DOCUMENT RESUME

ED 107 241 IR 001 981

AUTHOR	Brecke, Fritz H.; And Others
TITLE	Algorithms: A New Tool for Educational Technology.
INSTITUTION	Arizona State Univ., Tempe.
SPONS AGENCY	Air Force Office of Scientific Research, Arlington, Va.
PUB DATE	Apr 75
NOTE	12p.; Paper presented at the Annual Meeting of the American Educational Research Association (Washington, D.C., March 30 through April 3, 1975)
EDRS PRICE	MF-$0.76 HC-$1.58 PLUS POSTAGE
DESCRIPTORS	*Algorithms; Learning Processes; *Learning Theories; *Logical Thinking; Models; Problems; *Problem Solving; Task Analysis; Teaching Techniques; Thought Processes

ABSTRACT

The concept of an algorithm derives from the physical sciences, but it has often been misunderstood and misapplied in the social sciences and in education. The theoretical and practical significance of algorithms stems from their applicability to problems of learning, instruction, and instructional design, and they may potentially provide the basis for the development of a paradigm or model of instruction. For the purposes of this paper, an algorithm may be defined as a strictly replicable procedure which always produces the correct result when applied by a user to a problem or class of problems. Examples of the specification and use of an algorithm are provided together with a discussion of the properties and characteristics of algorithms. Some areas for further investigation and clarification are also suggested. (DGC)

Handout to accompany

ED107241

Algorithms: A New Tool for Educational Technology*

Fritz H. Brecke
Southern Illinois University, Edwardsville
Vernon S. Gerlach
Brian D. Shipley
Arizona State University

Introduction

The behavioral sciences frequently adopt terms, concepts, even theories, of the older and more formalized "hard" sciences. Unfortunately, the early stages of the adoption process are often characterized by vague, imprecise, or even inappropriate applications to the newer discipline. The use of the concept algorithm by educators and psychologists provides a recent example of this phenomenon. An algorithm is a strictly replicable procedure. (The term is defined in the paper much more precisely.) The theoretical and practical significance of algorithms stems from their applicability to problems of learning, instruction, and instructional design (including goal specification and evaluation). Algorithms provide a highly effective tool for task analysis. They offer many advantages for educators concerned with the development of performance, planning, or communication aids. Theoretically, they may even provide the basis for the development of a paradigm or model of instruction.

Important as these contributions may be, they only hint at the enormous potential of algorithms. Unfortunately, two problems confront the researcher or the developer who is interested in the area: (1) Much of the literature is of European origin and has not yet been translated into English. (2) The literature on algorithms applied to our discipline is rapidly increasing. Even though it seems quite clear that the concept may be a useful (even powerful!) one for instructional designers, the literature reveals that use of the term ranges from the enthusiastic vagueness of a fadist to the highly limited precision of a scientist.

*This paper was presented to the Convention of the American Educational Research Association, Washington, D.C., March 31, 1975. The paper is abstracted from a technical report written for the US Air Force Office of Scientific Research under Grant # OSR71-2128, awarded to Arizona State University.

Consequently, there is a need for a comprehensive analysis of the literature, as well as for a taxonomic cleansing and sharpening of definitions.

In the context of instruction, algorithms may be used to represent subject matter or they may be subject matter themselves. Learners may use an algorithm as a learning aid to acquire a certain skill or as a generalized study strategy to acquire a whole range of skills. Algorithms may also be used to represent, develop and describe teaching strategies; they may serve as a basis for curriculum planning and they may be used as a basis for the design of instructional materials.

Definition and Example

At the present time, the literature contains an abundance of statements regarding algorithms which might be considered definitions and/or lists of properties. Unfortunately, none has been discovered which can be considered complete and precise. Consequently it is necessary to look at a number of definitions and statements in order to arrive at a satisfactory resolution of this problem. Even then, the definition is not one which is mathematically rigorous.

This being true, let us for the purpose of this afternoon's presentation begin with a very simple, easy to understand statement. This statement is neither complete nor precise. However, as we examine its faults and shortcomings, we will be building a working vocabulary which should enable us to handle the concept with ever-increasing sophistication and discipline.

To begin with, then, an algorithm is a strictly replicable procedure; it is a procedure which always produces the correct result when applied by a user of a defined class of users to any problem of a defined class of problems.

