BULLETIN OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 79, Number 3, May 1973

BOOK REVIEWS

The Art of Computer Programming, Volume 1. *Fundamental Algorithms* and *Volume* 2. *Seminumerical Algorithms* by Donald E. Knuth. Addison-Wesley, Reading, Mass., 1969 (second printing) and 1969.

1. These books are the first two of eight volumes on the construction and analysis of computer algorithms. The entire set, consisting of twelve chapters and a compendium volume covering the first ten chapters, will provide comprehensive coverage of "nonnumerical programming", computer programming which relies but very little on classical numerical analysis. This area of computing is surprisingly broad and accommodates not only the fundamentals but also many of the active research problems of modern computing. The first seven volumes have the following outline:

Volume 1. *Fundamental Algorithms*
  Chapter 1. Basic Concepts
  Chapter 2. Information Structures
Volume 2. *Seminumerical Algorithms*
  Chapter 3. Random Numbers
  Chapter 4. Arithmetic
Volume 3. *Sorting and Searching*
  Chapter 5. Sorting Techniques
  Chapter 6. Searching Techniques
Volume 4. *Combinatorial Algorithms*
  Chapter 7. Combinatorial Searching
  Chapter 8. Recursion
Volume 5. *Syntactical Algorithms*
  Chapter 9. Lexical Scanning
  Chapter 10. Parsing Techniques
Volume 6. *Theory of Languages*
  Chapter 11. Mathematical Linguistics
Volume 7. *Compilers*
  Chapter 12. Programming Language Translation

These are important books. They are written by a computer scientist *and* mathematician who has performed high-level research in both fields (e.g. [**1**] and [**2**]). As he says in the preface to Volume 1, "... parts of these books may be thought of as 'a pure mathematician's view of computers'." He is concerned with the application of mathematical tools to programming and with the use of computers in exploring mathematical conjectures. "I wish to show that the connection between computers and mathematics is far deeper than ... traditional relationships would imply."

The author is also an excellent teacher. He has much to say and says it with a fine blend of humor, finesse, and scholarship.

The books can be studied profitably at each of several levels. First, the flowing textual style of the author permits the books to be read and enjoyed in an easy chair atmosphere. Of particular merit in this respect are the many historical notes and digressions. For instance, it is interesting to know that the factorial notation $n!$ in common use today was introduced by a relatively unknown mathematician Christian Kramp in an 1808 algebra text; Euler used $[n]$ while Gauss wrote $\pi(n)$. Secondly, the books contain rigorous, detailed, and chiefly constructive proofs of the important theorems. Thus, they are entirely suitable for serious study at one's desk. Finally, the prodigious problem and answer sections and the extensive bibliographical references contained in the text provide an excellent starting point for serious research. All exercises are given a difficulty rating by the author. Solutions, or suggestions leading to a solution, for most exercises with a rating of 40 (a "term project") or below are given in the text. Many interesting "research problems" (50 rating) are included. For instance, "Is there any way to evaluate a permanent of a general $n \times n$ matrix using a number of operations which does not grow exponentially with n?" is a research problem from §4.6.4 on Evaluation of Polynomials.

Both volumes are organized into chapters, sections, subsections, etc. Exercises are given at the end of each numbered section. About 20% of each volume is devoted to answers to the exercises. Both volumes have an index to notations, an appendix of tables, and a superb general index and glossary. Volume 2 rather extravagantly has as an appendix a copy of most of §1.3 on the MIX computer and MIXAL assembly language. The indices to notations include both mathematical and computer-oriented symbolism. They are quite complete although the symbol "$\approx$", which is used informally throughout both texts as "approximately equal to", does not appear. The tables, including for instance a forty digit representation of important numerical constants, are useful. The reviewer would like to have seen some of the excellent tables (e.g. the table of the ten largest prime numbers less than 2^n, $15 \leqq n \leqq 64$, and 10^m, $6 \leqq m \leqq 16$, which appears in §4.5.4 on Factoring into Primes) accessible in the appendix also. However, the strength of such criticism is certainly diminished by the all-inclusiveness of the indices. This applies as well to the lack of a bibliography. By using the comprehensive indices it is quite easy to track down any reference or fact which the author has discussed.

