DOCUMENT RESUME

ED 258 847 SE 045 846

AUTHOR Sleeman, D.
TITLE An Attempt to Understand Students' Understanding of Basic Algebra.
INSTITUTION Stanford Univ., Calif. Dept. of Computer Science.
PUB DATE 84
NOTE 36p.; A product of the Heuristic Programming Project. For related documents, see SE 045 844-845.
PUB TYPE Reports - Research/Technical (143)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Algebra; Artificial Intelligence; *Cognitive Processes; Comprehension; *Computer Assisted Instruction; Junior High Schools; Mathematics Education; *Mathematics Instruction; Mathematics Skills; *Models; Problem Solving; *Secondary School Mathematics
IDENTIFIERS *Intelligent CAI Systems; Mathematics Education Research

ABSTRACT

This paper reports the results obtained with a group of 24 14-year-old students when presented with a set of algebra tasks by the Leeds Modelling System (LMS). These same students were given a comparable paper-and-pencil test and detailed interviews some 4 months later. The latter studies uncovered several kinds of student misunderstanding that LMS had not detected. Some students had profound misunderstandings of algebraic notation; others used strategies such as substituting numbers for variables until the equation balanced. Additionally, it appears that the student errors fall into several distinct classes: namely, manipulative, parsing, clerical and "random". LMS and its rule database have been enhanced as the result of this experiment, and so LMS is now able to diagnose the majority of the errors encountered in this experiment. Finally, the paper gives a process-orientated explanation for student errors, and re-examines related work in cognitive modelling in the light of the types of student errors reported in this experiment. Misgeneralization is a mechanism suggested to explain some of the mal-rules noted in this study. (Author)

AN ATTEMPT TO UNDERSTAND STUDENTS' UNDERSTANDING OF BASIC ALGEBRA*

D. SLEEMAN
Heuristic Programming Project**
Computer Science Department
STANFORD University
CALIFORNIA 94305.

* ACKNOWLEDGEMENTS: To Mr. M. McDermot and students of Abbey Grange School, Leeds, for providing fascinating sets of protocols. To Pat Langley, Kurt VanLehn, Stellan Ohlsson, Jaime Carbonell, Jim Greeno, K. Lovell and Alan Bundy for discussions about this work. Kurt VanLehn, Peter Jackson, Stuart MacMillan, Pat Langley, Stellan Ohlsson, William Bricken, and an anonymous referee made valuable comments on an earlier draft of this paper. To the University of Leeds for granting me sabbatical leave. To the Sloan Foundation for providing me with some financial support. And to CMU and Stanford for providing wonderful environments in which to complete this study.

**This work was initiated while the author was at the University of Leeds, U.K.

AN ATTEMPT TO UNDERSTAND STUDENTS' UNDERSTANDING OF BASIC ALGEBRA

ABSTRACT

This paper reports the results obtained with a group of 24 14-year-old students when presented with a set of algebra tasks by the Leeds Modelling System, LMS. These same students were given a comparable paper-and-pencil test and detailed interviews some 4 months later. The latter studies uncovered several kinds of student misunderstandings that LMS had not detected. Some students had profound misunderstandings of algebraic notation; others used strategies such as substituting numbers for variables until the equation balanced. Additionally, it appears that the student errors fall into several distinct classes: namely, manipulative, parsing, clerical and "random".

LMS and its rule database have been enhanced as the result of this experiment, and so LMS is now able to diagnose the majority of the errors encountered in this experiment.

Finally, the paper gives a process-orientated explanation for student errors, and re-examines related work in cognitive modelling in the light of the types of student errors reported in this experiment. Misgeneralization is a mechanism suggested to explain some of the mal-rules noted in this study.

1. INTRODUCTION

The impetus for work in Intelligent CAI has two major sources; firstly, the practical aim of producing teaching systems which are truly adaptive to the needs of the student and secondly the "theoretical" interest involved in formulating these activities as algorithms. It has been argued by Hartley and Sleeman 1973 that an intelligent teaching system requires access to: knowledge of the task domain; a model of the student's behaviour; a list of possible teaching operations; and means-ends guidance rules which relate teaching decisions to conditions in the student model.

During the last decade a number of systems have been implemented which include some or all of these databases. In particular during the last 10 years a number of systems have been implemented which attempt to provide supportive learning environments intended to facilitate learning-by-doing. These systems include SOPHIE (Brown, Burton & de Kleer 1982), GUIDON (Clancey 1982), WEST (Burton & Brown 1982), WUMPUS (Goldstein 1982), and PSM-NMR (Sleeman & Hendley 1982); such systems have been called Coaches or Problem Solving Monitors. In this paper, we address a particular aspect of the problem of inferring a model from the student's behaviour on a set of tasks, [1]. We shall outline the results of a recent experiment with 24 14-year-old students who were considered to be of average ability. The issue to be considered in this paper is whether the models inferred by the Leeds Modelling System, LMS, can be given a cognitive interpretation, and whether it is possible to say something about the nature of the processes used by a student given the model inferred by LMS.

1.1 The Leeds Modelling System, LMS.

In common with BUGGY (Brown & Burton, 1978) LMS uses a generative mechanism to create models from a set of primitive components. Without a generative facility, the ability of a system to model complex errorful behaviour is severely limited. However, the use of such a mechanism also causes difficulties, since such an algorithm can readily lead to a combinatorial explosion. For example, if there are N primitive rules in a domain where

---

1. For a more detailed discussion of this, and related issues see the Introductory essay to Intelligent Tutoring Systems, (Sleeman & Brown 1982).

the rule-order is significant, then there are N factorial, N!, models to be considered. BUGGY and LMS are similar in that BUGGY uses a collection of primitive bugs from which to generate models, while LMS uses mal-rules, incorrect rules, observed in the analysis of earlier protocols. On the other hand, whereas BUGGY uses heuristics to limit the size of its model space, a major feature of the LMS work has been the formulation of the search to focus each task-set on particular rule(s). [2]. As has been demonstrated (Sleeman & Smith 1981, and more particularly by Sleeman 1983a) this technique drastically reduces the number of models to be considered at each stage. [3]. Before considering the results of this experiment, we briefly review the production system representation which has been used for student models and explain the main features of the production system interpreter used to execute these models.

Figure 1a gives a set of Production Rules, used with LMS, which are sufficient to solve linear algebraic equations of one variable. Figure 1b gives a set of mal-rules for this domain which have been observed in protocols analysed earlier. A task-set is a set of 5-7 tasks which highlights the use of one or more domain-rules; Figure 2 gives a typical task for each of this domain's task-sets and the rules which each set focusses on. Further, Figure 2 shows the exact format of tasks presented by LMS; this format has also been used in all subsequent interactions with the students.

[Figures 1 and 2 about here]

In this work, a model is an ordered list of rules. Order is significant, as the interpreter executes the action of the first rule in the model whose conditions are satisfied by the state (i.e., the task or the partially solved task). In this way we are able to capture precedence which is important in this subject domain. The match-execute cycle continues until no further rules fire. Figure 1c shows pairs of correct and "buggy" models executing typical tasks. LMS infers a model for each task which the student works, producing summary model(s) for each task-set. If the student's behaviour is random or conforms to a previously unencountered mal-rule then LMS returns a null model (see Sleeman 1982 for more details). LMS presents tasks to a student until its example bank is

---

2. Examples of task-sets are given in Figure 2.

---

3. Initially, we made the assumption that the domain was hierarchical and so we have referred to the stages as levels: thus modelling proceeds by first considering level 1, then 2, etc.

exhausted or until the student opts to "retire".