To illustrate: one algorithm for the addition of two common fractions having natural number denominators can be expressed in this way.

a: Are the denominators identical?
- If yes, add the numerators, write the sum over the denominator, reduce to lowest terms if necessary.
- If no, determine whether one denominator is a multiple of the the other.
  - If yes, factor the larger denominator into two factors with the smaller denominator as one factor; multiply the other numerator by the second factor; add numerators, sum over denominators, reduce.

If no, determine whether the denominators are multiples of a common factor.
If yes, form a common denominator by multiplying the common factor with each of the unique factors; multiply each numerator with the unique factor of the denominator of the other fraction; add the numerators; sum over denominator; reduce.
If no, multiply each numerator by the denominator of the other fraction; add the numerators; sum over denominator; reduce if necessary.

This replicable procedure can be presented in the form of a list structure, or a decision logic table, or a logic diagram, or a flowchart, or any of an unknown number of presentation modes. The flowchart is well-known because of its use in a large number of disciplines; furthermore, one rarely if ever encounters an algorithm which cannot be represented by a flowchart. Consequently, this afternoon we shall present our algorithms in flowchart form. The algorithm for adding fractions is shown in flowchart form on the next page of your handout.

Characteristics of Algorithms

Deterministic. An algorithm must result in a predictable outcome. The algorithm for adding fractions must produce the correct sum every time it is applied to a given pair of addends.

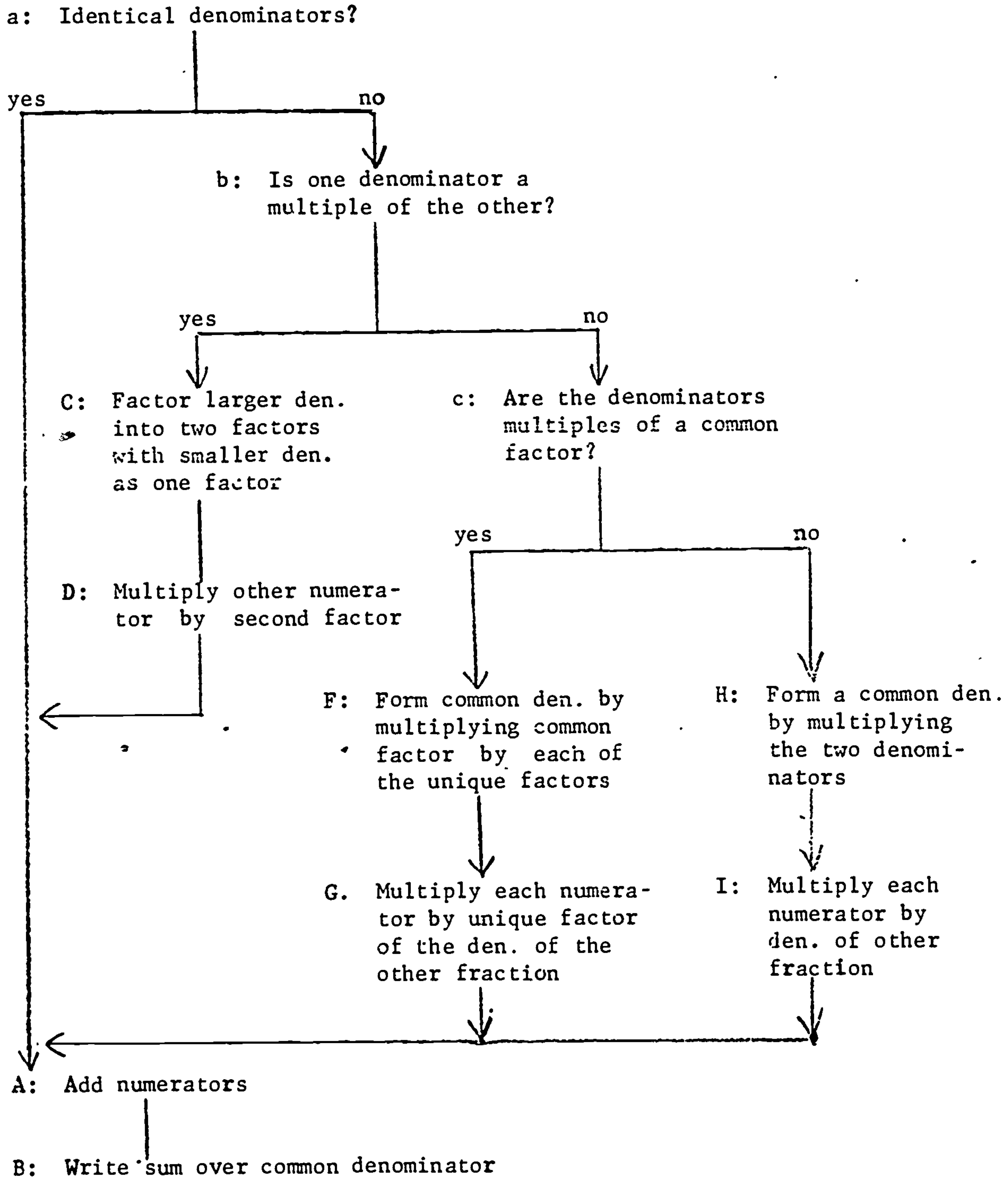
Generalizable. An algorithm must provide a procedure sufficiently general so that the solution to any problem of a class of problems can be obtained.

