditions of the latter. Thus, they define the notion of *hierarchical rewriting* by extending the conventional one to specify that the rewriting substitution maps conditional variables in terms of the nonconditional algebra, and the condition reduces to *true*. This contribution is a clear and rigorous introduction to conditional rewriting and describes practical tools for solving conditional equations.

In *From Unification in Combination of Equational Theories to a New AC-Unification Algorithm*, Kirchner expounds a general method for equation solving in equational algebras. The process of E-unification is extrapolated form the classical term unification method developed by Martelli and Montanari. The latter method consists of a transformation system manipulating multi-equations that may be viewed as congruence classes of terms. Transformations consist of two operations: *decomposition* and *merging*. Decomposition creates new multi-equations by equating corresponding subterms of terms of a congruence class. Merging coalesces multi-equations that share equal variable terms. This proceeds nondeterministically until either a clash of function symbol occurs in a multi-equation, or no operation can apply, at which point all multi-equations are in normal form, i.e., contain each at most one nonvariable term. Kirchner generalizes this technique to E-unification by extending the notion of systems of multi-equations to *disjunction systems* corresponding to alternative systems of multi-equations, and introducing a third

transformation—an operation called *mutation* which takes the theory E into account. Mutation allows transforming a system of multi-equations into a (possibly disjunctive) system which preserves all solutions. Of course, this operation depends on the particular equational theory being considered, and may not exist for arbitrary ones. (The general problem of arbitrary E-unification is indeed undecidable.) Although general and complete equation-solving techniques like *narrowing* may be used in many cases, special purpose mutations for particular theories, when they exist, are more useful, as they can exploit idiosyncratic aspects of the axioms. In particular, Kirchner gives a complete mutation for associative-communicative unification. Besides being simple and general, Kirchner's method allows the clean combination of equational theories in the case where the subtheories form a partition (i.e., do not share function symbols) and do not contain collapsing equations of the form $t = x$ (where t is nonvariable). For those collapse-free partitioned theories, a combined unification algorithm is obtained from the unification algorithms of the subtheories. This contribution is crisp reading, offering rigorous and practical tools for general equation solving in equational varieties.

In *Inductive Completion by Ground Proof Transformation*, Küchlin studies the possibility of using completion for inductive proofs in the initial algebra of an equational variety without explicit induction. Indeed, first-order equational reasoning is not sufficient in general to prove the validity problem for ground terms that may require induction—a second-order axiom. However, as was originally remarked by Musser and further expounded by many others, the Knuth-Bendix completion method may be used to prove a universally quantified theorem $s = t$, given a theory $\mathcal{A}$ that may otherwise require induction. The technique, known as *inductionless induction*, relies on searching for a counter-example of the theorem by attempting completion of $\mathcal{A} \cup \{s = t\}$. If the theorem is valid in the initial algebra, no rule should be generated in the process. A test was devised by Jouannaud and Kounalis called *inductive reducibility* which allows one still to conclude validity of the theorem if the rules introduced by the completion process have a left-hand side that does not alter ground normal forms. The test is an overkill, as it implies attempting *all* possible induction schemas and thus fails to terminate should only one of these inductions diverge, even if it were not

necessary to the proof. Fribourg sharpened this test, remarking that for inductive proofs it is sufficient to use and possible to detect only those positions that lead to the needed proof. He devised a purely syntactic test for this—*complete superposability*—which amounts to checking only for *ground* confluence, as opposed to general confluence as proposed by Jouannaud and Kounalis. Küchlin, on the other hand, developed a general test for ground confluence in investigating ground completion. This contribution relates Fribourg's test to ground completion, thereby inheriting a complete correctness proof for inductive completion. Küchlin extends and simplifies Fribourg's theory. The outcome of his work is a uniform proof procedure for the decidability of an equational theory, the validity problem in all models (general), and in the initial model (ground).

In *Lazy Unification Algorithms for Canonical Rewrite Systems*, Martelli, Rossi, and Moiso present practical algorithms for unification modulo an equational theory. In the case where the equational theory is represented by a canonical set of rewrite rules (i.e., confluent and finite terminating), they extend the well-known nondeterministic scheme introduced by Martelli and Montanari for syntactic unification (i.e., with an empty theory) to express a *narrowing—semantic unification* modulo term rewriting—as a nondeterministic process of elementary transformations of equations. From that, they derive a *lazy* narrowing algorithm that rewrites only by need, using an almost outermost strategy. Further, they remark that by viewing Horn clauses as rewrite rules, this equation-solving procedure becomes a complete interpreter for Horn clause logic, where functional expressions are also accommodated. As a result, they obtain a first-order language for logic programming *cum* functional programming. They give many examples and express their algorithms in Prolog, making them readily implementable. Comparison with other similar algorithms is also discussed. This contribution is a clear and useful one which will benefit the reader eager to experiment with logic and equational programming.

In *Equations in Words*, Perrin surveys important results pertaining to solving systems of word equations in the free monoid and the free group. This notion of equation solving is more general than that used in term-rewriting proof theory, as a solution is defined as a word homomorphism rather than a substitution. He reviews the main definitions and theorems, then presents algorithms—the