*A SYSTEM WHICH ALLOWS STUDENTS TO EXPLORE ALGORITHMS

D H Sleeman

Computer Based Learning Project
Department of Computer Studies
The University of Leeds, England, UK

Abstract

This paper discusses the problems of making algorithms 'transparent'. The approach taken has been to define a Formal Language for the problem domain which describes the main steps in the algorithm under discussion. This Formal Language has been used in a system which allows the student to enquire about selected parts of the algorithm and in a facility which comments on the student's explanations of certain parts of the algorithm. Some indication is given as to how the system's facilities can be extended to answer a further range of questions, but it is concluded that in order to make substantial progress in this pursuit a single representation for algorithms is required which can be both executed and used as a basis for explanations.

1. Introduction

There are clearly two major driving forces in the field of generative CAI (Computer Assisted Instruction); namely, the practical one of implementing systems which are capable of producing teaching material, and secondly the interest of the AI (Artificial Intelligence) practitioners who seek to replicate various facets of intelligent behaviour (teaching clearly being one such activity), Generative CAI systems have used essentially two types of knowledge bases: Semantic Nets and Algorithms. In this paper we shall be concerned only with the latter type and we shall see that, given an algorithm which is able to solve problems in a particular domain, it enables the system to offer a range of teaching modes. These capabilities can be divided into three levels of sophistication which are represented by the ability to:

(i) Tell the student whether or not his solution is correct.

(ii) Show the student how the algorithm would solve the same problem (this is essentially showing the student the trace of the algorithm).

(iii) Debug the student's explanation of how the algorithm, or part of the algorithm works.

*This is a revised version of a paper given at the SSRC "Computers in Education" Seminar held at the University of Warwick, UK, July 1976.

The main problem with level one systems is deciding whether or not the answer given by the student is equivalent to that generated by the algorithm. In general, this is a non-trivial task: but so far areas have been chosen where the answer required is a number or a simple literal and hence this difficulty has been avoided (see for instance Uhr[1]). Peplinskil2] has implemented a second level system which gives students linear equations to solve and requires them to respond with the value for the unknown. After a number of unsuccessful attempts, the program presents the student with all the steps which the algorithm performs in solving the problem. In this paper we shall discuss a system which presents the student with the trace of the algorithm and then allows him to discuss certain aspects of this information. Further, we wish to consider the third and more challenging problem of designing a system to comment on (debug) the student's explanations.

The system we have implemented assumes that the students are familiar with the algorithm but may need some assistance in extracting the essential information from that presented for a particular problem, and that the students may need some help in Processing this information. Suppose that the given algorithm contains *N' sequential processes, P1 to Pn: suppose that the essential feature of the first N-1 processes is that each calculates the value of a variable, say V1 to Vn-1, and further suppose that the essential feature of the Nth process is to calculate the value of a further variable, Vn using the values of the variables V1 to Vn-1. If the student requests help about this algorithm, it seems reasonable to present him with a TRACE which consists of N-I facts (variable values) and a single DEDUCTION, the value of the Nth variable. (This information is referred to as the System's TRACE and this facility as the HELP mode.) Clearly, the form of the TRACE will depend upon the algorithm being discussed and will usually be a series of interspersed FACTS and DEDUCTIONS. On the other hand, whatever the form of the TRACE it is possible that the HELP provided would not be at a sufficient level of detail and so the student would want to ask some subsidiary questions about the various statements. This can be viewed as the student asking for more detailed information about one of the processes. Ideally, the student would be able to further query this second-level information and so on until all his queries are answered.

Essentially, this facility has been designed to help the student determine the short-comings in his understanding of the algorithm (i.e. to help him 'debug' his algorithm). However, we can attempt to do this more directly by getting him to outline the essential features of the algorithm (or section of the algorithm) and for the system to comment on the student's explanation. There are a number of results which indicate that this is a powerful teaching approach (for instance see Pask [3]). Further, designing such a system is a challenging problem from the point of view of AI/ Computer Science. Looking again at the problem of Following the Student's Reasoning from the perspective introduced when discussing the HELP mode, we could say that the objective is to infer from the student's response a series of FACTs and DEDUCTIONS and to compare these with the TRACE of the algorithm when it attempts to solve the same problem.