The 1980 Experiment

In 1980 an experiment was run with a group of 15-year-old students and a very close agreement was achieved between LMS's diagnosis and those made by a group of investigators who gave the students individual interviews on analogous tasks (Sleeman 1982). In one important respect LMS and the investigators differed. The design of LMS was such that if the student did not make an error with say XTOLHS when it was introduced, then LMS assumed XTOLHS would be used successfully at all subsequent levels. This experiment showed that this was not a valid assumption. For example, some students who were able to correctly work tasks of the form:

$$3 * X = 4 * X + 9$$

had trouble on the following type of task:

$$12 * X = 2 * (4 * X + 5)$$

where they appeared to forget to change the sign of the X-term when the side is changed, and thus we have seen 20 * X = 10 returned as an answer.

It was, in fact, easy to remove this assumption from LMS's code, but unfortunately the modification led to an explosion in the number of models to be consi... ed, and so a reformulation of the algorithm was carried out (Sleeman 1983a).

As a result of this experiment, we believed that students' behaviour on algebra could be largely explained in terms of manipulative mal-rules, namely mal-rules in which one of the substeps is omitted.

2. AN OVERVIEW of EARLIER RELEVANT WORK in COGNITIVE MODELLING

BUGGY (Brown & Burton 1978) analysed the responses which students gave to multi-column subtraction tasks. The system reported a diagnosis for each student in terms of correct procedures, or procedures which had some of their substeps replaced by incorrect variants, called bugs.

Young & O'Shea 1981 point out that although BUGGY produces models that behave functionally as the students, those models are not very convincing as psychological models. Many of the bugs appear to be very similar (many are connected with borrowing from zero) yet this relationship is not made clear. More particularly, Young & O'Shea show that some of the BUGGY data can be analysed more simply in terms of certain competences being omitted from the ideal model.

Repair theory (Brown & VanLehn 1980) is a further attempt to provide a psychological explanation for the same data. Here Brown & VanLehn take a correct procedure for performing subtraction and apply a deletion operator to the procedure. This perturbed procedure is then used to solve tasks. When it encounters an impasse, such as a situation where it is about to violate a precondition (e.g., attempting to take a number from 0), a repair is applied to the perturbed procedure, and it attempts to continue solving the task. This process also uses critics to throw out some repairs which are considered impossible at a given impasse.

More recently VanLehn 1983a has suggested a variant of repair theory, which does not delete steps from procedures - as it is argued that the blocking, or inhibition, of the deletion operator was unprincipled. Secondly this version overcomes the difficulty that certain core procedures cannot be generated easily by rule deletion. Instead, VanLehn has suggested a series of core procedures, which correspond to the various stages of instruction (c.f., Sleeman & Smith 1981). From this perspective an impasse occurs when the student encounters a sub-task which he has not learnt, or has forgotten.

Both variants of repair theory explain what Brown and his coworkers have called bug migration, namely that with the same type of task, the student may display different bugs both during the same test-period and between different tests. Moreover, VanLehn 1981 has analysed protocols in which it was possible to generate all the bugs in an observed migration class by applying different repairs to a common (partially learnt) core procedure. So VanLehn suggests consistent bugs can be explained by supposing the student stores the "patch" and merely uses it with the next task. The explanation for bug migration is that the patch is not retained and that one of the repairs is selected randomly.

The Illinois group (Davis, Jockusch & McKnight 1978) ha reported algebra students overgeneralising from instances, using an "old" oper ... ead of a more recently introduced one [4], and regressing under cognitive load. Ma.. 1982 has further analysed those students' performances and has suggested a number of high-level schema which explain series of observed errors. These include her "extrapolation principle" which explains why a student who has seen the legal transformation:

(A * B) ^ C ==> A ^ C * B ^ C

would then write:

(A + B) ^ C ==> A ^ C + B ^ C

She also discusses the confusion which seems to arise between the notations of arithmetic and algebra. For instance, she argues that as 3 3/4 is to be interpreted as 3 + 3/4 it is not unreasonable that the student should interpret the algebraic expression, 3X as 3 + X, [5], [6].

---

4. + instead of *, * instead of Exponential.

---

5. Although this explanation would explain some of our observations, students AB17 and AB18 in the Leeds study gave an alternative and more comprehensive explanation for their actions, see Section 5.1.2a.

---

6. Similarly, our earlier work provided an additional data point for the 1981 experiment. As a result of our 1980 experiment, see Section 1.1, we believed that students' behaviour on algebra could be largely explained in terms of "manipulative" mal-rules (where a manipulative mal-rule is a variant of a correct rule and has one substep replaced by an inappropriate or incorrect step, see Section 5.1.1 for further discussion.)

3. THE 1981 EXPERIMENT WITH LMS

The 1981 experiment was carried out with the revised modeller, LMS-II, [7], but with the same data-base of rules and tasks as used in the 1980 experiment. This group of 24 students, average age ' years 3 months, were judged to be of average ability at mathematics: however the results were dramatically different from the earlier group's, [8]. Indeed many of their difficulties were not diagnosed by LMS-II and had to be analysed by the investigator. This analysis was made very difficult because it had been assumed that students would at most make one or two minor manipulative errors, e.g., changing side and not sign, and so LMS had been designed to allow the student to input his or her final answer, and as many intermediary steps as he chose. In Figure 3 we give sample summaries produced by LMS-II for students' online interaction, together with the mal-rules which the investigator suggested were appropriate for each task-set. In Figure 4, we summarize the complete set of new mal-rules which the investigator considered explained the students' behaviour with LMS-II.

Note that by stating that a protocol can be explained by a mal-rule, say, of the form

$$M * X \Rightarrow M + X$$

(Figure 3a), we do not wish to imply that given a problem of the type

$$3 * X + 4 * X = 5$$

that the student would produce the response:

$$3 + X + 4 + X = 5$$

Indeed, we have seen several students write

$$X + X = 5 - 3 - 4$$

and when asked to provide intermediary steps they have said categorically that there were none as the above was done in "one step". Nevertheless we are happy to accept that both forms are explained by the mal-rule: the first form however requires that several additional rules are executed in order to get it into the sta.e given by the "second"

---

7. Revised to remove the assumption that if an error is not made with a rule at the stage it is first introduced, that the student will use the rule correctly on all subsequent occasions, see Section 1.1.

---

8. Most of these students had been introduced to algebra several years earlier in their middle schools; further, the high school had retaught algebra - virtually from the beginning - in the year before the experiment took place.

student. (It should be noted that the mal-rules given in Figures 3d and 3e are more comprehensive and carry out several housekeeping steps. The differences between basic and comprehensive mal-rules are significant when one tries to perform remedial instruction, as it is important to ensure that the grain of the instruction matches the student's.)

[Figures 3 and 4 about here]

As a result of analysing these summaries, a number of questions were raised:

-What is the crucial difference between the task-sets which a particular student is and is not able to correctly solve?

- Does the student's perception of algebraic tasks vary from one task-type to another?

Unfortunately, as the school vacation intervened, it was not possible to meet with the students again until September (1981). Because of the time that had elapsed, the students were given a paper-and-pencil test which covered comparable tasks to those used by LMS. These tests were analysed in detail by the investigator, and as a result of this certain students were given detailed diagnostic interviews. The next sections give more details of these stages.

3.1 The September Paper-and-Pencil Test

From a comparative review of the May and September data, see Sleeman 1983c for the details, we concluded:

1. The performance was generally considerably better in September than in May. (Note no additional teaching in Algebra had been given, however the students had presumably done some self-study in preparation for their June examinations.)

2. A considerable number of tasks were not solved on the written test (whereas LMS insisted the student gave a response to each question).

3. Some students who appeared to have "wild" rules on particular tasks in May, seemed to solve this type of task correctly in September, e.g., A85.

