

MATHEMATICAL STRUCTURES FOR COMPUTER SCIENCE

*A Modern Approach
to Discrete Mathematics*

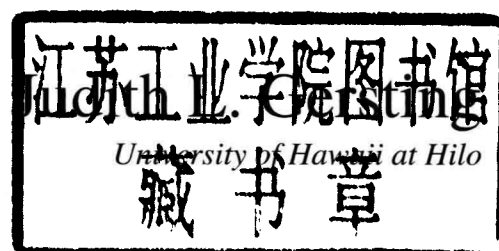
Sixth Edition

Judith L. Gersting

MATHEMATICAL STRUCTURES FOR COMPUTER SCIENCE

A Modern Approach to Discrete Mathematics

SIXTH EDITION



W. H. Freeman and Company New York

Now there are 0101_2 favorite discrete structures:

$(\text{Adam} \wedge \text{Francine}), (\text{Jason} \wedge \text{Cathryn} \rightarrow \text{Samantha})$

Publisher: Craig Bleyer

Acquisitions Editor: Terri Ward

Executive Editor Statistics and Mathematics: Ruth Baruth

Associate Editor: Brendan Cady

Executive Marketing Manager: Sarah Martin

Senior Media Editor: Victoria Anderson

Cover Designer: Blake Logan

Cover Art: Fine Line Illustrations

Text Designer: Lisa Douglass

Project Editor: Penelope Hull

Illustrations: Fine Line Illustrations

Illustration Coordinator: Bill Page

Production Coordinator: Julia DeRosa

Composition: Black Dot Group

Printing and Binding: Quebecor World Book Group

Library of Congress Control Number: 2006926756

ISBN-13: 978-0-7167-6864-X

ISBN-10: 0-7167-6864-X

© 2007, 2003, 1999, 1993, 1987, 1982 by W. H. Freeman and Company. All rights reserved.

Printed in the United States of America

Second printing

W. H. Freeman and Company
41 Madison Avenue, New York, NY 10010
Houndmills, Basingstoke RG21 6XS, England
www.whfreeman.com

Mathematical Structures for Computer Science

Preface

Discrete structures plays an ever more prominent role in computer science education. Computing Curricula 2001 (CC2001) was a joint undertaking of the Institute for Electrical and Electronic Engineers/Computer Society (IEEE/CS) and the Association for Computing Machinery (ACM). The final report may be found at <http://www.acm.org/sigcse/cc2001/>. CC2001 identifies a computer science *body of knowledge*, a set of knowledge units “for which there is a broad consensus that the material is essential to an undergraduate degree in computer science.” CC2001 describes discrete structures as “foundational material for computer science” and devotes 43 hours of the computer science body of knowledge to discrete structures, more hours than to any other area.

This Sixth Edition of *Mathematical Structures for Computer Science* covers all the topics in the CC2001 discrete structures core and quite a bit more. All topics suggested for a one-semester intensive discrete structures course—and virtually everything suggested for a two-semester version of a discrete structures course—are covered here.

Binding together what otherwise appears to be a collection of disjoint topics are the following themes:

- Importance of logical thinking
- Power of mathematical notation
- Usefulness of abstractions

Of all computer science courses, discrete structures is probably the least appreciated in real time and the most appreciated given more perspective. Two years after the fact, students report, “Every course I had after that used material from discrete structures.”

Content Changes in the Sixth Edition

Major changes include the following:

- Reorganization of Chapter 2 to make Recurrence Relations a separate section (Section 2.5), with the addition of material on linear second-order recurrence relations and on divide-and-conquer recurrence relations
- New section on number theory (Section 3.7)
- New section (Section 4.5) on “the mighty mod function,” giving a number of applications of the mod function in hashing, cryptography, identification codes, and modular arithmetic designs, among others
- Restoration, at the request of some reviewers, of a section (Section 6.5) on articulation points in graphs that had been deleted from the Fifth Edition.

New Exercises have also been added to each section. It appears that exercises, like chocolate chip cookies, are something you just can't have too many of. Answers to all Practice Problems are given at the back of the book, as are answers to all starred Exercises.

A complete **Solutions Manual** is available to instructors from the publisher.

