ILLINOIS

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

PRODUCTION NOTE

University of Illinois at
Urbana-Champaign Library
Large-scale Digitization Project, 2007.

Technical Report No. 376

THE COMPUTER AS A TOOL FOR LEARNING THROUGH REFLECTION

Allan Collins
Bolt Beranek and Newman Inc.

John Seely Brown
Xerox, Palo Alto Research Center

March 1986

Center for the Study of Reading

TECHNICAL REPORTS

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
174 Children's Research Center
51 Gerty Drive
Champaign, Illinois 61820

BOLT BERANEK AND NEWMAN INC.
10 Moulton Street
Cambridge, Massachusetts 02238

The National Institute of Education
U.S. Department of Education
Washington, D.C. 20208

CENTER FOR THE STUDY OF READING

Technical Report No. 376

THE COMPUTER AS A TOOL FOR LEARNING THROUGH REFLECTION

Allan Collins
Bolt Beranek and Newman Inc.

John Seely Brown
Xerox, Palo Alto Research Center

March 1986

University of Illinois
at Urbana-Champaign
51 Gerty Drive
Champaign, Illinois 61820

Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, Massachusetts 02238

The work upon which this publication is based was performed pursuant to Contract No. 400-81-0030 of the National Institute of Education. It does not, however, necessarily reflect the views of this agency. To appear in H. Mandl and A. Lesgold (Eds.), Learning issues for intelligent tutoring systems. New York: Springer, in press.

The Computer as a Tool for Learning through Reflection

A unique aspect of computers is that they not only represent process, but they also naturally keep track of the actions used to carry out a given task, so that the process and its trace can become an object of study in its own right. One effect of this can be seen vividly in the sciences where computers and computational languages have improved our ability to develop and test process theories of complex natural phenomena. Before powerful computers became readily available as scientific tools, process models were expressed in mathematical languages, such as differential equations--languages primarily effective in capturing a static snapshop of a process. Computation provided formal languages that are more flexible than mathematics, but just as precise. In part because computation is itself dynamic, it provides an ideal medium for representing and testing richer, more varied, and more detailed theories of process. The use of this medium for process modelling has radically changed the nature of many current theories in both the physical and social sciences. Particularly in the arena of the cognitive sciences, computational techniques have proved to be powerful tools for both experimental and theoretical investigations of mind.

The computational revolution in the sciences has a parallel in education. With a computational medium it becomes possible, and often easy, to capture directly the processes by which a novice or an expert carries out a complex task. Properly

abstracted and structured, this process trace or audit trail can become a useful object of study for students who are trying to learn how to improve their performance on a task. By comparing the details and structure of their own performance with that of more expert performers, they can discover elements that need improving. In a sense, the expert's audit trail provides an accessible example of the situated use of general reasoning strategies. Likewise, an audit trail of their own performance provides an object of study from which students can hone important self-monitoring and other metacognitive strategies.

It is because of its ability to record and represent process that we conjecture that the computer can become a powerful tool for learning through reflection, a new form of intellectual bootstrapping. We suggest that the revolution in discovery learning heralded by Logo (Papert, 1980) will not fully materialize, unless there is a way for students to study and explore their own problem-solving efforts. The students' problem-solving processes--their thrashings, false starts and restarts, and partial successes--should not be left implicit. A major value in solving problems occurs when students step back and reflect on how they actually solved the problem and how the particular set of strategies they used were suboptimal and might be improved. Of course, this ideal scenario seldom transpires, in part because students are not really motivated to do so and in part because the current problem-solving medium (i.e., paper and

pencil) does not really lend itself to this activity. Our claim here is that the computational medium, properly structured, can provide a powerful, motivating, and as yet untapped tool for focusing the students' attention directly on their own thought processes.

This paper reports on several steps in the direction of reflective learning. We will begin by considering a familar skill, tennis, to illustrate the power and possibilities of reflective media for learning.

Types of Reflection

Let us consider the pedogogical strengths and weaknesses of different ways of representing a tennis swing and the different ways of reflecting on that representation:

Imitation. The tennis coach can imitate a student's swing, highlighting those aspects of the swing that are correct or incorrect, while verbally describing the crucial properties of the swing as it progresses. He can slow the swing down and even stop at critical moments. However, imitations have their limitations as a pedogogical device. For one, there are always distortions in any imitation and the student may focus on them as the relevant features. For another, from a model of a swing, the student cannot be sure how much or exactly how to correct a particular movement. Nor can the student easily engage in a fine-grain analysis of his own swing: He may miss critical

relationships that can only be seen in an abstracted replay or spatial reification.

