DOCUMENT RESUME

ED 277 349 IR 012 439

TITLE Technology Report.

INSTITUTION Mid-Continent Regional Educational Lab., Inc., Kansas City, Mo.

SPONS AGENCY Office of Educational Research and Improvement (ED), Washington, DC.

PUB DATE 30 Nov 85

NOTE 14p.

PUB TYPE Reports - Research/Technical (143)

EDRS PRICE MF01/PC01 Plus Postage.

DESCRIPTORS *Algorithms; Elementary Education; Field Tests; *Problem Solving; *Quasiexperimental Design; Reinforcement; Research Methodology; *Skill Development; *Training Methods

IDENTIFIERS *Conceptual Models; Likert Scales; *LOGO Programing Language

ABSTRACT

In order to determine the extent to which the LOGO programming language can be used as a reinforcer for general problem solving ability, this field study used a production theory approach to problem solving as a conceptual model, and then translated the model into a LOGO oriented framework. The methodology tested the use of an algorithm in a quasi-experimental fashion by presenting it to students, and then determining the extent to which it increased their ability to solve LOGO problems. Four subjects, one each from grades 4, 5, 6, and 8, were presented with standard LOGO problems. (All students had received at least 4 hours of instruction in using LOGO commands.) Students were then rated using a Likert scale. The results indicate that a general problem solving algorithm does not significantly change the problem solving ability of students as it relates to LOGO problems, which implies that a "stronger" algorithm approach should be developed for teaching problem solving for different types of problems. However, the results also suggest ways that LOGO problem solving techniques might generalize to more specific algorithms because of LOGO's highly visual nature, recursive features, and provision of immediate feedback. For the reinforcement of these aspects, LOGO appears to be a highly useful teaching tool. Five references are listed. (DJR)

TECHNOLOGY REPORT

Submitted to the Office of
Educational Research and Improvement

by
The Mid-continent Regional Educational Laboratory

November 30, 1985

This report was prepared by the Mid-continent Regional Educational Laboratory, a not-for-profit educational laboratory. The activities reported herein were performed pursuant to a contract from the National Institute of Education, Department of Education. However, the opinions expressed herein do not necessarily reflect the position or policy of the National Institute of Education, and no official endorsement by the National Institute of Education should be inferred.

INTRODUCTION

The purpose of this report is to summarize the findings of the field testing efforts to determine the extent to which the programming language LOGO can be used as a reinforcer for general problem solving ability. As described in the Progress Report on Technology Activities submitted in December, 1984 there is a long standing debate among problem solving theorists as to the utility of general verses specific problem solving methods (Polson and Jeffries, 1985).

General problem solving techniques have their roots in the "weak" method approach to information processing and from divergent-production theories of cognition (Polson and Jeffries, p. 448 and p. 426). Specific problem solving methods original from the "strong" method approach within information processing. Weak verses strong methods differ primarily in terms of generalizability and specifically.

Weak methods like those developed by Polya (1985) are algorithms with very general components. For example, Polya's algorithm had four basic steps: 1) understand the problem, 2) devise a plan, 3) carry out the plan and 4) look back. The assumption underlying weak methods is that they are generalizable. That is, Polya assumes that his four steps could be applied to different types of problems in mathematics, different types of problems in science, etc. The emphasis in weak methods, then, is the transferability of the algorithms; within weak methods there is very little specific content knowledge taught.

Strong methods, on the other hand, emphasize specificity the assumption here is that for each problem there is highly specific declarative and procedural knowledge necessary to solve the problem . To illustrate, consider the following algorithm developed for a computer simulations program called ACT, Adaptive Control of Thought (Anderson, 1983):

P1 IF the goal is to do an addition problem
THEN the subgoal is to itcrate through the columns of the problem.

P2 IF the goal is to iterate through the columns of an addition problem
and the rightmost column has not been processed
THEN the subgoal is to iterate through the rows of that rightmost column
and set the running total to 0.

P3 IF the goal is to iterate through the columns of an additional problem
and a column has just been processed
and another column is to the left of this column
THEN the subgoal is to iterate through the row of this column to the left
and set the running total to the carry.

P4 IF the goal is to iterate through the columns of an problem
and the rightmost column has not been processed
and there is a carry
THEN write out the carry
and POP the goal.

P5 If the goal is to iterate through the columns of an addition problem
and the last column has been processed
and there is no carry
THEN POP the goal.