4. Some students whose behaviour had been "random" or "wild" in May had now settled to use mal-rules consistently, e.g., student A818.

5. One student, A87, gave multiple values in an equation where X occurred more than once.

6. Many of the students made the common precedence error, namely given a task of the form:

$$2 + 3 * X = 11 \quad \text{they return} \quad 5 * X = 11.$$

As a result of this comparison it was decided to interview all those who appeared on the written test still to have major difficulties, but not to interview those who had <u>only</u> common "precedence" errors, or those who had had major difficulties which appeared to have "cleared up".

<u>3.2 The Interviews</u>

The interviews proved to be remarkably revealing as the students without exception were extremely articulate. These dialogues were recorded; Figures 5&6 have been reconstructed from the tapes and the worksheets.

The investigator presented the student with a series of tasks and asked him, or her, to work each one explaining as he went along exactly what he was doing. In some cases the investigator asked the student to tell him which of two alternative forms were correct and frequently asked the student to explain why. The tasks presented were different for each student, and were based on the difficulties noted in the individual's September test. The interviewer thus started each session with a list of task-types to be explored, but often generated particular tasks as a result of answers given to the planned tasks.

The following is a summary of the main features noted during the interviews:

1. Some students "searched" for solutions (i.e., tried different values for X). (Section 3.2.1).

2. One student computed a separate value for each X given in the equation. (Section 3.2.2).

3. One student maintained that there were a number of quite distinct ways of solving an equation; even when it is demonstrated that each approach led to different answers. (Section 3.2.3).

4. Some students have "hard", consistent, mal-rules. (Section 3.2.4).

5. Some students have the correct rules and can explain why it is not permissible to perform the illegal transformation, including the illegal transformations that the student appeared to use in May. (Section 3.2.5).

Each of these points is discussed in the following sub-sections.

3.2.1 Searching for Solutions

Searching for a solution appears to be a very common way of solving equations with students beginning algebra, and presumably arises because the initial equations presented could be solved in this way. When given an equation of the form:

$$3 * X + 2 = 14$$

such students substitute X=1, then X=2, then X=3.... until the equation balances. (See Sleeman 1983c for further details of student AB11's protocol). [9].

Further, student AB11 solved tasks of the form:

$$3 * X = 2$$

as X = -1, explaining she subtracted 3 from both sides.

It is indeed intriguing to watch students changing their approach when solving tasks of this latter form depending on whether the task is solvable by "search". Students do not appear

9. Indeed, in a more recent test with 100 13-year-olds, it appears that about 95% of them use this approach. Intelligent Tutoring Systems and teachers should suspect that a student is using a naive algorithm if he appears to be unable to solve tasks where the variable is a negative integer, large-integer or non-integer. The teacher should be concerned because the naive algorithm is only applicable to a sub-set of algebraic equations, and hence should be deemed a significant weakness, and one to be remedied. It seems clear that the use of simplistic tasks leads to a naive algorithm which causes major conceptual difficulties on more advanced tasks.

to believe that all equations of the same form should be solved in the same way. (Clearly this point should have been discussed in an interview with those students.) Such students are often unable to solve correctly equations which contain multiple Xs; they attempt to <u>guess</u> values for all the Xs in the equation, see the next subsection.

3.2.2 Multiple Values for X

In this subsection, we report a student who has a very strange, but nevertheless very <u>consistent</u> algorithm, for solving tasks involving 2 Xs. When student A87 was originally working at the terminal, she was heard to mutter:

"If this X was 2, then it would work if this second X was 4".

Not only was this student consistent in both the paper-and-pencil exercise and in the interview, she was able to explain what she was doing. Given the task:

3 * X + 2 * X = 12

She gave the following explanation:

"What I do is take the 3 and I make the first ... ual to 2, so I write:

3 * 2 "

When asked by the interviewer why the "first" X is equal to 2, she explains that it's the next number along, and then added "I think this is the wrong thing to do, but that's what I do".

She then continued "... and then I write down the + 2 making

3 * 2 + 2

I then work this out, this is equal to 8 and so the second X is

12 - 8, that is 4".

She then completed the solution and gave the 2 values for X, and so the final state of her worksheet was:

3 * 2 + 2 + 4 = 12

X = 2

X = 4

She used this algorithm consistently on 9 tasks. see Sleeman 1983c, [10].

3.2.3 Alternative Algorithms

Although student AB17 was able to solve several task-types correctly, he was easily "distracted" and quite unable to tell the investigator why the investigator's "alternatives" were illegal. On some tasks the student suggested several illegal solutions, and again was really unable to distinguish between them. (See Figure 5 for details). On the other hand, this student did give as an aside a rationale for his "method", namely "collecting all the Xs to the LHS and all the numbers to the RHS", which will be discussed in Section 5.1.2a.

[Figure 5 about here]

3.2.4 "Hard"/Consistent Mal-Rules

Many of the students used consistent mal-rules. Just over half of the 24 students we saw mis-handled precedence in equations of the form:

$$2 + 3 * X = 9$$

Part of a protocol for one such student is given in Figure 6.I, [11]. Figure 6.II is part of the protocol produced by the student discussed in Section 3.2.3, where he consistently applies

---

10. Initially, we had supposed this to be a very idiosyncratic algorithm, but subsequently noted that a variant was used extensively by 13-year-olds. For example we have seen:

$$3 * X + 4 * X = 3$$

"solved" as:

$$3*1 + 4*0 = 3, \text{ making } X = 1 \text{ and } X = 0.$$

Similarly,

$$3 * X + 4 * X = 98$$

has been "solved" as:

$$3*22 + 4*8 = 66 + 32 = 98.$$

Moreover, in "complicated" cases the two sides often are not "balanced". Thus we have seen

$$3 * X + 4 * X = 100$$

"solved" as:

$$3*30 + 4*2 = 100$$

and when asked the student explained that "this one did not work out exactly".

Note that these students frequently solve tasks of the form:

$$3 * X + 4 = 10$$

by "search".

---

11. Recently we have discovered that 90% of a sample of 13-year-olds had precedence difficulties with arithmetic expressions involving " + " and "*" operators.

a further intriguing transformation to a complete set of tasks. In order to understand this protocol fully we have suggested that a normalization step takes place between stages 1 and 2 of say protocol a). That is, we suggest that when the student applies the mal-rule to the original task, this results in an "unusual" form which the student then "normalizes" before continuing to process the rest of the task. (See Sleeman 1983c for a lengthier discussion of "normalization").

[Figure 6 about here]

Student AB18, Figure 6.III, is remarkably consistent with his mal-rules over a whole range of task-types. Note the application of his algorithm to task c) which involves 3 X-terms. (To give him justice, he realises that he had got tasks d) - g) wrong as he noticed that the equations did not balance when he substituted his answers back in). Further, having worked task h), he noticed that when he moved the 4 across to the right hand side, he changed the sign. So he suggested that when he moved the X (associated with 2 * X) to the LHS, he should also change its sign. He said:

"X - X is 0, and so the LHS became 0 and the RHS did not"

and so realized that this proposed solution was impossible. However, for good measure he also worked task i) with the "revised" algorithm.

In the course of our discussion, this student also gave the basis for his "algorithm" which is discussed in more detail in Section 5.1.2a.

3.2.5 "Saved Souls"

In September student AB5 worked correctly tasks which she had got consistently wrong in May, namely task-sets 7 and 8. For task-set 8 she appeared to use mal-rule:

M * X = N * X + P ==> X + X = M + N + P.