Replay. Alternatively, the student's swing can be videotaped from different angles and replayed and discussed. The tape can be played as often as the student wants, sped up or slowed down, or stopped in critical places for detailed discussion with the coach. The replay is accurate in its reproduction of the student's behavior. It has high physical fidelity and captures not only the swing itself but also the follow-through, the angling of the ball off the strings of the racquet and so forth, so that the student sees the swing in context. Given split screen technologies, students can even compare themselves to video recordings of experts, and attempt to abstract how to alter their movements to better approximate the important aspects of the experts' swings.

The last notion highlights one of the fundamental limitations of exact replay for use in reflective learning. It is often difficult for students to know what to pay attention to unless a coach points out the important properties as they watch the replay. Indeed, without the student possessing a relevant set of distinctions about the process being observed, he is hard-pressed to meaningfully remember or compare his performance with that of the expert, nor can he readily modify his performance to bring about the desired effects once he knows what they are. However, there are ways to focus the student's attention and to

help set the state for their constructing a useful set of distinctions with which to observe and remember expert performance.

Abstracted replay. Suppose a reflective material is taped to critical points (e.g., the shoulder, elbow, wrist, handle, racquet head), and the motion of these different points recorded during the swing, perhaps from two angels (e.g, the side and the front). Such an abstracted replay attains both accuracy and the unambiguous highlighting of critical features, thus focusing the student's attention on the important parameters of the swing. Abstracted replay thus turns on the notion of "cognitive fidelity" rather than physical fidelity. This is especially crucial when there is too much data for the student to absorb in a full replay or imitation. The highlighting made possible through abstraction conveys information in a way that no verbal explanation can. Of course, if critical features (such as leg positions) are left out, information is lost to the student that is available in the full replay condition.

As with the replay condition, comparison of the student's swing with that of the expert depends on the student either remembering the expert's or using a side-by-side comparison with split screens. If a good abstraction can be constructed, it becomes possible to overlay the student's swing with a trajectory of an expert's swing.

Spatial reification. The trajectory of the critical points of a swing, say from the side angle or from other angles, can be plotted in a graph. This gives a static representation of the process unfolding in time that can be inspected and analyzed in detail. A spatial reification has many of the same properties as an abstracted replay, but because the dimension of time is now spatially represented, the student can analyze critical relationships over time more easily and can directly refer back to prior parts of the process. For example, the relative height of the racquet head at beginning, middle, and end of the swing can be easiy seen from the side plot. Students can directly compare their plot with a plot of expert performance without relying on memory. But again some critical features may be lost at the expense of others being reified. For example, the timing of the swing is only implicit in the above representation scheme.

As a general principle, multiple representations are helpful. Students should be able to inspect their performance in different ways, so it makes sense to provide them capabilities for seeing full replays, abstracted replays, or spatial reifications. A critical ingredient of the Reciprocal Teaching Method (Palincsar & Brown, 1984) is that the students are able to compare their performance with expert performance in terms of the difficulties they are currently having and the distinctions they currently hold. This suggest showing simpler abstractions of their performance at earlier stages of learning.

Ideally, a coach could diagnose where the student is having difficulty and abstract those elements critical to overcoming the difficulty. For example, a student who is dropping his racquet head might see a replay where the relative position of the wrist and racquet head is highlighted, whereas a student who is bending his elbow too much might see a replay that highlights the positions of the shoulder, elbow, and wrist. This linking of correction to diagnosis is what gives coaching in general and the Reciprocal Teaching Method in particular much of their leverage.

Reflection on the Process of Problem Solving

Two recently developed tutoring systems utilize reifications of the student's problem solving process as a major pedogogical device: Algebraland and Geometry Tutor.

Algebraland (Brown, 1985). Students are given algebraic expressions to solve for a particular variable; in Figure 1 they are to solve for N. They manipulate both sides of the equation by selecting an algebraic operator from the menu at the bottom right and a term in the equation in the record window on which the operator is to be applied. In Figure 1, the student first distributes 4 across (2 + N), and then divides both sides by 4. In a special search space window, the program automatically forms a tree that represents the various problem-solving steps, halts, and continuations that the student has thus far taken in attempting to solve the problem. If the student becomes stuck, he can return to an earlier node in the solution path by simply

pointing at it, and begin a new path that he hopes will lead to a solution. This branching process causes the resulting search space window to be a tree rather than just a single chain of nodes. The record window records each state (i.e., node) the student reached in the current solution path, and the algebraic operation that was used to move from one state to another in that chain.

Insert Figure 1 about here.

