DOCUMENT RESUME

ED 089 691 IR 000 403

AUTHOR Merrill, Paul F.
TITLE Algorithmic Organization in Teaching and Learning: The Literature and Research in the USA.
INSTITUTION Florida State Univ., Tallahassee.
PUB DATE Apr 74
NOTE 27p.; Paper presented at the American Educational Research Association Annual Meeting (Chicago, Illinois, April through 19, 1974)

EDRS PRICE MF-$0.75 HC-$1.85 PLUS POSTAGE
DESCRIPTORS *Algorithms; Cognitive Processes; *Educational Research; *Instruction; *Learning; Learning Processes; Literature Reviews; Problem Solving; Research Methodology; State of the Art Reviews; Task Analysis; *Teaching; Teaching Procedures
IDENTIFIERS Retrogression; United States of America; USA

ABSTRACT

A review is presented of the role of algorithms in learning and instruction. The paper first describes an algorithm as a procedure guaranteed to produce a correct result and lists its features as: 1) having a reasonably finite number of unambiguously defined operations; 2) having zero or more inputs from a specified domain; 3) having outputs with a relationship to the inputs; and 4) having operations sufficiently simple to be completed in a finite period. The second section reviews the use of algorithms to describe cognitive processes in learning and problem-solving; and the third, their use in developing instructional strategies. The fourth part of the paper examines the use of algorithms in task analysis; and the following section, their application to instructional materials. The concluding segment discusses research issues related to the use of algorithms in learning and instruction, including the order in which the operations of and paths through algorithms are best taught, the demonstration of the interrelationships among an algorithm's components, and methods of teaching students to synthesize the steps of an algorithm, including the retrogression approach. (Author/PB)

Algorithmic Organization in Teaching and Learning: The Literature and Research in the USA

Paul F. Merrill
Florida State University

An algorithm is a sequence of operations for solving a problem or performing a task which is certain to produce the correct result. Algorithms have had a profound impact on highly quantitative disciplines such as mathematics and computer science. The procedure commonly taught in elementary school for performing long division is an example of a mathematical algorithm. The computer program utilized to compute the gross and net pay for salaried employees is also an algorithm. However, algorithms have also been found to be of considerable value in many other fields. The purpose of this paper is to review and analyze the role of algorithms in learning and instruction within the United States. The general characteristics of algorithms will be described in the first section, and later sections will review the role of algorithms in the following major areas:

(1) The use of algorithms to describe cognitive processes in learning and problem solving,

(2) The use of algorithms to develop and describe complex instructional strategies,

(3) The use of algorithms in task analysis,

(4) The use of algorithms in instructional materials,

(5) Issues and research questions related to algorithms in learning and instruction.

General Characteristics of Algorithms

The term algorithm is generally used to describe a procedure which is guaranteed to produce the correct result. According to Knuth (1968), an algorithm has five important features:

(1) The operations or steps of an algorithm must be unambiguously defined,

(2) An algorithm should produce the correct result in a reasonably finite number of steps,

(3) An algorithm should have zero or more inputs from a specified set of objects or domain,

(4) An algorithm should have one or more outputs having a specified relationship to the inputs, and

(5) The operations or steps of an algorithm should be sufficiently basic so that they can be done precisely in a finite length of time.

However, there are many situations in which such a procedure is not available or is too expensive in terms of the number of steps and time required to achieve a guaranteed correct solution. In such cases, a simplified procedure can often be developed which utilizes shortcuts and rules of thumb which will generally lead to an adequate solution. Heuristic procedures have all of the properties of algorithmic procedures except that they are not guaranteed to always produce the correct solution, and the steps of the procedure may not always be specified in a precise and unambiguous manner. A recipe for baking a cake would be an example of a heuristic procedure. Although this distinction between heuristic and algorithmic procedures is an important one, and most instructional applications involve heuristic procedures, for the purpose of this paper the term algorithm will be utilized in a general sense to refer to both types of procedures.

Algorithms may vary widely in their degree of complexity. Some algorithms may be linear in nature where the same sequence of operations is followed each time the algorithm is executed. In contrast, other algorithms may have decision points where the results or outputs from previous operations are tested or evaluated to determine if certain specific conditions have been satisfied. If the specified condition is satisfied, then one set of operations is performed, while a different set of operations is performed if the specified condition is not satisfied. Such decision points are often referred to as branches. Obviously, there can be a large number of distinct paths through an algorithm which contains decision points.

There may be several different algorithms available for solving a particular problem or performing a given task. Given several algorithms for accomplishing the same task, it is possible to determine which algorithm would be best under given conditions. This determination can be made by comparing the available algorithms on such criteria as efficiency (the length of time required to perform the algorithm or the number of steps executed), economy of memory or storage requirements (Bruner, 1966), simplicity, and the correspondence of the difficulty or complexity of the operations specified in the algorithm to the capability of the individual or machine performing the algorithm.

The Use of Algorithms to Describe Cognitive Processes in Learning and Problem Solving

With the advent of high-speed computer technology, several investigators have attempted to construct computer algorithms or programs which would enable a machine to solve problems normally thought to require intelligence (Newell, Shaw, & Simon, 1963). This early work in artificial intelligence research naturally led to an effort to develop computer algorithms which would simulate the way a human being would perform a given task. Such a computer program was then considered to represent a model of the researcher's hypotheses concerning the cognitive processes underlying the given task. By executing such an algorithm on a computer, predictions of the model could be generated and compared to actual human behavior and subsequently revised to reduce discrepancies between the predictions and actual behavior. The expression of a model of cognitive processes as a computer algorithm constrains the researcher to express his model in a complete and precise manner. These computer simulation models attempt to explain cognitive processes at an information processing level and are based on the premise that human thought processes are composed of elementary symbol manipulation operations. Therefore, an algorithm consisting of an ordered sequence of these basic operations would be a model of the corresponding cognitive process.

Feigenbaum and Feldman (1963) indicate that the following major steps are involved in the development of a computer simulation algorithm or model.

(1) Select a relevant task,

(2) Observe the behavior of individuals performing the task who are asked to "think aloud" and describe what they are doing while performing the task,

(3) Write a preliminary computer algorithm based on the "protocol" data from the previous step,

(4) If insufficient information is available to complete the model, reanalyze old data and/or conduct additional experiments,

(5) Execute the computer simulation with the same task originally given to human subjects,

(6) Compare the output of the computer program with the behavior of the human subjects,

(7) Attempt to identify the sources of error in the computer algorithm and make appropriate revisions.

