20

A GENERATING SYSTEM FOR CAI TEACHING OF SIMPLE ALGEBRA PROBLEMS

by

Carol Peplinski

Computer Sciences Technical Report #24

May 1968

Introduction

Computer-assisted instruction is a field with many and varied possibilities, several of which are only gradually becoming apparent. One of these is a system which generates a set of questions to be asked of a student, then compares the student's answer with its own and tells him whether he is right or wrong. This sort of system allows the student to do as many problems as necessary to master the subject matter in question and further gives him immediate knowledge of results. Thus a situation where a wrong answer has been reinforced because the student did an entire set of questions without realizing he wasn't doing the work correctly is avoided.

A generation program can be built most easily when the subject is rather basic with obvious logical connections. Uhr[1] describes a program which generates elementary arithmetic problems as well as problems involving simple transformations, e.g., from one language to another. Suppes[2] writes of a system giving elementary school children practice in basic relationships between numbers.

This paper describes a drill and practice system which generates problems at a somewhat different level, first year algebra. It does not attempt to teach the student from the beginning, but rather assumes that he has heard an explanation of how to do the problems in question. It offers its own explanation only when the student has given a wrong answer. If he never answers incorrectly, the student will be given no explanations. The program aims to give its users practice in working algebra problems with explanation when necessary, but does not present an entire algebra course.

Program Description

Overview

The system begins by asking the student for his name, then follows with questions regarding the type of problem the student desires to work. He replies by typing either one of several designations of type or the word OPTION, which results in a description of kinds of problems generated and their respective calls. When the student has made his choice, the computer generates an algebra problem to which it expects an answer. As soon as it has received one, or in the case of quadratic equations, two, it compares the student's answer to that generated along with the equation. If they agree, another problem is generated; if they disagree, the program explains a method of solution and generates another problem. After a predetermined number of problems (five) have been worked, the program compares the number right to the total number generated. If eighty per cent or more of the problems have been solved correctly, the program generates more difficult problems, degree of difficulty being determined by the size of numbers in the equations. This continues until eighty per cent of the most difficult problems of the specified class have been answered correctly. The student is then asked whether he would like to do more problems; if so, he can type the appropriate call and continue.

The system automatically keeps an internal record of each student's success with each kind of problem. The number done correctly is stored under each student's name, so that the information is available for the teacher's scrutiny at any time upon reception by the computer of the proper calling word, RESULTS.

Detailed Description

The system as it now stands generates either linear or quadratic equations. The linear equations can be divided roughly into two groups: those with x-terms on one side of the equals sign and those with x-terms on both sides. When there are x-terms on only one side of the equals sign, there is also an option of having the answers limited to integers. In all types of linear equations, the student determines how many x-terms will appear on each side of the equals sign. If the number chosen is greater than 1, the student will gain practice in combining terms. If the number is 1, the problem will be of form $ax+b = c$ or $ax+b = cx+d$, depending on the option chosen. In any case the program begins by generating problems with only one x-term on each side and gradually adds more as the student solves the easier ones, until finally the specified maximum has been reached.

All of the quadratic equations now being generated by the system are solved by factoring into a form specified by the call. Options include equations to be solved by taking the square root of both sides or by completing the square on the left and then taking the square root, equations which are the difference of two squared terms, and basic factoring problems of form $ax^2 + bx + c = 0$. In each case, the coefficient of the x^2 term may be specified to be 1, or the equation may require combination of terms to be reduced to one of the above forms.

As soon as the program has identified which type of problem it has been asked to generate, it branches to the appropriate section of code. If it is to generate linear equations, it sets the maximum size of coefficients, then calls the function GENERATE, which returns a combination of x-terms and constants obtained from the random number generator together with the sum of the s-terms and constants. Each output of GENERATE may be used as one side of an equation, so it is called twice if the call has specified x-terms on both sides of the equation, once if on only one side. If there are x-terms on only one side, a random number is generated for the other side. The program keeps track of how many x-terms there are to be on each side, beginning with one and gradually adding them until the specified number has been reached.

The returned totals of x-terms and constants are then used to compute the answer to the problem. This is stored in four separate variables, one each for the sign, the integral answer, the numerator, and the denominator of a fraction. After the answer has been computed, the problem appears before the student on an output device. When the student's answer has been read, it too is broken into four variables, each of which is then compared to those calculated by the computer. If all agree, the word CORRECT is printed and the program branches to generate the next problem. If either the sign or the integer portion of the answer disagrees, the program branches to an explanation immediately with the function comparing the answers returning the value of the correct answer in a form suitable for use in the explanation. Disagreement between the fractional portions of the answers requires further tests since the computer often arrives at anxwers which are not in lowest terms.