2. There are essentially three levels of language used in these books for the description and analysis of algorithms. At the lowest level, the author has introduced an "assembly programming language", MIXAL,

which is associated with his hypothetical machine MIX. This computer language is used for a detailed, precise presentation of some of the basic algorithms. The primary algorithmic language, however, is an informally defined, step-by-step language which blends certain common programming language features such as assignment statements and functional notation with English commentary. Such language is also used for remarks attached to the MIXAL programs. At a still higher level, the author has used standard mathematical symbolism—scripting, summation notation, display, etc.—for the general textual algorithmic discussion and analysis.

In the reviewer's opinion, most readers will not read programs written in MIXAL; they will perhaps only skim their remarks. For many, the slight gain in preciseness and analyzability is not worth the difficulty in reading such detailed language. On the other hand, the use of MIXAL is certainly of interest to students of computer science. Writing a MIX simulator and MIXAL assembler is an excellent term project which has indeed been tackled by individuals and classes at several universities. The author's idea of defining a model computer and language as a basis for the description of algorithms is good, but it must not be carried too far. As one would expect there are many fewer MIXAL programs in Volume 2 than in Volume 1—the space allocated in Volume 2 for the reprint of the MIX and MIXAL description could have been put to better use.

The less formal, principal presentation of algorithms is excellent. Each step is labeled and is clearly explained using simple symbolism and English commentary. Each algorithm is preceded by a concise yet intelligible description of its notation and basic idea. Furthermore, the text itself provides additional clarification and, for some, a detailed analysis. Thus, understanding the algorithms is an easy matter.

The important and difficult question of notation becomes particularly sticky in a work such as this which involves mathematics *and* computer programming. Knuth has well met the challenge by using and naturally blending notational conventions from both fields. Actually, it is the reviewer's contention that conventional mathematical symbolism need be extended only very little to incorporate effectively the new concepts required to express computer algorithms. It is just a matter of time before notations such as von Neumann's assignment operator, $\rightarrow$, (or the more fashionable left arrow, $\leftarrow$), and Iverson's floor function notation, $\lfloor x \rfloor$, and ceiling function notation, $\lceil x \rceil$, are accepted and used by mathematicians and computer scientists alike. Similarly, it is just a matter of time before computers will be taught the two-dimensional symbolism of mathematics. Thus, computer algorithmic language will merely be absorbed into mathematics.

There are a few slightly confusing notations in Knuth's algorithmic

language. The right double arrow, $\Rightarrow$, is used for logical implication whereas the left double arrow, $\Leftarrow$, is used for a special "stack" storing operator. A form such as BRANCH(P) is used both for functional evaluation and "field referencing". There is further confusion in Volume 1 where a similar form using square brackets, e.g. TAG[I], denotes the Ith element of the sequential stored list TAG. Such notation is of course a carry-over from the one-dimensional programming languages such as Fortran and Algol. Fortunately, Volume 2, which is less concerned with implementation matters, incorporates the more mathematically natural subscript form for sequence component referencing.

With regard to mathematical symbolism itself, Knuth has introduced some interesting notations. He uses $\left[{n \atop m}\right]$ for the positive Stirling numbers of the first kind and $\left\{{n \atop m}\right\}$ for Stirling numbers of the second kind. These are suggestive and readable notations and blend well with the common diplayed binomial coefficient notation $\binom{n}{m}$. They could reasonably be recommended for general acceptance (although the square bracket form *is* often used for the *Gaussian coefficients*—the number of m-dimensional subspaces of a vector space of dimension n over a finite field). Knuth uses $x^{\bar{n}}$ for the rising factorial $x \cdot (x + 1) \cdot \cdots \cdot (x + n - 1)$ and $x^{\underline{n}}$ for the falling factorial $x \cdot (x - 1) \cdot \cdots \cdot (x - n + 1)$. These are certainly more suggestive than constructions commonly in use, e.g. $(x)^n$, $x^{(n)}$, etc., but still can be confused with other forms, e.g. x raised to the power of n-bar. Perhaps some entirely new but unambiguous notation should be invented, à la Kramp and Iverson, for these important functions. It seems likely that the need for precision in computer language will influence mathematical language to some extent in the coming years.