Web Site

Online Study Guide

A Web site for the book may be found at www.whfreeman.com/gersting, accessed using the password "logic." It functions as an online study guide, offering for each chapter a chapter overview, section overviews, and sample problems with audio commentaries outlining the step-by-step solution process. The sample problems (not contained in the text) are representative of many of the end-of-section Techniques. Each Technique that has a corresponding Web page example is marked with the icon .

Each sample problem on the Web first states the problem, and then, as the student navigates the pages, the solution unfolds step-by-step much as the student would be expected to write it. A compressed audio file is also part of each Web page after the initial problem statement. The audio file contains a first-person stream-of-consciousness thought process about that step of the solution—why it occurred to the narrator to try it, why it looked promising, what knowledge was being called upon to suggest that it should come next, and so on. Students see perfect and complete worked-out proofs in the textbook and often also see them performed by the instructor. Yet when a student goes home and tries to produce a solution alone, he or she is unsure where to start or how to think about the problem or how to discern any pattern to enable a guess as to what to do next. Consequently the student gives up in frustration. The purpose of the audio narration is to share the "secret picture" that mathematicians use to solve problems.

To access the problems, after you go to www.whfreeman.com/gersting, select a chapter, log in with the password "logic," select a chapter section, then select a sample problem and follow its step-by-step process with the "Next" button. Several sample problems are provided for each section of each chapter.

PowerPoint Slides

PowerPoint slides accompanying each section of the text are available on the Web site at www.whfreeman.com/gersting.

Acknowledgments

My thanks to the reviewers of this edition, as well as to reviewers of earlier editions, all of whose help is greatly appreciated.

Elizabeth S. Adams, *James Madison University*

Kemal Akkaya, *Southern Illinois University*

Charles Ashbacher, *Mount Mercy College*

Terry J. Bridgman, *Colorado School of Mines*

David Casperson, *University of Northern British Columbia*

Adrienne Decker, *SUNY Buffalo*

Steve Donaldson, *Samford University*

Mordechai S. Goodman, *Dominican University*

Michael A. Gray, *American University*

Jerrold R. Griggs, *University of South Carolina*

Joseph Hobart, *Okanagan College*
 Mark Jacobson, *University of Northern Iowa*
 Lisa A. Jamba, *University of Northern
 Florida*
 Tim Lin, *Cal Poly*
 David Luginbuhl, *Western Carolina
 University*
 Damian Lyons, *Fordham University*
 Mariana Maris, *Arizona State University*
 Mikel D. Petty, *University of Alabama in
 Huntsville*

Amar Raheja, *Cal Poly*
 J. Ben Schafer, *University of Northern Iowa*
 Ali Shaykhian, *Florida Institute of
 Technology*
 Shunichi Toida, *Old Dominion University*
 William J. Weber, *Southeast Missouri State
 University*
 Eric Westlund, *Luther University*
 Hua Yan, *Borough of Manhattan Community
 College*
 Yu Zhang, *Texas A&M Corpus Christi*

My deepest thanks go to my husband, John, ever my most ardent supporter and dearest friend.

Note to the Student

As you go through this book, you'll encounter many new terms and new ideas. Try reading with pencil and paper at hand and work the Practice Problems as you encounter them. They are intended to reinforce or clarify terminology or a method just introduced; answers are given at the back of the book. Pay attention also to the Reminders that point out common pitfalls or provide helpful hints. Be sure to visit the Web site at www.whfreeman.com/gersting, using the password "logic," for detailed, worked-out solutions to additional example problems tied to the Techniques at the end of each section. The Web site solutions are accompanied by audio files that explain each step.

You may find at first that the thought processes required to solve the exercises in the book are new and difficult. Your biggest attribute for success will be perseverance. Here's what I tell my students: "If you do not see at first how to solve a problem, don't give up. Think about it some more; be sure you understand all the terminology used in the problem, play with some ideas. If no approach presents itself, put it aside and think about it again later. Repeat this process for days on end. When you finally wake up in the middle of the night with an idea, you'll know you are putting in the right amount of effort for this course." Mathematical results don't spring fully formed from the foreheads of mathematical geniuses (well, maybe from mathematical geniuses); but for the rest of us, it takes work, patience, false starts, and **perseverance**.

Enjoy the experience!