P6 IF the goal is to iterate through the rows of a column
and the top row has not been processed
THEN the subgoal is to add the digit of the lower row to the running total.

P8 IF the goal is to iterate through the rows of a column
and the last row has been processed
and the running total is a digit
THEN write the digit
and delete the carry
and mark the column as processed
and POP the goal.

P9 IF the goal is to iterate through the rows of a column
and the last row has been processed
and the running total is of the form "string digit"

```
THEN write the digit
     and set carry to the string
     and mark the column as processed
     and POP the goal.

P10  IF the goal is to add a digit to another digit
     and a sum is the sum of the two digits
THEN the result is the sum
     and mark the digit as processed
     and POP the goal.

P11  IF the goal is to add a digit to a number
     and the number is of the form "string digit"
     and a sum is the sum of the two digits'
     and the sum is less than 10
THEN the result is "string sum"
     and mark the digit as processed
     and POP the goal.
P12  If the goal is to add a digit to a number
     and the number is of the form "string digit"
     and a sum is the sum of the two digits
     and the sum is of the form "1 digit*"
     and another number sum* is the sum of 1 plus string
THEN the result is "sum* digit*"
     and mark the digit as processed
     and POP the goal.
```

Of course the terminology and format used above would not be part of a classroom instructional model, yet this example illustrates the specificity inherent is strong methods. That is, the intent of strong methods is to provide a high level of detail in the facts (declarative information) and processes (procedural information) necessary to solve different problem types. Ideally a strong algorithm should provide a student with a solution if the algorithm is followed precisely. The strength, then of strong methods is their specificity; their weakness is their lack of generality. That is, a strong method approach to problem solving would require the identification and teaching of many and varied algorithms for problem solving.

THE STUDY

The intent of the present study was to test the effects of a general problem solving algorithm on student ability to solve LOGO problems and to determine if programming languages such as LOGO have any inherent characteristics which would make them useful tools for teaching general problem solving strategies. To do this a production theory approach to problem solving was used. Specifically Anderson's (1983) construct of a production was used to develop a general problem solving algorithm. That general production theory problem solving algorithm is reported below.

1. Identify what is missing: Missing data can take the form of:

   a) a missing antecedent in a specific production

   b) a missing consequent in a specific production

   c) a missing production in a production network

   d) a missing network or plan of control within a network

2. Decide whether: a) the missing information can be inferred or b) outside information should be obtained. If b, exit the problem solving algorithm and return when information is obtained. If a ask and answer the question: "In what prior situation have I made a similar inference?"

3. Test out the inference

4. Determine if the inferred missing data complete the production or production network.

The model above was considered "conceptual" in that it was intended to identify all of the components of a problem from a production theory perspective.

The next step was to translate the conceptual model into a more student friendly LOGO oriented framework. That framework is reported below:

Step #1 What does the final product look like?

Step #2 Where are you now?

Step #3 Is there a procedure you have to use?

Step #4 If you don't know a procedure, where can you find one?

Step #5 Try the procedure?

Step #6 Where are you now?

Step #7 If you are not finished go back to Step #3

This algorithm was tested in a quasi-experimental fashion by presenting it to students and then determining the extent to which it increased their ability to solve LOGO problems. For subjects were used for the study --one from each of the following grade levels 4, 5, 6 and 8. These students were first presented with standard LOGO problems. All students had received at least four hours of instruction using LOGO commands. Students were then rated using a Likert scale (H - M - L) on the following characteristics:

- ability to describe the problem
- ability to identify the procedures necessary to solve the problem
- ability to solve the problem successfully

The results of the Likert ratings for the four students are reported below:

	L	M	H	
Ability to describe problem	4th, 8th	5th, 6th		
Ability to identify procedures	4th, 6th 8th	5th		
Ability to successfully solve problem	8th	4th	5th, 6th	

In addition to the ratings the four students were characterized on the following way, relative to their problem solving behaviors:

4th grader: jumps into problems, tries solution without much forethought, does not check the accuracy of his steps.

5th grader: is persistent when working through a problem, takes time when approaching a problem, can describe why she is performing certain steps.

6th grader: thoughtful approach to problems, can describe why he performs certain steps, is persistent, checks results.

8th grader: lack of persistence, tries solutions without much forethought, becomes discouraged easily.

