DOCUMENT RESUME

ED 248 856 IR 011 305

AUTHOR Cramer, Stephen E.
TITLE The Instructional Value of Wrong Answers.
PUB DATE 7 Sep 84
NOTE 10p.; Paper presented at the 1984 MUG/USG Conference (Atlanta, GA, September 7, 1984).
PUB TYPE Guides - Non-Classroom Use (055) -- Reports - Descriptive (141) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; *Courseware; *Design Requirements; Error Patterns; *Feedback; *Instructional Development; *Material Development; Teacher Response; Teaching Methods
IDENTIFIERS *Interactive Video

ABSTRACT

In early computer assisted instruction (CAI), negative feedback often insulted students and/or provided no useful knowledge. In classroom settings, teachers use the following approaches in dealing with students' wrong answers: (1) ask the question again, louder and slower; (2) ask the question again, using different words; (3) back up and reteach the past three minutes/hours/days; and (4) keep still, listen to the students and let their behavior reveal where the source of the misunderstanding lies. The latter approach suggests that students answer correctly or incorrectly for a reason and that observation of their behavior can indicate ways to help them correct their misunderstandings. For example, using algorithms to describe student behavior can help minimize student errors in learning a procedure. Designing CAI materials for procedural learning should include two steps: construction of the correct production system or algorithm and construction of potential "buggy" or incorrect procedures, followed by incorporation of instructional error checking into the program. This paper concludes with the description of an interactive video program developed to teach procedures for diagnosing reading problems which included three incorrect algorithms. Responses to students' choice of incorrect algorithms were positively phrased and offered hints. Responses for correct answers explained why the answer was correct. Three references are listed. (LMM)

THE INSTRUCTIONAL VALUE OF WRONG ANSWERS

by

STEPHEN E. CRAMER

Educational Research Laboratory

Department of Educational Psychology

University of Georgia

Presented at the

1984 MUG/USG Conference

Atlanta, Georgia

September 7, 1984

This presentation is going to deal with the concept of negative feedback in computer assisted instruction, in other words, the process by which the program tells the student he has chosen the wrong answer to a question. In the earliest CAI attempts, negative feedback often looked like this:

Computer: WHAT IS THE CAPITAL OF PARAGUAY?

Student: Rio di Janero.

Computer: (BUZZ!!) WRONG, DUMMY!

We soon realized that there was no need to insult our students, and so revised our negative feedback procedures to look like this:

Computer: WHAT IS THE CAPITAL OF PARAGUAY?

Student: Rio di Janero.

Computer: (Beep) SORRY, TRY AGAIN.

The tone of those two messages is quite different, but they share one quality: the amount of information they provide. In both cases, the only thing that student learns is that Rio is not the capital of Paraguay.

Let's digress a bit and talk about how a live classroom teacher deals with wrong answers or students who don't seem to understand. In my long career as an educator and educatee, I have identified four approaches to this problem, which may be familiar to you:

Approach 1: Say it again, louder and slower.

Approach 2: Say it again, using different words. This is a more useful strategy than you might think, since the words you

routinely use to conceptualize an idea may not be the same ones I use.

Approach 3: Back up and reteach the past three minutes/hours/days.

Approach 4: Keep still and listen to the student and let his behavior tell you where the source of the misunderstanding is. This is the approach that we'll be taking today.

The methods I'll be talking about have certain underlying assumptions which I'd like to make clear. First, we need to assume that behavior is not random, that students answer questions right or wrong for a reason (This is not the same as saying that behavior is rational, because it frequently isn't). Second, we assume that we can learn about our students from observing their behavior. Third, that we respect our students as people, who have a right to their own opinions, even if we don't agree with them. Finally, for the purposes of the current discussion, let's assume that what we are teaching a procedural skill of some kind --medical diagnosis, student evaluation, mathematical operations, etc-- and not pure facts.

I'm going to begin by talking about the process of algorithmization of behavior, using the subtraction of two-digit numbers as an example skill. Brown and Burton (1978), two cognitive psychologists, did some of the initial work in this area. They examined mistakes that students consistently made in solving problems, like 763 - 541, which require "borrowing". They developed a process model, which can be presented as a sequential program or a flowchart, and tried out various "bugs" to see what answers they would give.

The next step was to construct an actual program, called BUGGY, which looked at students' performance and classified their mistakes according to the bug found. This information could then be reported to the teacher or the student for remediation purposes. How well did it work? Out of 1325 children studied, BUGGY identified a bug or bugs associated with the consistent errors made by 74% of them. Although the program could not completely explain performance, it certainly gave useful information for instruction.

A second approach to defining the algorithmization of subtraction was taken by Young and O'Shea (1981). In their words, "...it is more fruitful to regard the child as faithfully executing a faulty algorithm than as wrongly following a correct one" (p. 154). Young and O'Shea looked at 344 2-digit subtraction errors of 10 year old children, and categorized 52% of them into 7 or 8 algorithm (37% were number fact errors).

They then developed a production system --a correct algorithm-- for subtraction. A production system is a set of rules of the form C ===> A, where C is the set of conditions which must be present in the problem space for action A to take place (or, as they say, to "fire"). They found that deleting steps from their production system could account for 84% of all the algorithm errors. It wasn't necessary to develop wrong steps at all.

This algorithmization theory provides an explanation for the case of a learner who is presented with a rule, say, borrowing in two-digit subtraction, who practices it in "consolidation" activities, but fails to apply it correctly in

new contexts. What the consolidation activities may have done is to prevent the inclusion of a crucial step in the alogrithm: "Is borrowing necessary in the problem?"