The explanation given is basically the same no matter what kind of linear equation has been generated. Tests are made, however, so that sections irrelevent to the specific problem are omitted. For example, if there is only one x-term on each side, the instruction to combine terms on each side of the equation is bypassed.

The section of code which generates quadratic equations begins by setting a series of indicators which tell the system which combination of options has been chosen. For example, if the call is DIFFSQSIM, problems which are the difference of two squares with the coefficient of the x^2 term not necessarily 1 and require simplification before they can be solved, an indicator is set to show that the problem is to require simplification. Another, the "square" indicator, says that both answers are to have the same absolute value. Others indicate that the equation is to factor int a sum and a difference and that the coefficient of x^2 need not be 1.

After all indicators have been set the program begins generating the equation itself. It chooses numbers a, b, c, and d at random within the constraints of the indicators so that the equation is of form $(ax+b)(cx+d)= 0$, then multiplies out the left side. In the example DIFFSQSIM, it would choose a at random (absolute value less than the upper limit placed on coefficients). Then, since the "square" indicator shows that the answers are to have the same absolute value, c is set to equal a. Next b is generated, and since another indicator shows that the answers are to have opposite signs, d is set to equal -b, and the numbers are multiplied to give the left side of a quadratic equation, $a^2x^2-b^2$. The variable holding the right side of the equation is set to 0 and both answers are calcualted by a function called for the purpose. They are returned in the same four variable form as the answers to the linear equations.

Then, since an indicator shows that the equations are to require simplification before factoring, the program branches to a section which adds random amounts to each term on both sides of the equation, leaving the answer unchanged.

Any one of the calls would be treated in a manner similar to DIFFSQSIM. SIMPLIFY does not require that both answers have the same absolute value, so b and d would by chosen independently of one another. Also, a would be chosen at random with c equaling 1. Later c would also be chosen at random. SQSIM would be identical to DIFFSQSIM except that the signs on both answers would be the same. COMPLSQSIM would involve the subtraction of the square of a number randomly chosen from the left side of the equation with appropriate adjustments to the stored answers.

When the equation has been placed in the form necessary for output, is is compared to the last equation presented to the student and replaced by another equation generated in its place if it is the same. This process is necessary mainly in the case of simple equations where both roots have the same absolute value since they require only one random number and the random number generator may easily produce the same number twice in succession. After the equation has passed this test it is presented to the student.

The process of checking to see whether the student's answers are correct is identical with the one used for linear equations. If a wrong answer is given, however, there are several possible explanations, depending on which kind of problem is being worked. After like terms have been combined the program branches to explain what kind of problem is in question and how it can be recognized. The equation is factored, answers are given, and another problem of the same type is generated.

Getting the machine to generate the equations themselves is quite simple involving only generating numbers and placing them in front of variables. Putting them in the usual form with only one sign before each number (i.e., eliminating the situation $x+-3$) and eliminating extraneous zeros (i.e., x^2+0x+4) require further routines.

Results

The system as so far described seems to handle the problems inherent in generating correct answers for its problems and then distinguishing right and wrong answers given it quite well. In simple problems where the answer is an integer recognizing the correct solution presents no great difficulty. However when the answer is a fraction the situation may become more complicated. If the answer generated by the machine is not in lowest terms several tests beyond simple comparison are necessary. The two answers required in a quadratic equation also complicate the situation since the program must be prepared to recognize them in either order and must not be confused if one answer is correct and the other is not. Figure 1 gives a partial listing of kinds of problems available as they are described to the student. Figure 2 presents interactions with simulated students, while Figure 3 gives an example of the form in which results are presented to the teacher.

Possible Improvements and Extensions

Several improvements to the system as now written are apparent. Something should be done to make sure that no equation reaches the student with all of its coefficients divisable by the same number. For example, the equation generated by FACTOR, $5x^2-5x-30=0$, should be reduced to $x^2-x-6=0$. It may be desirable to have a few equations of this sort left unchanged, but in that case the explanation should begin by instructing the student to divide both sides by 5. At present, all explanations factor the equations into the form in which they were generated. Thus the example would be factored to $(5x+10)(x-3)$ rather than $(x+2)(x-3)$.