The tree in the search space window is a reification of the student's problem-solving process. Students can see exactly where they backed up, where they reached the same state twice, where they were getting farther away from a solution, and so on. The structured representation of partial solution paths provides an opportunity to reflect on problem-solving and evaluation strategies in the context of their use, a context that reveals where they worked well and where they may have led the student astray. For example, reflecting on a choice point where the branch (i.e., operator) first chosen proved to be a counter-productive, but where a different branch taken at that choice point (chosen at a later time) proved to be productive, provides grist for considering what features the decision process for that choice point should have focused on. That is, the student should ask himself what properties of the algebraic expression

comprising the node could have alerted him to a better strategic choice?

Countless learning activities can be constructed around this reified problem space. For example, a student or team can be asked to study another student's (or team's) problem space with the aim of finding a shorter solution path to the problem. Among other things this kind of exercise helps to make explicit that there is no single "right" solution; there are many solutions some of which are shorter and perhaps more elegant than others. Indeed, games can be constructed that turn on this simple idea. Alternatively, using a menu-based annotation editor, such as shown at the bottom left of Figure 1, a student might be asked to annotate the reasons why he made certain choices (see Bundy, 1983), a simple and rewarding excercise if the annotation menu has built into it strategic terms that can be readily selected and joined to the links in the reified problem space (personal communication, Carolyn Foss, a Stanford graduate student who is writing a thesis on the role of reflection in the development of metacognitive skill and impasse-driven learning). Finally, students can examine their own floundering in order to formulate self-monitoring strategies that would help to detect and prune non-productive approaches to similar problems.

Geometry Tutor (Anderson, Boyle, & Reiser, 1985). In another learning environment involving reflection, this one for learning the skill of doing proofs in geometry, students are

given a diagram of the problem at the top left of the screen and a set of "givens" at the bottom of the screen (see Figure 2). In this example, the goal is to prove the statement at the top of the screen. Students can work either forward from the givens (forward chaining) or backward from what is to be proved (backward chaining) as shown in the middle panel of the figure. The system alternates operators and states in the diagram it constructs. Again as seen in the bottom panel there is a trace of the problem solving process. Although it is impossible to tell the order of the steps taken, the student can see dead ends and look for other possible proofs.

Insert Figure 2 about here.

As Anderson, Boyle, Farrell, and Reiser (1984) point out, geometry proofs are usually presented in a fundamentally misleading way. Proofs on paper appear to be linear structures that start from a set of givens and proceed step by step (with a justification for each step) to the statement to be proved. But this is not at all how proofs are constructed by mathematicians or by anybody else. The process of constructing proofs involves an interplay between forward chaining from the givens and backward chaining from the goal statement. Yet, the use of paper and its properties encourage students to write proofs as if they were produced only by forward chaining--starting with the givens

at the top of the page and working downward to the goal in a two column linear format (left column for the derived statements, right column for the logical justifications). If students infer that they should construct proofs this way, they will fail at any long proof. Properly designed computational learning environments can encourage students to proceed in both directions, moving forward, exploring the givens, and moving backwards, finding bridges to the goals.

The representations in Algebraland and Geometry Tutor are abstractions of the problem solving process in terms of "problem spaces." Both systems show the states in the problem space that the student reached and the operators used to reach each of those states. Simply seeing the steps toward a solution reified in this way helps to create a problem space as a mental entity in its own right. This, in turn, makes it possible, for both teachers and students, to characterize problem-solving strategies in terms of abstractions that refer to properties concretely manifested in the refied problem space. For example, in geometry it is a good strategy to forward chain at the beginning of a problem in order to understand the implications of the givens. Similarly, if you are stuck in backward chaining, and do not see a way to connect your backward chain to any of the givens, then either go back to forward chaining or go back to the goal state again and try backward chaining along a different path.

These problem solving strategies are what are known as "metacognitive" strategies (Flavell, 1976; Brown, 1978); students must learn them if they are to control their problem solving processes. Metacognitive strategies are what people use to detect and control "floundering," i.e., moving through the problem space without getting closer to the goal. Figure 3 shows the problem space of one of Foss's subjects floundering while using Algebraland. The problem was to solve the equation for V. When the student first got to the state 1/V=1/F-1/U, he tried a whole series of different operations (e.g., multiplying by 1, dividing by 1, subtracting 1, etc.). In that sequence he even tried the operation that eventually led to success (i.e., mutiplying by V), but he failed to see that this step was a good one. The student was obviously floundering at the time: He was just trying operations without any clear plan and without considering where they might lead. As a result he was carrying out operations without apparently getting closer to the goal. Suddenly however, he started over and solved the problem systematically as seen in the window on the right hand side of the screen image.

Insert Figure 3 about here.

