Mathematics for Computing

G. P. McKeown
V. J. Rayward-Smith

School of Computing Studies and Accountancy, University of East Anglia

First published 1982 by
THE MACMILLAN PRESS LTD
London and Basingstoke
Companies and representatives
throughout the world

Filmset in 10/12 Monophoto Times by
MID-COUNTY PRESS, LONDON SW15
Printed in Hong Kong

ISBN 0 333 29169 7
ISBN 0 333 29170 0 pbk

Macmillan Computer Science Series

Consulting Editor
Professor F. H. Sumner, University of Manchester

S. T. Allworth, *Introduction to Real-time Software Design*

Ian O. Angell, *A Practical Introduction to Computer Graphics*

G. M. Birtwistle, *Discrete Event Modelling on Simula*

T. B. Boffey, *Graph Theory in Operations Research*

Richard Bornat, *Understanding and Writing Compilers*

J. K. Buckle, *The ICL 2900 Series*

Robert Cole, *Computer Communications*

Derek Coleman, *A Structured Programming Approach to Data**

Andrew J. T. Colin, *Fundamentals of Computer Science*

Andrew J. T. Colin, *Programming and Problem solving in Algol 68**

S. M. Deen, *Fundamentals of Data Base Systems**

J. B. Gosling, *Design of Arithmetic Units for Digital Computers*

David Hopkin and Barbara Moss, *Automata**

Roger Hutty, *Fortran for Students*

H. Kopetz, *Software Reliability*

A. M. Lister, *Fundamentals of Operating Systems, second edition**

G. P. McKeown and V. J. Rayward-Smith, *Mathematics for Computing*

Brian Meek, *Fortran, PL/I and the Algols*

Derrick Morris and Roland N. Ibbett, *The MU5 Computer System*

John Race, *Case Studies in Systems Analysis*

Peter Wallis, *Portable Programming*

I. R. Wilson and A. M. Addyman, *A Practical Introduction to Pascal*

* The titles marked with an asterisk were prepared during the Consulting Editorship of Professor J. S. Rohl, University of Western Australia.

To Mary and Sheila

Preface

The problems that early computers solved were mostly mathematical. Since then, as everyone knows, the use of computers has greatly diversified such that today the majority of applications are non-numerical. Thus, while it is immediately clear that a good mathematical background is essential for the obviously mathematical areas of computing, such as numerical techniques, computer simulation or the theory of computation, the question arises as to whether mathematics is now important in computing as a whole. We believe the answer to this question to be an unqualified yes. The last decade or so has seen the development of computing from little more than a 'bag of tricks' into a science. As in any scientific or engineering discipline, mathematics is the medium through which the underlying concepts and principles of computer science may be understood. Mathematical notation is now found in virtually every branch of computing, from the mathematical areas mentioned above, through compiling techniques, data structures and algorithm design and analysis, to data base systems. Without the ability to appreciate and apply mathematical concepts and techniques, the aspiring computer scientist cannot hope to grasp the fundamental principles of computing, principles that will still be relevant even if the particular programming skills that he has learnt become obsolete.

This book is designed for two types of user. The first is the student starting tertiary education in computing, who will need to develop a reasonable mathematical maturity in order to cope with the use of mathematical notation in subsequent computer science courses. We assume that no such student will be without a reasonable 'A'-level (or equivalent) in mathematics. The book provides a basis for a course equivalent to about one-third of the first year of study in a degree programme.

The second category of reader for whom the book is designed is the practising computer scientist who needs a reference book on his shelf to which he can go when he needs a definition and an example of some concept only vaguely remembered. For this reason, after most definitions in the book, one or more examples are given to illustrate the new term.

While the material in this book is essentially traditional mathematics, it has been given a computer science flavour through the use of algorithms.

The algorithm is the core concept in computing, but one which normally has little place in traditional mathematics. Nevertheless, an algorithm is often the best way of describing well-known mathematical techniques.

