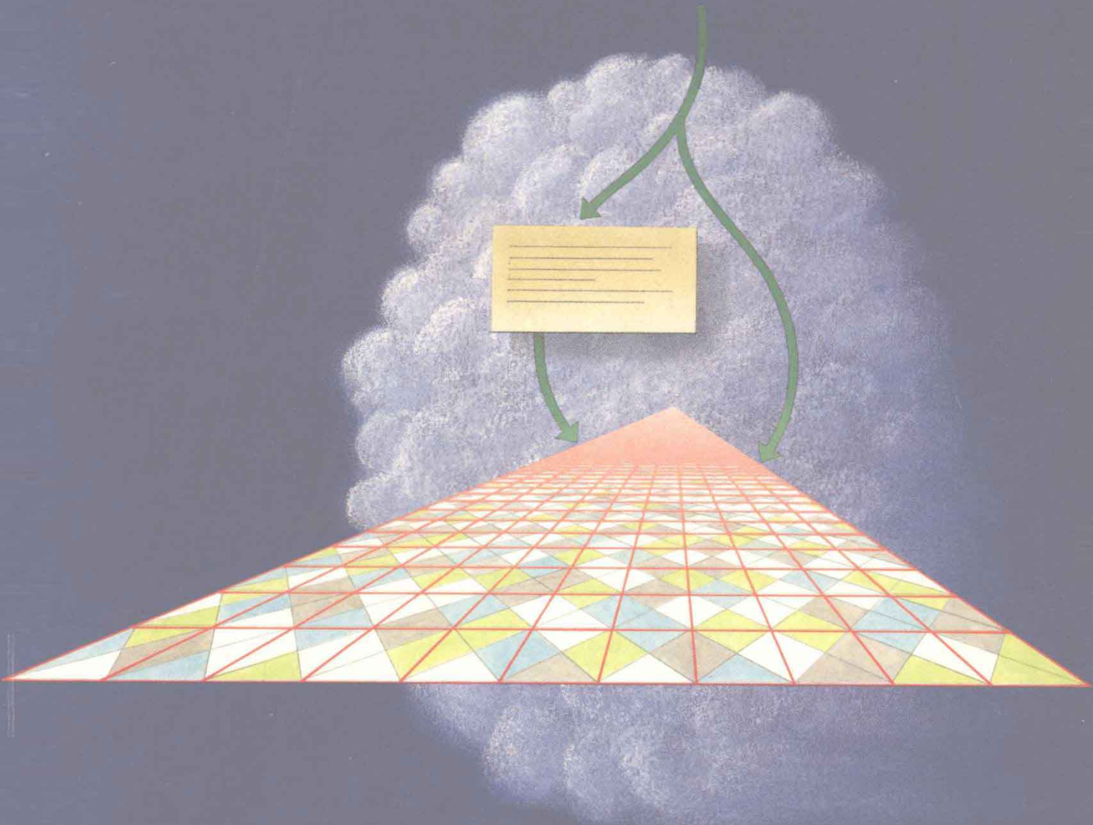
ALGORITHMICS

The Spirit of Computing



DAVID HAREL

ALGORITHMICS
The Spirit of Computing

DAVID HAREL
The Weizmann Institute of Science, Rehovot, Israel

Not by might, nor by power, but by my spirit
Zechariah 4: 6

for the spirit of the . . . creature was in the wheels
When those moved, these moved;
and when those stood still, these stood still
Ezekiel 1: 20, 21

ADDISON-WESLEY PUBLISHING COMPANY
Wokingham, England · Reading, Massachusetts · Menlo Park, California
New York · Don Mills, Ontario · Amsterdam · Bonn · Sydney
Singapore · Tokyo · Madrid · Bogota · Santiago · San Juan

Cover design by John Gibbs from an idea suggested by the author.
Illustrations by Chartwell Illustrators.
Typeset by MCL Computerset Ltd., Ely, Cambs.
Printed in Great Britain by T J Press (Padstow) Ltd., Cornwall.