This approach has been utilized to develop computer algorithms to simulate human behavior in logic problems (Newell & Simon, 1963), verbal learning (Feigenbaum, 1961), concept formulation (Hunt & Hovland, 1961), binary choice experiments (Feldman, 1963), and so forth.

In their classic book, A Study of Thinking, Bruner, Goodnow, and Austin (1956) describe several concept attainment strategies utilized by subjects under a variety of experimental conditions. The strategies utilized by subjects were analyzed in terms of their relationship to specified ideal concept attainment strategies. In essence, these investigators view concept attainment behavior as a sequence of decisions or algorithm designed to obtain information useful in solving a problem. Figure 1 shows a flowchart representation of an algorithm for one of the identified ideal concept attainment selection strategies labeled as the conservative focusing strategy. This algorithm or strategy utilizes a positive instance as a focus, and additional cards are selected which alter one attribute value of the focus card at a time. If the change yields a positive instance, the attribute is considered irrelevant. Bruner and his colleagues (Bruner, et al., 1956) conducted several studies to determine the relationship between strategies or algorithms utilized by subjects and the cognitive strain inherent in different experimental situations. In general, they found that the type of strategy utilized by a subject reflected the nature of the experimental situation, and that strategies differed markedly in terms of the demands they placed on memory and inference capabilities and their efficiency in the utilization of information available in each selected instance.

Insert Figure 1 about here

The fundamental notion that knowledge should be represented in terms of rules or algorithms has been used by Scandura (1973) as the basis for a comprehensive theory of structural learning. The major premise of this theory is that all human behavior is basically rule-governed and that rules provide a more appropriate basis for analyzing complex human learning than do associations. According to this formulation, conceptual and association-governed behaviors are considered to be special cases of rules. It is assumed that behavior is caused by rules and not by overt stimuli, and that stimuli merely provide the occasion for rule using.

Scandura (1973) defines a rule as an ordered triple (D,O,R), where D refers to a domain of stimuli, O refers to an operation, and R refers to a range of responses. Thus, rules are considered to be functions where each stimulus in the domain is paired with exactly one response in the range by a connecting operation. Although the operators are generally treated as indivisible wholes, they are actually composed of several steps in a procedure or algorithm. It is assumed in Scandura's structural learning theory that individuals are goal-seeking information processors, and that in a given goal situation, an individual will apply an appropriate rule if he has at least one available. If the individual does not have a learned rule available, control will shift to a higher order goal of deriving a new rule to satisfy the original goal. When the higher order goal has been achieved, control will shift back to the original goal, and the newly derived rule will be applied to solve the initial problem. According to this theory, learning is considered to be a problem-solving process where higher order rules operate to generate new rules.

The Use of Algorithms to Develop and Describe Complex Instructional Strategies

The design, management, and description of complex instructional systems is greatly facilitated through the utilization of algorithms. Algorithms have been utilized to design and describe complex instructional systems at many different levels from an overall system to a single lesson plan. Yee, Schores, and Skuldt (1970) described the need to develop designs for educational institutions which specify the objectives and interrelate all facets of the system to achieve the objectives. They suggested that systematic flowcharting of educational objectives and processes would facilitate preplanning, management of the system, and reduce uncertainty and error. Flowcharts could be based on a taxonomy of hierarchical sets where a set consists of those processes and objectives at a given level of the instructional system. Thus, flowcharts could be developed at the lesson level, the unit level, the course level, the major level, and the school level with an overall flowchart to show the interrelationship between levels. Walter (1971) advocates the use of algorithmic flowcharts in the development and description of instructional strategies for individualized learning modules. An instructional strategy is defined by the author as a general plan which specifies the sequence of instruction, options available to the student, and the criteria for modifying the instructional sequence. The development of an instructional strategy flowchart would facilitate the specification of the relationship between component parts of the instructional module, and would insure that all conditions which may arise during the use of the module are accounted for. The completed flowchart would also serve as a guide for the potential user in utilizing the components of the module. Figure 2 shows an instructional strategy flowchart for a drill and practice module on four arithmetic operations.

Insert Figure 2 about here

Hansen and his colleagues (Hansen, Brown, Merrill, Tennyson, Thomas, & Kribbs, 1972) have described a set of adaptive instructional models that incorporates complex algorithms. An adaptive instructional model is considered to be a description of a complex instructional strategy which specifies a set of training decision rules. The student's motivation and progress in the course is optimized through the assignment of instructional material, media, and incentive rewards according to his individual characteristics and performance within and prior to the instructional program. The proposed adaptive instructional models include strategies by which student characteristics are matched with a catalog of instructional alternatives under the control of a computer-based algorithm. Thus, these adaptive instructional models were designed to make the assignment of instructional procedures contingent on the interactive characteristics of a given task and the characteristics and performance of the individual learner.

Several different systems approach models for the design of individualized instructional materials have been proposed in recent years (Dick, 1969; Briggs, 1970; Bunderson, 1970). In general, these models are algorithmic in nature and describe the step-by-step procedures for developing individualized instructional material. These systems models are basically similar and tend to differ mainly in areas of emphasis and amount of detail. A representative model described by Dick (1969) contains the following steps: (1) Identify a problem; (2) conduct a task analysis; (3) describe entry behavior; (4) state behavioral objectives; (5) develop evaluation instruments; (6) determine instructional sequence; (7) select appropriate media and instructional procedures; (8) develop instructional materials; and (9) conduct formative and summative evaluation. The formative evaluation step entails the utilization of student performance data in the revision of the products of any prior steps in the process.

Scandura and his colleagues (Scandura, 1973a; Scandura, 1963b; Ehrenpreis and Scandura, 1972) have proposed an algorithmic approach to curriculum construction based on a theory of structural learning. This approach is based on the following ideas:

(1) Behavior may be accounted for by the invention of a finite set of rules which can be used to generate the given set of behaviors;

(2) Higher order rules may operate upon other rules to generate another set of rules;

(3) A school curricula may be defined in terms of a finite number of behavioral objectives which correspond directly to a finite set of rules.

