Book Reviews

K. O. Geddes, S. R. Czapor and G. Labahn
Algorithms for Computer Algebra. Kluwer Academic Publishers, 1992, 585 pp hardbound, ISBN 0-7923-9259-0

While in the past computers were used mainly to perform numerical computations, nowadays the problems of precise and symbolic computations have become more common. Although there are many systems able to manipulate symbolic mathematical objects, the basic ideas behind these systems are often similar. This book gives a survey of basic algorithms used in algebraic computation. The book can be used as a textbook for computer algebra courses at either an advanced undergraduate or a graduate level.

The first chapter explains requirements for systems performing symbolic computations and shows possible ways of exploitation of such systems. The second chapter introduces basic notions of general algebra ranging from rings and fields to power series. The third chapter describes ways of representing basic symbolic objects. Various normal forms for multiprecision numbers, polynomials, rational functions and power series are described and their advantages and disadvantages compared. Chapter 4 deals with basic operations over polynomials, rational functions and power series. The Chinese remainder theorem and related algebraic notions are presented in Chapter 5 and algorithms for polynomial interpolation and inverting homomorphisms based on the Chinese remainder theorem are presented. Chapter 6 deals with Hensel construction and its modifications. It also shows how Hensel construction can substitute for the Chinese remainder algorithm and notably improve the performance of algorithms.

Chapter 7 concentrates on polynomial greatest common divisor computation. The greatest common divisor algorithm is often used by other computer algebra algorithms and therefore it is not surprising that this problem is studied in detail. Polynomial factorization is another problem that is often encountered in computer algebra. The famous Berlecamp factorization algorithms and its improvements are studied in Chapter 8. Chapter 9 gives algorithms for solving systems of equations; both linear and nonlinear systems are considered. A different approach to solve systems of nonlinear equations is given in Chapter 10 where Buchberger's algorithm for transforming an arbitrary ideal basis into a Gröbner basis and applications of this algorithm are presented.

The last two chapters are dedicated to symbolic computing of indefinite integrals. While Chapter 11 covers an integration of rational functions, Chapter 12 describes the Risch algorithm for integrating elementary functions.

The algorithms in the book are written in an easy-to-read Pascal-like language and are accompanied by suitable examples. The background of algebra required is covered and mathematical results that form the basis for the algorithms are precisely proved. Every chapter of the book is followed by a set of suitable exercises. Unfortunately, the analysis of the complexity of the algorithms is not given and this makes comparing the efficiency of algorithms hard.

Since there are still not many books on the subject of computer algebra, this book should be considered by everyone with an interest in symbolic computation.

Vlado Danč´k
Warwick University

D. A. Wolfram
The Clausal Theory of Types. Cambridge University Press, 1993, 124 pp hardbound, ISBN 0-521-39538-0

The Skolem–Herbrand–Godel theorem for first order logic (SHG Theorem, for short), the resolution principle and the effectiveness of unification together constitute the theoretical foundation for automated theorem-proving in first order logic and logic programming. This book attempts to provide a similar foundation for a higher order logic called the Clausal Theory of Types (CTT for short) defined by the author. Higher order analogues of the SHG theorem and resolution principle are proved and the equational unification of CTT terms with built-in equational theories is discussed in the book.

In what sense is CTT a higher order logic? It allows for embeddable predicates and enriches first order logic in the following ways: the terms may include λ-abstractions, the formulas include equality between terms and quantification can be over variables of propositional type. CTT is obtained as a restriction of Church's Simple Theory of Types (the restriction being for technical reasons).

The organization of this slim volume (105 pages of main text and 18 pages of bibliography and index) is as follows. Chapter 1 (14 pages) traces the development of logic programming to the SHG theorem. Chapter 2 (9 pages) defines the simply typed λ-calculus, α-conversion, β, η-reductions and normal forms. Chapter 3 (18 pages) defines CTT, its models and proves the SHG theorem for CTT. Chapter 4 (44 pages) is an extensive treatment of higher order unification theory. Chapter 5 (17 pages) proves the resolution theorem for CTT and shows the least model/least fixed-point semantics and a breadth-first procedure as operational semantics for CTT based logic programs.

I have specified the textual length of each chapter because it corresponds well with the depth of treatment accorded to the respective topics. Chapter 2 is more a set of definitions needed later on than a sketch of λ-

calculus. Chapters 3 and 5 systematically confirm our expectations about the higher order extension, working out exactly as much detail as required for that.