Contents in Brief

	Preface	xv
	Note to the Student	xvii
CHAPTER 1	Formal Logic	1
CHAPTER 2	Proofs, Recursion, and Analysis of Algorithms	88
CHAPTER 3	Sets, Combinatorics, Probability, and Number Theory	186
CHAPTER 4	Relations, Functions, and Matrices	285
CHAPTER 5	Graphs and Trees	401
CHAPTER 6	Graph Algorithms	475
CHAPTER 7	Boolean Algebra and Computer Logic	534
CHAPTER 8	Modeling Arithmetic, Computation, and Languages ..	592
APPENDIX A	Derivation Rules for Propositional and Predicate Logic	685
APPENDIX B	Summation Notation	687
APPENDIX C	The Logarithm Function	690
	Answers to Practice Problems	693
	Answers to Selected Exercises	727
	Answers to Self-Tests	789
	Index	797

Contents

	Preface	xv
	Note to the Student	xvii
CHAPTER 1	Formal Logic	1
	1.1 Statements, Symbolic Representation, and Tautologies	2
	Connectives and Truth Values	2
	Tautologies	8
	Logical Connectives in the Real World	10
	An Algorithm	11
	<i>Exercises 1.1</i>	14
	1.2 Propositional Logic	21
	Valid Arguments	21
	Derivation Rules for Propositional Logic	24
	Deduction Method and Other Rules	27
	Verbal Arguments	29
	<i>Exercises 1.2</i>	31
	1.3 Quantifiers, Predicates, and Validity	35
	Quantifiers and Predicates	35
	Translation	38
	Validity	41
	<i>Exercises 1.3</i>	43
	1.4 Predicate Logic	49
	Derivation Rules for Predicate Logic	49
	Universal Instantiation	51
	Existential Instantiation	51
	Universal Generalization	52
	Existential Generalization	53
	More Work with Rules	54
	Verbal Arguments	58
	Conclusion	59
	<i>Exercises 1.4</i>	60
	1.5 Logic Programming	63
	Prolog	64
	Horn Clauses and Resolution	66

Recursion	69
Expert Systems	72
Exercises 1.5	72
1.6 Proof of Correctness	74
Assertions	75
Assignment Rule	76
Conditional Rule	79
Exercises 1.6	82
Chapter 1 Review	84
On the Computer	87

CHAPTER 2 **Proofs, Recursion, and Analysis of Algorithms . . 88**

2.1 Proof Techniques	89
Theorems and Informal Proofs	89
To Prove or Not to Prove	90
Exhaustive Proof	91
Direct Proof	92
Contraposition	93
Contradiction	95
Serendipity	96
Common Definitions	97
Exercises 2.1	98
2.2 Induction	100
First Principle of Induction	100
Proofs by Mathematical Induction	102
Second Principle of Induction	107
Exercises 2.2	112
2.3 More on Proof of Correctness	118
Loop Rule	118
Euclidean Algorithm	121
Exercises 2.3	124
2.4 Recursive Definitions	129
Recursively Defined Sequences	129
Recursively Defined Sets	132
Recursively Defined Operations	134
Recursively Defined Algorithms	135
Exercises 2.4	140
2.5 Recurrence Relations	147
Linear First-Order Recurrence Relations	147
Expand, Guess, and Verify	147
A Solution Formula	149
Linear Second-Order Recurrence Relations	156
Divide-and-Conquer Recurrence Relations	161
Exercises 2.5	164

2.6 Analysis of Algorithms	169
The General Idea	169
Analysis Using Recurrence Relations	172
Upper Bound (Euclidean Algorithm)	174
Exercises 2.6	175
 Chapter 2 Review	 181
On the Computer	183
 CHAPTER 3 Sets, Combinatorics, Probability, and Number Theory	 186
3.1 Sets	187
Notation	187
Relationships Between Sets	189
Sets of Sets	191
Binary and Unary Operations	192
Operations on Sets	194
Set Identities	196
Countable and Uncountable Sets	199
Exercises 3.1	202
3.2 Counting	211
Multiplication Principle	212
Addition Principle	215
Using the Principles Together	216
Decision Trees	218
Exercises 3.2	220
3.3 Principle of Inclusion and Exclusion; Pigeonhole Principle	225
Principle of Inclusion and Exclusion	225
Pigeonhole Principle	230
Exercises 3.3	231
3.4 Permutations and Combinations	233
Permutations	233
Combinations	235
Eliminating Duplicates	238
Permutations and Combinations with Repetitions	239
Exercises 3.4	246
3.5 Probability	252
Introduction to Finite Probability	252
Probability Distributions	254
Conditional Probability	256
Expected Value	257
Average Case Analysis of Algorithms	260
Exercises 3.5	261