Since this book is intended primarily to be of use on a general first mathematics course in a computer science degree programme, it is not constrained to just discrete mathematics. We believe that calculus contains a wealth of results with applications in computing, and for this reason a substantial amount of calculus is presented in the third chapter. Although this chapter is called 'Calculus' it does, in fact, contain some finite mathematics. In particular, the section on series includes both the finite and the infinite cases.

G. P. McKEOWN
V. J. RAYWARD-SMITH

ACKNOWLEDGEMENTS

The authors wish to thank Miss Gillian Hall, who produced all of the computer-generated figures in the book. The figures were generated using the GINO-F general-purpose graphics package on a PRIME 400 computer.

The authors also wish to thank Mr E. W. Haddon for his constructive criticism of the text.

Finally, the authors offer their warmest thanks to Mrs Jane Copeman and to Mrs Jane Loughlin for their excellent and patient typing of a difficult manuscript.

Contents

Preface ix

1 Foundations 1

1.1 Propositional logic 1
1.2 Set theory 9
1.3 Numbers 19
1.4 Complex numbers 37
1.5 Functions 52

2 Linear Algebra 73

2.1 Vectors 73
2.2 Matrices 90
2.3 Systems of linear equations 105
2.4 The solution of systems of linear equations 121
2.5 Determinants 141

3 Calculus 160

3.1 Sequences 160
3.2 Series 173
3.3 Continuous real functions 196
3.4 Differentiation 214
3.5 Integration 235

4 Probability 260

4.1 Introduction 260
4.2 Conditional probability. Multi-step experiments 271
4.3 Independent trials. Discrete probability distributions 287
4.4 Continuous probability distributions 306
4.5 Independent random variables 320
4.6 Computer sampling 333

5 Algebraic Structures 343

5.1 Relations 343
5.2 Digraphs 355

5.3 Groups and semigroups 368
5.4 Rings, fields and vector spaces 377
5.5 Boolean algebras 390

References 406

Solutions to Selected Exercises 407

Index 419

1 Foundations

1.1 PROPOSITIONAL LOGIC

Throughout this book, the mathematical reasoning is presented in the English language, suitably augmented by a collection of special symbols. These symbols are defined not only as a shorthand tool but also for the sake of clarity and precision. In this opening section, notation is introduced to show how a complex statement in English is constructed from simple statements and how, given the truth value of these simple statements, the truth value of the complex statement can be determined.

A simple statement may be represented by a *statement letter*, either an upper-case letter of the roman alphabet (A, B, C, ...) or such a letter with an integer subscript (A_1, A_2, ...). The simple statement

It is raining

might thus be represented by the statement letter, R, while H might be chosen to represent

Today is a holiday

Complex statements are constructed from simple statements using connectives such as: 'not', 'and', 'or', 'implies'. For example, a complex statement constructed from the above simple statements using the connectives 'not' and 'and' is

It is not raining and today is a holiday

Whether this statement is true or not will depend on the truth values of the simple statements used in its construction. For this example, the complex statement constructed from R and H is true if and only if R is false and H is true.

Connectives, Statement Forms and Truth Tables

A statement is either a simple or a complex statement and is represented by a capital, script letter of the roman alphabet ($\mathscr{A}$, $\mathscr{B}$, $\mathscr{C}$, ...).

Given a statement $\mathscr{A}$, $(\sim\mathscr{A})$ [read: not $\mathscr{A}$] represents the statement that is true if $\mathscr{A}$ is false and is false if $\mathscr{A}$ is true. If true is denoted by T

and false by F, then this definition can be summarised in a tabular form

$\mathscr{A}$	$(\sim\mathscr{A})$
T	F
F	T

Such a table is known as a *truth table.*

Given statements $\mathscr{A}$ and $\mathscr{B}$, $(\mathscr{A}\wedge\mathscr{B})$ [read: $\mathscr{A}$ and $\mathscr{B}$] represents the statement that is true if and only if both $\mathscr{A}$ and $\mathscr{B}$ are true. In terms of a truth table