The real 'meat' of the book lies in Chapter 4. The author first proves the completeness of a procedure for equational unification of CTT terms and shows that the problem is undecidable. He then considers higher order unifiability and pre-unifiability; these are again undecidable, but he discusses heuristics. He moves on to higher order matching (given two terms t and t' of the same type, is there a substitution γ such that $t' = \gamma t$?). This has been conjectured by G. Huet to be decidable, but the problem remains open. The author presents extensive evidence which suggests a positive result for CTT terms: he presents a terminating procedure for matching conjectured to be complete; he also discusses other approaches like the Plotkin–Statman conjecture (relating this problem to that of deciding λ-definability), some NP-hard third order matching problems, Zaionc's (1985) idea of regular unification and second-order monadic unification. Thus this chapter also constitutes an excellent survey on this subject.

The bibliography, containing 204 reference items, is quite comprehensive and ought to be of great value to researchers in this area. But excessive referencing can also be obtrusive: do we need a reference to Huet's thesis (written in French) to support a claim that composition of substitutions is associative (page 25)?

My major criticism of the book is that it is *not* self-contained. While every definition needed in any proof can be found in the book, other books and papers are needed to make sense of the material here. Andrews' book (Andrew, 1986) is a pre-requisite for understanding Chapters 2 and 3, and the important ideas of Chapter 4 need recourse to Huet's papers and Statman (1982) for a complete understanding. Material from these sources could easily have been included to add substance to this slim volume.

Further, there are very few examples. The book has 145 definitions as opposed to 24 examples, most of which highlight specific definitions used in proofs rather than illustrate concepts. Given that the book is about extending results from first order logic to a higher order one, we can reasonably expect examples of CTT formulas with specific higher order features and instances of reasoning in this logic. Indeed, in the entire book there is only *one* example giving a CTT formula (example 3.27, showing its conversion to normal form). All the rest illustrate local technical details (subsets of $\sim$ relation, applying the MATCH procedure to nodes, etc.). Such a purely formal treatment, and the lack of proofs of base material (strong normalization theorem for λ-calculus, the many theorems of Huet and Statman quoted in the book) do not make for pedagogic use. This book is more in the nature of an extended research article on CTT meant for researchers in automated theorem proving and logic programming, rather than a textbook for a graduate course (as claimed on the back cover).

The production of the book is excellent. There are very few mistakes. (Page 55, line 5 from bottom, 'equational unification' should be 'pre-unification'.)

REFERENCES

Andrews, P. B. (1986) *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof.* Academic Press, Orlando.

Statman, R. (1982) Completeness, invariance, and λ-definability. *J. Symbolic Logic*, **47**, 17–26.

Zaionc, M. (1985) The set of unifiers in typed λ-calculus as regular expression. In *Rewriting Techniques and Applications, Lecture Notes in Computer Science 202.* Springer, Berlin.

R. RAMANUJAM
The Institute of Mathematical Sciences, Madras

GÉRARD HUET AND GORDON PLOTKIN (editors)
Logical Environments. Cambridge University Press, 1993, £35.00. 338 pp hardbound, ISBN 0-521-43312-6

This book is a collection of papers about computer systems that provide facilities for the development of formal (i.e. mathematical) proofs about hardware and software, as well as about mathematical systems proper. The concept of a logical environment can be construed in the above sense (i.e. as a software system composed of algorithms and representations) or as a formal (logical) theory within which such reasoning can take place. When implemented, the environment contains, in addition to theorem-proving software, databases of axioms and of theorems: typically, users can add to the collection of theorems available within a system.

The book reflects many of the issues raised by logical environments. It contains a section on the general concept of a logical framework and the issues raised by them, a section on the algorithms needed to support environments, a section on foundational issues, and a final section describing experiments that have been performed using two such environments (LEGO from Edinburgh and ALF from Göteborg).

The general theoretical viewpoint of logical environments is constructive logic. That is, the logics typically used and/or researched are of a constructive nature: a proof of a property is interpreted as a construction of an (abstract) object with the required property. This interpretation is clearly very close to the concept of an algorithm: proofs are ways of building things. The theory of constructive systems is an active research area, and this collection contains chapters on the foundations of such logics.

The field may be dominated by the constructivist approach, but the chapter by Matthews, Smaill and Basin outlines their work of a *classical* system due to Feferman called FS_0. In common with other work in this area (that of Constable *et al.*, and the work on the Edinburgh LF), work on FS_0 is aimed at producing a framework within which to construct other logics. The kinds of logic used to provide a framework are examples