LOGIC AND DISCRETE MATHEMATICS

A Computer Science Perspective

WINFRIED KARL GRASSMANN
JEAN-PAUL TREMBLAY

LOGIC AND DISCRETE MATHEMATICS

A Computer Science Perspective

WINFRIED KARL GRASSMANN

Department of Computer Science
University of Saskatchewan

JEAN-PAUL TREMBLAY

Department of Computer Science
University of Saskatchewan

江苏工业学院图书馆
藏书章

PRENTICE HALL, *Upper Saddle River, New Jersey 07458*

Library of Congress Cataloging-in-Publication Data

Grassmann, Winfried K.
Logic and discrete mathematics : a computer science perspective / Winfried Karl Grassmann, Jean-Paul Tremblay.
p. cm.
Includes bibliographical references and index.
ISBN 0-13-501206-6
1. Computer science — Mathematics. I. Tremblay, Jean-Paul
II. Title.
QA76.9.M35G725 1996
005.1'01'5113—dc20 95-38351
CIP
AC

Acquisitions editor: *Alan Apt*
Editorial/production supervision: *Maes Associates*
Copy editor: *William O. Thomas*
Cover designer: *Bruce Kenselaar*
Manufacturing buyer: *Donna Sullivan*

© 1996 by Prentice-Hall, Inc.
Simon & Schuster/A Viacom Company
Upper Saddle River, New Jersey 07458

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-501206-6

PRENTICE-HALL INTERNATIONAL (UK) LIMITED, *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sidney*
PRENTICE-HALL CANADA INC., *Toronto*
PRENTICE-HALL HISPANOAMERICANA, S.A., *Mexico*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
SIMON & SCHUSTER ASIA PTE. LTD., *Singapore*
EDITORIA PRENTICE-HALL DO BRASIL, LTDA., *Rio de Janeiro*

To my wife Louise Grassmann and
to my children Stephanie and Bettina Grassmann
—W.K.G.

To my wife Deanna Tremblay and
to my grandchildren
Robert, Leanne, Lisa, and Nicole Tremblay
—J.P.T.

Preface

Most universities require that a course in discrete mathematics be taken by every undergraduate computer science student, and rightly so. Indeed, discrete mathematics gives the appropriate theoretical foundations for computer science, foundations that are not only beneficial for doing theoretical computer science, but also for the practice of computer science as evidenced by the recent proliferation of books on formal methods. The areas covered in a course in discrete mathematics vary, but they traditionally include logic, sets, relations, functions, and graphs. All these topics are included in this book. Moreover, this book reflects several recent trends in computer science. In particular, it gives a more thorough exposure to logical reasoning than most other texts. It also shows how to use discrete mathematics and logic for specifying new computer applications and how to reason about programs in a systematic way. The book contains chapters on languages and grammars, the Z specification language, and relational databases. There is a chapter describing Prolog, a programming language based on logic, and a section discussing Miranda, a language based on functions. In all chapters, numerous examples relate the mathematical concepts to problems in computer science. We found that such examples are essential for keeping the student motivated.

The outline of the book is as follows. Chapter 1 covers propositional calculus. We also define what is to be understood by a formal derivation. In fact, formal derivations are introduced as refinements of the type of logical reasoning we use in daily life. The chapter also contains an extensive discussion on algebraic manipulation of logical expressions.

Chapter 2 discusses predicate calculus. We cover predicates and quantifiers, and we try again to relate this theory to everyday thinking. Chapter 2 also introduces unification, a topic that is needed in order to understand logical languages such as Prolog

此为试读,需要完整PDF请访问: www.ertongbook.com

(covered in Chapter 4) and resolution theorem proving (described in Chapter 11). Chapter 2 also introduces equational logic. Equational logic is concerned with the manipulation of mathematical equations, and this is obviously of fundamental importance. As it turns out, equational logic is closely related to functions, and indeed, without functions, equational logic would not have attained the dominance in mathematical reasoning which it now has. This motivated us to introduce functions at this point.

Chapter 3 deals with induction and recursion, which is an important part of logic, mathematics, and, above all, computer science. We discuss several proof methods, including mathematical induction, strong induction, proof by recursion, and structural induction. The notions of matrices and sums are also introduced. As an application, recursive programming techniques are explored. This chapter also includes material on recursive functions and decidability. Though recursive functions are very important, they constitute advanced material which may be omitted.

Chapter 4 covers Prolog. The coverage is sufficient to allow the reader to write nontrivial Prolog programs. The connections between logic and Prolog are also explored. A section in Chapter 4 explains what kind of logical statements can be translated into Prolog. Another section shows how to use Prolog to program manipulations involving logical expressions. The study of Prolog also reinforces the concept of recursion, since almost every nontrivial Prolog program makes use of recursion.

Computer scientists deal with many different types of objects, which can in turn be combined in different ways. To understand these compound objects and their manipulation, the student has to understand the concepts of sets and relations, which are discussed in Chapter 5. The important operations on these constructs are described by numerous examples, many of which are directly related to computer applications.