Knuth also has radical tendencies in certain terminological areas. Most notable in this respect is his use of the unqualified term "tree" to imply both "rooted" and "ordered". A linear graph without cycles is then a "free tree". Such terminology stems, of course, from the way in which trees are customarily represented within computers. (The organization of modern computer memories is such that all items are implicitly ordered by the addressing scheme.) The use of this computer-oriented terminology is practical today and is to be welcomed by computer scientists. However, due to the importance of inherently unordered structures (sets) in mathematics, it is unlikely that such change of emphasis will make significant inroads outside of the computer field. A term used by Knuth which could be profitably adopted by both mathematicians and computer scientists is de Bruijn's "multiset". A *multiset* is an (unordered) set of objects in which each object has an associated multiplicity.

3. Volume 1 divides, roughly by chapter, into two more or less independent courses of study. Chapter 1, plus §2.3.4 on Basic Mathematical

Properties of Trees but perhaps minus §1.3 on MIX and §1.4 on Some Fundamental Programming Techniques, provides a useful introduction to discrete (combinatorial) mathematics and computing. The remainder discusses the rudiments of programming and data structures for non-numerical computing. Within a university curriculum these areas can be treated as independent semester courses or blended into a full year's work. The range of levels for courses that could be designed from this text is quite broad, covering most of graduate and undergraduate school.

The mathematical sections discuss induction; basic operations and functions such as summation, exponentiation, and logarithms; elementary number theory; fundamental combinatorial numbers and tools such as binomial coefficients, Fibonacci numbers, generating functions, asymptotic representations; and certain properties of (free) trees and their enumeration. Of particular merit is the section on the handling of finite sums. Most mathematics students learn these manipulations in a rather haphazard manner on their own. Knuth's systematic presentation is exceptionally complete and furnishes the reader with useful constructive tools of finite mathematics. The sections on combinatorial numbers contain derivations of many important combinatorial identities useful in algorithmic analysis. (It is rather strange that Knuth, who has generally taken great pains to acknowledge contributions, does not refer to the identity

$$\sum_k \binom{r}{k}\binom{s}{n-k} = \binom{r+s}{n}$$

by its common name—the Vandermonde convolution.)

One aspect of Knuth's discussion of generating functions is misleading. This is the implication that one must generally find a "proof" independent of a derivation using generation functions unless one has concerned himself with convergence questions. It would be preferable to point out that results obtained by equating coefficients after formal algebraic manipulations of generating functions are valid independent of convergence [3]. Knuth does discuss the formal manipulation of power series as the last section of Volume 2.

A course on programming designed from Volume 1 would, after an introduction to computing via a study of MIX and MIXAL and to the important concept of subroutine, focus on the way in which information is structured for efficient computational processing. To Knuth this means a data organization based on the concept of a linked "List", essentially a rooted directed graph with atoms of information and pointers (links to successor nodes) stored sequentially at each node. More precisely, he defines a *List* recursively as a *finite sequence of zero or more atoms or Lists*. The items of a List (sometimes called "fields") are often given names

to simplify their reference. These structures *are* indeed of fundamental importance in the construction of computer algorithms, and while Knuth was certainly not the first to recognize their importance, he was, via this volume, the first to give a systematic and detailed treatment of their properties and use. The important topics of this programming course are the basic terminology associated with linear and linked lists, representation of trees within the computer, traversing of trees and Lists, Lists and garbage collection, and dynamic storage allocation.

A criticism regarding Knuth's handling of information structures that is occasionally heard from computer scientists is that his examples are too simple and do not well enough prepare the student for the more complex problems of the real world of computing. This criticism is at least partially valid as the section on multilinked structures is quite short in comparison, say, to the sections on binary trees. However, we must remember that Knuth is attempting to lay a strong foundation in the general area of information structures, to relate them to familiar mathematical structures, and to introduce us to the mathematical analysis that can be applied to computer algorithms. In these aims he has succeeded commendably.