The need for another similar improvement becomes apparent in many cases when the correct answers are fractions. While the program recognizes fractions in lowest terms as correct, it does not reduce its own answers. Some routine should be added to make sure that all fractions handled have been reduced, thus forcing the student to reduce his answers.

The explanations printed in case of a wrong answer vary in understandability. They are usually quite clear when the problem in question is simple, taking things step by step and finally arriving at an answer for the linear equations. The explanations of simple factoring problems do little beyond giving the answer, but since factoring is largely a matter of trial and error it is hard to see what else might be done. Problems which are the difference of two squares can be solved systematically and thus can be explained without too much trouble, but problems which factor into $(ax+b)^2=0$ are rather inadequately handled. The explanation as it now stands says several things which, while true and possibly useful, will probably only confuse a student who doesn't know how to solve th problem. This explanation is probably the one aspect of the program which most needs improvement. The explanation of problems to be solved by completing the square also has a rather confusing sentence, the second, but this could be dealt with by taking the process being explained in more steps.

The system at present assumes that a student will be able to work up to a level of doing eighty per cent of his problems correctly in one sitting. It might be a good idea to add a provision for the student to stop when he is tired and then restart at a later time, the level of difficulty, type, and number of problems yet to be done being stored by the computer. Something similar to the system described by Suppes, in which a student who is successful in less than sixty per cent of the problems he attempts moves down a level, might also be incorporated. This would call for an ordering of levels of difficulty among the different types of quadratic equations beyond what is now operational.

The descriptions of problems stored for the teacher's reference need improvement, since all levels of difficulty are stored under the same title. This is especially true of the linear equations.

Several features not included in the running system come to mind readily and probably should be added. A good explanation of how to do each kind of problem might appear as soon as the type has been stated. Another obvious addition is problems to be solved by use of the quadratic formula. This would require an entirely new generating routine, since all equations generated by existing routines are factorable. Some conbination of types of quadratic equations would also be valuable, since a student must be able to decide which method of solution to use when he meets a quadratic equation in a practical situation. Some method of random choice of type could probably be used to accomplish this, although some rewriting of the present quadratic generator might prove necessary to make a gradual increase in difficulty of problems possible.

Discussion

The program described in this paper has its main similarities to those described by Suppes and Uhr in the basic idea of generating questions and their answers rather than giving each student the same problems. Since the subjects are different the questions themselves have little in common. Both Suppes and Uhr offer the student a second chance when he has once given a wrong answer. The described system does not, on the theory that a failure with an algebra problem is likely to be due either to an arithmetic error in combining terms or a lack of understanding of the method to be used. In the first case the error is not fundamental, should not often recur, and will probably take the student longer to find by checking back in his calculations than it merits. In the latter case any second answer will be nothing but guessing. The present system also makes no effort to store problems the student has been unable to work, as does Uhr's.

Summary

The system described above presents algebra problems of various types, individual problems within each type becoming more difficult as more and more are done successfully. It reads the student's answers and tells him if they are correct, then generates another problem or explains the problem answered incorrectly, whichever is appropriate. It keeps a record of each student's successes and failures available for the teacher's reference upon demand and allows the student to progress to the next level of difficulty when he has done eighty per cent of the problems presented on his level correctly.

Suggested extensions include a wider range of problem types and an increased and improved system of explanations. Increased possibilities of switching students from one level to another are also suggested.

