SWITCHING AND FINITE AUTOMATA THEORY

COMPUTER SCIENCE SERIES

SWITCHING AND FINITE AUTOMATA THEORY

Second Edition

ZVI KOHAVI
Departments of Electrical Engineering
and Computer Science
TECHNION–Israel Institute of Technology

McGraw-Hill Book Company
New York St. Louis San Francisco Auckland Bogotá Düsseldorf
Johannesburg London Madrid Mexico Montreal New Delhi Panama
Paris São Paulo Singapore Sydney Tokyo Toronto

SWITCHING AND FINITE AUTOMATA THEORY

567890 DODO 83210

This book was set in Modern 8A by Bi-Comp, Incorporated.
The editors were Julienne V. Brown and Madelaine Eichberg;
the production supervisor was Charles Hess.
New drawings were done by Fine Line Illustrators, Inc.
R. R. Donnelley & Sons Company was printer and binder.

Library of Congress Cataloging in Publication Data

Kohavi, Zvi.
Switching and finite automata theory.

(McGraw-Hill computer science series)
Includes bibliographical references and index.
1. Sequential machine theory. 2. Switching theory.
I. Title.
QA267.5.S4K64 1978 629.8'9 77-13703
ISBN 0-07-035310-7

SWITCHING AND FINITE
AUTOMATA THEORY

McGRAW-HILL COMPUTER SCIENCE SERIES

EDWARD A. FEIGENBAUM: *Stanford University*
RICHARD W. HAMMING: *Naval Post Graduate School*

ALLEN: Anatomy of LISP
BELL AND NEWELL: Computer Structures: Readings and Examples
DONOVAN: Systems Programming
GEAR: Computer Organization and Programming
GIVONE: Introduction to Switching Circuit Theory
GOODMAN AND HEDETNIEMI: Introduction to the Design and Analysis of Algorithms
HAMACHER, VRANESIC, AND ZAKY: Computer Organization
HAMMING: Introduction to Applied Numerical Analysis
HAYES: Computer Architecture and Organization
HELLERMAN: Digital Computer System Principles
HELLERMAN AND CONROY: Computer System Performance
KAIN: Automata Theory: Machines and Languages
KATZAN: Microprogramming Primer
KOHAVI: Switching and Finite Automata Theory
LIU: Elements of Discrete Mathematics
LIU: Introduction to Combinatorial Mathematics
MADNICK AND DONOVAN: Operating Systems
MANNA: Mathematical Theory of Computation
NEWMAN AND SPROULL: Principles of Interactive Computer Graphics
NILSSON: Problem-solving Methods in Artificial Intelligence
ROSEN: Programming Systems and Languages
SALTON: Automatic Information Organization and Retrieval
STONE: Introduction to Computer Organization and Data Structures
STONE AND SIEWIOREK: Introduction to Computer Organization and Data Structures: PDP-11 Edition
TONGE AND FELDMAN: Computing: An Introduction to Procedures and Procedure-Followers
TREMBLAY AND MANOHAR: Discrete Mathematical Structures with Applications to Computer Science
TREMBLAY AND SORENSON: An Introduction to Data Structures with Applications
TUCKER: Programming Languages
WATSON: Timesharing System Design Concepts
WEGNER: Programming Languages, Information Structures, and Machine Organization
WIEDERHOLD: Database Design
WINSTON: The Psychology of Computer Vision

Preface

The subject of switching and finite automata theory needs no introduction. It has become a part of every computer science and electrical engineering curriculum, and rightly so. It provides techniques useful in a wide variety of applications and helps develop *a way of thinking* that leads to understanding of the structure, behavior, and limitations and capabilities of logical machines. In this book I have tried to cover the whole subject, starting with introductory material and leading to the more advanced topics, assuming a minimal technical background on the part of the reader. I did not attempt to provide detailed techniques for the design of specific circuits, but rather to formulate methods and to develop algorithms that can be applied to a broad class of problems. For once such general principles are understood, the relevance of specific procedures and their applicability to given problems are a matter of engineering decisions.

I have endeavored to provide a logical and rigorous presentation with a minimum of formalism. Accordingly, theorems are proved and algorithms are carefully developed, but only after an intuitive understanding of the procedures involved has been achieved by means of illustrative examples. Throughout I have assumed that the reader of this book is a computer scientist, a logical designer, or a communication or control engineer. Applications and examples are drawn accordingly from these fields. Most of the material is not new, although many subjects appear for the first time in an introductory text. A listing of the main sources upon which I have drawn, as well as some historical notes, is provided at the end of each chapter. These references are just the basic ones, and should not be considered a comprehensive bibliography.

The book is divided into three parts. The first part, which consists of Chapters 1 and 2, provides some introductory background. The second part is devoted to combinational logic, and the third part is concerned with finite automata. The book is organized so that many of its chapters cover specific topics and they are not prerequisite for subsequent chapters.

In this category are Chapters 6, 7, 8, 11, 12, 13, 14, 15, 16, and their selection in a course outline depends on the preferences of the instructor. Sections marked with a star (★) may be omitted without loss of continuity.

