tables needed for bottom up parsing. This material is not particularly basic or mathematically elegant, but unfortunately is needed to design bottom up parsers.

Chapter 8 is a theoretical study of deterministic parsing that explores in more depth some of the concepts introduced in Volume I. Chapters 9, 10, and 11 cover the material on "compiling" indicated by the subtitle. The topics discussed are translations, bookkeeping and optimization. In general this material is not as well understood theoretically as parsing theory, but the presentation is at a higher level of rigor than in most other compiler books. For example, the section on optimizing the code generated for arithmetic expressions without common subexpressions proves the optimality of the generated code under a variety of cost criteria including program length and number of accumulators used.

In the chapters on "compiling" the authors tended to be somewhat more selective in their choice of material than in the chapters on parsing which are almost encyclopedic in their completeness.

One possible source of confusion for some readers occurs occasionally throughout the book and is more prevalent in Chapter 9. The authors do not distinguish carefully enough in some places what is "true" in a mathematical sense about compilers and what is current practice and only "true" about some compilers in existence today.

The book contains many illustrative examples and a wide variety of exercises graded for difficulty.

In summary, this two volume work sets a new standard in books on compiler design.

PHILIP M. LEWIS
Corporate Research and Development
General Electric Company
Schenectady, N.Y. 12305


Historians tell us that in ancient times the usual mode of operation of a Computer System Manufacturer was for an electronic engineer to emerge from his laboratory one day and tell the waiting programmer "Look! Here's our new model!" The engineer then disappeared to work on the next-but-one model, leaving the programmer to work around any deficiencies or awkwardness in the hardware design which had been handed to him. In modern times, the idea that a machine should be designed to fulfill a function, that the intended use of the machine should supply constraints on its design (rather than the converse), has received at least lip service in most quarters. Some manufacturers have even adopted it as a way of life.

Professor Organick's book reflects this modern approach. In response to the question "What do we want the system to do?" he answers "We want it to execute a (time-invariant) algorithm, expressed in a block-structured language, calling upon data and other items from a (time-varying) environment." Thus, in the smallest view, the hardware processor functions by maintaining a pair of pointers, one to the algorithm (the instruction pointer) and one to the environment. As a block or procedure in the algorithm is entered, or exited, the environment changes with the scope of the variable names. Chapters 2 and 3 introduce and elaborate these concepts, by asking at each step "What information must be available to enable the algorithm to be executed?"

However, the idea of an algorithm as a sequence of procedural steps is regarded as inadequate for the 1970's. Rather, it is proposed that a task be defined in this manner, and that the definition of an algorithm be generalized, so that it consists of a structure (often nested) of interdependent, normally asynchronous tasks. Chapters 4 and 5 develop the requirements for implementing this generalization, that is, requirements for multitasking. The treatment is still on the level of asking what information must (somehow) be made known, to what task, and when.

In traditional treatments, the introduction of interrupts represents a dramatic new concept to be grasped by the reader. However, Professor Organick regards interrupts as "merely unexpected procedure calls," so the treatment of interrupts in Chapter 6 can be quite brief. The emphasis here is on software interrupts, and the author covers some of the philosophical problems encountered, such as interrupting a "sleeping task."

Chapter 7 discusses storage control strategies and Chapter 8 is a discussion of the pros and cons of the computer organization which has emerged from the considerations of earlier chapters. It will be no surprise to readers of this review that a computer organized in this manner is commercially available, and is known as the Burroughs B6700. Chapter 9, which is written by J. S. Cleary and has the nature of an appendix, reveals some of the hardware details of the B6700 implementation. As a further example of the flavor of the book, we note that it is only in this last chapter that we are told that some of the information necessary for the execution of the algorithm is kept in primary storage (main memory), and some in temporary storage (the processor registers).

The publication of this book is a highly significant event, reflecting the relocation of the hardware/software interface which has occurred in recent years, and the trend toward "higher level language processors." Readers who expect an exposition on the level of the machine's instruction set will be disappointed. Such readers should consult Bell and Newell's excellent book (see review B72-3). Organick's book might well be regarded as complementary to that of Bell and Newell. For example, a student baffled by Bell and Newell's comment that the language ALGOL is the antecedent of the B5000, will understand that comment fully after reading Organick's book.

Some minor negative comments should be made. The emphasis on multitasking may be too great, in view of the author's admission that "as of 1972 there has hardly been amassed any abundance of applications programming experience with tasking...". The concept of interrupts as "merely unexpected procedure calls" is too superficial for critical real-time problems. The comment in Chapter 2 that languages like FORTRAN "can be regarded as degenerate examples of block structured languages" implies a promise which is broken in Chapter 8, which discusses why FORTRAN programs do not execute rapidly on the machine described.

