Handbook of Logic in Computer Science

Volume 1
Background: Mathematical Structures

Edited by
S. ABRAMSKY, DOV M. GABBAY,
and T. S. E. MAIBAUM

OXFORD SCIENCE PUBLICATIONS

ISBN 0 19 853735 2

Handbook of Logic in Computer Science

Volume 1

Background: Mathematical Structures

Edited by

S. ABRAMSKY
Professor of Computing Science

DOV M. GABBAY
Professor of Computing Science

and

T. S. E. MAIBAUM
Professor of Foundations of Software Engineering

Imperial College of Science, Technology and Medicine
London

Volume Co-ordinator

DOV M. GABBAY

CLARENDON PRESS · OXFORD
1992

Oxford University Press, Walton Street, Oxford OX2 6DP
Oxford New York Toronto
Delhi Bombay Calcutta Madras Karachi
Kuala Lumpur Singapore Hong Kong Tokyo
Nairobi Dar es Salaam Cape Town
Melbourne Auckland Madrid
and associated companies in
Berlin Ibadan

Oxford is a trade mark of Oxford University Press

Published in the United States
by Oxford University Press Inc., New York

A catalogue record for this book is available from the British Library

Library of Congress Cataloging in Publication Data
(Data available on request)

ISBN 0–19–853735–2

Printed in Great Britain by
Biddles Ltd,
Guildford and King's Lynn

HANDBOOK OF LOGIC IN COMPUTER SCIENCE

Editors

S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum

HANDBOOKS OF LOGIC IN COMPUTER SCIENCE
and
ARTIFICIAL INTELLIGENCE AND LOGIC PROGRAMMING

Handbook of Logic in Computer Science

Volume 1 Background: Mathematical structures
Volume 2 Background: Computational structures
Volume 3 Semantic structures
Volume 4 Semantic modelling
Volume 5 Theoretical methods in specification and verification
Volume 6 Logical methods in computer science

Handbook of Logic in Artificial Intelligence and Logic Programming

Volume 1 Logical foundations
Volume 2 Deduction methodologies
Volume 3 Nonmonotonic reasoning and uncertain reasoning
Volume 4 Epistemic and temporal reasoning
Volume 5 Logic programming

Preface

We are happy to present the first volumes of the *Handbook of Logic in Computer Science*. Logic is now widely recognized to be one of the foundational disciplines of computing and has found applications in virtually all aspects of the subject, from software engineering and hardware to programming language and artificial intelligence. There is a growing need for an in-depth survey of the application of logic in computer science and AI. The *Handbook of Logic in Computer Science* and its companion, the *Handbook of Logic in Artificial Intelligence and Logic Programming* have been created in response to this need.

We see the creation of the Handbook as a combination of authoritative exposition, comprehensive survey, and fundamental research exploring the underlying unifying themes in the various areas. The intended audience is graduate students and researchers in the areas of computing and logic, as well as other people interested in the subject. We assume as background some mathematical sophistication. Much of the material will also be of interest to logicians and mathematicians.

The tables of contents of the volumes were finalized after extensive discussions between handbook authors and second readers. The first two volumes present the background logic and mathematics extensively used in computer science. The point of view is application oriented. The other four volumes present major areas in which the methods are used. These include Volume 3 — Semantic Structures; Volume 4 — Semantic Modelling; Volume 5 — Specification and Verification; and Volume 6 — Logical Methods.

The chapters, which in many cases are of monographic length and scope, are written with emphasis on possible unifying themes. The chapters have an overview, introduction, and a main body. A final part is dedicated to more specialized topics.

Chapters are written by internationally renowned researchers in the respective areas. The chapters are co-ordinated and their contents were discussed in joint meetings.

Each chapter has been written using the following procedures:

1. A very detailed table of contents was discussed and co-ordinated at several meetings between authors and editors of related chapters. The discussion was in the form of a series of lectures by the authors to everyone present. Once an agreement was reached on the detailed table of contents the authors wrote a draft and sent it to the editors and to other related authors. For each chapter there is a second reader (the first reader is the author) whose job it has been to scrutinize the

chapter together with the editors. The second reader's role is very important and has required effort and serious involvement with the authors.
Second readers for this volume are:

Chapter 1: Valuation Systems and Consequence Relations — M. Fourman
Chapter 2: Recursion Theory — M. Hennessy and J. Tucker
Chapter 3: Universal Algebra — A. Poigne and M. Fourman
Chapter 4: Category Theory — E. Wagner and M. Fourman
Chapter 5: Topology — H. Barendregt
Chapter 6: Model Theory — I. Hodkinson and D. Gabbay.

2. Once this process was completed (i.e. drafts seen and read by a large enough group of authors), there were other meetings on several chapters in which authors lectured on their chapters and faced the criticism of the editors and audience. The final drafts were prepared after these meetings.