3.6 Binomial Theorem	266
Pascal's Triangle	266
Binomial Theorem and Its Proof	267
Applying the Binomial Theorem	269
Exercises 3.6	270
3.7 Number Theory	272
Fundamental Theorem of Arithmetic	272
More on Prime Numbers	276
Euler Phi Function	277
Exercises 3.7	279
Chapter 3 Review	281
On the Computer	283

CHAPTER 4 Relations, Functions, and Matrices 285

4.1 Relations	286
Binary Relations	286
Properties of Relations	289
Closures of Relations	291
Partial Orderings	293
Equivalence Relations	295
Exercises 4.1	301
4.2 Topological Sorting	311
Exercises 4.2	316
4.3 Relations and Databases	319
Entity-Relationship Model	319
Relational Model	320
Operations on Relations	323
Null Values and Three-Valued Logic	326
Database Integrity	328
Exercises 4.3	329
4.4 Functions	331
Definition	332
Properties of Functions	337
Onto Functions	337
One-to-One Functions	338
Bijections	339
Composition of Functions	339
Inverse Functions	341
Permutation Functions	343
How Many Functions	345
Equivalent Sets	348
Order of Magnitude of Functions	349

More on Analysis of Algorithms	352
<i>Exercises 4.4</i>	354
4.5 The Mighty Mod Function	362
Hashing	363
Computer Security	366
Cryptography	366
Hashing for Password Encryption	372
Miscellaneous Applications	373
Identification Codes	373
Generating and Decomposing Integers	375
Modular Arithmetic Designs	376
<i>Exercises 4.5</i>	378
4.6 Matrices	382
Terminology	382
Matrix Operations	384
Boolean Matrices	388
<i>Exercises 4.6</i>	390
Chapter 4 Review	396
On the Computer	399
 CHAPTER 5 Graphs and Trees	 401
5.1 Graphs and Their Representations	402
Definitions of a Graph	402
Applications of Graphs	405
Graph Terminology	407
Isomorphic Graphs	410
Planar Graphs	413
Computer Representation of Graphs	418
Adjacency Matrix	418
Adjacency List	420
<i>Exercises 5.1</i>	423
5.2 Trees and Their Representations	433
Tree Terminology	433
Applications of Trees	435
Binary Tree Representation	437
Tree Traversal Algorithms	438
Results About Trees	442
<i>Exercises 5.2</i>	444
5.3 Decision Trees	451
Searching	452
Lower Bounds on Searching	454
Binary Tree Search	455

Sorting	457
<i>Exercises 5.3</i>	459
5.4 Huffman Codes	462
Problem and Trial Solution	462
Huffman Encoding Algorithm	464
Justification	466
Applications of Huffman Codes	468
<i>Exercises 5.4</i>	469
Chapter 5 Review	472
On the Computer	474
 CHAPTER 6 Graph Algorithms	 475
6.1 Directed Graphs and Binary Relations; Warshall's Algorithm	 476
Directed Graphs and Binary Relations	477
Reachability	479
Warshall's Algorithm	483
<i>Exercises 6.1</i>	487
6.2 Euler Path and Hamiltonian Circuit	490
Euler Path Problem	490
Hamiltonian Circuit Problem	495
<i>Exercises 6.2</i>	496
6.3 Shortest Path and Minimal Spanning Tree	499
Shortest-Path Problem	499
Minimal Spanning Tree Problem	505
<i>Exercises 6.3</i>	507
6.4 Traversal Algorithms	513
Depth-First Search	513
Breadth-First Search	515
Analysis	518
Applications	519
<i>Exercises 6.4</i>	521
6.5 Articulation Points and Computer Networks	524
The Problem Statement	524
The Idea Behind the Algorithm	525
The Algorithm Itself	526
<i>Exercises 6.5</i>	529
Chapter 6 Review	531
On the Computer	532