The existence of this book means that ideas long buried in fragmentary form in the manuals of the manufacturer, or in one or two conference papers, are now readily available in coherent form, to teachers and students. It is to be hoped that both groups take advantage of this new opportunity.

A. C. L. BARNARD
Dep. of Inform. Sci.
University of Alabama in Birmingham
Birmingham, Ala. 35294


Pattern Recognition Techniques does just what the doctor ordered; it serves as a broad informal introduction for engineers, computer scientists, and biologists to the field of pattern recognition. All necessary background is developed within the text, which is illustrated in terms of character recognition. It would be difficult to find a more extensive bibliography. This greatly contributes to the usefulness of this book.

CHAPTER I—MASK MATCHING

Chapter I begins with an introduction to character recognition via optical mask matching. Treatment is given to several machines which recognize characters printed in a single font. The concepts of best match, reject threshold, substitution error, and success rate are introduced. In the interest of speed, electronic mask matching techniques are presented using photocell masks and resistance measuring devices. Biarizing circuits are added to couple the photocell signal to the output. Both analog and digital devices are discussed. The concepts of maximization and minimization for selection are both introduced. Peephole masks and negative weights end the chapter. The author is very successful in this chapter in establishing a back-
CHAPTER II—PREPROCESSING FOR CHARACTER RECOGNITION

Chapter II is devoted to the techniques of conversion from visual to electrical patterns, binarization, alignment, smoothing, edge detection, and thinning. The conversion from visual to electrical patterns includes a discussion of the Nipkow disk, the Vidicon tube, photocell arrays, the flying spot scanner, and facsimile transmission. Although not a great deal of detail is given on any one of these systems, the author's liberal use of references would allow ample opportunity for anyone wishing to find out more. Binarization is discussed with emphasis given to the focus-defocus technique and measurement of limb width. There are several printing errors in the discussion of limb width which make comprehension somewhat difficult.

The discussion of alignment includes: (a) vertical alignment by measurement; and (b) vertical alignment by exhaustive trial. This technique requires the use of flip-flop shift registers and is ably demonstrated by example; (c) horizontal alignment and segmentation of text. Various techniques of horizontal alignment such as right justification and pitch are used; (d) size and prospective normalization. This includes the technique of normalization by perspective transformation described by Nagy and Tuong [1].

Various rules for smoothing, edge detection and thinning are given with examples. Some difficulty is encountered in the example used to describe Sherman’s [2] thinning technique because it is dependent on the starting point and the example does not illustrate this point.

CHAPTER III—LINEAR TECHNIQUES

This chapter provides an excellent introduction to the technique of linear discriminant functions. After an introduction to elementary probability, Bayes' Law is developed with a simple example. The next few sections closely follow the development given in Nilsson [3]. This includes the maximum likelihood approach to the classification of binary valued patterns followed by a discussion of the multivariate normal discriminant function with unequal, equal, and identity covariance matrices. The fixed increment adaptive procedure is presented in both the two case and the many case problems. Pattern error and its minimization by the method of steepest descent are briefly considered. Also included is a fine discussion of the advantages and disadvantages of one/many, many/many, and one/one dichotomization. The chapter closes with a brief discussion of the Karhunen-Loeve expansion technique, and two separate procedures for calculation of the scatter matrix are included. A brief discussion is also given regarding the problem of distinguishing variability between classes from variability within classes.

CHAPTER IV—PIECEWISE LINEAR TECHNIQUES, THE METHOD OF POTENTIALS, AND STOCHASTIC APPROXIMATION

Chapter IV, as the heading indicates, is a collection of various techniques. Beginning with a discussion of the need to develop other than purely linear discriminant functions, the author develops the idea of subclasses within a region. Linear discriminant functions are developed for each of the subclasses; such discriminant functions are called piecewise linear discriminant functions. As another example of a piecewise technique, the author develops the nearest neighbor concept as a classical example of a piecewise technique as well as an example of a nonparametric procedure. Finshein and Fischler's method of automatically splitting up each class into a set of sub-classes is included. The method of potentials as an attempt to approximate unknown probability descriptions is given considerable attention. The polynomial discriminant function developed by Specht [4] as an approximation to the potential function is presented adequately enough for the reader to make use of the procedure. Various error correction procedures applicable to these methods are presented. The chapter ends with a discussion of stochastic approximation in pattern recognition, although the author comments that this technique suffers from the serious drawback of being exceedingly slow to converge. The stochastic procedure is important because it necessarily converges to the minimum error weight vector even when the classes are not linearly separable. The author has done an excellent job of including enough essentials regarding the various procedures to permit their usage without further investigation and without being bogged down in proofs.