References

1. Uhr, Leonard <u>The Automatic Generation of Teaching Machine Programs</u> (1965).

2. Suppes, Patrick "The Uses of Computers in Education," <u>Scientific American</u>, 215 (September 1966) pp. 207-220.

Figure 1. Part of the explanation of problem types.

```
HELLO. LET≠S DO SOME ALGEBRA PROBLEMS. FIRST TYPE YOUR NAME ON THE TYPEWRITER.
STUDENT
NOW TYPE THE NAME OF THE KIND OF PROBLEM YOU WISH TO DO. IF YOU WOULD LIKE AN EXPLANATION OF THE
POSSIBLE TYPES. TYPE
THE WORD OPTION.
OPTION

TYPE OF PROBLEM                                                              PLEASE TYPE

QUADRATIC EQUATIONS OF FORM
     A(1)X**2 + A(2)X + A(3) = 0 WHICH FACTOR TO
            (X-B)(X-D)                                                        FACTOR1
            (AX-B)(CX-D)                                                      FACTOR
            (X-B)(X-B)                                                        SQ1
            (AX-B)(AX-B)                                                      SQ
            (X-B)(X+B)                                                        DIFFSQ1
            (AX-B)(AX+B)                                                      DIFFSQ

     A(1)X**2 + A(2)X + A(3) = B(1)X**2 + B(2)X + B(3) WHICH FACTOR TO
            (X-B)(X-D)                                                        SIMPLIFY1
            (AX-B)(CX-D)                                                      SIMPLIFY
            (X-B)(X-B)                                                        SQSIM1
            (AX-B)(AX-B)                                                      SQSIM
            (X-B)(X+B)                                                        DIFFSQSIM1
            (AX-B)(AX+B)                                                      DIFFSQSIM

QUADRATIC EQUATIONS TO BE SOLVED BY COMPLETING THE SQUARE OF FORM

     A(1)X**2 + A(2) + A(3) = 0 WHICH FACTOR TO
            (X-B)(X-B) = E**2                                                 COMPLSQ1
            (AX-B)(AX-B) = E**2                                               COMPLSQ
     A(1)X**2 + A(2)X + A(3) = B(1)X**2 + B(2)X + B(3) WHICH FACTOR TO
            (X-B)(X-B) = E**2                                                 COMPLSQSIM1
            (AX-B)(AX-B) = E**2                                               COMPLSQSIM

NOW TYPE THE CALL FOR THE KIND OF PROBLEM YOU WOULD LIKE TO SOLVE.
```

Figure 2. Problems and explanations generated by the program.
Linear equations.

```
-4X -1 +5X +4   = -3
YOUR ANSWER IS -6/1
GENERATED ANSWER IS -6 0/1

YOU HAVE MADE AN ERROR. THE PROBLEM IS
-4X -1 +5X +4   = -3
FIRST COLLECT ALL TERMS CONTAINING ≠X≠ AND ALL CONSTANTS ON THE LEFT. THIS IS
1X +3 = -3
SUBTRACT 3 FROM BOTH SIDES, OBTAINING
1X = -6

+4X   = -4X -4
YOUR ANSWER IS -1/2
GENERATED ANSWER IS -0 4/8
CORRECT.


-3X +6   = +8X
YOUR ANSWER IS 11/12
GENERATED ANSWER IS 0 6/11

YOU HAVE MADE AN ERROR. THE PROBLEM IS
-3X +6   = +8X
SUBTRACT 8X FROM BOTH SIDES. THIS PRODUCES
-11X +6 = 0
SUBTRACT 6 FROM BOTH SIDES, OBTAINING
-11X = -6
NOW DIVIDE BOTH SIDES BY -11. AND X = 6/11
```

```
+8X +2 +5X -5 -5  = +2X +7 +3X +4 +6X -8
YOUR ANSWER IS 5 2/4
GENERATED ANSWER IS 5 1/2

YOU HAVE MADE AN ERROR. THE PROBLEM IS
+8X +2 +5X -5 -5  = +2X +7 +3X +4 +6X -8
FIRST COLLECT ALL TERMS CONTAINING ≠X≠ AND ALL CONSTANTS ON THE LEFT. THIS IS
13X -8 = +2X +7 +3X +4 +6X -8
THEN DO THE SAME ON THE RIGHT. THIS RESULTS IN
13X -8 = 11X +3
SUBTRACT 11X FROM BOTH SIDES. THIS PRODUCES
2X -8 = 3
SUBTRACT -8 FROM BOTH SIDES, OBTAINING
2X = 11
NOW DIVIDE BOTH SIDES BY 2. AND X = 5 1/2
```