Designing CAI Materials

Now we can begin to talk about designing computer-assisted instructional materials. Remember that we are discussing procedural learning: how to do something, not merely learning facts.

The first step has to be the construction of the (a?) correct production system or algorithm by the instructor. This is often a revealing process, since the instructor almost certainly has enough expertise that the process is automatic. In fact, he probably uses a different algorithm from the one that the student is expected to learn.

The next step is to construct potential buggy procedures. There are two ways to do this, one procedure-driven and the other data-driven.

In the procedure driven approach, we begin by constructing buggy procedures. This is most easily and directly done by taking the correct algorithm we developed and either omitting steps, as in Young and O'Shea's model, or making some of the steps incorrect, ala Brown and Burton.

We would then run the buggy procedure and note the answer which is obtained. This is basically an introspective process, and requires some practice to do well. Somewhat in the same vein as "If I were a set of car keys, where would I hide?" we need to ask ourselves, for example, "If I were assessing reading ability and didn't know about consonant clusters/word callers/hi

- lo reading materials, how would I answer this question?" The answers thus obtained are the ones we will be expecting from the students, and the ones which we will deal with in the program.

A data-driven approach to buggy algorithm construction works like the procedure-driven approach, but in reverse. We make a list of the answers to this or a similar question obtained from past students, either on tests or in class discussions. We then work backwards from there to construct the buggy algorithm they must have been using. Obviously, this approach might not be best for a brand new instructor, or a brand new subject.

This process of analysing student mistakes and procedures may be new to some instructors, especially if they have been working on the assumption that wrong answers are due to faulty performance, not faulty procedures. This method is, however, a basic one for skilled instructors who interact well with their students, and interaction is the name of the game in CAI.

It might also be noted that the two techniques outlined above are the basic procedures for creating multiple-choice foils for any examination.

Now we are ready to actually build the instructional error checking into the program. Of course, it goes without saying that a CAI program will always include basic error trapping routines --to accept only alpha or only numeric characters, for example-- but perhaps it needs to be noted that we of course always tell the learner why his response was unaccteptable, rather than leaving it to him to figure out how he is to answer. And of course, it costs no more to be polite; "I SAID A NUMBER,

TURKEY!" is not an appropriate error message.

Now let's consider the case of a program designed to teach the procedure of diagnosis of reading problems, since that's the most recent thing I've written. This uses an interactive video format which shows scenes of a reading specialist giving an informal reading inventory to a third grader. She asks the child to read lists of words, one for each grade, and stops her after the third grade list like the Our question to the student is to explain why she has stopped at this point.

In prior analysis, we devised three buggy algorithms, which have been validated by our instructional experience; i.e., students really do make these mistakes. The buggy algorithms are:

Buggy algorithm 1: Stops testing on words in isolation after child reads list at his/her current grade level.

Buggy algorithm 2: Stops testing when the latency of the child's response increases to a certain point.

Buggy algorithm 3: Stops testing when the child appears to be having difficulty.

After some appropriate amount of instruction, we might present a question like the following:

Why did the examiner stop testing April on words in isolation?
(a) April is in the third grade.
(b) April mispronounced more than 5 words in the third grade list.
(c) April was reluctant to proceed.
(d) More than 10 seconds elapsed between April's responses.

A student answering (a) would then receive a message like this:

April is in the third grade; however, grade level has no bearing on when to stop word reconition testing.

A student answering (c) would receive the message:

Although April slowed down considerably, you should continue. Information gained about word recognition strategies at or near frustration level is important for later remedial decisions.

A student answering (d) gets this message:

The directions for this particular IRI do not specify a time limit. Some inventories do, though.

Note two things about these responses. First, they are phrased positively, and avoid using the word "WRONG." They instruct the student, and justify the preferred response. They also comment on the aspect of the situation that the student has focused on, and support his observation, if not his conclusion.

Second, some people might consider them "hints." Are they? Absolutely. A hint, after all, is a merely a piece of information that fills in a gap in a cognitive structure or supplies the missing or incorrect step in an algorithm. But, it supplies it when the learner needs it, and we all know the positive effects of need on motivation.

To continue, we need a response for the correct answer, also:

You should stop on the list where the child mispronounces 5 or more words. Good answer.

Why does "Good answer" appear at the end? Because we want the learner to read these messages. It's not enough for the learner to answer the question right, we need for him to know why that's the right answer.

There are various possibilities for expansion of this technique. We might, for example, wish to acknowledge what the student does know with a message beginning:

Your answer would be correct if April....

Or, we might want to press home the importance of the material to be learned with a message which details the consequences of the previous choice, for example:

Your suggestion that April be diagnosed as learning disabled may have the effect of labelling her for years to come. Please reconsider such an extreme step and answer the question again.

To conclude, it is important for us as developers of interactive learning programs to learn about the buggy algorithms our students are likely to use, and to build error traps for these algorithms into our programs. Certainly, it is impossible to forsee all of them, but a good teacher grabs for any opportunity to transmit information to his students, even wrong answers.

REFERENCES

Alverman, D., and Cramer, S.E. (1984). Diagnosing reading problems with the informal reading inventory [Interactive videotape computer facilitated instruction package]. Athens, GA: Instructional Resources Center, University of Georgia.

Brown, J.S., and Burton, R.R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science, 2, 155-192.

Young, R.M., and O'Shea, T. (1981). Errors in children's subtraction.