Therefore, an algorithmic approach to curriculum construction is viewed as a process of identifying a finite set of rules and higher order rules which account for the desired behaviors expressed in terms of behavioral objectives. This approach generally consists of the following steps:

(1) Identify a set of tasks in behavioral terms;

(2) Specify a set of rules which the learner must know in order to generate the behaviors specified;

(3) Identify higher order relationships which may exist among the rules by searching for parallels or invariance among the different rules;

(4) Eliminate those rules which may be derived from the higher order rules.

Ehrenpreis and Scandura (1972) report the results from two studies which show that the algorithmic approach to curriculum construction described above was feasible, and that a curriculum defined in terms of rules and higher order rules provides an adequate basis for instruction and facilitates transfer to new tasks.

Scandura's algorithmic approach to curriculum construction does not describe in sufficient detail the instructional procedures for teaching the rules that are identified. However, such procedures have been described by Gagne (1970), Evans, Homme, and Glaser (1962), and P. F. Merrill (1972). Each of these procedures is algorithmic in nature and specifies an instructional strategy or paradigm for teaching rule-governed behavior. Merrill's paradigm is a synthesis and extension of the procedures proposed by Gagne and Evans, et al., and specifies a sequence of displays or frames. Each display contains some of the following components: (1) behavioral objectives; (2) verbal statements of the rule; (3) examples of the rule; (4) partial statements of the rule; (5) incomplete examples; (6) prompts (verbal descriptions of how an example relates to the rule); and (7) feedback concerning the correctness of a student's response to a partial problem or incomplete example.

According to this instructional strategy, supporting stimuli such as objectives, rules, examples, and prompts are faded gradually until the student is able to solve problems and state the rule with minimal support. Provision is made for students to skip highly prompted frames and some of the fading frames based on his performance on preceding response frames.

The Use of Algorithms in Task Analysis

The purpose of conducting a task analysis is to determine the sub-skills or component operations which are required to learn and/or perform a task and to identify the interrelationships between the sub-tasks. The most widely accepted model for task analysis has been proposed by Gagne (1970). He states that in analyzing a terminal objective, it is possible to define a hierarchy of sub-skills such that lower order skills or behaviors would generate positive transfer to skills at a higher level. Such an analysis may be performed by starting with the terminal behavior and identifying subordinate skills by asking the question: "What would an individual already have to know how to do in order to learn the new capability simply by being given verbal instructions?" This question is asked recursively of each sub-behavior identified until the assumed student entry behaviors are determined.

P. F. Merrill (1971) has proposed an algorithmic approach to task analysis. According to this approach, information obtained from observing an expert perform the terminal task is used to outline an algorithmic procedure of the task. The individual being observed is asked to "think aloud" while performing the task, and detailed notes are recorded concerning the information or objects operated upon, the specific operations being performed, the results of each operation, and all decision points encountered. The algorithm generated, based on this data, could be empirically tested by having an individual follow the algorithm in attempting to perform the task. The algorithm should reveal the output/input relationships between sub-operations of the task wherein the results or outputs of initial operations are utilized as part of the inputs for succeeding operations. Thus, an algorithmic analysis should reveal the performance sequence of the sub-operations of a task.

Scandura (1971, 1973) also has proposed an algorithmic approach to task analysis. According to the structural theory of learning, all behavior may be generated by rules, and a rule is essentially considered to be an algorithm for generating a set of responses from a corresponding set of stimuli. Thus, an algorithmic task analysis consists of decomposing a rule into its component steps. These steps and their ordered relationships may then be represented in terms of a flow diagram or a directed graph where decision rules are represented by points and operating rules by arrows. Any given step of the algorithm may be broken down further into a sub-algorithm whose steps may in turn be further broken down into a sub-subalgorithm. Scandura (1970) suggests that this breakdown corresponds directly to the hierarchies obtained by conducting the analysis suggested by Gagne (1970).

The algorithmic analysis described above also provides an efficient method for determining those rules or parts of rules which a subject can perform. It is hypothesized that a rule may be broken down into simple enough steps that every subject in a given population will be able to perform each step of the rule in an all-or-none fashion. It is also hypothesized that each path through a procedure or algorithm of a rule may be performed in an all-or-none fashion. Each path through the algorithm effectively partitions the range and corresponding domain of the algorithm into a set of mutually exclusive equivalence classes of stimuli and corresponding responses. Therefore, it is possible to determine which path of an algorithm a student has learned by testing him on one item selected from each equivalence class. According to the above assumptions, success on any item from one equivalence class implies success on all other items in the same class. Figure 3 shows examples of a flowchart, directed graph, paths, and corresponding instances of an algorithm for generating the "next" numeral in base three. Implicit in this approach is the assumption that the paths of an algorithm can be partially ordered according to difficulty. The most direct path would be the least difficult while paths which included the steps of another path would be more difficult. The results of several studies reported by Scandura and Durnin (1971) provide considerable support for the hypotheses that: (1) Success on one item in an equivalence class implies success on other items in the class, and (2) success on a higher order path which includes steps from a lower order path implies success on the subordinate path. The authors suggest that this approach has important implications for computer-assisted instruction, diagnostic testing, and sequential testing.

Insert Figure 3 about here

The Use of Algorithms in Instructional Materials

Davies (1970) suggests that algorithms may be used to improve the communication process. He argues that continuous prose inadequately expresses the complex, logical interrelationships involved in rules and regulations. Changing continuous prose to an algorithm would make rules and regulations intelligible by exactly delineating the components of the decision-making process. Diagramming such algorithms through the use of flowchart symbols has proven to be very valuable in improving communication. Unskilled people can learn to do complex tasks if they are given an algorithm to follow as a learning and/or job aid. Through the use of algorithms, individuals can make simple decisions one at a time without having to worry about previous decisions. Lewis, Horabin, and Gane (1967) compared the intelligibility of a capital gains tax regulation presented in three different forms. The results showed that time and errors were greatly reduced by following an algorithmic form as compared to the official prose or a simplified prose form. Horabin (1974) has described several examples of the application of algorithms in training situations.