Resultivity. This term comes from Landa, a distinguished Russian psychologist and educator. This property is reflected in the fact that an algorithm always converges on a specific sought-for result, which is always obtained in the presence of the appropriate data set. This property of an algorithm, however, does not assume that algorithms result in the obtaining of the derived result with all data sets belonging to the defined class. It is possible that the algorithm will be applicable to certain sets of data; and, in that case, the process of carrying out the algorithm will either halt suddenly, or it will never end. For example, the algorithm for adding fractions could "break down" if some fractions consisted of literal numbers.

Automata. While this term is not a characteristic in the ordinary sense, it does help us understand by contrast. When we say "automaton," we generally think of some real, tangible manifestation of the execution of an algorithm. When we talk of an algorithm, per se, we are generally thinking of the process. A computer solving addition of common fraction problems might be an automaton; the program which controls the computer is an algorithm.

Al: Algorithm for the addition of two fractions

Domain: Any set of two fractions with natural number denominators
Range: Sum of any set of the domain
Entry skills: Can factor natural numbers



Procedure. All algorithms are a subset of the set "procedure." Whenever a procedure satisfies the three criteria of generalizability, replicability, and resultivity (Landa), it is an algorithm. This definition, however, does little to help us unless these three terms are operationally defined. The authors have adapted the concepts "domain" and "range" in an effort to operationalize the terms mentioned.

Domain. The domain of an algorithm is the precise specification of the area or limits of its applicability. (It is assumed that the class always has at least one member; that is, domain is never an empty set.) Again, the algorithm for addition of fractions provides an illustration: the domain is any set of two fractions with natural number denominators.

Range. The application of an algorithm always leads to some uniquely determined result which is a member of a set of possible results or outputs or output words. Range refers to the set of possible results. In the fraction addition example, the range is the sum of any set of the domain.

User. The intended users of an algorithm must be described in terms of minimum skill or knowledge required for correct execution of the procedure. In the case of our example, this is labeled "Entry skills: can factor natural numbers." As long as one is dealing with a machine, this specification is a relatively simple matter. As soon as we are involved with human users, this criterion becomes extremely difficult to specify on an a priori basis. This difficulty leads us to the consideration of a number of additional terms.

Quasi-algorithm. This is a term introduced by Bung. Quasi-algorithms are procedures which are explicit for, and can be carried out by, a specified set of human beings; algorithms are procedures which are explicit for, and can be carried out by automata. Since all procedures which can be carried out by automata can also be carried out by human beings but not vice versa, it follows that all algorithms are quasi-algorithms but not all quasi-algorithms are algorithms. The set of all algorithms is therefore a subset of the set of all quasi-algorithms.

Syntactic aspects. The macrostructure of an algorithm represents its syntax. This is illustrated by means of the familiar Euclidean algorithm on the next page. The syntactic structure of this algorithm is shown on the next succeeding page.

Semantic aspects. When we speak of the semantic aspects, we refer to the meaning of the verbal elements associated with the symbols of the syntactic skeleton. Essentially, we are concerned with the ancient question which provides the raison d'etre for information theory: How precisely do the . . . symbols convey the desired meaning?