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A}\wedge\mathscr{B})$
T	T	T
T	F	F
F	T	F
F	F	F

Further connectives, $\vee$ [or], $\supset$ [implies] and $\equiv$ [if and only if, sometimes written iff], can also be defined using truth tables

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A}\vee\mathscr{B})$	$(\mathscr{A}\supset\mathscr{B})$	$(\mathscr{A}\equiv\mathscr{B})$
T	T	T	T	T
T	F	T	F	F
F	T	T	T	F
F	F	F	T	T

The definition of $(\mathscr{A}\supset\mathscr{B})$ may seem a little strange at first — the idea that $(\mathscr{A}\supset\mathscr{B})$ should be true whenever $\mathscr{A}$ is false can be puzzling. However, consider the following example. Let $\mathscr{A}$ be the statement

It is a sunny day

and let $\mathscr{B}$ be the statement

It is daytime

No one would dispute that statement $\mathscr{A}$ implies statement $\mathscr{B}$, even though $\mathscr{A}$ is, in fact, often false. However, $\mathscr{A}$ can never be true when $\mathscr{B}$ is false. In general, therefore, $(\mathscr{A}\supset\mathscr{B})$ is taken to be true when either $\mathscr{A}$ is false or both $\mathscr{A}$ and $\mathscr{B}$ are true. If $\mathscr{A}$ is true and $\mathscr{B}$ is false then $(\mathscr{A}\supset\mathscr{B})$ is taken to be false.

The five connectives ($\sim$, $\vee$, $\wedge$, $\supset$, $\equiv$) are used to combine simple statements into complex statements whose truth values will usually depend on the truth values of the constituent simple statements. A statement may thus be represented by an expression composed of

statement letters and connectives. Such an expression is called a statement form. For example, the statement

It is not raining and today is a holiday

can be represented by the statement form $((\sim R)\wedge H)$.

A *statement form* can be formally defined as follows.

Definition 1.1.1
(a) All statement letters (capital roman letters and such letters with numerical subscripts) are statement forms.
(b) If $\mathscr{A}$ and $\mathscr{B}$ are statement forms, then so are $(\sim\mathscr{A})$, $(\mathscr{A}\wedge\mathscr{B})$, $(\mathscr{A}\vee\mathscr{B})$, $(\mathscr{A}\supset\mathscr{B})$ and $(\mathscr{A}\equiv\mathscr{B})$.
(c) Only those expressions that are determined by (a) and (b) are statement forms.

In ordinary algebra, $a+b\times c+d$ may be used to represent $((a+(b\times c))+d)$ since there is a convention for the restoration of parentheses based on the priorities of operators. There is a similar convention in propositional logic which works according to the following rules.

(a) The connectives

$$\sim,\ \wedge,\ \vee,\ \supset,\ \equiv$$

are ranked from left to right, with $\sim$ having the highest priority.
(b) If there is only one connective in a statement form then parentheses can be omitted by association to the left.
(c) The outer parentheses can be omitted.

Example 1.1.1
Removing parentheses from $((A\supset(\sim B))\supset(C\vee(\sim A)))$ results in the following steps

$$(A\supset(\sim B))\supset(C\vee(\sim A))$$

$$A\supset(\sim B)\supset(C\vee(\sim A))$$

$$A\supset(\sim B)\supset C\vee(\sim A)$$

$$A\supset\sim B\supset C\vee\sim A$$

Of course, it is not always possible to remove all the parentheses from a statement form and indeed, for the sake of readability, it is often not desirable to remove the maximum number of parentheses.

Given any statement form, $\mathscr{A}$, and an assignation of truth values to the individual statement letters occurring in $\mathscr{A}$, a truth value for $\mathscr{A}$ can be deduced. It is possible to tabulate all possible such assignations of truth values to the individual statement letters, giving the resulting truth values of the statement form.