Although we are talking about the discussion of algorithms in general, we will take examples from a system which helps students interpret NMR (Nuclear Magnetic Resonance) spectra. The student is presented with a spectrum and a molecular formula, and he has to postulate a maolecular structure which explains the spectrum given the constraints of the molecular formula and the various rules which apply in this physical domain. It is usual to consider the interpretation of NMR spectra as P sub-problems, namely the specification of P groups which constitute the molecule. A distinctive feature of the interpretation procedure is that it is not always possible to specify a unique solution for a particular sub-problem. In these cases one could select a 'solution' to a sub-problem, and proceed until the problem is solved or one realizes that a basic constraint has been violated: after a violation, one backtracks and selects an alternative solution to a sub-problem. Essentially, we are suggesting that a depth-first search is a 'natural' way to solve these tasks [4].

In order to discover the difficulties which students experience when they first encounter this task, we asked a dozen or so such students to work a series of problems recording all the stages of their solutions. On analysing these protocols we noted that the students were often unaware that a solution to a sub-problem violated a constraint and so they continued solving the problem after they had given an.unacceptable solution to a sub-problem. Secondly, it was noted that many students were unable to handle the backtracking inherent in the search and that they frequently omitted to return the 'resources' used in the steps through which they had backtracked. As a result of this analysis we implemented a system which constrains the student to follow a consistent strategy (namely depth-first search), which tells him whether his solution to a sub-problem is acceptable and provides him with a number of SOCRATIC facilities. (We refer to the student's solution to a sub-problem as an assertion.) The SOCRATIC facilities includes the HELP facility, which as indicated above provides the student with a hint as to how to proceed in solving the next sub-problem. The EXPLAIN facility, which is only available after an incorrect assertion, points out why the last assertion is unacceptable. This explanation may merely involve pointing out the rule which has been violated or it may be necessary to demonstrate that given the last assertion it is not possible to find a solution which is consistent with the data and which does not violate a physical rule. (To reach this conclusion it is necessary to produce all possible solutions to the problem and hence this facility needs access to the algorithm, BUILD, which can perform an exhaustive search of the problem space.) The reader is referred to [4] for further discussion of these facilities and for an extensive system's protocol.

2. The Extended HELP Facility

In order to implement both the extended HELP and the FSR facilities we defined a FL (Formal Language) which describes the major steps in the algorithm (including the topics which we wish the student to discuss). Thus in this formulation, giving the student HELP consists of executing the algorithm, BUILD, in a special mode which creates a TRACE: the TRACE is then returned to the HELP process which in turn calls a further algorithm that converts the FL statements into reasonable NL (Natural Language). The FL to NL conversion algorithm is discussed by Hendley[5]. Figure 1 gives a protocol for the HELP facility showing the basic HELP and the subsidiary queries which can be answered by the system: Figure 2 shows the FL statements which correspond to these various TRACEs. In this system the FL statements have been embedded in the algorithm and these statements, together with appropriate values are appended to the TRACE when certain conditions are satisfied. (Essentially a parameter indicates which, if any, of the TRACEs is to be created.)

If we now interpret Figures 1 and 2 in terms of the more general schema given in Section 1, we see that here we have three processes. P1 corresponds to a process which determines the group's valency, P2 determines the group's interaction and P3 determines the selection of the group. Thus Dialogue (a) of Figure 1 corresponds to the 'result' of PI, the 'result' of P2 and the consequent decision made by P3. Dialogue (b) shows in more detail how P1 calculates its result: similarly, Dialogue (c) gives a more detailed explanation for the result of P2. Finally, Dialogue (d) provides more details about how P3 reached its result (this process makes use of the results from P1 and P2 and 'globally' available information). These dialogues also bring out two further points which were not mentioned in our initial discussion in Section 1. Firstly, the dialogues show how the system is able to cope with decision making, i.e. how it handles conditional statements. We see in these protocols that the conditional statement is presented as a FACT, the relevant variable values are given as a series of FACTs followed by the DEDUCTION. (For example, see Dialogues (b) and (c) of Figure 1.) Secondly, these same dialogues show that we are dealing with two types of FACTs, namely equations and particular values of variables. So far we have only allowed the student to initiate more detailed discussion

a)
*HELP PLEASE
PARSES TO:(HELP)
THE VALENCY OF THIS GROUP IS 1 OR 2
AND THE INTERACTION OF THIS GROUP IS 2.
SO THE LIST OF POSSIBLE GROUPS IS (C H 2).