Moreover, when presented with a fallacious alternative during the September interview, she was able to spot it and to say why it was wrong. For example, "not able to add a number to an X term", "not able to separate a number from an X term", etc. (see Sleeman 1983c for more details). In May, this student showed a lack of understanding of basic algebraic notation which appeared to be remedied by September. To see whether this was the case the investigator also presented tasks from sets 12 and 13 of Figure 2, i.e., tasks of the form:

$$M + N * X + P * X = L \quad \text{and} \quad M * X = N + P * \langle Q * X + R \rangle$$

All of which she worked correctly and was able to verbalise the stages she went through. An equation which contained an "unusual" variable, AA, was also presented and again this was worked correctly. Similarly several other students showed substantial "progress", and again this was associated with the ability to <u>explain</u> what they were doing.

4. SUBSEQUENT UPGRADE of LMS and its DATABASE

The set of rules and mal-rules used in the 1981 experiment has subsequently been enhanced to include the additional manipulative- and parsing- mal-rules confirmed by the student interviews. [12]. Additionally, the code of LMS has been extended to deal with mal-rules which have a somewhat different character from manipulative rules. The extended LMS with the enhanced database is able to diagnose the majority of the errors encountered both in the on-line sessions carried out in May 1981 (see Figure 3) and in the interviews (see Figures 5&6) Sleeman 1983c.

5. COGNITIVE MODELLING and an INTERPRETATION of the RESULTS of the 1981 EXPERIMENT.

There is a steadily growing body of data about how school and college students solve algebra tasks, Paige & Simon 1966, Lewis 1980, Davis, Jockusch & McKnight 1978, Kuechemann 1981, Sleeman 1982 and Sleeman 1983c. The major thrust of this paper is the analysis of the experimental results reported here in terms of related work in cognitive modelling; Section 2 gives a summary of this earlier work.

<u>Pertinent Observations from this Experiment</u>

1. Students appear to regress under cognitive load, see Sleeman 1983b for details.

---

12. Some of the mal-rules noted in Figure 4 were not seen in the interviews. I now believe these were an artifact of the first version of LMS which <u>forced</u> the student to give a response to each question. LMS has now been modified so that the student can give up on any task.

2. There appears to be a number of clearly identifiable types of errors. (Section 5.1).

3. Students use a number of alternative "methods" to solve tasks of the same set. (Section 5.2).

5.1 Classification of observed student errors

We propose a classification of students' errors observed in this experiment, namely, manipulative, parsing, clerical and "random". Practically, this classification is of considerable importance as it enables one to give appropriate remedial instruction for the several types of error. In case of "manipulative" mal-rules it would appear that the student basically "knows" the rule, but due to cognitive overload or inattention, is omitting substep(s). The parsing errors appear to arise from a profound misunderstanding of algebraic notation, and so have to be remediated very differently. Additionally, in this section we suggest mechanisms for several of these error-types.

5.1.1 Manipulative Errors

We define a manipulative mal-rule to be a variant on a correct rule which has one sub-stage either omitted or replaced by an inappropriate or incorrect operation, c.f., Young & O'Shea 1981. For example, MNTORHS is a mal-rule which captures the movement of a number to the other side of an equation where the student omits to change the sign of the number. We suggest that all the mal-rules reported in Figure 1b and those numbered 7-9 in Figure 4 fall within this category. Also in Sub-section 5.1.1b we briefly discuss the (apparently) related phenomena of confusion of operands.

a) A schema for generating manipulative mal-rules

In Figure 4 we report 3 new (manipulative) mal-rules, variants of SOLVE, SIMPLIFY and BRA3 respectively, [13], which can be explained by a deletion mechanism, as can the "original" mal-rules given in Figure 1b. Note that this schema would ALSO generate many mal-rules which we have not yet observed. In the next paragraph we suggest why some of the possible mal-rules are not observed.

A variant on SOLVE. The student realizes he has a task in which the SOLVE rule should be activated and forgets to apply one of the operations, namely dividing by M. SOLVE has three principal actions: noting down N, the divide symbol and M, and so this mal-rule could be said to be omitting some of the principal steps. It appears that students have an idea about the acceptable FORM of answers and so given:

M * X = N, they do NOT produce X = /M or X = N/

A variant on SIMPLIFY. Examples of the two rules given here, which have occurred reasonably frequently are:

X = 6/4 => X = 3/4

X = 6/4 => X = 6/2

Again we argue that the above observations can be explained if we assume that this rule has several principal steps including calculate the common factor, divide the "top" by the common factor, and divide the "bottom" by the common factor. Each of these mal-rules corresponds to one of the latter steps being omitted.

A variant on BRA2. We have seen the task:

6 * X = 4 * <2 * X + 3> => 6 * X = 4 * X + 12

BRA2 is a more complex rule with several steps and so one might expect to find a

---

13. The variant on SOLVE reported in Figure 4 is:

M * X = N => X=N

Two variants on SIMPLIFY reported in Figure 4 are:

M * X = N => X = (N/F)/M
M * X = N => X = N/(M/F)

where F is a factor of M and N.

The variant on BRA2 reported in Figure 4 is:

M*<N * X + P> => M * X + M*P

correspondingly larger number of mal-rules. This is indeed true. This "new" mal-rule also conforms to the pattern noted above, as it can also be explained by the omission of one sub-action.

b) Confusion of Operands.

We have noted errors of the following form:

5 * X = 12 ==> X = 2 2/12

where clearly one operand is confused for another. Norman 1981 explains such slips by saying that they are a consequence of a noisy processor.

c) "Grain size" and manipulative mal-rules.

There is a very real sense in which detailed analyses of manipulative mal-rules allows one to infer the substeps processed by students, and this in turn allows one to predict the set of mal-rules that will be encountered in a domain. (Bearing in mind the idea of acceptable form outlined above). Further, one might argue that the representation of the tasks should be at this "lower" level: the justification for the representation chosen (see Figures 1a&b) is that this appears to be more consistent with the collected verbal and written protocols for students solving these tasks.

The schema discussed above for generating manipulative mal-rules by omitting, or modifying, one substep is thus consistent with Young and O'Shea's modelling of subtraction. Further, we believe that confusion of operands can be seen as a variant of this same mechanism.

5.1.2 Incorrect Representation of the Task or Parse Errors.

We have categorized the first 6 sets of Mal-Rules in Figure 4 as ones which summarize what happens when a student "mis-sees", or mis-parses, an algebraic equation. We assert that many of the students whom we interviewed carried out steps of the computations in ways which would not fall within the definition given earlier for manipulative mal-rules. Below, we give typical protocols for two students working the task 6*X = 3*X + 12:

6*X = 3*X + 12	6*X = 3*X + 12
9*X = 12	X + X = 12 + 3 - 6
X = 12/9	2*X = 9
X = 4/3	X = 9/2

When we pressed the "first" student for an explanation of how the original equation was transformed into the second, i.e., 9*X = 12, the student talked about moving the 3*X term across to the left hand side. Thus the interviewer concluded the student was using a variant of the correct rule, namely a manipulative mal-rule. When the "second" student was pressed he simply asserted that the change from the original equation to the second form "was all done in one step". Hence the interviewer concluded it was a very different type of mal-rule involved and not a simple variant on the correct rule. Thus the interviews provided essential additional information as, of course, the second student's protocol could be explained by the use of MXTOLHS and the mal-rule:

$$M * X \Rightarrow M + X$$

which some people might wish to argue constitutes a manipulative mal-rule (replacing the * operator by the + operator). This investigator maintains that such a transformation reveals a profound misunderstanding of algebraic notation and so should be considered as a parsing mal-rule.