Chapter 6 deals with several topics regarding functions, such as algorithmic analysis and computational complexity. These concepts are illustrated with a number of important examples. In addition to the topics mentioned, the language Miranda is introduced as an example of a functional language.

Chapter 7 gives an overview of graphs and trees, concepts that are fundamental to computer science. In addition to introducing basic terminology dealing with paths, reachability, and connectedness, the emphasis in this chapter is on computing paths, minimum paths, minimum weighted paths, spanning trees, and minimum spanning trees for weighted graphs. This chapter concludes with a discussion of the application of graphs to project planning and management.

Chapter 8 is about Z, a language applying concepts from logic and set theory for specifying and analyzing requirements in software development.

Chapter 9 shows how to use logic, and to a certain extent sets and functions, to reason about programs. As in Chapters 1 and 2, we start with commonsense reasoning, and we introduce formal correctness proofs as a refinement of normal reasoning.

Chapter 10 deals with context-free grammars as a vehicle for defining the syntax of programming languages and their use in syntax analysis. Emphasis is given to an LL(1) parser generator system.

Chapter 11 gives additional information about derivations. In particular, it shows how to use natural deduction and the techniques of resolution theorem proving.

The book concludes with a discussion of relational databases in Chapter 12. This chapter first focuses on the relational data model and its associated relational algebra. An alternative way of describing queries based on predicate calculus is also given. Finally, an overview of a structured query language for specifying queries is given.

This text would appeal to a student in computer science at the freshman (second term), sophomore, or junior level. Preliminary versions of this book were used in a one-semester course given to second-year computer science students. In this course, most of the material contained in Chapters 1 through 6 was covered. To this, we selectively added topics from Chapters 7 through 12. However, the book contains enough material for a full year course, but in discrete mathematics, such courses are not normally offered. Whatever the length of the course, it should lay the mathematical foundations for many classes, including classes in database design, syntactical analysis and parsing, artificial intelligence, programming languages, logic programming, functional languages, and computer hardware.

The application chapters allow the instructors to shift the emphasis according to their interests. In all cases, the material of Chapters 1, 2, and 5 is essential, and most instructors want to cover Section 6.1, which discusses functions. With respect to the other chapters, we would like to suggest the following four scenarios, which can be combined or altered, depending on the preferences of the instructor.

Emphasis on Procedural Programming For procedural programming, recursion is very important, and this is covered in Chapter 3. The same chapter contains a discussion of recursive functions, and it shows what can and what cannot be solved by computers. Section 6.3 gives an introduction to computational complexity which should prove useful to the student. The instructor may also want to cover Chapter 7, which discusses graphs and trees, and Chapter 9, which shows how to prove that a program is correct.

Emphasis on Logic and Logic Programming Since recursion is very important for all types of logic programming, the instructor should cover recursion, which is described in Chapter 3. Chapter 4 then gives an extensive treatment of Prolog. At this point, Chapter 10, grammars and languages, and Chapter 11, which shows resolution theorem proving, provide suitable topics.

Emphasis on Functions and Functional Programming To provide a functional programming emphasis, all of Chapter 3 should be covered. From Chapter 6, Sections 6.1 and 6.3 are important. Of course, Section 6.5, Miranda, is central under this scenario. For the remaining topics, the instructor may select material from Chapter 8, requirement specification, Chapter 9, correctness proofs, and Chapter 10, grammars and languages.

Emphasis on Information Systems The slant to systems analysis requires some knowledge of graphs and trees as it is discussed in Chapter 7, which in turn have the first two sections of Chapter 3 as a prerequisite. Other relevant material can be found in Chapters 8 (specification in Z) and 12 (relational database systems).

The text is geared to the student, and every effort has been made to explain the material as clearly as possible. In general, we motivate all concepts and derivations by means of examples, preferably examples from computer science. This approach allows for an informal introduction to the topic that can be formalized later. This gradual approach helps the students to appreciate the value of formal arguments and prepares them for their later studies in computer science.

As teaching and learning aids, the book includes more than 300 examples and more than 550 problems with detailed solutions provided for all even-numbered problems.

We would like to thank Wayne Mackrell, Christy Kenny, David Haugen, Allan Rempel, Jacob Wickland, and Michael Zaleski for their help in entering the manuscript and suggesting improvements to the text. Wayne and Christy have also assisted us in formulating solutions to the problems and in producing the computer-generated diagrams. A special thank you to Cyril Coupal who suggested the title of the book. Deanna Tremblay proofread many versions of the manuscript during its period of preparation.

We would like to thank our colleagues for their valuable help. Jim Greer read and commented on the first three chapters. Eric Neufeld and Grant Cheston made important suggestions for which we are very grateful. John Cooke read and commented on the relational database chapter.

Finally, we thank the students who used earlier versions of these notes and who made many valuable comments.

Winfried Karl Grassmann
Jean-Paul Tremblay

Contents

此为试读,需要完整PDF请访问: www.ertongbook.com

此为试读,需要完整PDF请访问: www.ertongbook.com