b)
INPUT YOUR OPTION.-VALENCY
IF THE LENGTH OF THE CHAIN IS 0
THEN THE VALENCY OF THIS GROUP IS 1
OTHERWISE THE VALENCY OF THIS GROUP IS 1 OR 2.
THE MOLECULE SO FAR IS ((C H 3)).
SO THE LENGTH OF THE CHAIN IS NOT 0.
SO THE VALENCY OF THIS GROUP IS 1 OR 2.

c)
INPUT YOUR OPTION:INTERACTION
IF THE INTERACTION OF THE LAST GROUP IS 0
THEN THE INTERACTION OF THIS GROUP IS UNKNOWN
OTHERWISE THE INTERACTION OF THIS GROUP IS THE
SPLITTING OF THE LAST GROUP - THE INTERACTION
OF THE PREVIOUS GROUP.
THE INTERACTION OF THE LAST GROUP IS NOT 0.
AND THE SPLITTING OF THE LAST GROUP IS 2.
AND THE INTERACTION OF THE PREVIOUS GROUP IS 0.
SO THE INTERACTION OF THIS GROUP IS 2.

d)
INPUT YOUR OPTION:SELECTION
THE REMAINING MOLECULAR FORMULA IS C 3 H 5 0.
SO THE LIST OF POSSIBLE GROUPS IS (C H 3) (0 H)
(C H 2) (C H 0) (C 0) (0).
THE LIST OF REMAINING SPECTRA IS (2 3) (3 0).
SO THE LIST OF POSSIBLE GROUPS IS (C H 3) (C H 2)
(C 0) (0).
THE VALENCY OF THIS GROUP IS 1 OR 2.
SO THE LIST OF POSSIBLE GROUPS IS (C H 3) (C H 2)
(C 0) (0).
THE INTERACTION OF THIS GROUP IS 2.
SO THE LIST OF POSSIBLE GROUPS IS (C H 2).

INPUT YOUR OPTION:CONTINUE

Figure_1_

A system protocol showing
the extended HELP facility.
(The student's responses are underlined)

a)
F V[TG] = 1 V 2
F I[TG] = 2
T PG = ((C H 2))

b)
F IF LENGTH(MOD=0 THEN v[TG] = 1 ELSE
V[TG] 1 v 2
F MOL = ((C H 3))
T LENGTH(MOL) # 0
T V[TG] = 1 V 2

c)
F IF I[LG] = 0 THEN I[TG] =U ELSE
I[TG] =S[LG]-I[PGI

F I[LG] # 0
F SLLG] = 2
F I[PG] = 0
T I[TG] = 2

d)
F RMFR = (C 3 H 5 0)
T PG = ((C H 3) (C H 2) (C 0) (0) (0 H) (C H 0))

F RSPC = ((2 3) (3 0))
T PG = ((C H 3) (C H 2) (C 0) (0))

F V[TG] = 1 V 2
T PG - ((C H 3) (C H 2) (C 0) (0))

F itTG] = 2
T PG = ((C H 2))

Figure_2

Formal Language statements, produced by
the BUILD algorithm in response to the
various requests for HELP in Figure 1.

a) Corresponds to a request for basic HELP
b) To a discussion of the group's valency
c) To a discussion of the group's interaction
and
d) To a selection of a group.

about DEDUCTIONS and variable values (i.e. it is not possible for a student to ask for an explanation of an equation).

3. Following the Student's Reasoning

So far three sections of the algorithm have been considered in this way: the splitting of a peak, the selection of a group and the justification of an assertion (i.e. a solution to a subproblem). These three tasks were selected because they were considered to be sufficiently different from each other, but to have similarities with other tasks which exist in the domain. However, only one of these tasks, the splitting of a peak, is discussed here.

(a) The splitting of this group is greater or equal to the interaction of the last group. The interaction of the last group equals 0 therefore the splitting of this group is greater than or equal to 0.

(b) The splitting of this group is greater or equal to 0 because the splitting of this group is greater or equal to the interaction of the last qroup and because the interaction of the last group is equal to 0.