Additional "evidence" for the distinction between manipulative and parsing mal-rules comes from an understanding of the likely representation of the equation for the two groups of students. Figure 7a gives a correct parse tree for the equation discussed above. Highly probable inadequate representations for the equation, which are consistent with observed mal-rules, are the linear algebraic string, i.e., the usual written form of an algebraic equation, and the not-so-deeply nested tree given in Figure 7b. These latter representations suggest that the student has failed to appreciate the semantics of algebraic expressions - and sees the solution of algebraic equations as a symbol manipulating task. We collected considerable evidence to support this view earlier, Sleeman 1982, and in the 1981 experiment (see Figure 6.IIa), where a student transformed:

$$2 * X + 4 * X = 12 \quad \text{into} \quad * X * X = 12 - 2 - 4.$$

a) Schema for "generating" Parsing mal-rules.

In the course of the interview student AB18 explained that he was carrying out the teacher-given algorithm of: "Collecting all the Xs on the left hand side and collecting all the numbers on the right hand side", and added that he was not really sure what to do about the "extra multiply signs". Student AB17 gave a similar explanation for his actions. This gives us a schema for generating mal-rules. In this section we explore this topic further.

For example given the task-type:

$$M * X + N * X = P$$

This schema gives the following "action sides" for mal-rules:

$$X + X = P - M - N$$

$$X + X = P + M + N$$

where in the second case the X coefficients are treated "specially", i.e., the coefficients of the Xs were taken across to the RHS of the equation but the signs were NOT changed. Additionally, there is the form given by student AB17, and quoted in Figure 6.II, namely:

$$* X * X = P - M - N$$

which he went on to "normalize" (see Section 3.2.4) to:

$$X * X = P - M - N$$

and its "complementary" form:

$$X * X = P + M + N$$

Similarly, given the task-type:

$$M * X = N * X + P$$

this schema creates the following forms:

$$X = N + P - M$$

$$X = N + P + M$$

$$X + X = N + P - M$$

$$X + X = N + P + M$$

$$X - X = N + P - M$$

$$X - X = N + P + M$$

For example on task h), student AB18 suggested the use of both the third and the fifth forms (see Figure 6.III).

As argued above, unlike the manipulative mal-rules, the parsing mal-rules cannot be explained by omitting a component. Neither does it seem that they can be explained by performing a repair to a core procedure, unless one is prepared to broaden one's view of a repair to include the schema which were observed with students AB17 and AB18, and the "extrapolation" procedure noted by Matz, [14].

---

14. Further, I suggest both these schema could have been created by the misgeneralization mechanism discussed in Section 5.2.

5.1.3 Clerical errors.

Analysing some of the protocols, one is happy with the explanation that some "slips" occur. For example:

$$10 * X = 25 \Rightarrow X = 25/18$$

$$2 * X = 6*5 \Rightarrow X = 18$$

In the first case the student has probably seen the "0" as an "8". In the second he has probably made an arithmetic error, [15]. DEBUGGY (Burton 1982) considers an answer to be a "number-bond slip" if the answer is within 2 of the correct one. The second slip given above could be explained if we had an analogous algorithm for the evaluation of multiplicative expressions. However, the first slip, a "visual" one, clearly could not be. So we suspect that to account for the variety of "slips" encountered in this domain a more sophisticated approach, c.f., that advocated by Norman 1981 would be necessary. However, we have not thought this worth investigating as clerical errors have so far been relatively infrequent.

5.1.4 "Wild"/Unexplained Errors.

Many of the mistakes not explained so far may be due to the consistent use of mal-rules which so far we have not identified, [16].

5.2 Using Alternative Methods or Bug Migration

Repair theory gives a neat explanation for the observed phenomena of bug migration in the domain of multi-column arithmetic, Brown & VanLehn 1980, namely that the student will use a related family of mal-rules, and possibly the correct rule, during a single session with a particular task-set.

---

15. Given the earlier definition of manipulative mal-rules, it appears that the errors reported above have a different nature. At the very least if they are instances of manipulative mal-rules, they are not algebraic manipulative mal-rules.

---

16. As this is clearly a very demanding task, there is a need to implement some computational device to assist the investigator. A preliminary system has been implemented which has already given several "explanations" not spotted by the investigator.

There seems to be an alternative explanation which should also be considered. Although a task-set may have been designed to highlight one particular feature, the student may spot completely different feature(s) and these may dominate his solution, [17]. Repair theory accounts for some bugs by hypothesizing that the student had not encountered the appropriate teaching necessary to perform the task. Suppose we make the converse assumption, that the appropriate teaching had been carried out, and further suppose that some students gain competence in this domain not by being told the rules but rather by inferring rules for themselves, by noting the transformations which are applied to tasks by the teacher and in texts, [18], [19]. It seems reasonable that the student's inference procedure should be guided by his previous knowledge of the domain, in this case the number system, and that the student normally infers several rules which are consistent with the example, and not just the correct rule. Indeed due to some missing knowledge the correct rule may not be inferred. (And so the fact that the student never uses the correct method along with several buggy methods is not evidence that he has NOT encountered the material before.) We shall refer to this process as Knowledge Directed Inference of rules, or mis-generalization for short.

Suppose, the student saw the following stages in an algebraic simplification:

3 * X = 6 ==> X = 6/3

Then he might infer

X = RHS number/LHS number OR X = LARGER number/SMALLER number

Further, we suggest that schema such as that articulated by students AB17 and AB18 could have been inferred by the process of mis-generalization.

---

17. Earlier Sleeman and Brown 1982 have argued: "......Perhaps more immediately, it suggests that a Coach must pay attention to the sequence of worked examples, and encountered task states, from which the student is apt to abstract (invent) functional invariances. This suggests that no matter how carefully an instructional designer plans a sequence of examples, he can never know all the intermediate steps and abstracted structures that a student will generate while solving an exercise. Indeed, the student may well produce illegal steps in his solution and from these invent illegal (algebraic) "principles". Implementing a system with this level of sophistication still presents a major challenge to the ITS/Cognitive Science community... "

---

18. Note I am not claiming that there is a single mechanism. Matz has provided another mechanism namely, that some students use an "extrapolation principle" to extend a method they know works in one context to further analogous contexts.

---

19. Independently, VanLehn 1983b has come to a similar conclusion, the Sierra system described in his thesis relies heavily on inference.

We will surmise how a student would use such a rule-set or schema. We will suppose that the abler students actively experiment with different "methods", and use their own earlier examples, examples worked by the teacher or in the text to provide discriminatory feedback. From our experiment with 14-year-old students we have direct evidence that some students, e.g., students AB17 and AB18, are aware of having a range of applicable rules and being unsure of when to select a particular method. That study did not provide any insights into the rule-selection processes used by those students. We could suggest the common default, i.e., that the process is random. However, studies in cognitive modelling have already discredited this explanation many times, so we will postulate that the process is deterministic but currently "undetermined". It is further suggested that tasks which show a rule is inadequate will weaken belief in the rule, but once a (mal)-rule is created it may not be completely eliminated, particularly if the "counter-examples" are not presented to the student for some period. Thus given this view point, the phenomena of bug-migration occurs because the student has inferred a whole range of rules and selects a rule using a "black-box" process. Given a further task, he again chooses a method and hence selects the same or an alternative algorithm, influenced partly by the relative strengths of the rules. That is, if the relative weights are comparable it is more likely that the student will select a different method for each task. If one weight dominates then it is likely that the corresponding method will be selected frequently. Further, if only one (mal) rule is generated by the induction process then this approach predicts that the student will consistently use that rule.

