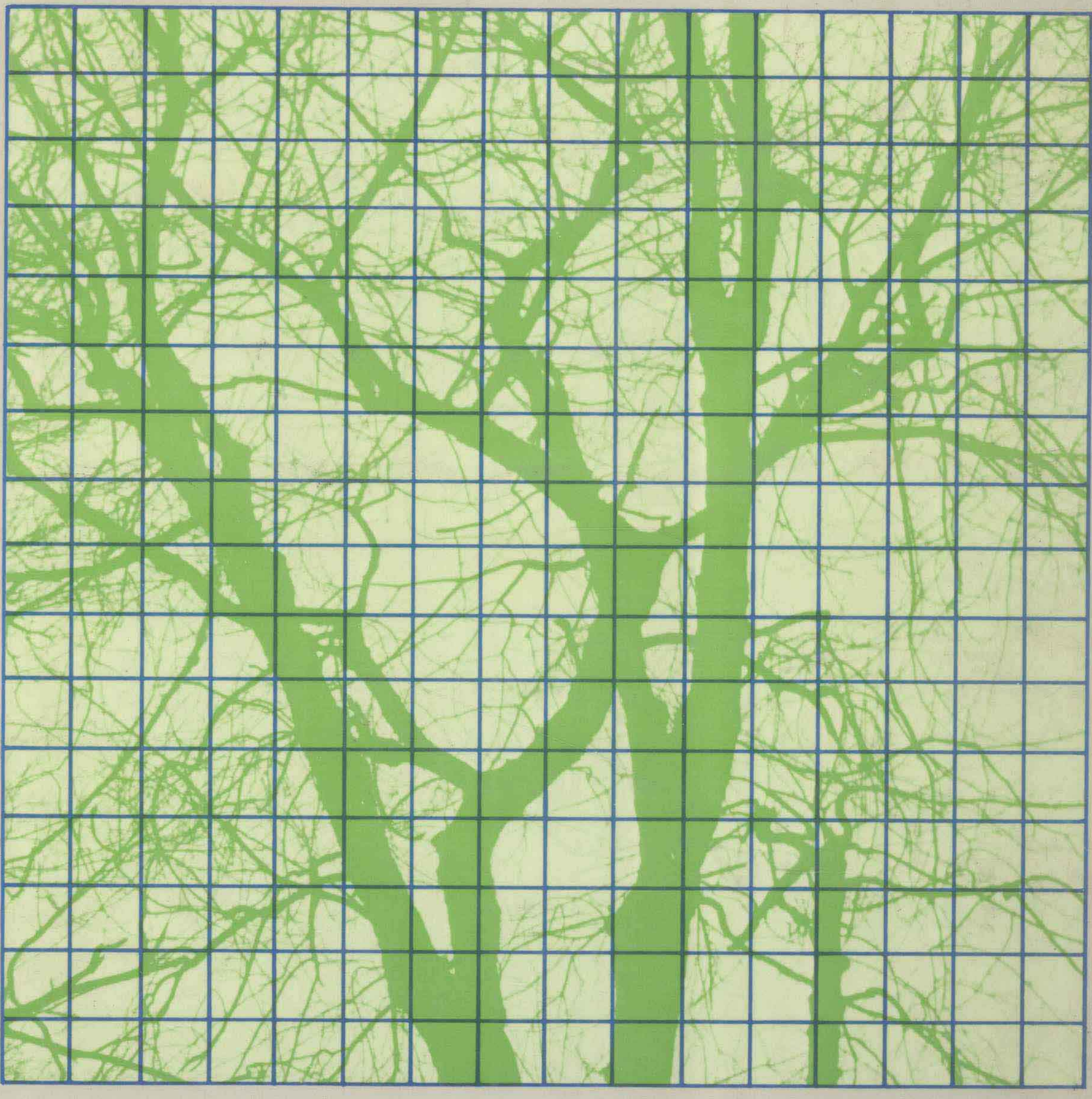
RICHARD JOHNSONBAUGH

Essential Discrete Mathematics



Essential Discrete Mathematics

MACMILLAN PUBLISHING COMPANY
NEW YORK
Collier Macmillan Publishers
LONDON

Macmillan Publishing Company
866 Third Avenue, New York, New York 10022

Collier Macmillan Canada, Inc.