(c) The splitting of this group is greater or equal to 0 because the interaction of the last group equals 0.

(d) The splitting of this group is 0 because the splitting of this group is equal to the interaction of the last group and the interaction of the last group is equal to 0.

Figure 3
Gives protocols for a student reasoning about peak splitting

3.1 FSR = SPL: Discussing the Splitting of a Peak

In Figure 3, we see protocols of students' reasoning about the splitting of a peak. These protocols were collected by modifying our original system so that the student was asked to type in an explanation for the value he had given for the splitting before the system told him whether or not his assertion was correct. These responses were then analysed 'manually' and we subsequently designed and implemented the FSR mode which comments on such explanations. In this mode the algorithm would return the following TRACE:

```
F    S[TG] >= I [LG]
F    I[LG]  = n
T    S[TG] >= n
```

that is the TRACE contains:

(i) a basic equation which holds for the problem-domain, FACTI.
(ii) a value for a particular variable, FACT2.
(iii) a deduction, D.

If we look again at the dialogues given in Figure 3 we see that:

argument (a) is essentially FACTI, FACT2 and D.
argument (b) is essentially D, FACTI and FACT2.
argument (c) is D and FACT2 (i.e. FACTI is omitted).
argument (d) is essentially an incorrect form of FACTI, FACT2 and a consistent, and hence incorrect deduction, D'.

For the moment, we have implemented a comparison algorithm for each of the tasks which can be considered by the FSR facility. The comparison algorithm for this task should obviously accept 'inverted' arguments such as Dialogue (b) of Fig.3. (In our present system the 'standardisation' of the order of the FL statements is performed by the NL to FL conversion algorithm [5]). On the other hand, the comparison algorithm should point out the inaccuracies of information given in arguments such as (d) and possibly the incompleteness of arguments such as (c). The comparison algorithm for this task makes the following 'tests' on the student's TRACE:

FACTI
(i) Checks that this argument includes the splitting of this group,
(ii) If necessary inverts the fact so that st [TG] is the 'subject'
eg. F I[LG] <= S[TG] is transformed to F StTG] >= ILLG].
(iii) If I[LG] is not mentioned in the (transformed) FACTI, then the system checks whether the item which is mentioned can be transformed into it. (Note, given this domain the Area of peak, the number of Hydrogens on a group and the Interaction of a group are numerically equal.)
(iv) Checks the relationship is >= .

FACT2 (This may be omitted: whether the complete argument is correct or incorrect this omission is pointed out to the user.)
(i) Checks to see whether I[LG], or its equivalent, is present,
(ii) Checks that a numerical value is given,
(iii) Checks the numerical value against that given in the TRACE.

DEDUCTION
(i) Checks that the splitting of this group, StTG] is the 'subject' of the conclusion.
(ii) Checks the correct relationship, in this case >= is given.
(iii) Checks that a numerical value is given.
(iv) Checks the numerical value against that given in the TRACE.

The protocol given in Figure 4 illustrates some of the features of the comparison algorithm discussed above. Dialogue (a) merely illustrates the ability of the NL to FL interface to cope with arguments where the DEDUCTION precedes the FACTs. In Dialogue (b) the system transforms FACTI into standard format and further checks that the area of the last group is related to the interaction of the last group: finally the system points out that

the value deduced for the splitting is inconsistent with the value given earlier. Dialogue (c) illustrates that the comparison algorithm rejects arguments which do not contain the (basic) FACT1. Having some simple criteria for rejecting completely irrelevant arguments is obviously very helpful as it saves a great deal of unnecessary analysis. On the other hand, one might argue with considerable justification that the comparison algorithm should accept the argument FACT2 and DEDUCTION. We shall discuss in Section 4 how this comparison algorithm could be made more powerful and more general purpose.

The reader will have noticed in Figure 4 that after the student has typed his arguments the system responds by typing a 'paraphrase' and asks the user whether it is acceptable. This 'paraphrase' is the NL equivalent of the FL statements which the system has extracted from the student's response. (In fact, the FL to NL algorithm used here is the same as that used by the HELP facility.) The reason for presenting the student with the •paraphrase' is that in the case of his input being mishandled by the system, this feature allows the student to prevent the analysis of an erroneous argument. Secondly, it was felt that the 'paraphrase' might help the student restate arguments which were mishandled by the system, and thirdly it was a very useful diagnostic during the debugging phase.