The literature also contains many examples of the use of algorithms in instructional materials for elementary, secondary, and higher education. Overholser (1966) describes a technique for introducing elementary school children to an algorithmic approach for solving arithmetic formulas. This technique involves the use of a relay game where one child performs an operation of the algorithm and relays his result to another child, who performs the next operation and relays his result to the next child. Kessler (1970) describes the use of a "how to" story game to introduce elementary children to the algorithmic analysis process of breaking down an activity into its component parts and the sequencing of those parts into logical order. The children are asked to write a story of how to accomplish some activity by placing each step of the activity on a separate card. A friend is asked to try to arrange the shuffled cards into the proper order for performing the activity. The author suggests that the "how to" story game is a valuable tool in helping children to develop logical thinking skills. Chilcote (1970) argues that encouraging elementary school children to write "how to" stories on cards and then use flowchart symbols to draw flowcharts of their stories will help them to learn to organize ideas and actions in logical sequences and thereby improve their creative writing skills.

Gust (1970) advocates that instructors use several different levels of algorithmic flowcharts in bookkeeping instruction. Thought pattern flowcharts could be used to capture and describe the thinking process utilized by a student. Operational flowcharts could be used to graphically portray the detailed processing of an information system. An overall systems flowchart would help students get the big picture of the bookkeeping cycle and would allow them to see how the detail fits into the larger framework of the total system. Weaver (1969) also advocates the use of algorithmic flowcharts in business to communicate, understand, and analyze aspects of an accounting system. He further suggests that having the employee flowchart the procedures he performs would facilitate the detection of employee conceptual errors by the supervisor.

M. D. Merrill and Boutwell (1973) describe the use of algorithms as prompts or as feedback in concept and rule instruction. In a concept learning task, an algorithmic search strategy could be presented to the student for identifying and searching for the relevant attributes in order to appropriately classify the given example. Algorithmic search strategies could also be utilized to identify which of several available rules should be applied in solving a particular problem. Harshbarger (1971) has written a statistics textbook organized around an algorithmic decision map for classifying statistical problems. The decision map may be utilized by the researcher to determine the appropriate method of solution for a given statistical problem.

M. D. Merrill, Barton, and Wood (1970) investigated the use of algorithms as feedback in a rule-using task. A step-by-step breakdown of the rule for solving the problem was presented as a "specific review" treatment. The presentation of this algorithmic feedback following incorrect responses decreased the amount of time required to learn the task although the *Ss* were given approximately 30 percent more material.

Issues and Research Questions Related to Algorithms in Learning and Instruction

One of the major issues related to algorithms which needs further investigation concerns the determination of an efficient instructional strategy or algorithm for teaching students a fairly complex algorithm. Although rules may be considered to be algorithms (Scandura, 1973), most instructional strategies for teaching rules view rules as indivisible wholes rather than as complex procedures made up of many component operations and decision points. Therefore, researchers need to investigate such questions as:

(1) In what order do you teach the individual operations of an algorithm?

(2) In what order do you teach the individual paths through an algorithm?

(3) How do you show students the interrelationship between the different operations and paths of the algorithm?

(4) How do you teach the student to put it all together?

(5) Would Gilbert's (1962) retrogression approach to establish chains be effective for teaching a complex algorithm?

A second major issue concerns whether or not the teaching of algorithms degrades the student by merely having him perform a set of mechanical operations. Although it is possible to have an individual utilize an algorithm in a rote fashion, it is also possible to utilize that same algorithm to help the student achieve an overall view and understanding of a complex process. There are several examples in the literature of the use of algorithms to promote understanding rather than rote learning. Silvey (1970) describes the use of a guided discovery approach to help fourth graders develop their own algorithm of the complementary method of subtraction. Lowry (1965) advocates teaching young children long arithmetic procedures designed to help the child see each step of the procedure. The child should also be able to justify each step according to his level of understanding of our numeric system and the principles governing the operations of numbers. Then the teacher should help the child work toward more efficient and shorter algorithms which he can also rationalize. The instruction should be individualized so each child can work with procedures he understands and can be encouraged to discover ways to shorten them with hints and guidance. This approach is contrasted with teaching the child an adult algorithm with little attempt to point out why the algorithm works. The Maryland Elementary Mathematics Inservice Program (Mueller, 1970) utilizes carefully structured games to introduce the properties of a mathematical system to inservice elementary teachers. The game rules, along with physical objects, are used to justify or demonstrate procedures of specific arithmetic algorithms. By analyzing the moves of the game, the student is able to identify a set of rules that are the physical demonstration of the mathematical properties of an abstract mathematical structure such as a field.

The final issue to be described in this paper concerns the most appropriate representation of a given algorithm. Bunderson (1970) describes in detail the importance of different types of representation of instructional content to the student's learning efficiency. Algorithms may be represented as (1) prose text, (2) numbered steps, (3) question lists, (4) branching booklets, (5) flowcharts, (6) directed graphs, (7) decision tables, etc. Lewis (1970) ably describes the relative merit of prose text, question lists, flowcharts, and decision tables. He argues that a prose description of an algorithm is very difficult to interpret and may hide contradictions, redundancies, or ommissions. Flowcharts are visual in nature and provide a pictorial image which is easy for the user to follow and greatly improves communication. However, flowcharts are laborious to draw and often difficult to alter. Question lists do not have the advantages of flowcharts but may be easier for some people to use because of their similarity to ordinary text. However, Lewis advocates the use of decision tables to represent algorithms for most applications. Decision tables are easier to draw, easier to change, and more compact than flowcharts. There also exists several techniques for evaluating the accuracy and completeness of a given decision table. For algorithmic analysis purposes, decision tables are greatly superior to flowcharts or question lists. Their clear separation of actions and conditions facilitate the identification and description of logically distinct rules or paths through the algorithm. Decision tables may very well be a more useful representation than directed graphs for determining the distinct paths through an algorithm.

Figure 4 shows a decision table representation of the algorithm represented by a flowchart and directed graph in Figure 3. The first table merely contains the first initial step and a link to the second table. The second table is divided into four quadrants by the double lines. The upper left quadrant specifies all the questions or decision points; the upper right quadrant identifies all the possible combinations of answers to the questions listed; the lower left quadrant shows all the operations or steps to be performed; and the lower right quadrant shows which operations should be performed for each combination of answers. Each column in the right half of the table represents a distinct path through the algorithm. Since Columns 3 and 4 lead to the same operations, Column 4 could be eliminated. Columns 2 and 3 of the table correspond directly to Paths II and I of Figure 3 respectively. Paths III and IV from Figure 3 correspond to combinations of Columns 1 and 2 and Columns 1 and 3.