Students were presented with the algorithm and the process was modeled for them. Students then practiced using the algorithm on a few structured examples. Students were then asked to use the algorithm when solving a set of LOGO problems which they worked on independently. After the students had solved the assigned set, their LOGO problem solving ability was again assessed. The post-treatment rankings are reported below.

	L	M	H
Ability to describe problem		4th, 8th	5th, 6th
Ability to identify procedures	4th, 6th, 8th	5th	
Ability to successfully solve problems		8th, 4th 8th, 4th	5th, 6th 5th, 6th

There results indicate that there was a slight increase in students' ability to frame the problems and a slight increase in their ability to correctly solve problems. However there was no apparent change in their ability to identify the procedures as processes necessary to solve the problems. These findings were consistent with the reported changes in problem solving characteristics. That is, use of the algorithm seemed to increase such general factors as students persistence and attention to detail but not their ability to identify specific procedures to be used in solving LOGO problems.

These findings support the importance of domain specific information in solving problems. Domain specific knowledge "consists of a well-formed semantic network of valid information as well as strategies for using this information" (Doyle 1983, p. 168). In other words, to successfully solve LOGO problems, students must know well the specific procedures and commands within the LOGO language. This assertion is quite consistent with the research of Heller and Reif (1984) --namely that without the factual knowledge relative to a content area, students have little success in acquiring more complex procedures for the content area.

Given the importance of domain specific knowledge and the apparent lack of effect of a general algorithm on problem solving ability, it would appear that more "strong" algorithms should be developed for teaching problem solving. This would mean that more algorithms would have to be developed for different problem types. That is, an algorithm or algorithms should be developed specific to problem solving in LOGO, in BASIC and other languages along with algorithms for specific types of mathematics problems and algorithms for specific types of science problems. Although different algorithms have been developed (e.g. Wickelgren in math and Gowin in science) they do not appear to be specific enough to enhance problem solving ability.

The results of this study also suggest ways that LOGO problem solving techniques might generalize to these more specific algorithms. By its nature LOGO is highly visual, highly recursive and provides immediate feedback. That is, the manipulation of the TURTLE on the screen provides students with a visual analog of their thinking. In a sense the movement of the TURTLE can provide a visual history of the students plans. This is generally lacking in any other type of problem solving situation.

As reported by Papert (1980), the individual most commonly associates with the development of LOGO, one of the most powerful aspects of the language is its recursive nature --the ability to use a set of commands over and over again simply by making the set of commands a procedure which calls itself. This allows students to see the long term effects of using a set of commands. This would be analogous to any problem solving situation in which an individual viewed the problem as a series of small steps organized into a set which was used repeatedly. For example, factoring an equation would have recursive components as would studying the incremental effects of adding one chemical solution to another. The recursive nature of LOGO then would seem to foster an awareness of the possible utility of breaking a problem into small steps which can be reiterated.

Finally, LOGO would appear to offer immediate feedback as to the correctness of a students hypotheses about the effects of various procedures. Again, this is not the case in most problem solving situations. Quite the contrary, in most situations a student must apply procedures in a linear fashion with no feedback as to the effect of the procedures until the problem is finished. Many times this general feedback (the problem is solved --not solved) offers no information as to which procedure in a sequence was incorrect.

SUMMARY

The results of this study indicated that a general problem solving algorithm does not significantly change the problem solving ability of students as it relates to LOGO problems. This implies, that a weak method approaches to instruction in problem solving has less utility that a strong method approach. However, where there might not be a truly robust general problem solving algorithm, there might be components of problem solving that are aspects of most strong methods. These aspects, might include attention to recursive components and immediate feedback as to the results of hypothesized procedures. For the reinforcement of these aspects LOGO appears to be a highly useful teaching tool.

REFERENCES

Anderson, J. (1983) The architecture of cognition. Cambridge, Mass: Harvard University Press

Doyle, W. (1983) Academic work. Review of Educational Research 53, 159-199.

Heller, J. & Reif, F. (1984) Prescribing effective human problem-solving processes: Problem description in physics. Cognition and Instruction, 1.

Papert, S. (1980) Mindstorms. New York: Basic Books Inc.

Polson, P.G. & Jeffries, R. (1985) Instruction in General Problem-Solving Skills: An Analysis of Four Approaches. In J.W. Segal, S.F. Chipman & R. Glaser (Eds). Thinking & Learning Skills, Vol. 1, Hillsdale N.J.: Erlbaum.