3.2 Natural Language to Formal Language Conversion Algorithm

At an early stage, we realized that it was not feasible for students to express their arguments in a FL as this was likely to interfere appreciably with their problem solving and distort their explanations. Hence it was decided that some form of NL interface was necessary. It was clear that keyword matching techniques would be inadequate for analysing students' arguments, but we were able to classify the topics encountered in these arguments as a series of semantic entities. Essentially this situation resembles that encountered in the SOPHIE System[6], and so we also implemented a Fuzzy Semantically driven parser. The grammar which our parser uses is the FL for the section of the algorithm which we expect the student to discuss. Nevertheless, the parser can accept sentences which are both irrelevant and which contain inappropriate relationships/values (eg. see Dia logue (c) of Figure 4). Similarly, this parser would accept the sentence:

The interaction of the last group is CH3.

Comments on a student's arguments are made by routines which have more specialised knowledge of the domain, namely the comparison algorithms. For example, given the above sentence this comparison algorithm would point out that a numerical value is expected. (The NL-FL programs used in this system are discussed in [5] and will be discussed more fully in a separate paper.)

4. Possible Enhancements to the Current System

We shall discuss three enhancements: the first applies to the extended HELP facility and the others to the FSR facility. Firstly, there are still many questions which a student may wish to ask about the algorithm which the system is unable to answer. The system could further use the TRACES which are already produced to answer questions such as:

Why was the CH3 group eliminated?

The answer to this question could be based upon the FACT which occurred in the TRACE immediately before this group was eliminated from the list of possible groups (see Figure 2(d)). And so in this case the system might respond:

Because the interaction of the group being considered is 2.

Further, using the FACTs in the TRACE it would be possible to answer questions such as:

What is the interaction of the CH2 group?

Analagous questions about the last or previous groups could be answered by accessing the relevant TRACE.

Secondly, the system could profitably be enhanced to tell the user whether or not his argument is consistent with an aspect of the assertion he has given. (In the case of the FSR facility discussed in Section 3.1, the system would compare the values for the splitting of the peak given in the student's assertion and in his subsequent explanation.) There are five cases for consideration:

(i) Correct assertion and correct explanation,
(ii) Correct assertion but incorrect explanation,
(iii) Incorrect assertion and correct, and hence incompatible, explanation,
(iv) Incorrect assertion and compatible explanation,
(v) Incorrect assertion and a further incorrect, but incompatible, explanation.

The third enhancement concerns the-implementation of a more general purpose comparison algorithm. It was pointed out in Section 3 that these particular tasks had been chosen for discussion because it was felt that they were fairly different from each other and yet representative of a number of other tasks which form part of the overall, BUILD algorithm. In Section 3.1 you will see that we categorize the three statements which form the TRACE for the task discussed as:

(i) The basic equation which holds for the problem domain,
(ii) The value for particular variable(s).
(iii) The deduction of a value of a variable using (i) and (ii).

The algorithms for determining the valency and the interaction of a group are analogous to that for determining the splitting of a peak as can be seen from the system protocol in Figure 1 and from TRACEs given in Figures 2(b) and 2(c). Thus there is a strong incentive to produce a more general

(a) PLEASE INPUT YOUR STMTS
:THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO 0 BECAUSE THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO THE INTERACTION OF THE LAST GROUP AND BECAUSE THE INTERACTION OF THE LAST GROUP IS 0

//THE 'MEANING'EXTRACTED FROM THE STUDENT'S ARGUMENT
THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO THE INTERACTION OF THE LAST GROUP.
AND THE INTERACTION OF THE LAST GROUP IS (0).
SO THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO (0).
IS THIS WHAT YOU INTENDED:YES

WE ARE CONSIDERING:
THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO THE INTERACTION OF THE LAST GROUP.
THIS STAGE IS FINE

WE ARE CONSIDERING:
THE INTERACTION OF THE LAST GROUP IS (0).
THIS STAGE IS CORRECT

WE ARE NOW CONSIDERING:
SO THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO (0).
GOOD YOUR ARG. IS ABSOLUTELY CORRECT