Insert Figure 4 about here

Summary

The purpose of this paper was to review and analyze the role of algorithms in learning and instruction within the United States. The term algorithm was defined and its characteristics were outlined. The use of algorithms in learning and instruction to describe cognitive processes in learning, to develop and describe complex instructional strategies, to analyze instructional tasks, and to facilitate communication were reviewed. Three major issues and research questions related to algorithms were discussed. M. D. Merrill and Boutwell (1973) have argued that directions for processing information in the form of algorithms may prove to be one of the most powerful instructional tools we have available. The logical process of breaking down an activity or process into its component parts and then developing a structure or sequence of those parts to reveal their interrelationship has great potential in many aspects of our lives. Let's harness this potential to facilitate the learning and instruction process.

References

Briggs, L. J. Handbook of procedures for the design of instruction. Pittsburth, Penn.: American Institute for Research, 1970.

Bruner, J. S. Toward a theory of instruction. Cambridge: Harvard University Press, 1966.

Bruner, J. S., Goodnow, J. J., & Austin, G. A. A study of thinking. New York: John Wiley and Son, 1956.

Bunderson, C. V. The computer and instructional design. In W. H. Holtzman (Ed.), Computer-assisted instruction, testing, and guidance. New York: Harper and Row, 1970.

Chilcote, E. Flowchart stories--A systems approach to creative writing. Grade Teacher, 1970, 88, 66-69.

Davies, I. K. Algorithms. Psychology Today, 1970, 3(11), 53-69.

Dick, W. New directions in learning. In D. Hansen, W. Dick, & H. Lippert (Eds.) Annual progress report. Tallahassee: CAI Center, Florida State University, 1969.

Ehrenpreis, W. & Scandura, J. M. The algorithmic approach to curriculum construction in mathematics: A field test. Structural Learning Series, Mathematics Education Research Group Report No. 64. Philadelphia: University of Pennsylvania, 1972.

Evans, J. E., Homme, L. E., & Glaser R. The ruleg system for the construction of programmed learning sequences. The Journal of Educational Research, 1962, 55, 513-518.

Feigenbaum, E. A. The simulation of verbal learning behavior. In E. A. Feigenbaum & J. Feldman (Eds.) Computers and thought. New York: McGraw Hill, 1963. Pp. 297-309.

Feigenbaum, E. A. & Feldman, J. (Eds.) *Computers and thought*. New York: McGraw Hill, 1963.

Feldman, J. Simulation of behavior in the binary choice experiment. In E. A. Feigenbaum & J. Feldman (Eds.) *Computers and thought*. New York: McGraw Hill, 1963. Pp. 329-346.

Gagne, R. M. *The conditions of learning*. (2nd ed.) New York: Rinehart and Winston, 1970.

Gilbert, T. F. Mathetics: The technology of education. *Journal of Mathetics*, 1962, *1*, 7-73.

Gust, E. D. Using flowcharting concepts in bookkeeping instruction. *Business Education Forum*, 1970, *25*, 45-46.

Hansen, D. N., Brown, B. R., Merrill, P. F., Tennyson, R., Thomas, D. B., & Kribbs, H. D. (Eds.) The analysis and development of an adaptive instructional model(s) for individualized technical training. Technical Report, Tallahassee: CAI Center, Florida State University, 1972.

Harshbarger, T. R. *Introductory statistics: A decision map*. New York: MacMillian Co., 1971.

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.

Hunt, E. B. & Hovland, C. I. Programming a model of human concept formulations. In E. A. Feigenbaum & J. Feldman (Eds.) *Computers and thought*. New York: McGraw Hill, 1963. Pp. 310-325.

Kessler, B. M. A discovery approach to the introduction of flowcharting in the elementary grades. *The Arithmetic Teacher*. 1970, *17*, 220-224.

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

Lewis, D. N., Horabin, I. S., & Gane, C. P. *Case studies in the use of algorithms*. London: Pergamon Press, 1967.

Lowry, W. C. Structure and the algorisms of arithmetic. *The Arithmetic Teacher*, 1965, *12*, 146-150.

Merrill, M. D., Barton, K., & Wood, L. E. Specific review in learning a hierarchical imaginary science. *Journal of Educational Psychology*, 1970, *61*, 102-109.

Merrill, M. D. & Boutwell, R. C. Instructional development: Methodology and Research. In F. N. Kerlinger (Ed.) *Review of Research in Education*. Itasca, Illinois: F. E. Peacock, 1973.

Merrill, P. F. Task analysis--An information processing approach. Technical Memo No. 21, Tallahassee, Fla.: CAI Center, Florida State University, 1971.

Merrill, P. F. Complex tutorial model: Rule learning and problem solving. In D. N. Hansen, B. R. Brown, P. F. Merrill, R. Tennyson, D. B. Thomas, & H. D. Kribbs (Eds.) The analysis and development of an adaptive instructional model(s) for individualized technical training. Technical Report, Tallahassee: CAI Center, Florida State University, 1972.

Mueller, F. J. Games and algorithms--A new view in elementary school mathematics for teachers, University of Maryland Mathematics Project. *The Arithmetic Teacher*, 1970, *17*, 342-346.

Newell, A., Shaw, J. C., & Simon, H. A. Empirical explorations with the logic theory machine: A case study in heuristics. In E. A. Feigenbaum and J. Feldman (Eds.) *Computers and thought*. New York: McGraw Hill, 1963. Pp. 109-133.

Newell, A., & Simon, H. A. GPS, A program that simulates human thought. In E. A. Feigenbaum & J. Feldman (Eds.) Computers and thought. New York: McGraw Hill, 1963. Pp. 279-293.

Overholser, J. S. Flowcharts for the elementary grades. The Arithmetic Teacher. 1966, 13, 59-63.

Scandura, J. M. Deterministic theorizing in structural learning: Three levels of empiricism. Journal of Structural Learning, 1971, 3, 21-53.