In Volume 2 we see a pronounced change of emphasis which is likely to prevail through Volume 7. Whereas the material of Volume 1 is quite generally applicable to the construction and analysis of algorithms, the topics of Volume 2 are more specialized. Also, those topics are covered in great detail and to a surprising depth. For instance, thirty-two pages are devoted to the analysis of Euclid's algorithm and twenty-three pages to a discussion of problems related to the efficient computation of x^n, n a positive integer. Much of this work is original to Knuth and in some ways Volume 2 appears to be a collection of research papers blended into a single volume. Indeed there is much here that should and will be ignored by beginning students or during early readings by professionals. (The word "irrelevant" has been used in relation to much of the material of Volume 2 [**4**].)

While Volume 2 is certainly less applicable as a textbook than Volume 1, or at least fits into a standard curriculum less smoothly, there is a certain attraction to its use in, say, an elementary number theory course. This is so because the number theoretic results and methods that arise in several contexts do so naturally in connection with actual computer applications. Two simple examples are (1) the primitive root modulo p^e criteria, which are important in determining multipliers for linear congruential pseudo-random numbers generating schemes, and (2) linear Diophantine analysis, which arises in the detailed discussion of "greatest common divisor" algorithms. An analogous claim relative to a course in elementary probability and statistics has some credence due to their application to the

study of random numbers in Chapter 3. However, there is considerable background material not covered in this single volume (much of it *is* in Chapter 1 though).

The material in Chapter 3 on Random Numbers is a detailed and reasonably complete treatise on the generation, statistical testing, and use of (pseudo) random numbers. The section on generation concentrates on linear congruential schemes, i.e.

$$X_{i+1} \leftarrow (aX_i + c) \mod m$$

for suitably chosen X_0, a, c, and m, since these are the ones that have been in most common use. The section on statistical testing discusses general tests such as the chi square test and the Kolmogorov-Smirov test, empirical tests such as the run test and poker test, and an important theoretical and empirical test known as the spectral test.

Actually the theory behind this last test has been clarified since this volume was written. The spectral test determines certain geometric properties of the lattice in n-space formed by the "random vectors"

$$(X_k, X_{k+1}, \ldots, X_{k+n-1}), \qquad k = 0, 1, \ldots .$$

The discussion of Knuth is based on early work by R. R. Coveyou and R. D. MacPherson and investigates the longest side of the "reduced cell" associated with this lattice. Recent investigations, Coveyou [**5**] and Beyer, Roof, and Williamson [**6**], have clarified the theory and have obtained the complete reduced cell. This is one of the few cases where material in these books is already obsolete.

Uniformly distributed random numbers—the sequence of X's of the linear congruential generator scheme—are basic data in "Monte Carlo" calculations, calculations which estimate or generate (i.e. select objects from a universe that follows) various involved statistical distributions. Pioneering study and application of Monte Carlo methods was made by the mathematicians John von Neumann and Stanislaw Ulam and the physicist Enrico Fermi in the early 1950's. Since then many people, e.g. Edmund Cashwell and Cornelius Everett [**7**] (a reference omitted by Knuth) and George Marsaglia, have contributed to the development of this important computational technique, although there exists even today a shortage of good books on the subject. (See Halton [**8**] for a review of the Monte Carlo method.) Knuth considers the generation of fundamental numerical distributions such as the normal, exponential, chi-square, and Poisson distributions; much of his discussion is based on the work of Marsaglia.

The final (starred) section of Chapter 3 is a research paper entitled "What is a random sequence?". This is an excursion into "higher mathe-

matics" wherein the author provides additional conditions that "completely equidistributed sequences" must satisfy in order that they become sufficiently haphazard to be called "random".

Chapter 4 contains a world of information, some useful, some less so. Of particular interest and use to computer scientists are the sections on positional number systems, multiple-precision arithmetic ("multi-word" or "extended-precision" would be better adjectives), and radix conversion, and the introductory parts of the sections on rational arithmetic and polynomial arithmetic. Of more mathematical interest are the sections on the analysis of Euclid's algorithm, factoring into primes, factorization of polynomials, and evaluation of powers and polynomials. Following are a few "random" thoughts that occurred to the reviewer while reading and discussing with colleagues this comprehensive chapter on arithmetic.

