J. Functional Programming **7** (6): 667–668, November 1997. Printed in the United Kingdom

Book review

Foundations for Programming Languages by John C. Mitchell, MIT Press, 1996.

This book presents a framework for the analysis of syntactic, operational and semantic properties of programming languages. The framework is based on a mathematical system called typed lambda calculus. The main features of lambda calculus are a notation for functions and other computable values, together with an equational logic and rules for evaluating expressions. The book is organized around a sequence of lambda calculi with progressively more complicated type systems. These are used to analyze and discuss relevant programming language concepts. The emphasis is on sequential languages, although many of the techniques and concepts also apply to concurrent programming languages.

The simplest system in the book is an equational system sometimes called universal algebra. This logic without function variables may be used to axiomatize and analyze many of the data types commonly used in programming. The next system is a lambda calculus with function types and, optionally, cartesian products and disjoint unions. When enriched with recursive definitions, this language provides a useful framework for studying operational and semantic properties of functional programs. When combined with algebraic data types, this system is adequate to define many conventional programming languages. In particular, with types for memory locations and stores, we may study traditional axiomatic, operational and denotational semantics of imperative programs. More advanced technical machinery, such as the method of logical relations, category theory, and the semantics of recursively defined types are covered in the middle chapters. The last three chapters of the book study polymorphic types, along with declaration forms for abstract data types and program modules, systems of subtyping, and type inference.

The book is written for upper-year undergraduates or beginning graduate students specializing in theoretical computer science, software systems, or mathematics. It is also suitable for advanced study or technical reference.

This is a well-written and technically sound presentation of a broad range of material, much of it of interest to functional programming enthusiasts. Various existing works such as (Loeckx and Sieber, 1984; Tennent, 1991; Gunter, 1992; Winskel, 1993) cover much of the same ground, as will the multivolume series (Abramsky *et al*., 1992), but I know of no single volume that is as encyclopedic in coverage. And although the book is about 850 pages long, it is being sold at a quite attractive price (£35, $60).

The preface suggests the book can be used as the basis for undergraduate courses and a suggested 'introductory' course is outlined. But this uses material selected from the first six chapters only, less than half of the book, and the remaining material is definitely at an advanced level. I expect that this book will prove to be more useful for advanced study by researchers and graduate students than as an undergraduate textbook.

Inevitably, there are important topics that are not treated: communicating processes, continuations, powerdomains, stable functions. But most of the topics that are treated are discussed very thoroughly. The index and bibliography are excellent, and there appear to be very few typos.

In summary, this is a valuable and important book that will be useful to many researchers and graduate students; however, I have reservations on its suitability as an undergraduate text. Perhaps the publisher should consider re-packaging a selection from the first six chapters to provide a more suitable volume for this purpose.

References

Abramsky, S., Gabbay, D. M. and Maibaum, T. S. E. (eds.) (1922) *Handbook of Logic in Computer Science*. Vols. 1–6. Oxford University Press.

Gunter, C. A. (1992) *Semantics of Programming Languages: Structures and Techniques*. MIT Press.

Loeckx, J. and Sieber, K. (1984) *The Foundations of Program Verification*. John Wiley & Sons/Teubner.

Tennent, R. D. (1991) *Semantics of Programming Languages*. International Series in Computer Science. Prentice-Hall.

Winskel, G. (1993) *The Formal Semantics of Programming Languages: An Introduction*. MIT Press.

R. D. Tennent

https://doi.org/10.1017/S0956796897002918 Published online by Cambridge University Press