LIBRARY OF CONGRESS CATALOGING-IN-PUBLICATION DATA

Johnsonbaugh, Richard,
Essential discrete mathematics.
Bibliography: p.
Includes index.
1. Mathematics—1961– 2. Electronic data processing—Mathematics. I. Title.
QA39.2.J66 1987 510 86-18028
ISBN 0-02-360630-4
Printing: 1 2 3 4 5 6 7 8 Year: 7 8 9 0 1 2 3 4 5 6

ISBN 0-02-360630-4

Preface

TO THE INSTRUCTOR

A course in discrete mathematics has become a standard offering in many mathematics and computer science departments. This book for a one-term course in discrete mathematics represents a different philosophy and organization from those of most discrete mathematics books now available. This new approach will make discrete mathematics more accessible to students and will make the subject more inviting to students who are required to take a course in discrete mathematics.

The features in this book include the following:

1. Topics are introduced in the order of increasing sophistication.
2. A course in computer science or computer programming is not a prerequisite. (The recommended mathematics prerequisite is college algebra.)
3. A review and a self-test with answers in the back of the book conclude each chapter.

4. Many examples are drawn from familiar, nonmathematical, nonthreatening settings.
5. Algorithms are described in ordinary English.
6. An optional *Student Study Guide* is available.
7. A number of bridges to previous work in mathematics, such as number systems, functions, and sequences, are provided.
8. Matrices and applications are included.
9. Logic is integrated into the main text.
10. Biographical sketches are included.

I will discuss some of these features in more detail.

So that students with a relatively modest background in mathematics can succeed in this course, topics are introduced in the order of increasing sophistication. As a prerequisite, I would recommend the equivalent of college algebra. Calculus and computer programming are not prerequisites.

The book begins with logic and circuits. Not only do these topics provide a gentle introduction to discrete mathematics, but also I have found that students enjoy analyzing and constructing circuits.

Chapter 2 deals with general mathematical concepts—sets, sequences, mathematical induction, functions, and matrices. Students will have seen some of this material before. I have taken a straightforward approach to these topics.

Chapters 3 and 4 deal with graphs and trees. Students enjoy this material and do not find it particularly abstract.

Beginning with Chapter 5 on algorithms, the level of abstraction begins to increase and more demands are made on the student for original thinking. At the minimum, one should cover Section 5.1, which discusses what an algorithm is, and Section 5.4, which deals with the time required to execute an algorithm. These topics are important in mathematics quite apart from their central role in computer science.

Permutations, combinations, and the pigeonhole principle form the subject matter of Chapter 6. A wealth of examples and discussion are provided to help the student master this tricky material.

Chapters 7 and 8 contain the most abstract topics in this book—recurrence relations, relations, and equivalence relations.

Algorithms are not formally treated until Chapter 5. (Algorithms are given prior to Chapter 5 but are not advertised as such.) Algorithms are stated in plain English and are also presented somewhat more formally as a stepwise process, but not in pseudocode. The latter description can be skipped if desired. Pseudocode is described in optional Section 5.2. Thereafter, pseudocode is presented only in optional sections flagged with †. Further, all computer science examples are given in a handful of optional sections marked with † that follow the regular sections. These optional sections are labeled Computer Notes. Exercises for Computer Notes sections are marked with the computer icon shown in the margin. Students who are taking a computer science course or who have programming experience should find the computer sections helpful.

Each chapter concludes with a chapter review section that lists the key ideas of the chapter and a self-test that includes questions from every section in the chapter. Answers to all self-test questions can be found in the Hints section at the end of the book. Students who have had programming experience may be interested in the computer exercises found at the end of each chapter.

For a taste of the examples that are drawn from familiar, nonmathematical, nonthreatening settings, sample the beginnings of Sections 1.1, 2.7, and 8.1; the beginning of Chapter 5; and Example 2.5.7.

Approximately one-third of the exercises have solutions in the Hints section at the back of the book. Exercises with solutions in the back of the book are marked with "H." An optional *Student Study Guide* is available. This *Guide* contains solutions to another third of the exercises in the book. In addition, the *Guide* contains detailed explanations of examples and solutions to problems together with tips on how to solve problems. The style is informal.

The basic operations on matrices are introduced in Section 2.7. Section 3.4 treats the matrix representation of graphs. Matrices of relations are covered in Section 8.3.

Logic is integrated into the main text. The first two sections of the book introduce fundamental concepts in logic. Two additional sections on logic, Proofs and Arguments, and Categorical Propositions, Sections A.1 and A.2, can be covered at any time as needed.