First printed 1987.
Reprinted 1987.

British Library Cataloguing in Publication Data
Harel, David
Algorithmics : the spirit of computing.
1. Electronic data processing
2. Computers
I. Title
004 QA76

ISBN 0–201–19240–3

Library of Congress Cataloging in Publication Data
Harel, David, 1950-
Algorithmics : the spirit of computing.

Bibliography; p.
Includes index.
1. Electronic data processing. 2. Algorithms.
I. Title.
QA76.H2833 1987 004 86-32038
ISBN 0–201–19240–3

ALGORITHMICS

To my dear parents,
Joyce and Harold Fisch,
for the first 20 years

and to Varda,
for all the rest

Preface

Read this, I pray thee
Isaiah 29: 12

This book tells a story. The story concerns the concepts, ideas, methods and results fundamental to computer science. It is not specifically about computer technology, nor is it about computer programming, though obviously it is heavily influenced by both.

The book is intended to fill a crucial gap in the literature related to the computer revolution. Scores of excellent books can be found on computers themselves, with details of their structure, workings and operation. There are also numerous books about the act of writing programs for computers in any of a growing number of languages. These books come at a wide range of levels, some aimed at people with no computer-related background at all, and some aimed at the most computer-literate professionals. In addition, there are many books on subjects peripheral to the technology, such as the social and legal aspects of the revolution, as well as books describing the relevance of computers to a variety of application areas. All this comes as no surprise. People are curious about computers, and want to learn how to put them to use. They are typically interested in specific kinds of computers, and often for specific purposes too.

Then there are textbooks. Indeed, computer science is a fast-growing academic discipline, with ever-larger numbers of potential students knocking at the doors of admission offices. Well-established academic disciplines have a habit of yielding excellent textbooks, and computer science is no exception. Over the years many comprehensive and clearly written textbooks have appeared, containing detailed technical accounts of the subjects deemed appropriate to students of computer science. However, despite the dizzying speed with which some of the *technological* innovations become obsolete and are replaced by new ones, the fundamentals of the *science* of computation, and hence many of the basic concepts that are considered important in a computer science curriculum, change slowly, if at all. Of course, new technologies and new languages require revisions in scientific emphasis, which are eventually reflected in the scientific literature. However, by and large, there is almost universal agreement on a core of fundamental topics that computer science students should be taught.

It would appear that anyone associated with computers ought to be aware of these topics, and not only those who have decided to spend three or four years getting a particular kind of academic diploma. Moreover, given that a revolution is indeed taking place before our very eyes, many of these topics, and the special ways of thinking that go with them, ought to be available to the enquiring person even if that person is not directly associated with a computer at all.

Books concerned primarily with computers or programming are intended to fulfil quite different needs. Computers are made of bits and bytes, and programming is carried out using languages with rigid rules of grammar and punctuation. Consequently, computer books often suffer from the 'bit/byte syndrome' and programming books from the 'semicolon syndrome'. In other words, the reader becomes predominantly involved in the principles of a particular computer or a particular programming language (or both). It would seem that things cannot be explained without first describing, in detail, either a machine or a medium for communicating with one (or both).

Many advanced textbooks *do* treat the fundamentals, but by their very nature they concentrate on specific topics, and do so at an advanced technical level that is usually unsuitable for the general reader. Even professional programmers and systems analysts might lack the background or motivation required to get through books aimed at full-time computer science students.

Curiously, there appears to be very little written material devoted to the *science* of computing and aimed at the technically oriented general reader as well as the computer professional. This fact is doubly curious in view of the abundance of precisely this kind of literature in most other scientific areas, such as physics, biology, chemistry and mathematics, not to mention humanities and the arts. There appears to be an acute need for a technically detailed, expository account of the fundamentals of computer science; one that suffers as little as possible from the bit/byte or semicolon syndromes and their derivatives, one that transcends the technological and linguistic whirlpool of specifics, and one that is useful both to a sophisticated layperson and to a computer expert. It seems that we have all been too busy with the revolution to be bothered with satisfying such a need.