CHAPTER V—POLYNOMIAL DISCRIMINANTS AND N-TUPLE METHODS

Chapter V extends the idea of linear discriminant functions to polynomial discriminant functions using orthonormal expansion techniques. N-tuple methods using the principle of maximum likelihood and the principle of nearest neighbor are investigated examples being given with different training set sizes. An interesting aspect of this chapter is the discussion of Bidose and Browning's degenerate form of the n-tuple nearest neighbor method. Bidose and Browning's method is shown to peak in performance at higher n than the maximum likelihood n-tuple method. This at first seems odd since the degenerate nearest neighbor method takes no account of joint occurrences of n-tuple states in the same training set pattern. The discussion by the author of the reasons for the improved performance of the method is very clear and easy to comprehend.

Polynomial discriminants as a procedure to separate classes which are not linearly separable is enhanced by an excellent discussion of selection procedures for transforming n element patterns into higher order patterns with product terms, etc. The chapter ends with a discussion of shifted peephole mask systems as an illustration of a procedure to generate discriminating hyperplanes.

CHAPTER VI—BOOLEAN AND SEQUENTIAL DECISION MAKING

The author demonstrates that the implementation of arbitrary Boolean functions is a pattern recognition problem. Two methods are given: the classical implementation via standard logic elements and the threshold logic procedure. Various advantages and disadvantages of each are thoroughly discussed. The concept of decision trees as program dominated sequences is illustrated as a procedure to reduce the number of tests which need be examined to classify. To carry this idea further, the concept of variable numbers of tests using the sequential probability ratio tests and cascaded decision making techniques are presented.

CHAPTER VII—FEATURES

In chapter VII, Dr. Ullmann covers a vast array of techniques useful in determining features of hand-written characters. Among the various techniques considered are zoned features, graph representation techniques, successively detected features, and cross-counting techniques. The most interesting section of the chapter is in the discussion of isomorphism. It is interesting to see how concepts of modern algebra can be used in a real situation. Although a great number of methods are discussed, use would have to be made of the references given for successful employment of any of the techniques.

CHAPTER VIII—CONTEXTUAL, LINGUISTIC, AND ARRAY TECHNIQUES

The concept of maximum likelihood in the contextual sense introduces chapter eight. This is similar to the idea of Markov sources where the probability of a particular event is influenced by previous events. Scene analysis is briefly discussed. The discussion of grammars is very complete, and is illustrated with examples. The chapter is concluded with analysis by synthesis and iterative array techniques. Continual reference is made in this chapter to methods of previous chapters by a chapter number and section number. It would have been better if descriptive names had been given to these techniques.

CHAPTER IX—COEFFICIENT ANALYSIS

Chapter IX contains concepts somewhat familiar to most engineers and applies them to character recognition. Higher moments, alit
scan techniques, and Fourier transformation head up the list. The development of the use of the Fourier transformation is most interesting and is applied to pattern recognition by Fourier optics. Speech recognition is introduced, and the differences in techniques required are fully discussed. The author does an excellent job in this chapter of relating known concepts to pattern recognition.

CHAPTER X—LEARNING

In chapter X the author discusses unsupervised learning systems to develop the classification characteristics. The whole chapter is devoted to various techniques of teaching the pattern classifier how best to achieve classification.

This book never discusses what Fukunaga [5] considers to be the most important part of pattern recognition, error determination. This would make it more difficult to use this book as text in a higher level course. One possible other shortcoming is the total lack of problems in the book. All in all, for persons interested in achieving an introductory background in pattern recognition, this book is unsurpassed.

REFERENCES


ROBERT C. GILLESPIE
West Virginia Tech.
Montgomery, W. Va.


This book will become a lethal weapon against skepticism toward the possibility of presenting programming as a precise discipline based on clear and simple principles. It treats what many regard as advanced topics in programming in so straightforward a manner, that there is little doubt that the "advanced topics" are in fact elementary notions. The material is presented simultaneously at many levels: the college sophomore will enjoy a hearthy repast, solid conceptual and practical aspects of algorithm design, and the college professor or industrial professional will enjoy many morsels from the philosophy of programming to practical advice on the best uses of each language construction. The material is presented in a fast-moving, unadorned style, reminiscent in places of sketchy lecture notes; indeed, fully fourteen chapters appear in the first 124 pages. Every concept is backed by an example; I counted approximately 45 sample algorithms or programs in the text and about 25 more in the Exercises. The sample programs are masterpieces of clarity, exemplifying how invariant assertions can be embedded in comments to make understanding and verifying the programs as simple as possible. Wirth leaves little doubt as to the power of this technique, not merely by discussing it but by doing it.