Quadratic equations.

```
4X**2 +9X -9 = 0
YOUR ANSWERS ARE X = -3, X = 3/4
GENERATED ANSWERS ARE  X = 0 3/4, X = -3 0/1
CORRECT.

1X**2 +9X +20 = 0
YOUR ANSWERS ARE X = 4, X = 5
GENERATED ANSWERS ARE  X = -4 0/1, X = -5 0/1

YOU HAVE MADE AN ERROR. THE PROBLEM IS
1X**2 +9X +20 = 0
FACTOR THE ABOVE EXPRESSION. OBTAINING
(1X +4)(1X +5) = 0
THEN 1X = -4 AND 1X = -5
```

```
1X**2 -14X +49 = 0
YOUR ANSWERS ARE X = 7, X = 7
GENERATED ANSWERS ARE  X = 7 0/1, X = 7 0/1
CORRECT.

+5X**2 +41X +78 = 2X**2 +7X -2
YOUR ANSWERS ARE X = -3 1/3, X = -3 1/3
GENERATED ANSWERS ARE  X = -3 1/3, X = -8 0/1

YOU HAVE MADE AN ERROR, THE PROBLEM IS
+5X**2 +41X +78 = 2X**2 +7X -2
FIRST COLLECT TERMS BY SUBTRACTING 2X**2, 7X, AND -2 FROM BOTH SIDES, THIS RESULTS IN
3X**2 +34X +80 = 0
FACTOR THE ABOVE EXPRESSION, OBTAINING
(3X +10)(1X +8) = 0
THEN 3X = -10 AND 1X = -8
DIVIDING TO SOLVE, X = -3 1/3 OR X = -8


25X**2 +40X +16 = 0
YOUR ANSWERS ARE X = 4/5, X = 4/5
GENERATED ANSWERS ARE  X = -0 4/5, X = -0 4/5

YOU HAVE MADE AN ERROR, THE PROBLEM IS
25X**2 +40X +16 = 0
NOTICE THAT THE CONSTANT TERM IS A PERFECT SQUARE, AND THAT THE COEFFICIENT OF
≠X≠ IS TWICE THE PRODUCT OF THE SQUARE ROOTS OF THE CONSTANT TERM, 4, AND THE COEFFICIENT OF
≠X**2≠, 5, THEN THE LEFT SIDE FACTORS INTO
(5X +4)(5X +4)
THEN 5X = -4
DIVIDING TO SOLVE, X = -4/5
```

```
4X**2 -3X -10 = 0
YOUR ANSWERS ARE X = 2, X = -1 1/4
GENERATED ANSWERS ARE  X = -1 1/4, X = 2 0/1
CORRECT.


4X**2 -25 = 0
YOUR ANSWERS ARE X = 2 1/3, X = -2 1/3
GENERATED ANSWERS ARE  X = -2 1/2, X = 2 1/2

YOU HAVE MADE AN ERROR. THE PROBLEM IS
4X**2 -25 = 0
THE EXAMPLE IS A DIFFERENCE OF TWO SQUARES. AND
THUS FACTORS INTO THE SUM AND DIFFERENCE OF THE
SQUARE ROOTS, AS FOLLOWS
(2X +5)(2X -5) = 0
THEN 2X = -5 AND 2X = 5
DIVIDING TO SOLVE, X = -2 1/2 OR X = 2 1/2


1X**2 -2X -15 = 0
YOUR ANSWERS ARE X = 4, X = 5
GENERATED ANSWERS ARE  X = 5 0/1, X = -3 0/1

YOU HAVE MADE AN ERROR. THE PROBLEM IS
1X**2 -2X -15 = 0
THIS PROBLEM MAY BE SOLVED BY COMPLETING THE SQUARE OF THE LEFT SIDE. BEGIN BY SUBTRACTING
-15 FROM BOTH SIDES.
THIS IS
1X**2 -2X = 15
NOW DIVIDE THE COEFFICIENT OF ≠X≠ BY TWICE THE SQUARE ROOT OF THE COEFFICIENT OF ≠X**2≠ AND
SQUARE THE RESULT.
THIS PRODUCES 1.
NOW ADD 1 TO BOTH SIDES AND TAKE THE SQUARE ROOT OF BOTH SIDES OF THE EQUATION. THEN
(1X -1) = + 4 OR -4
THUS, 1X -1 = 4 OR 1X -1 = -4
THEN X = 5 OR -3
```

Figure 3. Record of results printed out on the teacher's demand. The program prints out the records kept for all students who have used it, not just the one shown here.

```
RESULTS
STUDENT
         FACTOR1     4/5
         FACTOR1     5/5
         FACTOR    4/5
         FACTOR    4/5
         FACTOR    4/5
         FACTOR    4/5
         SIMPLIFY1     4/5
         SIMPLIFY1     4/5
         SIMPLIFY    5/5
         SIMPLIFY    4/5
         SIMPLIFY    4/5
         SIMPLIFY    4/5
         SQ1    4/5
         SQ1    4/5
         SQ    4/5
         SQ    4/5
```