Pragmatic aspects. The pragmatic aspect is that characteristic which requires that an algorithm be operationally definable. One method of meeting this requirement is to demonstrate that a specified class of users can execute the algorithm in such a manner that an acceptable outcome results. It may be necessary to define the pragmatic aspect of an algorithm in terms of probability.

Domain:	Any set of two whole numbers
Range:	The greatest common factor of any set
Entry skills:	Can divide and multiply whole numbers

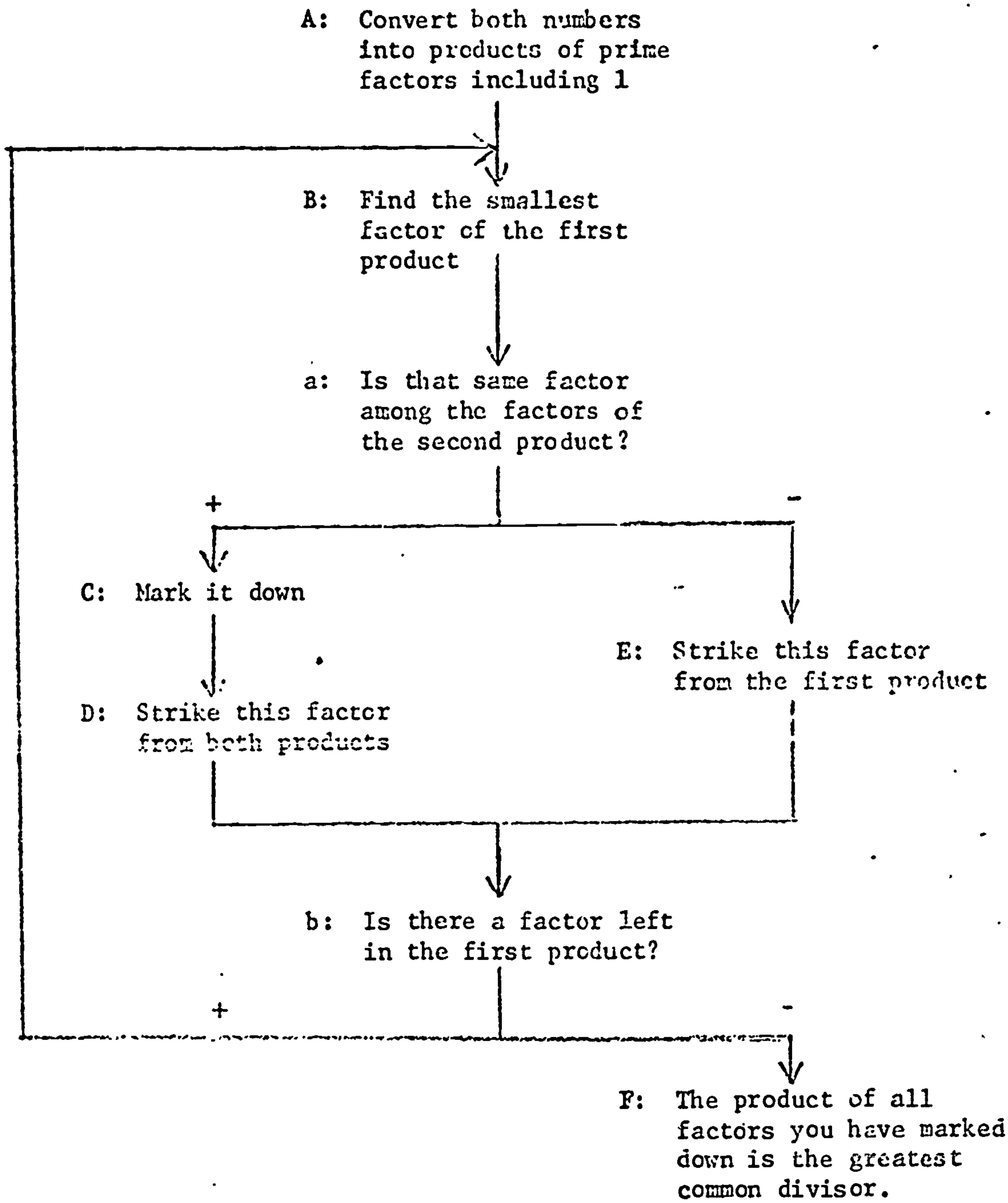


Figure 1: The Euclidean Algorithm

(Version 1)