Scandura, J. M. Structural learning and the design of educational material. Educational Technology, 1973, 13(8) 7-13. (a)

Scandura, J. M. Structural learning: I. Theory and research. New York: Gordon and Breach, 1973. (b)

Scandura, J. M. & Durnin, J. H. Assessing behavior potential: Adequacy of basic theoretical assumptions. Structural Learning Series, Mathematics Education Research Group Report No. 62. Philadelphia: University of Pennsylvania, 1971.

Silvey, I. M. Fourth graders develop their own subtraction algorithm. The Arithmetic Teacher, 1970, 17, 233-236.

Walter, K. A. Authoring individualized learning modules: A teacher training manual. Rockville, Maryland: Montgomery County Public Schools, 1971.

Weaver, D. H. Flowcharting as a teaching/learning tool. Business Education World. 1969, 49, 18-20.

Yee, A. H., Schores, J., & Skuldt, K. Systematic flowcharting of educational objectives and processes. A. V. Communication Review, 1970, 18(1), 72-83.

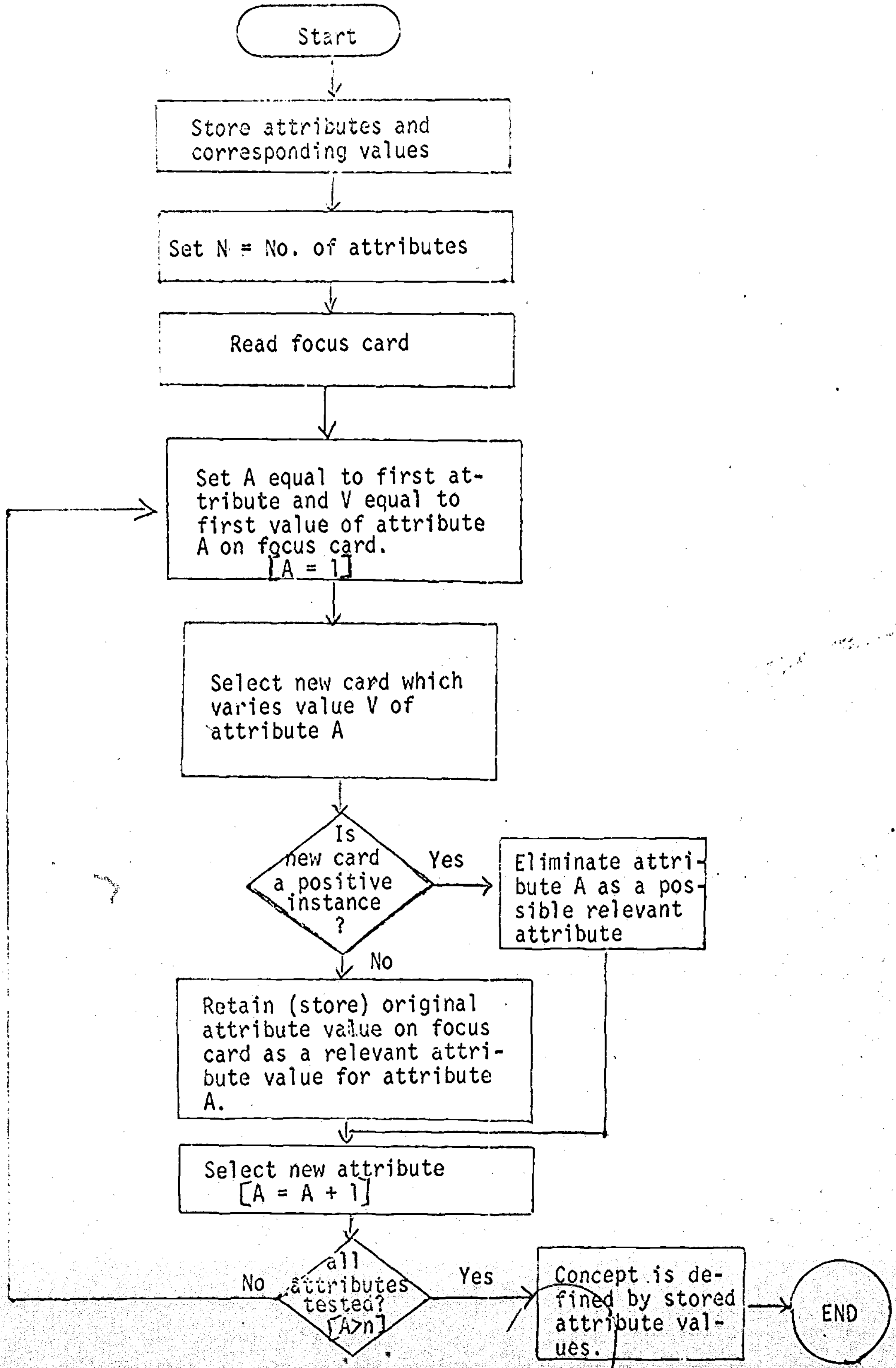
Start
Store attributes and corresponding values
Set N = No. of attributes
Read focus card
Set A equal to first attribute and V equal to first value of attribute A on focus card. [A = 1]
Select new card which varies value V of attribute A
Is new card a positive instance ?
Yes
Eliminate attribute A as a possible relevant attribute
No
Retain (store) original attribute value on focus card as a relevant attribute value for attribute A.
Select new attribute [A = A + 1]
all attributes tested? [A>n]
No
Yes
Concept is defined by stored attribute values.
END

Figure 1. Algorithm of conservative focusing selection strategy in Concept Attainment. Based on Bruner et al. (1956) p.87.