This book is an attempt in this direction. Its objective is to present a readable account of some of the most important and basic topics of computer science, stressing the fundamental and robust nature of the science in a form that is virtually independent of the details of specific computers, languages, and formalisms.

* * *

This book grew out of a series of lectures given by the author on 'Galei Zahal', one of Israel's national radio channels, between October 1984 and January 1985. It is about what shall be called **algorithmics** in this book – that is, the study of algorithms. An algorithm is an abstract recipe, prescribing a process that might be carried out by a human, by a computer, or by other means. It thus represents a very general concept, with numerous applications. Its principal interest and use, however, is in those cases where the process is to be carried out by a computer.

The book can be used as the basis of a one-semester introductory course in computer science or as a text for a general computer science literacy course in science and engineering schools. Moreover, it can be used as supplementary reading in many kinds of computer-related educational activities, from basic programming courses to advanced graduate or undergraduate degree programs in computer science. The material covered herein, while not directly aimed at producing better programmers or system analysts, can aid people who work with computers by providing an overall picture of some of the most fundamental issues relevant to their work.

* * *

The preliminary chapters discuss the concept of an **algorithmic problem** and the **algorithm** that solves it, followed by cursory discussions of the **structure** of algorithms, the **data** they manipulate, and the languages in which they are programmed. With the ground thus set, Part Two of the book turns to some general methods and paradigms for algorithmic design. This is followed by two chapters on the analysis of algorithms, treating, respectively, their **correctness** and **efficiency** (mainly time efficiency), including techniques for establishing the former and estimating the latter. Part Three of the book is devoted to the **inherent limitations** of effectively executable algorithms, and hence of the computers that implement them. Certain precisely defined problems, including important and practical ones, are shown to be **provably** not solvable by any computers of reasonable size in any reasonable amount of time (say, the life time of a person), and never will be. Worse still, it is shown that some problems are provably not solvable by computers at all, even with unlimited time! In Part Four of the book the requirements are relaxed – for example, by employing concurrent activities or coin tossing, in order to overcome some of these difficulties. Finally, the relationship of computers to human intelligence is discussed, emphasizing the 'soft' *heuristic*, or intuitive, nature of the latter, and the problems involved in relating it to the 'hard' scientific subject of algorithmics.

The book is intended to be read sequentially, not to be used as a reference. It is organized so that each chapter depends on the previous ones, but with smooth readability in mind. Most of the material in the preliminary

Part One should be familiar to people with a background in programming. Thus, Chapters 1 and 2 and parts of Chapter 3 can be browsed through superficially by such readers. Starting with Chapter 4, however, the material becomes increasingly harder as it proceeds, until Chapter 12 is reached, which is of somewhat lighter nature.

** Certain sections contain relatively technical material and can be skipped by the reader without too much loss of continuity. They are set in smaller type and are enclosed between asterisks. It is recommended, however, that even those sections be skimmed, at least to get a superficial idea of their contents. **

Whenever appropriate, brief discussions of the research topics that are of current interest to computer scientists are included. The text is followed by a section of detailed bibliographic notes for each chapter, with 'backward' pointers connecting the discussions in the text with the relevant literature.

* * *

It is hoped that this book will facilitate communication between the various groups of people who are actively involved in the computer revolution, and between them and those who, for the time being, are observers only.

David Harel
Pittsburgh
February 1987

Write the vision, and make it plain upon tablets,
that he[†] *who reads it may run*
Habakkuk 2: 2

[†] Relying on no less an authority than The Bible itself, the masculine forms 'he', 'him', 'his', etc., will be used whenever the more precise, but far more awkward, forms 'she/he', 'her/him', 'hers/his', etc., are intended.

