Some thoughts on Programming

By

RICHARD GOODMAN

THERE are, I gather, a number of terms which we, with our mystique, have been bandying about which are not altogether clear to some of you. The first thing I would like to do is to make quite clear what is meant by a *linear* programme and what is meant by a *branching* programme. I shall of necessity have to be rather dogmatic about this. I would remind you that both Professor Stolurow and I have emphasised that the tendencies are at the present moment not to be partisan about this division between types of programme, but to use whatever type of programme is best suited to the purpose.

The basic ideas underlying a linear programme are as follows. I am now going to be, I hope, a fairly orthodox Skinnerian for a change. Suppose we want to elicit a response R, which may be quite a complicated one, given a stumulus S. Suppose, for instance, we want a student when presented with a quadratic equation (S), to solve that equation (R). If you recall the film yesterday, Skinner waited until his pigeon had begun to turn before he pressed the button and rewarded the bird with a succulent seed; so we must start with the initial behaviour repertoire of the student. Let's assume that we start by eliciting a response R1 by means of a stimulus S1; this corresponds roughly to the first frame of our programme. It will probably be necessary to embed within the frame certain cueing or prompting material so that the probability of a wrong response is radically restricted. The student responds correctly (R1). We now build up another frame, using this response and additional stimulating information (S2) and elicit a second response (S2). In this way we get a series of stimulus-response units, S1–R1, S2–R2 . . . until towards the end we are able to *fade* our cues or prompts allocated, and elicit by an unprompted stimulus the desired criterion response R that we have been working for. The idea behind linear programming is that you cope with individual differences by making the learning steps sufficiently small so that on the average only a very small percentage of students will make an incorrect response at any particular time. (The figure suggested is about 5 per cent.) Furthermore the whole sequence must be so organically organised that, even if you should make an incorrect response to any one item, not only is your response immediately corrected but you will be required to make the response again, as it were further up the line.

When we apply this method to verbal learning what do we get? May I show you the first few frames of a *Temac* programme teaching the symbolic notation used in matrices? The first frame begins "When we see a lot of houses side by side we say that they are in a row". We start off, you see, with something with which we assume everybody is familiar and suppose that when you see a lot of houses side by side you will in fact call them a row of houses. Then the frame continues "A lot of numbers placed side by side might also be said to form a . . . ." Notice that "a lot of houses" is paralleled by "a lot of numbers" placed side by side. This is known as formal or structural cueing. Here is the cue and you are required to fill in "row". You are then presented with the numbers 8, 7, 9, 6, 5, 4, 3. Previously we had "a lot of numbers placed side by side", but now we have some specific integers placed side by side and are required to recognise that they form a *row*. But we have also to ensure that we apply the term "row" (the row of a matrix) also to a particular set of numbers, which may not be integers, but are real numbers, decimal numbers, if you like. So we come on to the numbers 1·30, ·56, ·79 etc. This too is a *row*, so we make sure that the word "row" will not be associated only with a row of integers. We now continue the house-building analogy—"Bricks laid one on top of the other form a column. Numbers can be written one below the other also. If they are so written they are said to form a . . ." The required response is also "column"; you have a *column* of numbers. Again we move forward: "The numbers which form a row or column are called the *elements*. The following row has four elements and the following column has four . . ." We proceed in this way to build up what is in fact a vocabulary, a mathematical vocabulary in this case.

Although you may start with such simple frames, you will find that as you proceed, your frames, your individual information items become more complicated. For example, in the particular programme at which we are looking, the 286th frame and the two succeeding frames are certainly rather more complicated. I don't want you to read these in detail but merely to ask you to notice that quite a number of responses are required in each of these particular frames. The frames tend to become longer as the learning process progresses. Thus we finally come to frames which present quite complicated information, and the learner is required to complete a simplex tableau. Notice the number of responses here, four in this frame, and, in this one, some dozen or more. Here you have your frame designed to elicit a very complicated response. This illustrates the way in which, in a linear programme, you gradually build up from a very simple to quite complicated responses.