Anderson, Boyle, Farrell, and Reiser (1984) argue that the system should prevent students from going off the optimal solution path so that they never flounder. They argue that

floundering leads to confusion, waste of valuable time, and loss of motivation. In contrast we argue that unless students flounder they won't ever have the opportunity to learn the kinds of metacognitive strategies suggested above. We need to create environments where students can flounder and where the system helps them profit from this floundering by making it explicit and, if need be, by having coaching systems highlight the floundering and help them discover or understand better metacognitive strategies grounded on their particular experience.

Perhaps a mixed pedagogical strategy would be ideal: When students are learning the use and meaning of basic domain operators for moving through a problem space, the system should prevent students from floundering. In this way, their time is being solely focused on mastering the basic tools of the trade. As students begin to tackle real problems, they need the elbow room to explore nooks and crannies of the problem space in order to gain insights into what makes a theorem true or a problem solvable. But during this phase, the system should attempt to provide students guidance on how to examine their own floundering, helping them to detect inherently useless exploration. In this way learning moves naturally from domain skills to metacognitive strategies.

Reflection on the Process of Writing

We can illustrate the educational potential of reflection on the writing process in the context of the NoteCards system

developed by Frank Halasz and Tom Moran (Brown, 1985). The NoteCards system is a multi-windowed authoring system based on the metaphor of the small notecards that writers sometimes use to capture, organize, and reorganize their thoughts. NoteCards allows a writer to create notes including text and sketches on a topic they plan to write about. These notes can be indexed however the writer wants by "filing" them in "fileboxes" by source, topic, etc. The writer can also create labeled links between notes that characterize the relationships between the ideas, e.g., comments, contradictions, elaborations, and so forth. The notes and their linkages to fileboxes or other notecards can be viewed in a link-icon browser, exemplified in Figure 4, using link-type selection as a mechanism for filtering the information in the notefile. Thus one might want to see only the cards that deal with the main thesis of the paper. Or one might want to view all the contradictions and support links for a given piece of text. The writer can also create an outline structure of the text and insert links to notes into it. Link icons that represent notecards can be moved freely around in the browser or in an outline allowing either local or global restructuring of the ideas for the paper.

Insert Figure 4 about here.

While the initial NoteCards system was under development a history graduate student used the system to write a paper on the deployment of NATO missiles in Western Europe. He read a number of documents and made notes on them in the system. After he had written about thirty notes and filed them in a topic hierarchy, he created a browser which reflected the structure of his initial thinking (see Figure 4). As he created more notes he changed the structure of the browser several different times. When he had written about 500 notes, he decided he was ready to start writing. He created a text outline for the paper and inserted footnote links to particular notes. He then rewrote each note, inserting it as text into the outline, adding bridging sentences and paragraphs as necessary. As he worked, he added new topics and subtopics to his outline. He proceeded in this way until he produced a complete draft.

It is now possible to look at the various structures he created while organizing and writing the paper (i.e., the notes, the various browsers, the outline). By adding a tracing program to the system, it would be possible to replay the actual process by which the paper was constructed, reflecting his strategies for producing a complex text based on many different sources.

People's strategies for writing vary widely. Some writers start with an outline and then produce notes or text to fill out the outline. Bereiter and Scardamalia (1985) argue that children tend to use a "knowledge telling" strategy, in which they write

the first thing they think of as the first sentence of a text, then the second thing they think of, and so on. More experienced writers tend to separate idea generation (e.g., producing notes) from actually writing text (Flower & Hayes, 1980), as did the graduate student in the study. While no one strategy is "correct," some are decidedly more effective than others.

The capability to record and replay the various notes, outlines, and pieces of text that students produce provides a new way for students to think about the process of writing. They might be able to look at the process by which different people produced articles in similar genres. Perhaps students might have access to models of how some classic texts of the future (i.e., by a future Shakespeare or Marx) were constructed using a system like NoteCards. Students could then systematically compare their writing process to a variety of different writers.

This possibility raises the issue of separating out for replay the critical aspects of the writing process. Students are not likely to spend the time to replay the entire process by which a text was produced, unless it is a short text. Instead they will want to see an abstracted replay or reification that highlights parts of the process.

The right set of abstractions (like the problem space abstraction in mathematical problem solving) is needed to characterize the writing process.[1] Then students could observe and analyze abstracted replays of the writing process as

practiced by themselves, other students, and more expert writers. An abstracted replay might use notes, outlines, browsers, and paragraph headings as elements in conjunction with operators such as rearrangement, deletion, and annotation as the level of process representation that students observe.