(b) PLEASE INPUT YOUR STMTS
:THE AREA OF THE LAST GROUP IS LESS THAN OR EQUAL TO THE SPLITTING OF THIS GROUP AND AS THE AREA OF THE LAST GROUP IS 0 THEN THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO 1

//THE'MEANING'EXTRACTED FROM THE STUDENT'S ARGUMENT
THE AREA OF THE LAST GROUP IS LESS THAN OR EQUAL TO THE SPLITTING OF THIS GROUP.
AND THE AREA OF THE LAST GROUP IS (0).
SO THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO (1).
ISL THIS WHAT YOU INTENDED:YES

WE ARE CONSIDERING:
THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO THE AREA OF THE LAST GROUP
UNEXPECTED TERM: YOU GAVE THE AREA OF THE LAST GROUP - I EXPECTED THE INTERACTION OF THE LAST GROUP
IHEREIS A TRANSFORMATION AND SO YOUR ARG. MAY BE ACCEPTABLE
THIS STAGE IS FINE

WE ARE NOW CONSIDERING:
THE AREA OF THE LAST GROUP IS (0).
THIS STAGE IS CORRECT

WE ARE NOW CONSIDERING:
SO THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO (1)
I EXPECTED A NUMERICAL VALUE IN THIS EXPRESSION CONSISTENT WITH YOUR EARLIER VALUE OF 0

(c) PLEASE INPUT YOUR STMTS
:THE VALENCY OF THIS GROUP IS 1 AND SO THE POSSIBLE GROUPS ARE CH3 CH2 CHO OH

//THE 'MEANING' EXTRACTED FROM THE STUDENT'S ARGUMENT
THE VALENCY OF THIS GROUP IS (1).
SO THE LIST OF POSSIBLE GROUPS IS ((C H 3) (C H 2) (C HO) (O H)).
IS THIS WHAT YOU INTENDED:YES

BASIC FACT OMITTED-FORM OF FACT EXPECTED:
THE SPLITTING OF THIS GROUP IS GREATER THAN OR EQUAL TO THE INTERACTION OF THE LAST GROUP.

system. Indeed, we propose implementing a sub-system to monitor the student's performance as he attempts to deduce the value of the variable. The system will comment when it thinks the student has introduced a circular argument or a totally irrelevant equation (or fact). On the other hand, when asked the system should be able to give advice about possible paths which the student has not explored. It seems fairly clear that such a system will be able to replicate the type of dialogue shown in Figure 4. Indeed, such a system should be more powerful as it has the potential to generate all possible solution-paths as opposed to the TRACE-based system which has access to the subset given in the TRACE(s). Further, we propose using this sub-system as a component of the enhanced FSR mode.

5. Some Thoughts About the Design of a Further System

A problem which merits further attention is that of producing a single data structure for representing algorithms which can be both executed and used to give an explanation of the algorithm. The conditional statements in the FL are very analogous to conditional statements in some programming languages (including BCPL, see Richards[7], the programming language used to implement the current system). Thus in response to the question about the valency of the group the system produced the FL statements shown in Figure 2(b). An inspection of the relevant part of the BCPL code which determines the valency of groups, shows that this algorithm is indeed very analogous to these FL statements. If such a unified representation were available for both execution and discussion then the student would be able, at least in principle, to ask the system for an explanation of any task which the algorithm can perform and similarly, the system could also ask the student to discuss any part of the algorithm. (As opposed to the current situation when we must decide in advance that certain aspects of the algorithm may be discussed and include appropriate FL statements in the algorithm.)

For the moment, let us assume that the algorithm is represented in a unified data structure, very probably a LISP-like structure. Secondly, we argue that this data-structure needs modifying to include relevancy tags in order to be usable with the HELP and FSR modes: and thirdly that the system will need an index to relate discussable topics to processes. (Given this extended data-structure it is clear that an interpreter will be needed in order to execute the algorithm.) In discussing the extended HELP facility earlier, we noted that the

Figure_4 (See opp.)