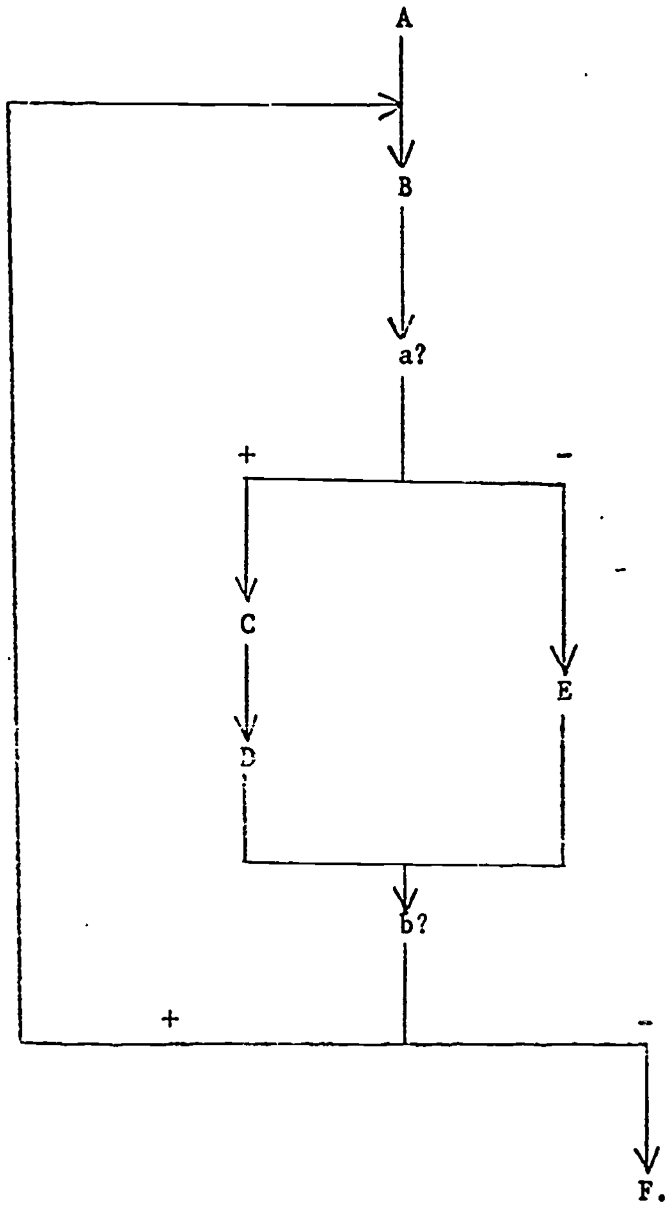


Figure 3 The syntactical structure

of the Euclidean Algorithm

(Version 1)

In summary, the authors have arrived at the following set of working definitions:

--Procedures in general are either explicit or non-explicit.

--A procedure is explicit if it contains a description of domain, range, and user; if it is formulated so that its elements (operators and discriminators) can be identified by the user.

--If a procedure is explicit, it is either a true algorithm or a quasi-algorithm.

--An explicit procedure is a true algorithm if it is *a priori* replicable.

--An explicit procedure is a quasi-algorithm if it is not *a priori* replicable, i.e., if predictability must be determined by empirical means.

--Quasi-algorithms vary in the degree to which the results of their application are predictable. Quasi-algorithms which are predictable with $p < .01$ are called quasi-algorithms of the first order, all others ($p > .01$) are called quasi-algorithms of the second order.