CHAPTER 7 Boolean Algebra and Computer Logic. 534

7.1 Boolean Algebra Structure.	535
Models or Abstractions	536
Definition and Properties	537
Isomorphic Boolean Algebras	541
What Is Isomorphism?	541
Isomorphism as Applied to Boolean Algebra	544
Exercises 7.1	546
7.2 Logic Networks	552
Combinational Networks	552
Basic Logic Elements	552
Boolean Expressions	553
Truth Functions	554
Networks and Expressions	554
Canonical Form	556
Minimization	558
Programmable Logic Arrays	560
A Useful Network	562
Other Logic Elements	564
Constructing Truth Functions	565
Exercises 7.2	566
7.3 Minimization	574
Minimization Process	574
Karnaugh Map	575
Maps for Three and Four Variables	576
Using the Karnaugh Map	578
Quine-McCluskey Procedure	582
Exercises 7.3	586
Chapter 7 Review	590
On the Computer	591

CHAPTER 8 Modeling Arithmetic, Computation, and Languages 592

8.1 Algebraic Structures	593
Definitions and Examples	593
Basic Results About Groups	600
Subgroups	603
Isomorphic Groups	606
Exercises 8.1	612
8.2 Finite-State Machines	618
Definition	618

Examples of Finite-State Machines	619
Recognition	622
Regular Sets and Kleene's Theorem	624
Machine Minimization.	626
Unreachable States.	626
Minimization Procedure	627
Sequential Networks and Finite-State Machines	632
<i>Exercises 8.2</i>	636
8.3 Turing Machines	646
Definition	647
Turing Machines as Set Recognizers	651
Turing Machines as Function Computers	653
Church-Turing Thesis	655
Decision Problems and Uncomputability	657
Examples of Decision Problems	658
Halting Problem.	659
Computational Complexity	660
<i>Exercises 8.3</i>	663
8.4 Formal Languages.	666
Classes of Grammars	672
Formal Languages and Computational Devices	675
Context-Free Grammars	676
<i>Exercises 8.4</i>	679
Chapter 8 Review.	682
On the Computer	684
 APPENDIX A Derivation Rules for Propositional and Predicate Logic	 685
APPENDIX B Summation Notation.	687
APPENDIX C The Logarithm Function	690
Answers to Practice Problems	693
Answers to Selected Exercises.	727
Answers to Self-Tests	789
Index.	797

Formal Logic

CHAPTER



CHAPTER OBJECTIVES

After studying this chapter, you will be able to:

- Use the formal symbols of propositional logic.
- Find the truth value of an expression in propositional logic.
- Construct formal proofs in propositional logic, and use such proofs to determine the validity of English language arguments.
- Use the formal symbols of predicate logic.
- Find the truth value in some interpretation of an expression in predicate logic.
- Use predicate logic to represent English language sentences.
- Construct formal proofs in predicate logic, and use such proofs to determine the validity of English language arguments.
- Understand how the programming language Prolog is built on predicate logic.
- Mathematically prove the correctness of programs that use assignment statements and conditional statements.

You have been selected to serve on jury duty for a criminal case. The attorney for the defense argues as follows:

If my client is guilty, then the knife was in the drawer. Either the knife was not in the drawer or Jason Pritchard saw the knife. If the knife was not there on October 10, it follows that Jason Pritchard did not see the knife. Furthermore, if the knife was there on October 10, then the knife was in the drawer and also the hammer was in the barn. But we all know that the hammer was not in the barn. Therefore, ladies and gentlemen of the jury, my client is innocent.

QUESTION Is the attorney's argument sound? How should you vote?

It's much easier to answer this question if the argument is recast in the notation of formal logic. Formal logic strips away confusing verbiage and allows us to concentrate on the underlying reasoning being applied. In fact, formal logic—the subject of this chapter—provides the foundation for the organized, careful method of thinking that characterizes any reasoned activity—a criminal investigation, a scientific experiment, a sociological study. In addition, formal logic has direct applications in computer science. The last two sections of this chapter explore a programming language based on logic and the use of formal logic to verify the