We suggest that many of the bugs encountered in the subtraction domain can be accounted for by this (inference) mechanism. For instance the Smaller-from-Larger bug where the smaller number is subtracted from the larger <u>independent</u> of whether the larger number is on top or the bottom row seems one such example, Brown & Burton 1978 and Young & O'Shea 1981. Brown & VanLehn 1980 report that because borrowing was introduced, with one group of students, using only 2-column tasks these students inferred that whenever borrowing was involved they should borrow from the left-most column, their "Always-Borrow-Left" bug. (So it appears important to ensure that the example set includes some examples to counter previously experienced mal-rules. Indeed it seems as if task-sets can be damaging if they are too preprocessed and contain too little "intellectual ruffage"; Michoner 1978 makes a similar argument.) Additionally, Ginsburg 1977, quotes several instances of young children inferring the name "three-ty" for 30, given the names for "40", "50", "60" and "3". So given the wealth of experimental evidence this alternative explanation should be given serious consideration.

Further, I have two philosophical reservations about repair theory. Firstly, by some mechanism not articulated all students acquire a common set of impasses, and moreover they consistently observe these. Secondly and more significantly, repair theory, which sets out to explain major individual differences at the task level, itself proposes a specific mechanism common to all students. [20]. On the other hand, mis-generalization predicts that the individual's initial knowledge profoundly influences the knowledge which is subsequently inferred, and captures the sense in which learners are active theory builders trying to find patterns, making sense out of observations, forming hypotheses, and testing them out.

5.3 Conclusions

Firstly, we have two explanations for some of the misunderstandings noted with algebraic notation. Namely, that given by Matz 1982 and that given by students, AB17 and AB18, see Section 5.1.2. Certainly, Matz's explanation explains some of our observations, but not all as in some cases the coefficients are treated "specially", and their sign is not changed when they are moved to the RHS. For example, we have observed:

$$3 * X + 4 = 12 \Rightarrow X = 12 + 3 - 4$$

i.e., the student changes the sign of the 4 but not the 3.

Secondly, there are two hypotheses which explain bug-migration: the one given by repair theory and the one put forward here, namely mis-generalization. Of course it is possible that each may be applicable in different situations.

Thirdly, several "algorithms" have been presented for creating student models. I believe these are suggestive about the students' organization of this knowledge and about the processes used when students solve (these) tasks. Repair theory suggests patches are made to incomplete core-procedures. Young and O'Shea suggest that it is adequate to take a correct procedure and merely delete substeps. The data for the algebra manipulative mal-rules can be adequately explained by either. However, Young and O'Shea's approach seems inadequate to explain the

---

20. Indeed, I am concerned that many theories of (child) development do not accept the possibility of there being significant individual differences in development, but merely in the individual's rate of progress and the level of his final maturation.

parsing mal-rules. Indeed, we have to extend the revised repair theory before the results reported here can be accommodated. (An analogous extension is needed to accommodate the Davis/Matz results). This paper claims that there are two very different types of malrules at large with algebra students - namely manipulative and parsing mal-rules. The existence of this second category of algebra errors, and many of the mal-rules collected in other areas, appear to be best explained by a further mechanism, namely misgeneralization, [21].

Fourthly, there is evidence that once _inferred_, rules are additionally _applied_ incorrectly. I suggest that the mechanism(s) described by repair theory, Young & O'Shea, and Section 5.1.1 are appropriate for the _application_ stage, whereas misgeneralization is a more plausible mechanism to explain rule _acquisition_.

---

21. The above comparisons of explanations (or theories) are important in that they remind us of the essentially pragmatic nature of Cognitive Science.

6. REFERENCES

J.S. Brown & R.R. Burton (1978). Diagnostic Models for procedural bugs in basic Mathematical Skills. in Cognitive Science, 2,2. pp155-192.

J.S. Brown & K. VanLehn (1980). Repair Theory: A generative theory of bugs in procedural skills. Cognitive Science, 4. pp 379-426.

J.S. Brown, R.R. Burton, & J. de Kleer (1982). Pedagogical, Natural Language and Knowledge Engineering Techniques in SOPHIE I, II and III. In Intelligent Tutoring Systems, edited by D. Sleeman & J.S. Brown. New York: Academic Press, pp 227-282.

A. Bundy (1982). Personal communication.

R.R. Burton & J.S. Brown (1981). An investigation of computer coaching for informal learning activities. In Intelligent Tutoring Systems, edited by D. Sleeman & J.S. Brown. New York: Academic Press, pp 79-98.

R.R. Burton (1982). Diagnosing bugs in a simple procedural skill. In Intelligent Tutoring Systems, ed. D.Sleeman and J.S. Brown, New York: Academic Press, pp 157-183.

W.J. Clancey (1982). Tutoring rules for guiding a case method dialogue. In Intelligent Tutoring Systems, edited by D. Sleeman & J.S. Brown. New York: Academic Press, pp 201-225.

R.B. Davis, E. Jockusch & C. McKnight (1978). Cognitive Processes in Learning Algebra. Journal of Children's Mathematical Behaviour, 2, pp1-320.

H.P. Ginsburg (1977). Children's Arithmetic: the Learning Process. New York: Van Nostrand.

I.P. Goldstein (1982). The genetic graph: a representation for the evolution of procedural knowledge. In Intelligent Tutoring Systems, edited by D. Sleeman & J.S. Brown. New York: Academic Press, pp 51-77.

J.R. Hartley & D.H. Sleeman (1973). Towards intelligent Teaching Systems, Int. J. Man-Machine Studies, 5, pp215-236.

D. Kuechemann (1981). Algebra in Children's Understanding of Mathematics: 11-16, edited by K.M. Hart, London: John Murray, pp102-119.

C. Lewis (1980). Kinds of Knowledge in Algebra. Paper presented to the AERA Annual Conference, 1980.

M. Matz (1982) Towards a process model for high school algebra errors. In Intelligent Tutoring Systems, edited by D. Sleeman and J.S. Brown, Academic Press, pp 25-50.

E.R. Michener (1978). Understanding Understanding Mathematics. Cognitive Science, 2, pp 361-383.

D.A. Norman (1981). Categorization of Action slips. Psychological Review, 88, pp1-15.

J.M. Paige & H.A. Simon (1966) Cognitive Processes in Solving Algebra Word Problems. In Problem Solving: Research, Method and Theory, edited by B. Kleinmuntz, Published by Wiley.

D.H. Sleeman & M.J. Smith (1981). Modelling pupil's Problem Solving. Artificial Intelligence, pp 171-187.

D. Sleeman (1982) Assessing competence in basic Algebra. In Intelligent Tutoring Systems, edited by D. Sleeman and J.S. Brown, Academic press, pp 186-199.

D. Sleeman & J.S. Brown (1982) Editorial in Intelligent Tutoring Systems, edited by D. Sleeman and J.S. Brown, Academic Press, pp 1-12.

D.H. Sleeman, & R.J. Hendley (1982). ACE: A system which Analyses Complex Explanations. In Intelligent Tutoring Systems, edited by D. Sleeman & J.S. Brown. New York: Academic Press, pp 99-118.

D.H. Sleeman (1983a). A Rule Directed Modelling system. In Machine Learning, R. Michalski, J. Carbonell, and T. M. Mitchell, (editors) Tioga Press, pp 483-510. .

D.H. Sleeman (1983b). An attempt to understand pupil's understanding of basic algebra. HPP working paper, 83-11.

D.H. Sleeman (1983c). Basic Algebra revisited: a study with 14-year-olds, Stanford University memo HPP, 83-9, and to be published in Int. Journal of Man-machine studies.

K. VanLehn (1981). Bugs are not enough: empirical studies of bugs, impasses and repairs in procedural skills. Xerox PARC, Cognitive and Instructional Sciences Tech. report, CIS-11.

