Third Edition

The third edition has been totally rewritten for clarity and accuracy. In addition, the following major changes have been made to the content:

- The discussion of logic programming has been shortened somewhat and the Prolog programs and their documentation have been removed to a freely available archive.
- The chapter on the $\mathscr{Z}$ notation has been removed because it was difficult to do justice to this important topic in a single chapter.
- The discussion of model checking in Chap. 16 has been significantly expanded since model checking has become a widely used technique for program verification.
- Chapter 6 has been added to reflect the growing importance of SAT solvers in all areas of computer science.

Notation

If and only if is abbreviated *iff*. Definitions by convention use *iff* to emphasize that the definition is restrictive. For example: A natural number is even iff it can be expressed as $2k$ for some natural number k. In the definition, *iff* means that numbers expressed as $2k$ are even and these are the only even numbers.

Definitions, theorems and examples are consecutively numbered within each chapter to make them easy to locate. The end of a definition, example or proof is denoted by ■.

Advanced topics and exercises, as well as topics outside the mainstream of the book, are marked with an asterisk.

Acknowledgments

I am indebted to Jørgen Villadsen for his extensive comments on the second edition which materially improved the text. I would like to thank Joost-Pieter Katoen and Doron Peled for reviewing parts of the manuscript. I would also like to thank Helen Desmond, Ben Bishop and Beverley Ford of Springer for facilitating the publication of the book.

Rehovot, Israel Mordechai (Moti) Ben-Ari

Contents

Chapter 1
Introduction

1.1 The Origins of Mathematical Logic

Logic formalizes valid methods of reasoning. The study of logic was begun by the ancient Greeks whose educational system stressed competence in reasoning and in the use of language. Along with rhetoric and grammar, logic formed part of the *trivium*, the first subjects taught to young people. Rules of logic were classified and named. The most widely known set of rules are the *syllogisms*; here is an example of one form of syllogism:

Premise All rabbits have fur.
Premise Some pets are rabbits.
Conclusion Some pets have fur.

If both premises are true, the rules ensure that the conclusion is true.

Logic must be formalized because reasoning expressed in informal natural language can be flawed. A clever example is the following 'syllogism' given by Smullyan (1978, p. 183):

Premise Some cars rattle.
Premise My car is some car.
Conclusion My car rattles.

The formalization of logic began in the nineteenth century as mathematicians attempted to clarify the foundations of mathematics. One trigger was the discovery of non-Euclidean geometries: replacing Euclid's parallel axiom with another axiom resulted in a different theory of geometry that was just as consistent as that of Euclid. Logical systems—axioms and rules of inference—were developed with the understanding that different sets of axioms would lead to different theorems. The questions investigated included:

Consistency A logical system is consistent if it is impossible to prove both a formula and its negation.
Independence The axioms of a logical system are independent if no axiom can be proved from the others.

M. Ben-Ari, *Mathematical Logic for Computer Science*,
DOI 10.1007/978-1-4471-4129-7_1, © Springer-Verlag London 2012

Soundness All theorems that can be proved in the logical system are true.
Completeness All true statements can be proved in the logical system.

Clearly, these questions will only make sense once we have formally defined the central concepts of *truth* and *proof*.

During the first half of the twentieth century, logic became a full-fledged topic of modern mathematics. The framework for research into the foundations of mathematics was called *Hilbert's program*, (named after the great mathematician David Hilbert). His central goal was to prove that mathematics, starting with arithmetic, could be axiomatized in a system that was both consistent and complete. In 1931, Kurt Gödel showed that this goal cannot be achieved: any consistent axiomatic system for arithmetic is incomplete since it contains true statements that cannot be proved within the system.

In the second half of the twentieth century, mathematical logic was applied in computer science and has become one of its most important theoretical foundations. Problems in computer science have led to the development of many new systems of logic that did not exist before or that existed only at the margins of the classical systems. In the remainder of this chapter, we will give an overview of systems of logic relevant to computer science and sketch their applications.

1.2 Propositional Logic

Our first task is to formalize the concept of the *truth* of a statement. Every statement is assigned one of two values, conventionally called *true* and *false* or T and F. These should be considered as arbitrary symbols that could easily be replaced by any other pair of symbols like 1 and 0 or even ♣ and ♠.

Our study of logic commences with the study of *propositional logic* (also called the *propositional calculus*). The *formulas* of the logic are built from *atomic propositions*, which are statements that have no internal structure. Formulas can be combined using *Boolean operators*. These operators have conventional names derived from natural language (*and*, *or*, *implies*), but they are given a formal meaning in the logic. For example, the Boolean operator *and* is defined as the operator that gives the value *true* if and only if applied to two formulas whose values are *true*.

Example 1.1 The statements 'one plus one equals two' and 'Earth is farther from the sun than Venus' are both true statements; therefore, by definition, so is the following statement:

'one plus one equals two' *and* 'Earth is farther from the sun than Venus'.

Since 'Earth is farther from the sun than Mars' is a false statement, so is:

'one plus one equals two' *and* 'Earth is farther from the sun than Mars'. ■

Rules of *syntax* define the legal structure of formulas in propositional logic. The *semantics*—the meaning of formulas—is defined by *interpretations*, which assign