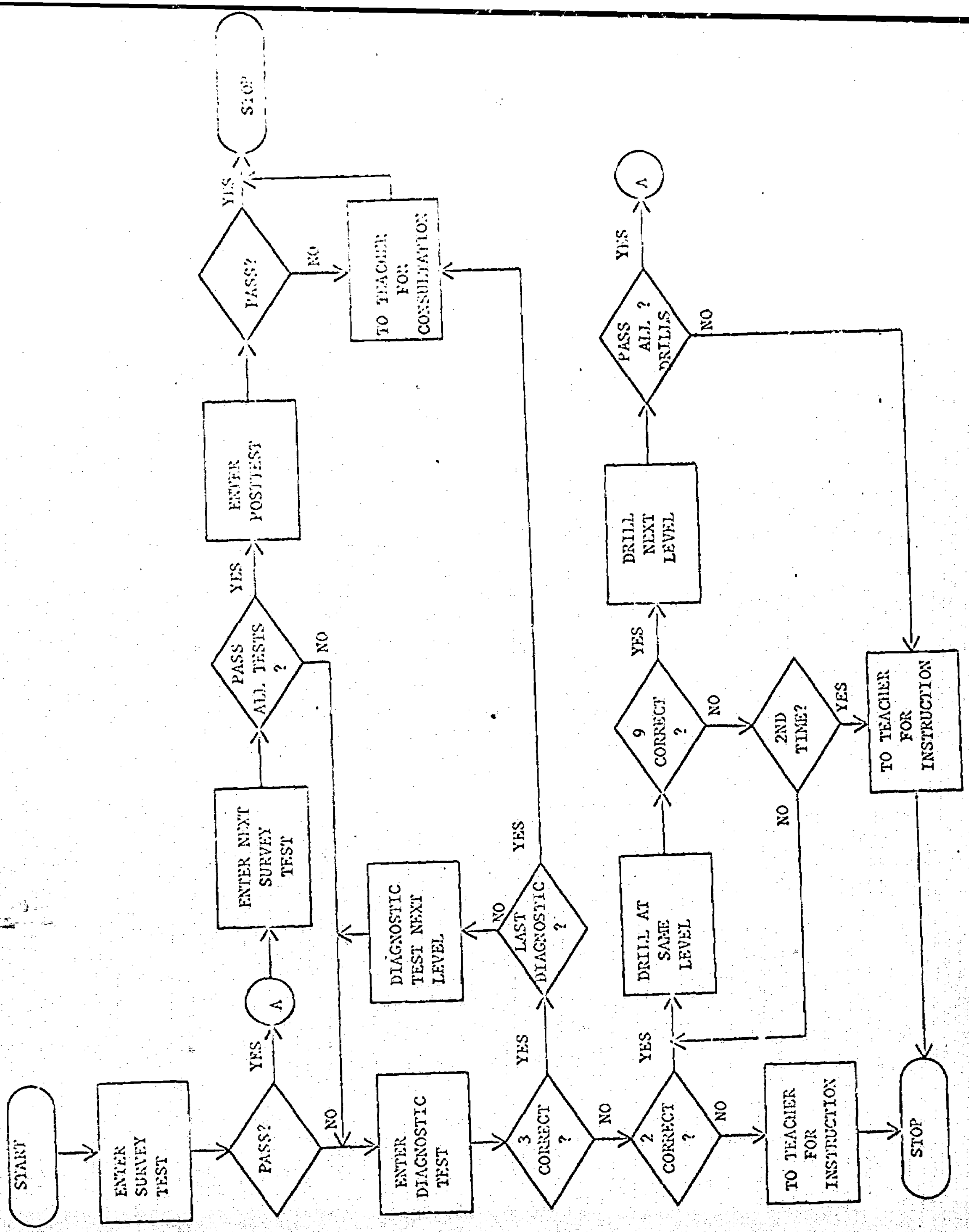

Figure 2. Algorithm of an instructional strategy for a drill and practice module on four arithmetic operations. Flowchart reproduced from Walter (1971) p. 95.

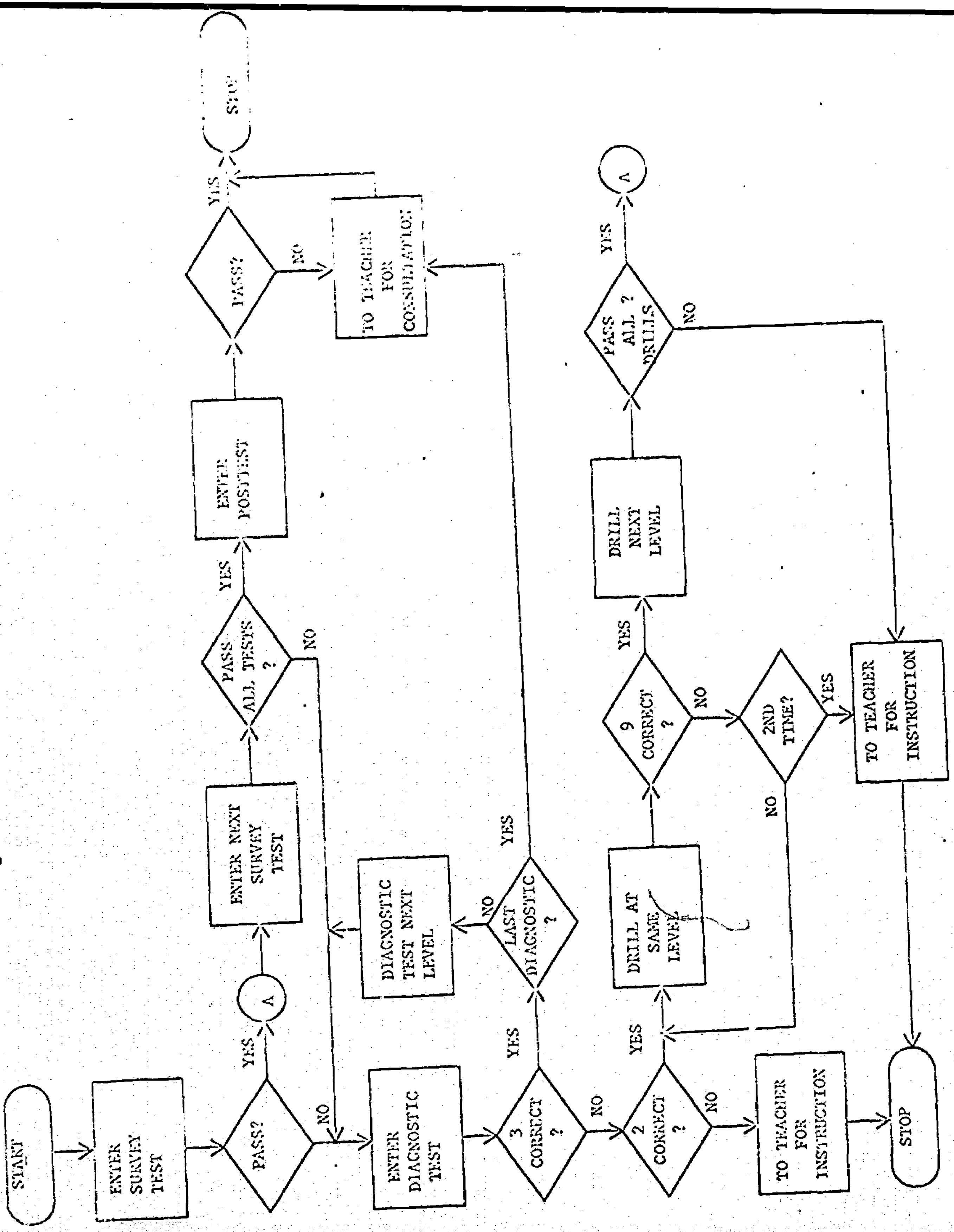

Figure 2. Algorithm of an instructional strategy for a drill and practice module on four arithmetic operations. Flowchart reproduced from Walter (1971) p. 95.