There are fifteen chapters. Chapter 1, a one-page introduction, tides the reader over to Chapter 2: the author evidently does not like long introductions. Accordingly, in Chapter 2 he gets down immediately to business, introducing the basic concept of an algorithm as a sequence of statements defining a pattern of behavior on the values of variables. Chapter 3 describes the basic components of a computing machine. Chapter 4 outlines the basic components of a computer system. Chapter 5 simultaneously presents some simple programs and shows how to include in them invariant assertions from which the correctness of the program can be deduced. Chapter 6 shows a simple condition which guarantees the termination of all iterations, one so simple as to have escaped the attention of many a programmer. Chapter 7 introduces basic programming language concepts—expressions, compound statements, conditional statements, repetitive statements, and selective statements. It also restates the rules of inference about program verification used in the previous two chapters. Chapter 8 is a masterful exposition, axiomatically, of the four data types Boolean, integer, character, and real. The difficult aspect of the proposition of real arithmetic is concisely treated. Chapter 9 discusses some very simple programs based on recurrence relations, showing the power and utility of the while statement. Unfortunately, all the examples are drawn from number theory or numerical analysis and do not illustrate the power of the while statement for other applications. Chapter 10 is a discussion of the data structure array and its use. Chapter 11 is a discussion of the data structure array and its use. Chapter 12 tells about procedures and functions; in ten short pages the author covers the motivation for procedures, scope of names, parameter passing, and function-procedures. The author eliminates the possibility of a student's confusion over the hairline distinction between "begin" blocks and "procedure" blocks (as in Algol or PL/1) by eliminating "begin" blocks as a method of limiting the scope of names. He also eliminates possible confusion over function procedures' side effects by prohibiting assignments to nonlocal variables within such procedures. However, the instructor is going to have to point out to the student that he may be confused by such things when he writes programming languages other than the one used here (Pascal). Chapter 13 is about number representations and conversions between bases. (It seemed to interrupt the progression of ideas, but I suppose every book on programming contains such a chapter.) Chapter 14 presents some text-processing examples, accomplished via array and file manipulations. Chapter 15 gives four examples of stepwise (sometimes called top-down) program development. The examples show very nicely the ease with which well-structured, nontrivial programs can be presented. The examples are: solving linear equations by Gaussian elimination, finding two solutions for the dio- phantine equation $x = 2^a + 2^b$, tabulating the first n prime numbers, and generating character strings in which no two immediately adjacent substrings are equal.

I have two criticisms of this book, each concerning what was left out rather than what is included. First, the programs appearing in the text and Exercises are heavily biased toward numerical and number-theoretic problems. My crude count of the numbers of examples in various categories is shown in the table:

<table>
<thead>
<tr>
<th>Category</th>
<th>Text</th>
<th>Exercises</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numerical or number-theoretic</td>
<td>27</td>
<td>34</td>
</tr>
<tr>
<td>Number conversion</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>File or textstring manipulations</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>Searching and sorting</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>TOTAL</td>
<td>45</td>
<td>52</td>
</tr>
</tbody>
</table>

(There are also 19 other Exercises which are straight exercises, proofs of simple propositions, or thought problems.) The heavy emphasis on numerical and number-theoretic programs will reduce the appeal of the book to engineering and business-oriented communities. And, except for Gaussian elimination, it is difficult to envision applications even in the scientific community of the examples developed stepwise in Chapter 15; to stress the utility of stepwise program development in practice, it seems, one should use examples of practical utility. However, the author's choice of examples is entirely consistent with his viewpoint; he has stated in the Preface, "exercises and examples have been selected as demonstrations of generally valid problems and methods of solution." His emphasis on numerical and number problems is also consistent with his intention that "this text is tailored for people who view a course in systematic construction of algorithms as part of their basic mathematical training."

My only criticism is that the book presents only a subset of Pascal, a language of which the author should be proud.1 In fact, the author seems to have taken great pains not to mention the name Pascal: there is no mention of it in the Preface, and the only textual reference I could find was in a footnote. I can surmise only that he has deemphasized Pascal because he wants the reader to regard a specific language merely as a tool for implementing the concepts of systematic programming—he specifically wants to