Acknowledgements

Therefore will I give thanks to thee
Psalm 18: 50

Parts of this book were written while I was visiting Digital Equipment Corporation's Systems Research Center in Palo Alto in the Summer of 1985 and Carnegie-Mellon University's Computer Science Department for the 1986/7 academic year. I would like to express my deepest gratitude for these opportunities to work on the book undisturbed.

T. Yuval, Managing Editor of the *Broadcast University* programs on the Israeli radio channel 'Galei Zahal', deserves special thanks for convincing me to prepare the lecture series out of which this book grew.

I am indebted to my colleagues at the Weizmann Institute, A. Pnueli, A. Shamir and S. Ullman, for discussions related to the material appearing herein. It is amazing how one's whole approach can benefit from being surrounded by researchers of such calibre.

I am most grateful to M. Ben-Ari, S. D. Brookes, A. Fiat and M. Vardi, as well as to the three anonymous reviewers commissioned by Addison-Wesley, who read significant parts of the manuscript, identified errors, and provided helpful and insightful feedback. I would also like to thank the following people, who all took time to read and comment on parts of the manuscript: H. Berliner, A. K. Chandra, N. Dershowitz, A. Heydon, C. A. R. Hoare, Z. Reisel, E. Roberts, S. Safra, D. Sherman, R. Sherman, B. Simons, D. Sleator, D. Tygar, P. Wegner and L. Zuck. In addition, collective thanks go to the many people I have pestered recently with questions pertaining to the bibliography.

I would like to thank Y. Barbut for his superb graphical work on the figures, and S. Fliegelman and C. Weintraub for helping me to struggle with the task of TEXing the draft manuscript. I am grateful to Addison-Wesley's S. Troth and A. King for being such effective sponsoring editors, and to D. Myson-Etherington for the endless hours she has devotedly spent on the project throughout its production.

As usual, my wife Varda and our children, Sarit, Hadas, Efrat and Yair, put up with everything in the best imaginable way, and provided the warm and loving environment that is so important for such an endeavour.

Table of Contents

Declare the things that are to come hereafter
Isaiah 41: 23

PART ONE

PRELIMINARIES

Now, these are the foundations

II Chronicles 3: 3

1 Introduction and Historical Review

OR

What's It All About?

Though thy beginning was small,
yet thy end will be very great
Job 8: 7

Computers are amazing machines. They seem to be able to do anything. They fly aircraft and spaceships, and control power stations and hazardous chemical plants. Companies can no longer be run without them, and a growing number of sophisticated medical procedures cannot be performed in their absence. They serve lawyers and judges who seek judicial precedents in scores of documented trials, and help scientists in performing immensely complicated and involved mathematical computations. They route and control millions of telephone calls in networks that span continents, and are used for map reading, typesetting, graphical picture processing and integrated circuit design. They can relieve us of many boring chores, such as keeping a meticulous track of home expenses, and at the same time provide us with exciting alternatives, such as realistic computer games or computerized music. Also, computers of today are hard at work helping design the even more powerful computers of tomorrow.

In short, computers are awesome and fascinating, but at the same time indispensable. For many of us they are also intimidating, and for others threatening. Whatever the case, we cannot afford to be indifferent to them. They are here to stay, and the remarkable growth in their rate of

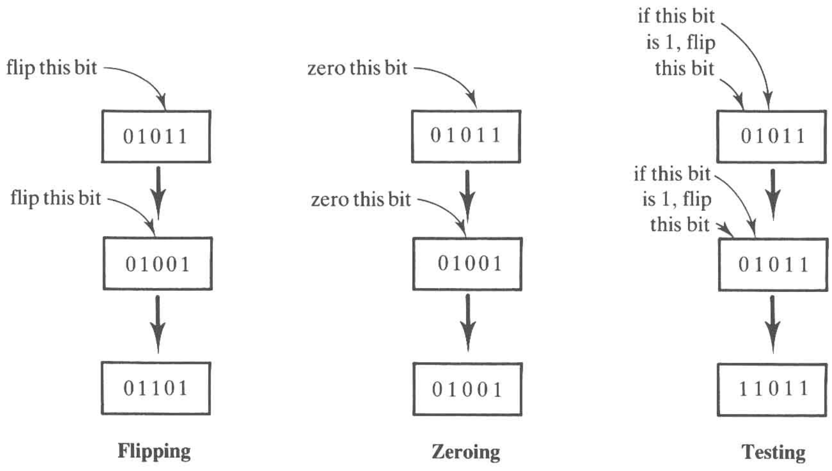


Figure 1.1 Operations on bits.