Example 1.1.2 The truth table for $A \supset \sim B \supset C \vee \sim A$

A	B	C	$\sim A$	$C \vee \sim A$	$\sim B$	$A \supset \sim B$	$A \supset \sim B \supset C \vee \sim A$
T	T	T	F	T	F	F	T
T	T	F	F	F	F	F	T
T	F	T	F	T	T	T	T
T	F	F	F	F	T	T	F
F	T	T	T	T	F	T	T
F	T	F	T	T	F	T	T
F	F	T	T	T	T	T	T
F	F	F	T	T	T	T	T

A more compact form of the truth table can be achieved by writing the truth values of statement letters immediately beneath them and writing the truth values constructed using a particular connective immediately beneath that connective. The final result is indicated by means of the symbol ↑. Using this technique for the above table results in a table of the following form.

A	$\supset$	$\sim$	B	$\supset$	C	$\vee$	$\sim$	A
T	F	F	T	T	T	T	F	T
T	F	F	T	T	F	F	F	T
T	T	T	F	T	T	T	F	T
T	T	T	F	F	F	F	F	T
F	T	F	T	T	T	T	T	F
F	T	F	T	T	F	T	T	F
F	T	T	F	T	T	T	T	F
F	T	T	F	T	F	T	T	F
				↑				

Tautologies and Contradictions

Definition 1.1.2
A statement form that is always true no matter what truth values are assigned to its statement letters is called a *tautology*. The truth table for a tautology thus has only T occurring in the final column calculated.

A *contradiction* is the opposite of a tautology, with only F appearing in the final column calculated.

An immediate consequence of these definitions is the result that $\mathscr{A}$ is a tautology if and only if $\sim \mathscr{A}$ is a contradiction.

Example 1.1.3
$(A \wedge B) \supset A$ is a tautology since its truth value yields a column consisting entirely of T values.

(A	∧	B)	⊃	A
T	T	T	T	T
T	F	F	T	T
F	F	T	T	F
F	F	F	T	F
			↑	

A complex statement in English that can be derived from a tautology by substituting English statements for statement letters, such that each occurrence of a particular statement letter is replaced by the same English statement, is said to be *logically true* (*according to propositional logic*). Similarly, a complex English statement arising from substitution into a contradiction is said to be *logically false* (*according to propositional logic*).

In example 1.1.3, it was shown that $(A \wedge B) \supset A$ is a tautology. If A is replaced by 'the sun shines' and B by 'the grass grows' then the result is the logically true statement, 'the sun shines and the grass grows implies the sun shines.'

Definition 1.1.3

Two statement forms, $\mathscr{A}$ and $\mathscr{B}$, are said to be *equivalent* if and only if $(\mathscr{A} \equiv \mathscr{B})$ is a tautology. $\mathscr{A}$ *implies* $\mathscr{B}$ if and only if $(\mathscr{A} \supset \mathscr{B})$ is a tautology.

Theorem 1.1.1

$\mathscr{A}$ and $\mathscr{B}$ are equivalent if and only if $\mathscr{A}$ implies $\mathscr{B}$ and $\mathscr{B}$ implies $\mathscr{A}$.

Proof There are two parts to this proof.

The first part is to show that $\mathscr{A}$ and $\mathscr{B}$ are equivalent if $\mathscr{A}$ implies $\mathscr{B}$ and $\mathscr{B}$ implies $\mathscr{A}$. If $\mathscr{A}$ is true then, since $(\mathscr{A} \supset \mathscr{B})$ is a tautology, it follows that $\mathscr{B}$ is true. If $\mathscr{A}$ is false then, since $(\mathscr{B} \supset \mathscr{A})$ is a tautology, $\mathscr{B}$ must be false. Hence, $(\mathscr{A} \equiv \mathscr{B})$ is a tautology.

