Theory and Practice of Logic Programming **1** (1): 123–125, January 2001.
Printed in the United Kingdom © 2001 Cambridge University Press

Book review

Logic in Computer Science: Modelling and Reasoning about Systems by Michael R. A. Huth and Mark D. Ryan, Cambridge University Press, 2000. ISBN 0521652006, £52.50, $80.00 (hardback), ISBN 0521656028, £9.95, $34.95 (paperback), xviii + 387 pages. `http://www.cs.bham.ac.uk/research/lics/`

1 In short

This is an excellent textbook on logic and formal methods which is very suitable for computer science students for at least two reasons.

First, because of the choice of the topics. The focus is on logic as a basis for specification and verification methods. The book discusses the whole range from logic to applications: propositional and predicate logic, temporal logic and more generally modal logic, program verification, model checking, and symbolic model checking using binary decision diagrams.

Secondly, because of the presentation. It differs from the one of many textbooks on logic in that it is more textual than mathematical, and more operationally than declaratively oriented.

As any good textbook, also this book is not only to be recommended for students but for anyone who is interested in applications of logic in computer science.

2 Contents

The book consists of the following chapters:

1. Propositional logic.
2. Predicate logic.
3. Verification by model checking.
4. Program verification.
5. Modal logics and agents.
6. Binary decision diagrams.

1. *Propositional logic*
   The syntax and semantics of classical propositional logic are given, and a proof system for natural deduction is presented. Soundness of the natural deduction proof system is shown in the usual way (many details are left as exercises). The proof of completeness is quite remarkable: the idea is to show that every tautology is derivable in the natural deduction proof system. The key point here is to encode every line in the truth table of the tautology as a sequent, and to build a derivation from all those sequents.

2. *Predicate logic*

   In the discussion of predicate logic, issues like bound and free variables are treated in an intuitive way. The syntax and semantics are explained, and the proof system of natural deduction is extended to the case of predicate logic. Soundness and completeness of the natural deduction proof system are stated but not proved. However, undecidability of predicate logic, which is not proved in some standard textbooks on logic, is shown here by a reduction to Post's correspondence problem.

3. *Verification by model checking*

   This chapter provides a very nice introduction to model checking. It presents the syntax and semantics of the temporal logic CTL (Computation Tree Logic). Two versions of a labelling algorithm are given, which computes for a given formula and a given model the set of states in the model where the formula is satisfied. The complexity of the labelling algorithms is explained, which brings the discussion to the state explosion problem: the size of the model is exponential in the number of variables. Several possible solutions are mentioned, one of which, using binary decision diagrams, is studied in Chapter 6. The chapter further contains sections on the model checker SMV (which is freely available via WWW), fairness, alternatives and extensions of CTL, and the fixed point characterisation of CTL.

4. *Program verification*

   The method discussed here is meant for sequential programs running on a single processor. A proof calculus for partial correctness, using Hoare triples, is discussed extensively, and its use is illustrated with a nice case study. Finally, total correctness is briefly discussed.

   This verification method is aimed at programs that may contain complex data-structures and may have an infinite state space. Another difference with model checking is that the subject of program verification using Hoare triples is not really justified from the point of view of applications used, or almost used, in industry.

   Having said that, two advantages of including this subject are first that it can be viewed as an application of predicate logic (model checking as discussed in Chapter 3 builds on propositional logic), and second that it provides a setting for exercises with the crucial notion of invariant.

5. *Modal logics and agents*

   To start with, the syntax and semantics of classical propositional modal logic are introduced. An important part of this chapter is devoted to the themes 'logic engineering' and 'reasoning about knowledge in a multi-agent system'. The main issue regarding the first theme is the following: given a particular mode of truth, how may we develop a logic capable of expressing and formalising that concept? Here this question is considered in the setting of some well-known versions of modal logic. The second theme illustrates the use of modal logic in a setting where different agents have different knowledge of the world (in spite of the clear exposition I still don't know exactly what an 'agent' is). The discussion is organised around the wise-men and muddy-children puzzles, which are completely formalised.

6. *Binary decision diagrams*
   The first half of this chapter is devoted to Binary Decision Diagrams (BDDs) in a general setting. So the representation of binary functions is discussed, followed by the introduction of BDDs, ordered BDDs, reduced BDDs, and some algorithms for reduced BDDs. Also some complexity issues are briefly mentioned. In the second half the use of BDDs for model checking is explained: in symbolic model checking sets of states of a model are represented as OBDDs. This makes the state explosion problem less dramatic. The chapter concludes with a section on mu-calculus.

3 To conclude

The book is supported by a WWW page which contains besides some basics also additional interesting information like pictures of the authors. What I really like is the WWW tutor. For each chapter there is a list of multiple choice questions that can be used to test whether the main concepts of the chapter are well understood. At the moment there are slightly over 50 exercises, all with answers, and many with an explanation of the answer. Of course the tutor can be extended and polished but already in its present form it can be quite useful for the student. The WWW page further contains some material intended for the instructor like the code of the SMV programs used in the book, and a list of the URLs that are given as references.

As already remarked above, the presentation is roughly speaking more textual than mathematical. The policy seems to be to avoid (mathematical) formalism as much as possible, which is a good idea for a textbook for computer scientists. At some points one could wonder whether this policy is taken maybe a bit too seriously. One such place is in the first chapter, where the introduction of the notion of valuation (later called assignments) is avoided at all costs. Also sometimes a second reading is needed to see where a definition starts and ends, because it is encapsulated in the text.

The book is currently in use in an introductory logic course at the Vrije Universiteit Amsterdam. It certainly seems very suitable for teaching. The important concepts are illustrated with many exercises, examples, and counterexamples. As an additional service, the instructor (if *bona fide*) can ask the publisher for sample solutions of selected exercises.

In the second and further printings, an option would be to replace the somewhat sensational cover picture by something more subtle which suits the book better.

Acknowledgements

I gratefully acknowledge discussions with K. R. Apt, M. Bognar, J. W. Klop and R. C. de Vrijer.

Femke van Raamsdonk