Several optional sections can be omitted without loss of continuity. For example, if only a brief introduction to trees is desired, one would cover only Sections 4.1 (Introduction) and 4.2 (Terminology and Characterizations of Trees). On the other hand, by covering all of the sections, binary trees, isomorphic trees, and games trees would also be discussed. In this way it is possible to tailor the topics to local needs.

Because it is important for students to work a large number of exercises, over 1700 exercises are included. Although several more challenging than average exercises are marked with a star, there are an ample number of exercises that directly check students' understanding of the concepts and examples presented in the book. Ends of proofs are marked with the symbol ■.

An *Instructor's Guide* and a program diskette for IBM and IBM-compatible personal computers are available at no cost to adopters of this book. The *Instructor's Guide* contains a discussion about how to use this book, sample exams, sample syllabi, and solutions to most exercises whose solutions are not given in either the *Student Study Guide* or the book itself. The program diskette, which is coordinated with optional sections of the *Student Study Guide*, contains (Turbo) Pascal programs with which students can experiment to obtain hands-on experience with concepts in the book. There is no charge for making copies of this diskette to distribute to students.

TO THE STUDENT

A reasonable question is, "What is discrete mathematics?" Discrete mathematics deals mainly with the analysis of finite collections of objects, unlike continuous mathematics, which is concerned with infinite processes. Sorting is an example of a problem that uses discrete mathematics: How does one arrange a finite set of objects in order? Problems in classical physics belong to the world of continuous mathematics, calculus in particular. One example is: Find the force of water at *every* point against a dam. Since there are infinitely many points on the dam, this problem is studied by continuous methods. Computer science is finite in nature—computers have finite memories, instructions are executed at finite time intervals, programs are finite, and so on—thus computer science finds discrete mathematics extremely useful. However, the applications of discrete mathematics are by no means limited to computer science; operations research, business, engineering, economics, chemistry, political science, and biology, among other disciplines, find discrete mathematics an indispensable tool.

This course will build on concepts that you have already studied. You have undoubtedly studied sets and functions and perhaps logic and mathematical induction previously. These topics are not peculiar to discrete mathematics. However, in discrete mathematics there are fewer problems that require you to plug into a formula than in some of the other mathematics courses you may have taken. The way to succeed is to work many exercises. Practice really makes perfect in discrete mathematics.

About a third of the problems are marked with an "H," which indicates that the solution is given in the back of the book. Before looking in the back of the book, make an honest effort to solve the problem on your own. You will learn far more by working on a problem on your own, even if you do not obtain a complete solution, than by looking at a solution by someone else.

Each chapter concludes with a chapter review. You might try defining each term listed and illustrating its use in a problem. After reviewing the chapter, you can test yourself by taking the chapter self-test. Answers to all self-test questions can be found in the Hints section at the end of the book.

Should you desire other study material, a *Student Study Guide,* written especially to accompany this book, is available. This *Guide* contains solutions to another third of the exercises in the book. Written in a relaxed and informal style, the *Guide* contains additional explanations of examples and solutions to problems along with tips on how to solve problems.

You need not have had any experience with computers to study this book. However, if you know something about computers or about programming, you might be interested in the optional Computer Notes sections, which indicate connections between discrete mathematics and computer science. Exercises that require you to have studied the Computer Notes sections are flagged with the computer icon shown in the margin. Also, there are computer exercises at the end of each chapter.

ACKNOWLEDGMENTS

In writing this book, I have been assisted by many persons, including Henry S. Tropp, Humboldt State University; Joe Chan, DePaul University; Kam-Chan Lo, Pansophic Systems Inc.; Jerrold Grossman, Oakland University; Donald J. Albers, Menlo College; Gregory Bachelis, Wayne State University; Gary Phillips, Oakton Community College; Jane Edgard, Brevard Community College, Florida; Susan Forman, Bronx Community College; and Sadie Bragg, Borough of Manhattan Community College. Special thanks go to the students of DePaul University who used various drafts of this book.

I am indebted to the Department of Computer Science and Information Systems at DePaul University and its chairman, Helmut Epp, for providing time and encouragement for the development of this book.