The second part is to show that $\mathscr{A}$ and $\mathscr{B}$ are equivalent only if $\mathscr{A}$ implies $\mathscr{B}$ and $\mathscr{B}$ implies $\mathscr{A}$, that is, if $\mathscr{A}$ and $\mathscr{B}$ are equivalent then it follows that $\mathscr{A}$ implies $\mathscr{B}$ and $\mathscr{B}$ implies $\mathscr{A}$. If $(\mathscr{A} \equiv \mathscr{B})$ is a tautology then, if $\mathscr{A}$ is true, $\mathscr{B}$ must be true and, if $\mathscr{A}$ is false, then $\mathscr{B}$ must be false. In either case, $(\mathscr{A} \supset \mathscr{B})$ and $(\mathscr{B} \supset \mathscr{A})$ are tautologies.

A more concise proof of this theorem can be achieved by rewriting it in propositional logic. The theorem states that $(\mathscr{A} \equiv \mathscr{B}) \equiv (\mathscr{A} \supset \mathscr{B}) \wedge (\mathscr{B} \supset \mathscr{A})$ and it is left as an exercise for the reader to check that this is a tautology by using a truth table.

Definition 1.1.4

Two English statements represented by $\mathscr{A}$ and $\mathscr{B}$ are said to be *logically equivalent* if and only if $\mathscr{A}$ and $\mathscr{B}$ are equivalent. The statement represented by $\mathscr{A}$ is said to *logically imply* the statement represented by $\mathscr{B}$ if and only if $\mathscr{A}$ implies $\mathscr{B}$.

Proving two English sentences to be logically equivalent is particularly important.

Example 1.1.4
Show that 'It is not raining or snowing so it is sunny' is logically equivalent to 'It is not raining and it is not snowing so it is sunny'.

If A represents 'It is raining', B represents 'It is snowing' and C represents 'It is sunny', then the first sentence is represented by $\sim(A \vee B) \supset C$ and the second sentence is represented by $(\sim A \wedge \sim B) \supset C$. The truth table below shows the two statement forms to be equivalent.

$\sim$	$(A$	$\vee$	$B)$	$\supset$	C	$\equiv$	$(\sim$	A	$\wedge$	$\sim$	$B)$	$\supset$	C
F	T	T	T	T	T	T	F	T	F	F	T	T	T
F	T	T	T	T	F	T	F	T	F	F	T	T	F
F	T	T	F	T	T	T	F	T	F	T	F	T	T
F	T	T	F	T	F	T	F	T	F	T	F	T	F
F	F	T	T	T	T	T	T	F	F	F	T	T	T
F	F	T	T	T	F	T	T	F	F	F	T	T	F
T	F	F	F	T	T	T	T	F	T	T	F	T	T
T	F	F	F	F	F	T	T	F	T	T	F	F	F
						↑							

Adequate Sets of Connectives

A truth table involving n statement letters $A_1, \ldots, A_n$ will consist of 2^n rows. If a truth value is arbitrarily assigned to each row, the following question naturally arises: can a statement form involving $A_1, \ldots, A_n$ and the connectives $\sim$, $\vee$, $\wedge$, $\supset$ and $\equiv$ be found whose truth values correspond to the assigned column of truth values? For example, consider a truth table involving three statement letters A_1, A_2, A_3 and assume the last column arbitrarily chosen as below.

A_1	A_2	A_3	
T	T	T	T
T	T	F	F
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	F
F	F	F	T

What statement form would then give a truth table with this last column? One way to construct such a statement is to use algorithm 1.1.1.

Algorithm 1.1.1

if last column has no T values **then** the statement form required is $A_1 \wedge \sim A_1$
else for each row i where the last column has value T **do**
  for j **from** 1 **in steps of** 1 **to** n **do**
    if A_j has value T **then**
      $U_j^i \leftarrow A_j$
    else
      $U_j^i \leftarrow \sim A_j$
    endif
  endfor
  $\mathscr{C}_i \leftarrow U_1^i \wedge U_2^i \wedge \ldots \wedge U_n^i$
endfor
The required statement form is given by combining the various $\mathscr{C}_i$ s using the connective $\vee$.
endif