K. VanLehn (1983a). On the Representation of Procedures in Repair Theory, in The Development of Mathematical Thinking ed H.P. Ginsburg. New York: Academic Press, pp 197-252.

K. VanLehn (1983b). Felicity conditions for human skill acquisition: Validating an AI-based theory. XEROX PARC tech. report CIS-21.

R. Young & T. O'Shea (1981). Errors in Children's Subtraction, Cognitive Science, 5, pp153-177.

Figure 1

a) RULES for the ALGEBRA domain (slightly stylized).

RULE NAME	CONDITION	ACTION
FIN2	(X = M/N)	((M N)) or ((M))
SOLVE	(M * X = N)	(X = N/M) or (INFINITY)
SIMPLIFY	(X = M/N)	(X = M'/N')
ADDSUB	(lhs M +|- N rhs)	(lhs [evaluated] rhs)
MULT	(lhs N * N rhs)	(lhs [evaluated] rhs)
XADDSUB	(lhs M*X +|- N*X rhs)	(lhs (M +|- N) * X rhs)
NTORHS	(lhs +|- M = rhs)	(lhs = rhs -|+ M)
REARRANGE	(lhs +|-M +|- N*X rhs)	(lhs +|- N*X +|- M rhs)
XTOLHS	(lhs = +|- M*X rhs)	(lhs -|+ M*X = rhs)
BRA1	(lhs < N > rhs)	(lhs N rhs)
BRA2	(lhs M*<N*X +|-P> rhs)	(lhs M*N*X +|- M*P rhs)

Where M, N and P are integers, lhs & rhs are general patterns (which may be null), +|- means either + or - may occur, and < and > represent standard algebraic brackets. For example, the MULT rule changes the state (3 * X = 5 + 3 * 4) to (3 * X = 5 + 12) where the variables in the pattern, lhs, rhs, M and N, are bound to (3 * X = 5 +), null, 3 and 4 respectively.
See Figure 1c for a complete trace of this and several other tasks.

b) Some MAL-RULES for the Domain

RULE NAME	CONDITION	ACTION
MSOLVE	(M*X = N)	(X = M/N) or (INFINITY)
MNTORHS	(lhs +|- M = rhs)	(lhs = rhs +|- M)
M2NTORHS	(lhs1 +|- M lhs2 = rhs)	(lhs1 +|- lhs2 = rhs -|+ M)
M3NTORHS	(lhs1 +|- M lhs2 = rhs)	(lhs1 +|- lhs2 = rhs +|- M)
MXTOLHS	(lhs = +|- M*X rhs)	(lhs +|- M*X = rhs)
M1BRA2	(lhs M *<N*X +|- P> rhs)	(lhs M*N*X +|- P rhs)
M2BRA2	(lhs M*< N*X +|- P> rhs)	(lhs M*N*X +|- M +|- P rhs)

Using the same conventions as above.

c) Pairs of correct and "buggy" models executing typical tasks.

i) shows the correct model (MULT ADDSUB SOLVE FIN2) and the "buggy" model (ADDSUB MULT SOLVE FIN2) solving 3 * X = 5 + 3 * 4.

[The first line gives the initial state and all subsequent lines give the rule which fires and the resulting state.]

	3 * X = 5 + 3 * 4		3 * X = 5 + 3 * 4
MULT	3 * X = 5 + 12	ADDSUB	3 * X = 8 * 4
ADDSUB	3 * X = 17	MULT	3 * X = 32
SOLVE	X = 17/3	SOLVE	X = 32/3
FIN2	(17 3)	FIN2	(32 3)

ii) shows the correct model (NTORHS ADDSUB SOLVE FIN2) and the "buggy" model (MNTORHS ADDSUB SOLVE FIN2) solving 4 * X + 6 = 19.

	4 * X + 6 = 19			4 * X + 6 = 19
NTORHS	4 * X = 19 - 6		MNTORHS	4 * X = 19 + 6
ADDSUB	4 * X = 13		ADDSUB	4 * X = 25
SOLVE	X = 13/4		SOLVE	X = 25/4
FIN2	(13 4)		FIN2	(25 4)

Figure 2 Typical tasks for each task-set and which rule(s) are being focussed on

Task-set	Rules Focussed On	Typical Task
2	SOLVE	6 * X = 7
3	ADDSUB	3 * X = 5 + 3
4	MULT	5 * X = 2 * 2
5	XADDSUB	2 * X + 3 * X = 10
6	NTORHS	2 * X + 4 = 16
7	REARRANGE	4 + 2 * X = 16
8	XTOLHS	4 * X = 2 * X + 3
9	BRA1	2 * X = 5 * <3 + 1>
10	BRA2	6 * X = 4 * <2 * X + 3>
11	ADDSUB&MULT	2 * X = 2 + 4 * 6
12	ADDSUB&XADDSUB	2 + 3 * X + 4 * X = 16
13	ADDSUB&BRA2	15 * X = 2 + 4 * <2 * X + 3>
14	MULT&XADDSUB	2 * 4 * X + 2 * X = 12
15	MULT&BRA2	14 * X = 2 * 3 * <2 * X + 3>

Figure 3

Protocols from which new mal rules were inferred. (Note the teacher specified the way in which the X-coefficient should be represented. NOTE too that some of the protocols are not totally consistent: the investigator has given the mal-rules which summarizes the student's behaviour in the majority of the tasks).

```
Task-set 5
Task is (2 * X + 4 * X = 12) Student's solution was (1 * X = 3)
Task is (2 * X + 3 * X = 10) Student's solution was (2 * X = 10 - 2 - 3)
Task is (3 * X + 2 * X = 11) Student's solution was (1 * X = 6 // 2)
Task is (2 * X + 6 * X = 10) Student's solution was (1 * X = 1)
Task is (3 * X + 4 * X = 9) Student's solution was (1 * X = 1)
Task is (2 * X + 4 * X = 3) Student's solution was (1 * X = -3 // 2)
Task is (4 * X + 2 * X = 4) Student's solution was (8 * X = 4)
```

Figure 3a. Protocol apparently showing M * X => M + X (student AB17).

```
Task-set 6
Task is (2 * X + 4 = 16) Student's solution was (1 * X = 8 // 3)
Task is (2 * X + 3 = 9) Student's solution was (1 * X = 9 // 6)
Task is (3 * X - 4 = 6) Student's solution was (1 * X = -6)
Task is (2 * X + 5 = 10) Student's solution was (1 * X = 10 // 7)
Task is (6 * X + 4 = 6) Student's solution was (1 * X = 3 // 5)
Task is (5 * X + 2 = 6) Student's solution was (1 * X = 5 // 7)
```

Figure 3b. Protocol apparently showing M * X + N => (M + N) * X (student AB20).

```
Task-set 7
Task is (4 + 2 * X = 16) Student's solution was (1 * X = 8)
Task is (2 + 4 * X = 14) Student's solution was (1 * X = 6)
Task is (3 + 5 * X = 11) Student's solution was (1 * X = -4)
Task is (4 + 6 * X = 11) Student's solution was (1 * X = -13)
Task is (4 + 5 * X = 6) Student's solution was (1 * X = -14)
Task is (5 + 2 * X = 8) Student's solution was (1 * X = -2)
```

Figure 3c. Protocol apparently showing M + N * X => M * N + X (student AB3).

```
Task-set 8
Task is (4 * X = 2 * X + 6) Student's solution was (1 * X = 6)
Task is (3 * X = 2 * X + 5) Student's solution was (1 * X = 5)
Task is (3 * X = -2 * X + 7) Student's solution was (1 * X = 4)
Task is (4 * X = 2 * X + 3) Student's solution was (1 * X = 9 // 2)
Task is (4 * X = -2 * X + 8) Student's solution was (1 * X = 5)
Task is (6 * X = 2 * X + 3) Student's solution was (1 * X = 11 // 2)
```

Figure 3d. Protocol apparently showing MX = NX + P => X + X = M + N + P (student AB1).

```
Task-set 7
Task is (4 + 2 * X = 16) Student's solution was (1 * X = 2)
Task is (2 + 4 * X = 14) Student's solution was (1 * X = 4 // 2)
Task is (3 + 5 * X = 11) Student's solution was (1 * X = 5 // 3)
Task is (4 + 6 * X = 11) Student's solution was (1 * X = 6 // 4)
Task is (4 + 5 * X = 6) Student's solution was (1 * X = -5 // 4)
Task is (5 + 2 * X = 8) Student's solution was (1 * X = 8 // 2)
```

Figure 3e. Somewhat erratic protocol: 3 responses conform to M + N * X = P => M * X = N (student AB7).