I am fortunate to be able to work with some of the best people in the publishing industry at Macmillan. John Schultz, Picture Editor, located

the photographs that accompany the biographical sketches. He also contributed to the biographies. I would like to thank Robert Freese, Book Designer, for his innovative design of the book. Elaine Wetterau, Production Supervisor, was instrumental in seeing that the publishing technicalities were executed properly. Bob Clark, Mathematics Editor, provided excellent reviewers and helped me keep track of myriad details. Gary Ostedt, Executive Editor, contributed many unique suggestions based on his wide experience in mathematics publishing and a great deal of warmly welcomed encouragement.

R. J.

Contents

†Can be omitted without loss of continuity.

> But I proved beyond a shadow of a doubt and with geometric logic that a key did exist.
>
> —from *The Caine Mutiny*

1

Logic and Circuits

Chapter 1 examines some concepts of **logic**, specifically, how we determine whether certain statements are true or false. Logical methods are used in mathematics to prove theorems and in computer science to prove that programs do what they are alleged to do. We will see that certain ideas in logic are closely related to the construction of circuits. The chapter concludes by showing how to design simple circuits like those used in digital computers.

1.1 Propositions

Which of the sentences are either true or false (but not both)?

(a) Benny Goodman has recorded classical music.
(b) The line "Play it again, Sam" occurs in the movie *Casablanca*.
(c) Earth is the only planet in the universe that has life.
(d) Buy two tickets to the Ungrateful Living concert for Friday.

Sentence (a) is true. Although Benny Goodman is best known for his jazz recordings, he recorded much classical music (e.g., the Weber Clarinet Concertos, numbers 1 and 2, with the Chicago Symphony Orchestra).

Although it is widely believed that the line "Play it again, Sam" occurs in *Casablanca*, it does not. The line that actually occurs is "Play it, Sam. Play 'As Time Goes By.' " Thus sentence (b) is false.

Sentence (c) is either true or false (but not both), but no one knows which at this time.

Sentence (d) is neither true nor false [(d) is a command].

A sentence that is either true or false, but not both, is called a **proposition**. Sentences (a)–(c) are propositions, whereas sentence (d) is not a proposition. Propositions are the basic building blocks of any theory of logic.

We will use lowercase letters, such as p, q, and r, to represent propositions. We will also use the notation

$$p: 1 + 1 = 3$$

to define p to be the proposition $1 + 1 = 3$.

In ordinary speech and writing, we combine propositions using connectives such as *and* and *or*. For example, the propositions "It is raining" and "I will take my umbrella" can be combined to form the single proposition "It is raining and I will take my umbrella." The formal definitions of *and* and *or* follow.

Definition 1.1.1. Let p and q be propositions.

The *conjunction* of p and q, denoted $p \wedge q$, is the proposition

$$p \text{ and } q.$$

The *disjunction* of p and q, denoted $p \vee q$, is the proposition

$$p \text{ or } q.$$

Propositions such as $p \wedge q$ and $p \vee q$, which result from combining propositions, are called **compound propositions**.

Example 1.1.2. If

$$p: 1 + 1 = 3,$$
$$q: \text{A decade is 10 years,}$$

then the conjunction of p and q is

$$p \wedge q: 1 + 1 = 3 \text{ and a decade is 10 years.}$$

The disjunction of p and q is

$$p \vee q: 1 + 1 = 3 \text{ or a decade is 10 years.}$$

The truth values of propositions such as conjunctions and disjunctions can be described by **truth tables**. The truth table of a proposition P made up of the individual propositions $p_1, \ldots, p_n$ lists all possible combinations of truth values for $p_1, \ldots, p_n$, T denoting true and F denoting false, and for each such combination lists the truth value of P.

Definition 1.1.3. The truth value of the compound proposition $p \wedge q$ is defined by the truth table

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

Notice that in the truth table in Definition 1.1.3 all four possible combinations of truth assignments for p and q are given.

Definition 1.1.3 states that the conjunction $p \wedge q$ is true only if p and q are both true; $p \wedge q$ is false otherwise.

Example 1.1.4. Let p and q be as in Example 1.1.2. Then p is false, q is true, and the conjunction $p \wedge q$ is false.

Example 1.1.5. Let

p: Benny Goodman has recorded classical music,
q: The Baltimore Orioles used to be the St. Louis Browns.

Then p and q are both true and the conjunction

$p \wedge q$: Benny Goodman has recorded classical music and the Baltimore Orioles used to be the St. Louis Browns

is also true.

Example 1.1.6. Let

p: $1 + 1 = 3$,
q: Chicago is in California.

Then p and q are both false and the conjunction

$p \wedge q$: $1 + 1 = 3$ and Chicago is in California

is false.