3. We attached great importance to group effort and co-ordination in the writing of chapters. The first two parts of each chapter, namely the Introduction–Overview and Main Body, are not completely under the discretion of the author, as he/she had to face the general criticism of all the other authors. Only the third part of the chapter is entirely for the authors' own tastes and preferences.

The Handbook meetings were generously financed by OUP, by SERC contract SO/809/86, by the Department of Computing at Imperial College, and by several anonymous private donations.

We would like to thank our colleagues, authors, second readers, and students for their effort and professionalism in producing the manuscripts for the Handbook. We would particularly like to thank the staff of OUP for their continued and enthusiastic support, and Mrs Jane Spurr, our OUP Administrator for her dedication and efficiency.

London The Editors
June 1992

Contents

Valuation Systems and Consequence Relations

Mark Ryan and Martin Sadler

Contents

1 Introduction

This chapter is an introduction to some of the basic concepts and machinery of logic, concentrating on the contemporary uses of logic in computer science. It is not comprehensive, since many topics (such as equational reasoning, algebras, categories and computability) are introduced in other chapters. In particular, we concentrate on the applications of logics as reasoning systems about computing rather than on the foundational aspects of computer science which logic addresses. This is not a standard introduction to classical first order logic and its model theory, such treatments being widely available elsewhere [Hamilton, 1978; Hodges, 1983; Makowsky,]. Rather, we attempt to bring together some topics from the logics currently used for reasoning about programs and about computer systems, and see them within a unifying framework at an appropriate level of abstraction.

Prerequisites. Most of the constructions in this chapter are based on 'naïve set theory', whose use we take for granted. We also assume some exposure to the use and manipulation of the usual propositional operators ($\wedge$, $\rightarrow$, $\vee$, $\neg$), and to predicates. Some familiarity with discrete mathematics, including the use of λ-notation, and the idea of proofs both *within* and *about* a formal system would be helpful.

1.1 Logics and computer science

It has been recognized for a long time that there is an intimate connection between logic and computer science:

> 'It is reasonable to hope that the relationship between computation and mathematical logic will be as fruitful in the next century as that between analysis and physics in the last. The development of this relationship demands a concern for both applications and mathematical elegance.' (J. McCarthy, 1965)

We begin this chapter by describing some aspects of the interconnections between logic and computer science. We will introduce the machinery of logic later, here we rely on the reader's intuitions about *truth*, *proofs* and *propositions.*

First we can distinguish between a 'foundational' aspect and a 'reasoning system' aspect of the connection. At the foundational aspect, logic is used to provide a model of the phenomenon of computation. For example: the *Curry–Howard isomorphism* is a description of the parallel between computation and proof transformation in intuitionistic logic. An idealized view of computation is adopted: it is taken to mean reducing typed lambda

terms to normal form. We can justify this idealization by appealing to the formal power of the typed λ-calculus and of Turing machines; see for example [Barendregt,] and [Phillips,] in this volume.

The key idea is the identification of *propositions* with *types*, and *proofs* with *terms*. A proposition is a type whose terms are the proofs of the proposition. This programme is called *intuitionistic type theory*, and started with P. Martin-Löf [Martin-Löf, 1975]. Reducing lambda terms to normal form (which is our model of computation) is the same as eliminating redundant steps in proofs. We return to this topic briefly later on.

Girard's *Linear logic* is a further step in this foundational direction. It abandons certain structural properties present in both classical and intuitionistic logic and by doing so provides a new approach to many issues in functional programming, such as lazy evaluation, side-effects, memory allocation and parallelism.

This is not, however, the level of interconnections between logic and computer science with which we are primarily concerned in this chapter; the interested reader should see other chapters in this Handbook. Instead, we concentrate on *logics as reasoning systems*—on their claim to be systems for making deductions from premises. Logic is used for describing and implementing systems which reason about a particular domain (e.g. in *specification theory* and *artificial intelligence*). Temporal logic is used to reason about domains in which time plays a key role, deontic logic for domains involving normative behaviour, default logic when only partial information is available, and so on. Specification theory provides many examples of this relationship between logic and computer science. The expressions in specification languages usually denote logical theories, often based on *multi-modal logics* (see [Stirling,] in this volume). Multi-modal logics have also been used to reason about concurrent systems and non-deterministic systems.

Already in this chapter we have referred to several different 'logics': intuitionistic logic, temporal logic, linear logic, deontic logic and others. There is a recurrent debate between *absolutism* and *pluralism*, whether there is 'one true logic' for reasoning about all domains or whether different logics should be studied for different domains. We attempt in this chapter to argue that computer science demands a pluralist view.

It is often stated that classical logic (the logic which is usually taken to be the 'one true logic' by absolutists) is capable of expressing arguments phrased in any 'non-standard' logic, and so removes the need for studying them. Indeed, we shall see in the last section that many logics can be embedded in classical logic in a precise sense. But to argue along these lines misses an important point. As V. Pratt says,

> 'There is a tradition in logic, carried over into computer science, to think of pure first order logic as a universal language. In fact