performance and range of applications shows no signs of coming to a halt.

It is all the more remarkable, therefore, that the digital computer, even the most modern and complex one, can be thought of as merely a large collection of switches. These switches, or **bits** as they are called in technical jargon, are not 'flipped' by the user, but are special, internal switches that are 'flipped' by the computer itself. Each bit can be in one of two positions, or, to put it another way, can take on one of two **values**, 0 or 1. Typically, the value of a bit is determined by some electronic characteristic, such as whether a certain point has a positive or negative charge.

Any computer can directly execute only a small number of extremely trivial operations, like flipping, zeroing, or testing a bit. Flipping changes the bit's value, zeroing makes sure that the bit ends up in the 0 position, and testing does one thing if the bit is already in the 0 position, and another if it is not (see Figure 1.1). Different computers differ in their size (i.e., the number of available bits), in the types of elementary operations they can perform, in the speed in which these operations are performed, in the physical media that embody the bits and their internal organization, and, significantly, in their external environment. This last item means that two computers, which are otherwise identical, might seem very different to an observer: one might be similar in appearance to a television set with a keyboard, and the other might be buried under the dials and knobs of an automatic knitting machine. In a sense, such peripheral but highly visible objects are far less important as components of the computer than the bits and their internal arrangement. It is the bits that 'sense' the external stimuli arriving from the outside world via buttons, levers, keys on a keyboard, electronic communication lines, and even microphones and cameras. It is the bits that 'decide' how to react to these stimuli, ultimately causing other stimuli to be sent back outside via displays, screens, printers, and even loudspeakers, beepers, levers and cranks.

How do they do it? What is it that spans the immense distance between such trivial operations on bits and the incredible feats that computers are capable of? The answer lies in the central concepts treated in this book: the **process**, and the **algorithm** that controls it and causes it to take place.

Some Gastronomy

Imagine a kitchen, containing a supply of ingredients, an array of baking utensils, an oven, a (human) baker, etc. Baking a delicious raisin cake is a process that is carried out *from* the ingredients, *by* the baker, *with* the aid of the oven, and, most significantly, *according* to the recipe. The ingredients are the **inputs** to the process, the cake is its **output**, and the recipe is the **algorithm**. In other words, the algorithm prescribes the activities that constitute the process. The recipes, or algorithms, relevant to a set of processes under discussion are often gathered under the general term **software**, whereas the utensils and oven are generally called **hardware**. The baker, in this case, can also be considered part of the hardware (see Figure 1.2).

In analogy with the simplicity of the bit operations that a computer

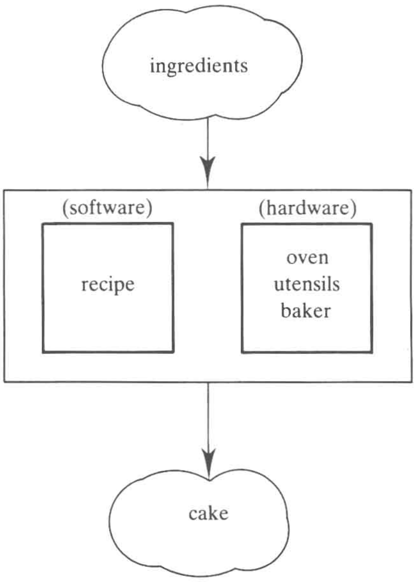


Figure 1.2 Baking a cake.