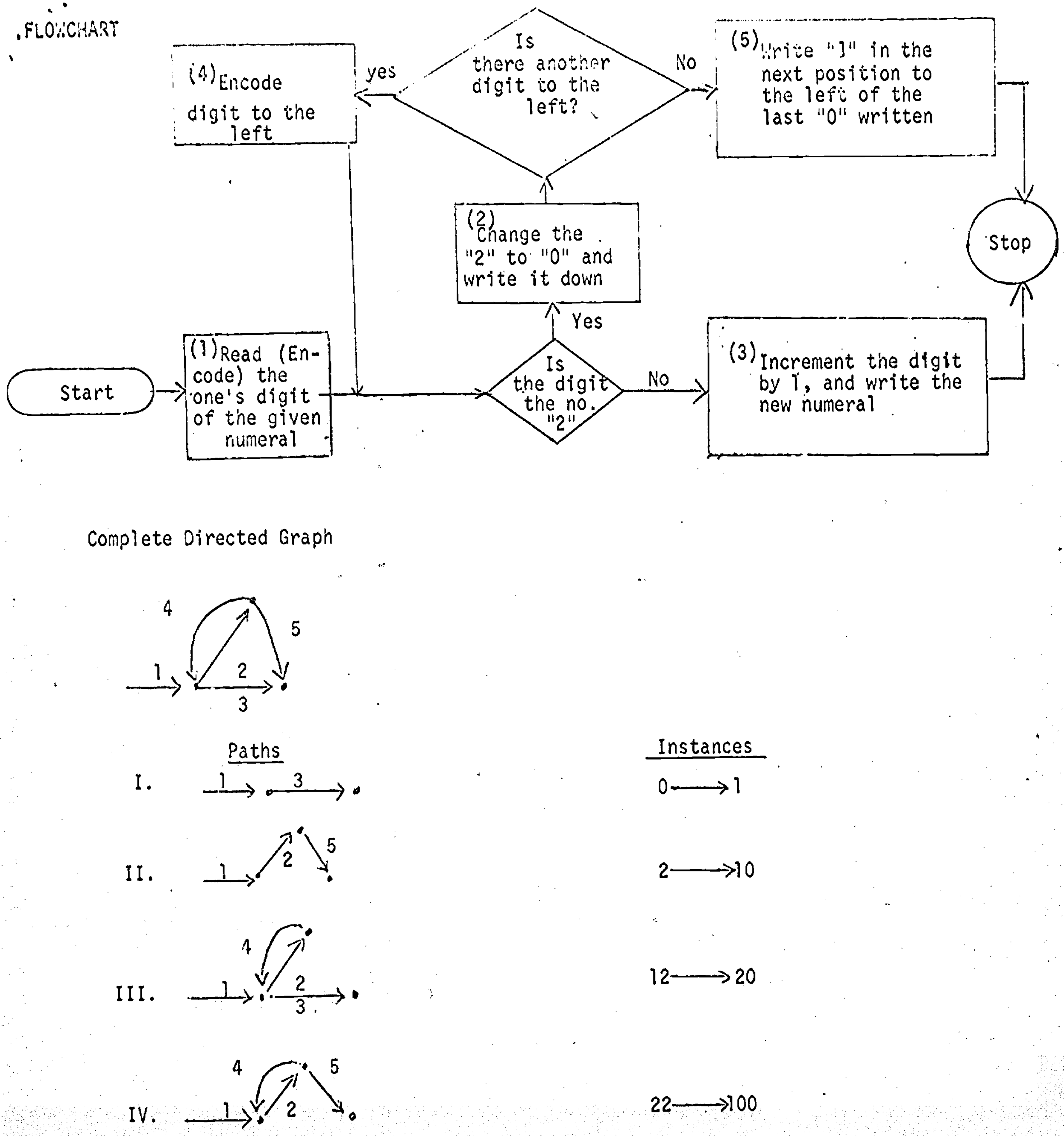
FLOWCHART
Start
(1) Read (Encode) the one's digit of the given numeral
Is the digit the no. "2"
Yes
No
(2) Change the "2" to "0" and write it down
Is there another digit to the left?
yes
No
(4) Encode digit to the left
(5) Write "1" in the next position to the left of the last "0" written
(3) Increment the digit by 1, and write the new numeral
Stop
Complete Directed Graph
1
2
3
4
5
Paths
I.
II.
III.
IV.
Instances
0 → 1
2 → 10
12 → 20
22 → 100

Figure 3. Flowchart, Directed Graph, Paths, and Instances of an algorithm for generating the "next" numeral in base three. Algorithm and directed graphs taken from Scandura (1973b), p. 184.

START TABLE	
A. Read (Encode) the one's digit of the given numeral	(1)
B. Go to INCREMENT TABLE	

INCREMENT TABLE		1	2	3	4
A. Is the digit the number "2"?		Y	Y	N	N
B. Is there another digit to the left?		Y	N	Y	N
C. Change the "2" to "0" and write it down	(2)	X	X		
D. Increment the digit by 1, and write the new numeral	(3)			X	X
E. Encode Digit to the left	(4)	X			
F. Write "1" in the next position to the left of the last "0" written.	(5)		X		
G. Go to INCREMENT TABLE		X			
H. Stop			X	X	X

Fig. 4. Decision Tables of algorithm for generating the "next" numeral in base three. Algorithm taken from Scandura (1973b), p. 184.