For the above example, $\mathscr{C}_1$ is $A_1 \wedge A_2 \wedge A_3$, $\mathscr{C}_5$ is $\sim A_1 \wedge A_2 \wedge A_3$, $\mathscr{C}_6$ is $\sim A_1 \wedge A_2 \wedge \sim A_3$ and $\mathscr{C}_8$ is $\sim A_1 \wedge \sim A_2 \wedge \sim A_3$. The required statement form is thus

$$(A_1 \wedge A_2 \wedge A_3) \vee (\sim A_1 \wedge A_2 \wedge A_3) \vee (\sim A_1 \wedge A_2 \wedge \sim A_3) \vee (\sim A_1 \wedge \sim A_2 \wedge \sim A_3)$$

The construction above shows that $\sim$, $\vee$ and $\wedge$ form an adequate set of connectives in the sense that every truth table corresponds to some statement form constructed using just these connectives.

A stronger result can be obtained by noting that $(\mathscr{A} \vee \mathscr{B})$ is equivalent to $\sim(\sim \mathscr{A} \wedge \sim \mathscr{B})$, so every occurrence of $\vee$ can be replaced using $\sim$ and $\wedge$. Hence just $\sim$ and $\wedge$ form an adequate set of connectives. Similarly, by noting the equivalence of $(\mathscr{A} \wedge \mathscr{B})$ and $\sim(\sim \mathscr{A} \vee \sim \mathscr{B})$ one can see that $\sim$ and $\vee$ also form an adequate set of connectives.

The connective $\vee$ is sometimes known as a *disjunction* and the connective $\wedge$ as a *conjunction*. If a statement form $\mathscr{C}$ can be written as $\mathscr{A} \vee \mathscr{B}$ then $\mathscr{A}$, $\mathscr{B}$ are known as *disjuncts*. Similarly, if $\mathscr{C}$ can be written as $\mathscr{A} \wedge \mathscr{B}$, $\mathscr{A}$ and $\mathscr{B}$ are known as *conjuncts*.

Definition 1.1.5
A statement form is in *disjunctive normal form* if it is a disjunction of one or more disjuncts, each of which is a conjunction of one or more statement letters and negations of statement letters.

Algorithm 1.1.1 shows that for every statement form there is an equivalent statement form in disjunctive normal form.

Exercise 1.1

1. Write the following as statement forms, using statement letters to stand for simple statements.

(a) If John is good, Mark is bad and if Mark is bad, John is good.
(b) A sufficient condition for John to be good is that Mark is bad.
(c) A necessary condition for John to be good is that Mark is bad.

2. Determine whether each of the following statement forms is a tautology, contradiction or neither.

(a) $A \equiv A \vee B$
(b) $(A \supset B) \wedge B \supset A$
(c) $\sim(A \supset (B \supset A))$
(d) $A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C))$
(e) $A \supset ((B \supset \sim A) \supset \sim B)$

3. Show that the following pairs of statements are equivalent.

(a) $\mathscr{A}$ and $\sim\sim\mathscr{A}$
(b) $\sim(\mathscr{A} \vee \mathscr{B})$ and $\sim\mathscr{A} \wedge \sim\mathscr{B}$
(c) $\sim(\mathscr{A} \wedge \mathscr{B})$ and $\sim\mathscr{A} \vee \sim\mathscr{B}$
(d) $\mathscr{A} \wedge (\mathscr{B} \vee \mathscr{C})$ and $(\mathscr{A} \wedge \mathscr{B}) \vee (\mathscr{A} \wedge \mathscr{C})$
(e) $\mathscr{A} \vee (\mathscr{B} \wedge \mathscr{C})$ and $(\mathscr{A} \vee \mathscr{B}) \wedge (\mathscr{A} \vee \mathscr{C})$

4. Find a statement form $\mathscr{A}$ constructed from the statement letters A, B, C corresponding to the following truth table.

A	B	C	$\mathscr{A}$
T	T	T	F
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	T
F	F	T	F
F	F	F	F

5. Define the connective | [alternative denial] by

$\mathscr{A}$	$\mathscr{B}$	$\mathscr{A} \mid \mathscr{B}$
T	T	F
T	F	T
F	T	T
F	F	T