LOGIC IN COMPUTER SCIENCE

Modelling and Reasoning about Systems

LOGIC IN COMPUTER SCIENCE

Modelling and Reasoning about Systems

MICHAEL HUTH
Department of Computing
Imperial College London, United Kingdom

MARK RYAN
School of Computer Science
University of Birmingham, United Kingdom

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

Published in the United States of America by Cambridge University Press, New York

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of
education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9780521543101

First published 2004
13th printing 2015

Printed in the United States of America by Sheridan Books, Inc.

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

ISBN 978-0-521-54310-1 Paperback

Contents

Cambridge University Press
978-0-521-54310-1 - Logic in Computer Science: Modelling and Reasoning about Systems
Michael Huth and Mark Ryan
Frontmatter
More information

Foreword to the first edition

by

Edmund M. Clarke
FORE Systems Professor of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Formal methods have finally come of age! Specification languages, theorem provers, and model checkers are beginning to be used routinely in industry. Mathematical logic is basic to all of these techniques. Until now textbooks on logic for computer scientists have not kept pace with the development of tools for hardware and software specification and verification. For example, in spite of the success of model checking in verifying sequential circuit designs and communication protocols, until now I did not know of a single text, suitable for undergraduate and beginning graduate students, that attempts to explain how this technique works. As a result, this material is rarely taught to computer scientists and electrical engineers who will need to use it as part of their jobs in the near future. Instead, engineers avoid using formal methods in situations where the methods would be of genuine benefit or complain that the concepts and notation used by the tools are complicated and unnatural. This is unfortunate since the underlying mathematics is generally quite simple, certainly no more difficult than the concepts from mathematical analysis that every calculus student is expected to learn.

Logic in Computer Science by Huth and Ryan is an exceptional book. I was amazed when I looked through it for the first time. In addition to propositional and predicate logic, it has a particularly thorough treatment of temporal logic and model checking. In fact, the book is quite remarkable in how much of this material it is able to cover: linear and branching time temporal logic, explicit state model checking, fairness, the basic fixpoint theorems for computation tree logic (CTL), even binary decision diagrams and symbolic model checking. Moreover, this material is presented at a level that is accessible to undergraduate and beginning graduate students. Numerous problems and examples are provided to help students master the material in the book. Since both Huth and Ryan are active researchers in logics of programs and program verification, they write with considerable authority.

In summary, the material in this book is up-to-date, practical, and elegantly presented. The book is a wonderful example of what a modern text on logic for computer science should be like. I recommend it to the reader with greatest enthusiasm and predict that the book will be an enormous success.

(This foreword is re-printed in the second edition with its author's permission.)

Preface to the second edition

Our motivation for (re)writing this book

One of the leitmotifs of writing the first edition of our book was the observation that most logics used in the design, specification and verification of computer systems fundamentally deal with a *satisfaction relation*

$$\mathcal{M} \vDash \phi$$

where $\mathcal{M}$ is some sort of *situation* or *model* of a system, and ϕ is a specification, a formula of that logic, expressing what should be true in situation $\mathcal{M}$. At the heart of this set-up is that one can often specify and implement algorithms for computing $\vDash$. We developed this theme for propositional, first-order, temporal, modal, and program logics. Based on the encouraging feedback received from five continents we are pleased to hereby present the second edition of this text which means to preserve and improve on the original intent of the first edition.

What's new and what's gone

Chapter 1 now discusses the design, correctness, and complexity of a SAT solver (a marking algorithm similar to Stålmarck's method [SS90]) for full propositional logic.

Chapter 2 now contains basic results from model theory (Compactness Theorem and Löwenheim–Skolem Theorem); a section on the transitive closure and the expressiveness of existential and universal second-order logic; and a section on the use of the object modelling language Alloy and its analyser for specifying and exploring under-specified first-order logic models with respect to properties written in first-order logic with transitive closure. The Alloy language is executable which makes such exploration interactive and formal.