The book is self-contained as a text. At present much of this material is customarily taught in graduate courses, but from my experience at the Polytechnic Institute of New York, the Massachusetts Institute of Technology, and the Technion most of it can be taught at the junior or senior level in computer science or electrical engineering departments. The book is intended as a text for a two-semester sequence. The first semester can be devoted to "classical" switching theory [Chapters 1, 3 (Sections 3-1 to 3-4), and 4 through 11], and the second semester devoted primarily to finite automata theory [Chapters 2, and 12 through 16]. Other organizations of the material into one-semester or two-semester courses are possible, keeping in mind the following prerequisites: Chapters 3 (excluding Section 3-5), 4, and 5 are prerequisite for the entire book. Chapters 9 and 10 (excluding Section 10-4) are prerequisite for Chapters 12 through 16.

In revising the book some chapters have undergone a major revision, while others required only minor changes. Chapters 5 and 9 have been updated to reflect the general use of integrated circuits in logical design. Chapters 8 and 13 have been revised and expanded to reflect the importance of testing circuits and of designing more reliable circuits. In an era where computers are used to execute algorithms and perform experiments, it is of utmost importance to establish that the computations will indeed terminate and to provide bounds for their length. The development of bounds on the length of various experiments has therefore been further emphasized by providing new proofs in appendixes and problems in Chapters 10, 13, and 14. The problem sets, which were originally quite extensive, have been further expanded. They range from simple numerical examples to natural extensions of the ideas presented in the text.

The first edition of this book has been used in many universities and departments and I received numerous helpful comments. I am grateful for all of them. My gratitude is due to M. Yoeli of the Technion and E. J. Smith of the Polytechnic Institute of New York, who stimulated my first interest in the subject of switching and automata theory. I also wish to thank I. Kohavi, I. Koren, S. Patil, R. Riesenfeld, and C. L. Seitz for their helpful comments and many valuable suggestions. I am indebted to the electrical engineering department at M.I.T. which provided a stimulating atmosphere for writing the first edition, and to the Technion and the computer science department at the University of Utah for making the revised edition possible.

Special gratitude is due to C. L. Liu of the University of Illinois for his thorough review of the entire manuscript and his invaluably discerning technical criticism, which significantly improved this book. I also thank Karen Evans for her excellent typing of the manuscript and for drawing the figures. Last but not least, I wish to thank my wife, Sima, for her help and understanding.

Zvi Kohavi

Contents

★ Sections marked with a star may be omitted without loss of continuity.

SWITCHING AND FINITE AUTOMATA THEORY

Part one
Preliminaries

1
Number Systems and Codes

This chapter deals with the representation of numerical data, with emphasis on those representations which use only two symbols, 0 and 1. Described are special methods of representing numerical data, affording protection against various transmission errors and component failures.

1-1 NUMBER SYSTEMS

Convenient as the decimal number system generally is, its usefulness in machine computation is limited because of the nature of practical electronic devices. In most present digital machines the numbers are represented, and the arithmetic operations performed, in a different number system, called the binary number system. This section is concerned with the representation of numbers in various systems and with methods of conversion from one system to another.

Number representation

An ordinary decimal number actually represents a polynomial in powers of 10. For example, the number 123.45 represents the polynomial

$$123.45 = 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0 + 4 \cdot 10^{-1} + 5 \cdot 10^{-2}$$

This method of representing decimal numbers is known as the *decimal number system*, and the number 10 is referred to as the *base* (or *radix*) of the system. In a system whose base is b, a positive number N represents the polynomial

$$N = a_{q-1}b^{q-1} + \cdots + a_0b^0 + \cdots + a_{-p}b^{-p}$$
$$= \sum_{i=-p}^{q-1} a_ib^i$$

where the base b is an integer greater than 1, and the a's are integers in the range $0 \leq a_i \leq b - 1$. The sequence of digits $a_{q-1}a_{q-2} \cdots a_0$ constitutes the *integral part* of N, while the sequence $a_{-1}a_{-2} \cdots a_{-p}$ constitutes the *fractional part* of N. Thus p and q designate the number of digits in the fractional and integral parts, respectively. The integral and fractional parts are usually separated by a *radix point.* The digit a_{-p} is referred to as the *least significant digit,* while a_{q-1} is called the *most significant digit.*

When the base b equals 2, the number representation is referred to as the *binary number system.* For example, the binary number 1101.01 represents the polynomial

$$1101.01 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

that is,

$$1101.01 = \sum_{i=-2}^{3} a_i2^i$$

where $a_{-2} = a_0 = a_2 = a_3 = 1$ and $a_{-1} = a_1 = 0$.

A number N in base b is usually denoted $(N)_b$. Whenever the base is not specified, base 10 is implicit. Table 1-1 shows the representations of integers 0 through 15 in several number systems.

The *complement* of a digit a, denoted a', in base b is defined as

$$a' = (b - 1) - a$$

That is, the complement a' is the difference between the largest digit in base b and the digit a. In the binary number system, since $b = 2$, $0' = 1$ and $1' = 0$. In the decimal number system the largest digit is 9. Thus, for example, the complement† of 3 is $9 - 3 = 6$.

Conversion of bases

Suppose that some number N, which we wish to express in base b_2, is presently expressed in base b_1. In converting a number from base b_1 to

† In the decimal system the complement is also referred to as the 9's *complement.* In the binary system it is also known as the 1's *complement.*