Shows two systems dialogues with the FSR-SPL mode. The student's responses are again underlined and the system prints a 'paraphrase' of what the student has typed in.

initial HELP gave the student a summary of the three processes involved and that as a result of subsidiary questions the student was presented with more details about the processes. This can be conceptualized as a particular data-element in a process having the highest relevance and that further discussion of a process reveals statements with the next level of relevance and so on. Similarly, when the FSR mode attempts to follow the student's argument it will have access to a data-base of 'highly relevant* equations and variables. If this initial search is unsuccessful, then the search will be continued with further equations and variables which are judged to be the next most relevant (hopefully this choice will be guided by the type of mis-match recorded in the earlier search as well as by the relevance tags in the data-structure). Ideally, such a facility would be able to:

(i) Comment on solutions which contain irrelevant and unnecessary steps (the comparison algorithm proposed in Section 4 will be able to do this to a limited extent).
(ii) Cope with arguments at various levels of detail (i.e. a 'plan' at the 'top' level or a more detailed 'plan' or a mixture of levels) .
(iii) Cope with changes of attention.
(iv) Decide whether or not the student is discussing the appropriate topic.

Further, the current system is unable to answer questions such as:

What is the valency of Carbon?

despite the fact that the system has this knowledge. Considering these various points it would seem that any future system should have the following data-bases :

(i) Global information (such as the valency of atoms and groups).
(ii) The algorithms (to be represented in the •unified' format).
(iii) Problem dependent information i.e. TRACES which correspond to each stage of the problem's solution.

6. Footnote

Readers who are familiar with SRI's Computer Based Consultant project [8] will have noted that both projects have similar objectives. Namely, to provide a supportive problem solving environment: in particular to provide advice when the user is unable to solve the next stage of a problem and to analyse the user's response in detail when he has made an error. The heart of SRI's Consultant is Sacerdoti's plan generator [9] which represents a plan as a procedural net, and uses Semantic knowledge about the user's problem to expand the net as necessary. Despite the very different representations used for algorithms in the two systems,they provide analogous facilities in similar ways. (For example, the plan expansion of SRI's system is paralleled by the relevancy levels in the system outlined above.) However, to date the SRI group has not reported work where their system comments directly on a student's explanation of an action. But if such a facility were to be attempted, a possible approach would be the analogue of the one discussed in this paper; namely the comparison of two procedural nets, one produced by the system and the other extracted from the student's explanation. (A more detailed comparison between the two systems will be included in a subsequent paper which reports progress on the system outlined in the last section.)

Finally, there are two crucial issues which have to be thoroughly investigated with both these systems; namely the level of support which the systems provide to their users, and secondly the range of algorithms which can be 'discussed' using these systems.

Acknowledgements

This work was started whilst I was associated with Professor R F Simmons' research group in Austin, Texas, and was partly supported by grant number NSF 509X. The facilities afforded by the CBL Project's Modular One system enabled me to continue this work. In particular I gratefully acknowledge the use of List Processing library routines implemented in BCPL by J A Self and A J Cole. Further, I am very grateful to Austin Tate (Edinburgh) for discussions about the planning aspects of this work and to R J Hendley who programmed the Natural Language handling routines.

References

1. UHR, L. Teaching Machine Programs that Generate problems as a function of interaction with Students, Proc. 24th Nat. Conf. (1969), 125-134.
2. PEPLINSKI, C. A Generative CAI system that teaches Algebra. Tech. Rep. 90, Computer Science Dept., Univ. of Madison, USA, (1970).
3. PASK, G. TEACHBACK in Cybernetics of Human Learning and Performance. (1975). Published by Hutchinson.
4. SLEEMAN, D H. A problem-solving monitor for a Deductive Reasoning Task. Int.J.Man-Machine Studies, 7, (1975), 183-211.
5. HENDLEY, R J. A Natural Language interface for an existing Problem Solving Monitor, BSc Dissertation, Centre for Computer Studies, Univ. of Leeds, UK, (1976).
6. BROWN, J S, BURTON, R R, and BELL, A G. SOPHIE: A Sophisticated Instructional Environment for teaching electronic troubleshooting (an example of AI in CAI), Bolt Beranek and Newman (Cambridge, Mass.), Report No.2790, (1975).
7. RICHARDS, M. BCPL Reference Manual. Technical Memo 69/1, Computer Laboratory, Cambridge, UK, (1969).
8. HART, P E. Progress on a Computer Based Consultant, Adv. Papers of IJCAI4, (1975), 831-841.
9. SACERDOTI, E D. The Non-linear nature of plans, Adv. Papers of IJCAI4, (1975), 206-214.