Algorithmic process, *algorithmic prescription*, and *algorithmic description*. These are terms used by Landa. A computer, for example, engages in an algorithmic process when it executes a program. If this program is in a form that the computer can read (such as punched cards, for example), then the program controls the process and it is an algorithmic prescription. If the program does control the process and if it can be used for communication only, it is an algorithmic description. A human brain may or may not function as deterministically as a computer, but humans are known to function at least occasionally in a lawful and predictable manner; when one does, he is engaged in algorithmic processes, or at least in quasi-algorithmic processes. If one does so intentionally and consciously by following an explicit procedure, he is following an algorithmic prescription. If he does so without conscious intent and awareness, his activity may be amenable to algorithmic description but it does not necessarily follow an algorithmic prescription. The rules of grammar, for example, are followed correctly both by people who know them and can state them and by people who cannot do so. Both kinds of people engage in algorithmic processes, but the rules of grammar are algorithmic prescriptions for the former only, even though they are algorithmic descriptions for both.

Application

Algorithms have both practical and theoretical significance for any institution which uses complex equipment or sophisticated procedures or which trains or employs human beings. Algorithms can be used to facilitate performance, planning, and/or communication. They offer the following advantages:

(1) The error potential attributable to misrepresentation is minimized, since the representational system used in an effective algorithm is always simple and unambiguous.

(2) The consumer of an algorithm needs only to process that information which is directly relevant to a given problem; he need not understand the total complex of rules which an algorithm represents.

(3) Algorithms are very helpful in performing a task analysis, since they "compel" the designer to communicate clearly and they tend to enable one to detect errors easily.

(4) Learners may use algorithms as an aid in the acquisition of a specific skill or as a generalized strategy for the acquisition of an entire range of skills.

(5) Algorithms may be used to represent, develop and design instruction (including curriculum planning, materials development, and evaluation).

Research

The concept "algorithm" as well as the theory of algorithms provides the interested researcher with a broad class of variables from which relevant and significant independent variables may be profitably selected for experimental study. Of even greater significance is the research promise which the subject holds for educators interested in general learning strategies, including transfer and generalization tasks.

Selected Bibliography to accompany

Algorithms: A New Tool for Educational Technology*

Fritz H. Brecke
Southern Illinois University, Edwardsville
Vernon S. Gerlach
Brian D. Shipley
Arizona State University

Anschuetz, H. Kybernetik. Wuerzburg: Vogel Verlag, 1970.

Bellman, R., Cooke, K. L., Lockett, J. A. Algorithms, graphs and computers. New York: Academic Press, 1970.

Bung, K. A model for the construction and use of adaptive algorithmic language programmes. In K. Bung (Ed.), Programmed learning and the language laboratory I. London: Longmac, 1967.

Frank, H. Kybernetische Grundlagen der Padagogik. Baden-Baden: Agis Verlag, 1969.

Glushkov, V. M. Theorie der abstrakten Automaten (G. Asser, Ed. and trans., K. Strahmel, trans.). Berlin: Deutscher Verlag der Wissenschaften, 1963.

Glushkov, V. M. Introduction to cybernetics (G. M. Kranc, Ed. and trans.). New York: Academic Press, 1966.

Horabin, I. S. Practical applications in education and training. In F. J. Frederick (Chm.) Algorithmic organization in teaching and learning. Symposium presented at the American Educational Research Association, Chicago, April, 1974.

Knuth, D. E. The art of computer programming, Vol. 1, fundamental algorithms. Reading, Massachusetts: Addison-Wesley, 1968.

Landa, L. N. Algorithmization in learning and instruction. Englewood Cliffs, N. J.: Educational Technology Publications, 1974.

Lansky, M. Learning algorithms as a teaching aid. RECALL: Review of educational cybernetics and applied linguistics, 1969, 1, 81-98.

Markov, A. A. Theory of algorithms. Washington: National Science Foundation, 1961.

*This paper was presented to the Convention of the American Educational Research Association, Washington, D. C., March 31, 1975. The paper is abstracted from a technical report written for the US Air Force Office of Scientific Research under Grant # OSR71-2128, awarded to Arizona State University.

Newell, A. & Simon, H. A. Human problem solving. Englewood Cliffs, N. J.: Prentice Hall, Inc., 1972.

Trakhtenbrodt, B. A. Algorithms and automatic computing machines. Lexington, Massachusetts: D. C. Heath and Company, 1963.

Weaver, W. Recent contributions to the mathematical theory of communication. In C. E. Shannon & W. Weaver, The mathematical theory of communication. Urbana, Ill.: The University of Illinois Press, 1949.

Wiener, N. Cybernetics (2nd ed.). Cambridge, Massachusetts: M.I.T. Press, 1948.