This kind of linear programme structure is quite effective when we are trying to teach association, to teach the syntax and semantics of a language, using "language" in the widest sense; but supposing you have got a problem-solving situation—a situation in which the problem is a real problem, and not, as often in text-books, a faked one, a problem that really does challenge the learner—then we must expect mistakes to be made. Look at it in a slightly different way: you may, with Skinner, deal with individual differences in certain fields by breaking down your material into such small steps that the individual differences between various learners are not apparently significant; but if, on the other hand, you are dealing with material in which individual differences and errors become unavoidable, important, and even significant, then you have to adopt a different kind of attitude and a different technique of programming. Here is a sequence of branching frames. They, too, should be overlapping, but for the sake of simplicity in drawing the diagram, I am not going to make them do so. But we stress that one should lead organically into its successor. This represents a sequence of information steps, or whatever you like to call them that you anticipate the "average" member of your group will follow with success. These are the "correct", main sequence frames.

Now in each frame you give a piece of information and at the end of that frame you test whether this information has been successfully communicated or not by posing a diagnostic question. When you have worked out and written down your answer, you are given a list of "plausible answers". One of these will be the correct answer, and one of them will be a "catch-all" answer; if, having worked out your answer to the problem, you select the correct answer, as matching the answer you have obtained, you will be directed to the next frame in the main sequence. There you will be told that your answer is correct and, to eliminate the possibility, or to reduce the effects of the possibility, that you have selected this right answer for the wrong reason, it is normal immediately to surmise the reasons why the answer chosen is correct. Then you are presented with a new quantum of information and are tested again. Suppose, however, your answer matches an "answer" which is the one arrived at as a result of a careless manipulative step (you may have mistaken a *minus* for a *plus*) you are taken to another frame where you are told that unfortunately you are incorrect, given just sufficient extra information to put you on the right track and then you are sent back to tackle the problem again. This is the function of the "Return" button in the *Autotutor* or *Grundytutor;* when you press it, it counts one error.

However, it may so happen that one of the listed "answers" may result from misunderstanding an idea or failure to acquire the skill taught in previous frames. In this case, the error is rather more serious and the student may be switched to a revisionary sequence of either a linear or branching structure, until he has shown that he is now not likely to make such a mistake again. He may then be taken back to the original frame to try again. On the other hand, it may be that even at this stage he still shows that he hasn't mastered the sequence and so may very well be taken back to some previous main sequence frame and be restarted there. If we remember that each test is diagnostic it may be that the student gives an answer very much more intelligent than the official "correct" answer; then you may whip him forward to some later frame in the main sequence ("accelerative branching"). Or alternatively, the answer may be such as to indicate that the particular student is worth while being given enrichment information that you would not normally give to the "average" successful student ("enrichment branching").

In many of the existing branching programmes (the primitive "intrinsic" programmes of the Crowder school) the method of presenting information in any frame is open to criticism, because it is served up in something like a miniature "lecturette" or conventional text-book section, and is thus perhaps as inefficient as most lectures and most text-book sections. There is no reason why, in such circumstances, one should not break this "lecturette" down into a sequence of linear frames each requiring construction response before requiring the student to answer a diagnostic-type question.

There is another point I would like to make in this connection—the importance of formulating the diagnostic question very carefully. If you use the primitive multiple-choice type of question, you have to formulate it in such a way that guessing the correct answer is made difficult. The primitive conventional Crowderian multiple-choice type of question is adequate if it admits only a "yes-no" answer. But, on the whole it is preferable to ensure that the student has to construct overtly his answer before the set of plausible answers is revealed to him. This leads to a more sophisticated type of questioning which requires him first to construct his answer, and only after he has done so match the answer with one from a "plausible", exhaustive set. This we call the *constructed-choice* method.

These, then are the most elementary kinds of programming possible without using a computer. Which you select as your model will be determined only after careful consideration, after you have effected the strategic and tactical breakdown of your material, after facing up to the question whether the largest population of students for which this programme is intended will exhibit a considerable spread or not, and after having determined whether the particular material to be programmed is syntax-semantic type or problem-solving recognition type. It may be that further analysis of material will be required so that you may find that certain phases are best dealt with by a relatively linear technique and certain phases are best done by a branching technique.