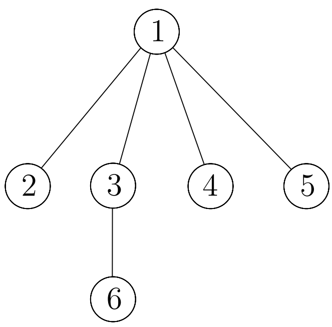
Chapter 3 has been completely restructured. It now begins with a discussion of linear-time temporal logic; features the open-source NuSMV model-checking tool throughout; and includes a discussion on planning problems, more material on the expressiveness of temporal logics, and new modelling examples.

Chapter 4 contains more material on total correctness proofs and a new section on the programming-by-contract paradigm of verifying program correctness.

Chapters 5 and 6 have also been revised, with many small alterations and corrections.

The interdependence of chapters and prerequisites

The book requires that students know the basics of elementary arithmetic and naive set theoretic concepts and notation. The core material of Chapter 1 (everything except Sections 1.4.3 to 1.6.2) is essential for all of the chapters that follow. Other than that, only Chapter 6 depends on Chapter 3 and a basic understanding of the static scoping rules covered in Chapter 2 – although one may easily cover Sections 6.1 and 6.2 without having done Chapter 3 at all. Roughly, the interdependence diagram of chapters is



WWW page

This book is supported by a Web page, which contains a list of errata; text files for all the program code; ancillary technical material and links; all the figures; an interactive tutor based on multiple-choice questions; and details of how instructors can obtain the solutions to exercises in this book which are marked with a $*$. The URL for the book's page is `www.cs.bham.ac.uk/research/lics/`. See also `www.cambridge.org/052154310x`

Acknowledgements

Many people have, directly or indirectly, assisted us in writing this book. David Schmidt kindly provided several exercises for Chapter 4. Krysia Broda has pointed out some typographical errors and she and the other authors of [BEKV94] have allowed us to use some exercises from that book. We have also borrowed exercises or examples from [Hod77] and [FHMV95]. Susan Eisenbach provided a first description of the Package Dependency System that we model in Alloy in Chapter 2. Daniel Jackson make very helpful comments on versions of that section. Zena Matilde Ariola, Josh Hodas, Jan Komorowski, Sergey Kotov, Scott A. Smolka and Steve Vickers have corresponded with us about this text; their comments are appreciated. Matt Dwyer and John Hatcliff made useful comments on drafts of Chapter 3. Kevin Lucas provided insightful comments on the content of Chapter 6, and notified us of numerous typographical errors in several drafts of the book. Achim Jung read several chapters and gave useful feedback.

Additionally, a number of people read and provided useful comments on several chapters, including Moti Ben-Ari, Graham Clark, Christian Haack, Anthony Hook, Roberto Segala, Alan Sexton and Allen Stoughton. Numerous students at Kansas State University and the University of Birmingham have given us feedback of various kinds, which has influenced our choice and presentation of the topics. We acknowledge Paul Taylor's LaTeX package for proof boxes. About half a dozen anonymous referees made critical, but constructive, comments which helped to improve this text in various ways. In spite of these contributions, there may still be errors in the book, and we alone must take responsibility for those.

Added for second edition

Many people have helped improve this text by pointing out typos and making other useful comments after the publication date. Among them, we mention Wolfgang Ahrendt, Natasha Alechina, Yasuhiro Ajiro, Torben Amtoft, Stephan Andrei, Miguel Carrillo Barajas, Bernhard Beckert, Jonathan Bowen, Jonathan Brown, Filip Bruman, James Caldwell, B. Chimbo, Ruchira Datta, Xiao Fan, Amy Felty, Valentin Goranko, Dimitar Guelev, A. Burak Gurdag, Ernst Moritz Hahn, Pascal Honoré, Rod Howell, Lei Jinjiang, Hirotsugu Kakugawa, Kamran Kashef, Anders Krogh, Markus Krötzsch, Jagun Kwon, Ranko Lazic, David Makinson, Alexander Miczo, Aart Middeldorp, Juan S. Morales, Robert Morelli, Lena Morgenroth, Prakash Panangaden, Aileen Paraguya, Frank Pfenning, Shekhar Pradhan, Hanspeter Schneider, Marek Sergot, Christian Sternagel, Koichi Takahashi, Petur Thors, Kazunori Ueda, Hiroshi Watanabe, Fuzhi Wang and Reinhard Wilhelm, Jin Yun, Harald Zankl.