<u>Figure 4.</u> <u>Summary of major new mal-rules encountered in recent experiments</u>. Sets 1 to 6 give "parsing" mal-rules and 7-9 additional manipulative mal-rules, and F in mal-rule 8 represents a common factor*.

1a.	M • X + N • X	=> M • X • N
		=> M • X + N
		=> M + X + N + X
1b.	M • X	=> M + X
2.	M + N • X =	=> M • N + X =
		=> M + N + X =
3.	M • X + N =	=> M + X + N =
		=> M • X • N =
		=> (M + N) • X =
4.	M • X = N•P	=> X = N
5.	M • X = N • X + P	=> X + X = M + N + P
6.	M • X = N + P	=> M • X = N
		=> M • X = P
7.	M • X = N	=> X = N
8.	M • X = N	=> X = (N/F)/M
		=> X = N/(M/F)
9.	M • <N • X + P>	=> M • X + M • P

---

* Note these rules could have been specified in exactly the same format as that used in Figures 1a&1b; the current form has been used for brevity. However, in that earlier notation rules 1b and 9 above would become:

CONDITION	ACTION
(lhs M • X rhs)	(lhs M + X rhs)
(lhs M • < N • X + P> rhs)	(lhs M • X + M • P rhs)

where lhs & rhs are general patterns.

Figure 5.

Protocol for a student who has a number of "Alternative Methods".

Student AB17 on task-set 6.

a) The task given was: 2 * X + 3 = 9

Student writes 1) 2X = 9 - 3
2) X = 3

Interviewer writes X = 9 - 3 + 2

Interviewer: Could you say whether your step 1) above or what I've just written is correct.

Student says he really could not.

b) The task given was: 2 * X + 4 = 16

Student writes 1) 2X = 16 - 4
2) 2X = 12
3) X = 6

Interviewer writes X = 16 - 4 - 2

Interviewer: Could you say whether your step 1) above or what I've just written is correct.

Student says his 1) probably is.

Interviewer: Can you say why?

Student: I'm afraid not.

Interviewer: Now look back at the last example, there I suggested a slightly different method there. Would that be possible here?

Student: That's right, it would.

Interviewer: Which of these do you think is correct?

Student: Really not sure. I often have a lot of methods to choose between, which makes it pretty confusing. I sometimes have as many as 5 or 6.

[And so this conversation continues. After this point the student voluntarily offers 2 or 3 solutions to each task, as in the next task.]

c) The task given was 4 * X = 2 * X + 6

Student writes 1) X = 2 - 4 + 6
X = 4

Then suggests the following reworking:

1) 4X = 2X + 6
2) 4X = 8X
Then Quits.

Interviewer: Which solution do you think is right?

Student: Oh, I'm not really sure.

Interviewer: If you were a betting man, which would you put your money on?

Student: Probably the first.

Figure 6
Three examples of very consistently used MAL-RULES.

I) Student AB11 on task-set 7.
a) The task given was: 4 + 2 * X = 16

Student writes 1) 6X = 16
2) X = 2.6666

b) The task given was: 2 + 4 * X = 14

Student writes 1) 6 * X = 14
2) X = 2.333

c) The task given was: 3 + 5 * X = 11

Student writes 1) 8 * X = 11
(and is told she can leave it in that form)
d) The task given was: 5 - 3 * X = 11

Student writes 1) 2 * X = 11
(and is told she can leave it in that form)

II) Student AB17 on task-set 6
a) The task given was: 2 * X + 4 * X = 12

Student writes 1) * X * X = 12 - 2 - 4
2) X ** 2 = 6
3) X = ROOT 6
b) The task given was: 2 * X + 3 * X = 10

Student writes 1) * X * X = 10 - 2 - 3
2) X ** 2 = 5
(and is told he can leave it in that form)
c) The task given was: 2 * X - 3 * X = 10

Student writes 1) * X * X = 10 - 2 + 3
2) X ** 2 = 11
(and is told he can leave it in that form)

III) Student AB18 on task-sets 5, 6, 7 and 8.
a) The task given was: 2 * X + 3 * X = 10

Student writes 1) 2 * X = 10 - 2 - 3
2) 2 * X = 5
3) X = 2.5
b) The task given was: 3 * X + 5 * X = 24

Student writes 1) X + X = 24 - 3 - 5
2) 2 * X = 16
3) X = 8
c) The task given was: 3 * X + 4 * X + 5 * X = 24

Student writes 1) X + X + X = 24 - 3 - 4 - 5
2) 3 * X = 12
3) X = 4
d) The task given was: 2 * X + 4 = 20

Student writes 1) X = 20 - 2 - 4
2) X = 14

e) The task given was: $3 \cdot X + 5 = 7$

Student writes
1) $X = 7 - 3 - 5$
2) $X = -1$

f) The task given was: $4 + 3 \cdot X = 14$

Student writes
1) $X = 14 - 3 - 4$
2) $X = 7$

g) The task given was: $5 + 6 \cdot X = 20$

Student writes
1) $X = 20 - 5 - 6$
2) $X = 9$

h) The task given was: $4 \cdot X = 2 \cdot X + 6$

Student writes
1) $2 \cdot X = -4 + 2 + 6$
2) $2 \cdot X = 4$
3) $X = 2$

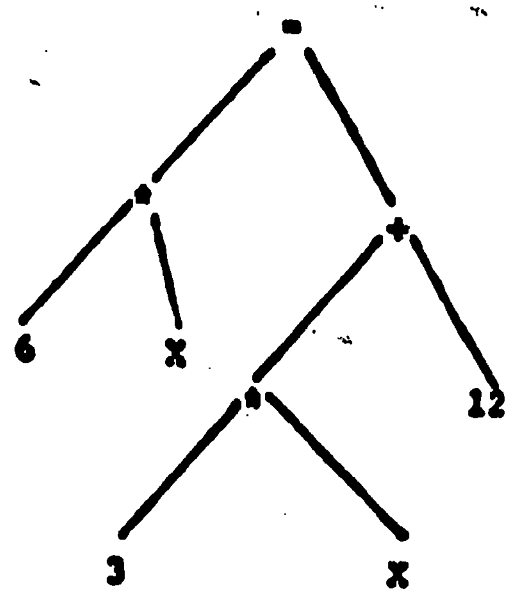
Student then wrote
1) $X - X = 2 + 6 - 4$
2) $0 = 4$
and QUITS.

i) The task given was: $5 \cdot X = 3 \cdot X + 6$

Student writes
1) $0 = 4$
and QUITS.

Figure 7

a) The correct parse tree for the equation $6 * X = 3 * X + 12$



b) A "two-level" representation for the same equation where [ ], following Bundy 1982, represents a "plus bag", that is, all the entities with the bag are operated on by the addition operator.