The section on floating point arithmetic contains a rather detailed analysis of the accuracy of "normalized" arithmetic, arithmetic in which the "fraction" f (would "coefficient" be a better word?) in the representation

$$x = f \cdot b^e, \qquad x \neq 0,$$

is maintained so that $1/b \leqq |f| < 1$. Knuth's goal is "to discover how to perform floating point arithmetic in such a way that reasonable analyses of error propagation are facilitated as much as possible." However, it is quite possible (even probable) that this goal will be more readily achieved by concentrating on "unnormalized" or "interval arithmetic"; these topics are only cursorily discussed by Knuth.

From the point of view of algorithm construction, the sections on factoring are particularly interesting. Knuth shows that a reasonably good algorithm for factoring very large integers into primes may be constructed by cleverly consolidating several special methods for factor searching and prime verification. All of the requisite number theoretical background material is discussed. The complete algorithm, which is based on work by John Brillhart and J. L. Selfridge, is illustrated using the number $2^{2^{14}} + 1$. A reasonable approach to polynomial factorization over the integers is based on work of E. R. Berlekamp, and others. Berlekamp has provided effective means of accomplishing the factorization modulo p. By using such factorizations for several primes and the Chinese remainder theorem one can find compatible factorizations modulo m, the product of the primes. Since modulo p factorizations *include* general factorizations, a compatible factorization is a general factorization for sufficiently large m. By finding upper bounds on the sizes of the coefficients of a general factorization one can then determine how large m must be for this approach to work.

The section on Evaluation of Powers is a research paper on addition

chains. An *addition chain for n* is a sequence

$$1 = a_0, a_1, \ldots, a_r = n$$

such that

$$a_i = a_j + a_k \quad \text{for some } k \leqq j < i.$$

Most work on addition chains is concerned with the calculation of $l(n)$, the length of the shortest addition chain for n. The relationship of these chains to computer algorithms is that an $l(n)$-chain gives a way to form x^n, arbitrary x and integral n, with fewest multiplications. Of special interest to mathematicians is the still unanswered "Scholz-Brauer conjecture"

$$l(2^n - 1) \leqq n - 1 + l(n).$$

Knuth's paper consolidates relevant mathematical and empirical work on addition chains; it is certainly of more interest to the mathematician than to the computer scientist. Recent computer work by Knuth (appearing in the second printing) has turned up the remarkable fact that $l(12509) < l^*(12509)$—an l^*-chain is one in which $j = i - 1$.

4. For various reasons, many of them nontechnical and extraneous, the interaction between mathematics and computer science over the past two decades has been less than one might expect. This has been and is unfortunate as each field can significantly benefit from and contribute to the other. The real importance of Knuth's work is that it represents a truly positive step towards eliminating the existing breach between mathematicians and computer scientists.

MARK B. WELLS

REFERENCES

1. D. E. Knuth, *A class of projective planes*, Trans. Amer. Math. Soc. **115** (1965), 541–549. MR **34** #1916.
2. D. E. Knuth and R. H. Bigelow, *Programming languages for automata*, J. Assoc. Comput. Mach. **14** (1967), 615–635.
3. John Riordan, "Generating functions," Chapter 3 in *Applied combinatorial mathematics*, edited by E. F. Beckenbach, Wiley, New York, 1964.
4. R. W. Hamming, Review (18,478) of *Seminumerical algorithms*. Vol 2, by D. E. Knuth, Comput. Rev. **11** (1970), 99.
5. R. R. Coveyou, "Random number generation is too important to be left to chance," in *Studies in applied mathematics*, SIAM, Philadelphia, Pa., 1969, pp. 70–111.
6. W. A. Beyer, R. B. Roof and D. Williamson, *The lattice structure of multiplicative congruential pseudo-random vectors*, Math. Comp. **25** (1971), 345–363.
7. E. D. Cashwell and C. J. Everett, *A practical manual on the Monte Carlo method for random walk problems*, Pergamon, New York, 1959. MR **21** #5269.
8. J. H. Halton, *A retrospective and prospective survey of the Monte Carlo methods*, SIAM Rev. **12** (1970), 1–63. MR **41** #2878.

Structure and Representations of Jordan Algebras, by Nathan Jacobson. American Mathematical Society Colloquium Publications, vol. 39. American Mathematical Society, Providence, R. I., 1968. x + 453 pp. $11.30 (Member Price $8.48).