Reflection on the Process of Reading

Reading is a very difficult task in which to apply reflection, because the process goes by very quickly. In spite of this, we would like to sketch the design of a system to tutor reading in which the kind of reflection we have described might be embedded, in order to show the range and power of this technique.

Researchers have proposed a number of methods for teaching reading that employ expert modelling as a component (Bereiter & Bird, 1985; Collins & Smith, 1982; Palincsar & Brown, 1984). Collins and Smith, for example, proposed that the teacher read aloud for the student in one voice while verbalizing her own thoughts about the passage in another voice. This technique results in something like a slow motion movie of the reading comprehension process. The teacher verbalizes many different kinds of thoughts: confusions over particular phrases, hypotheses about what a passage means, predictions about what will come later, summaries of what the text says, descriptions of ideas provoked by the text, guesses about the author's intentions, evaluations of the writing, and reevaluations of any

of the above as they occur. In short, the goal of expert modelling in this proposal is to verbalize all the thoughts a skilled reader might have while reading.

There have also been several attempts in recent years to build computer-based systems that help people to learn to read (Collins, 1985). One class of systems provides interactive help to novice readers as they read texts: for example, systems that will pronounce any word or sentence that the reader indicates by pointing to it on the screen. We imagine extending systems like this so that the student tries to read the passage aloud. His reading is tape recorded and can be played back at any time. In addition the student would have access to tapes of well known people with different accents and backgrounds (e.g., Vanessa Redgrave, Martin Luther King, and Ricardo Montalban). Thus students can compare how they read the passage to how more expert readers read the passage. Such a system might also ask questions at critical junctures in the student's reading to see what hypotheses, evaluations, and so on he had formed as an active problem-solver trying to comprehend the passage.

In the Stone Soup fable by Aesop shown in Figure 5, we have indicated questions that might be interjected while the student reads, as well as answers an expert might give to each question. In our proposed design, the system would verbally ask the reader each question when they had finished reading the prior sentence. The answer would be recorded. The student then could ask to hear

answers to the same question by the same experts who were recorded reading the passage. At any time students could go back and replay either their own tapes or the expert tapes, and even rerecord themselves for a second try.

Insert Figure 5 about here.

One of the goals in this system design is to make direct comparison possible between what the student and the expert produce in the same situation. Thus the student sees how an expert deals with the same problem he has just tried to solve. Brown and Palincsar (1985) argue that this is one of the critical reasons for the success of the Reciprocal Teaching Method. In Reciprocal Teaching the expert modelling is initiated when the student has difficulties producing a question or a summary for a text, and the teacher intervenes to help provide one. Initially, the teacher, as expert, provides a complete model of how to do the task and gradually turns over more and more of the task to the student, aiding him with leading questions, evaluation of the student's efforts, and encouragement. We do not have the technological capability to do the kind of individual shaping that teachers do in Reciprocal Teaching, but technology can provide expert models to students struggling with problems of pronunciation or interpretation of text.

Conclusion

The recording and replaying of the processes people use to perform tasks such as reading, writing, and problem solving, has the capability to make these processes objects of reflection, annotation, and communication. Using imitation, replay, abstracted replay, and reification, student's can begin to think about, talk about, and experiment with their learning and problem-solving processes in a way not previously possible.

By way of summary, we can briefly reiterate some of the reasons why reflection is important to learning:

(1) Students can compare their own process to the way more expert performers carry out the process.

(2) With reification, it is possible to reconfigure a process representation so that students can see separate aspects of the process together and can view the process itself from perspectives they have not seen before.

(3) Students can derive abstractions about the process by comparing multiple performances simultaneously.

(4) Abstractions can be constructed in a form that is critical to developing good metacognitive strategies.

When we design learning environments for any subject, be it history, language, or physics, we should consider how to record and abstract the problem-solving processes students use in these learning environments. We should then provide students with facilities for replaying and observing their own performance and the performance of other students. And finally we should provide process models of more advanced performance that students can compare to their own process.

References

Anderson, J. R., Boyle, C. F., Farrell, R., & Reiser, B. J. (1984). Cognitive principles in the design of computer tutors. In Proceedings of the Sixth Annual Conference of the Cognitive Science Society (pp. 2-9). Boulder, CO: University of Colorado.

Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). Intelligent tutoring systems. Science, 228, 456-468.

Bereiter, C., & Bird, M. (1985). Use of thinking aloud in identification and teaching of reading comprehension strategies. Cognition and Instruction, 2.

Bereiter, C., & Scardamalia, M. (1985). Cognitive coping strategies and the problem of "inert knowledge." In S. F. Chipman, J. W. Segal, & R. Glaser (Eds.), Thinking and learning skills (Vol. 2). Hillsdale, NJ: Erlbaum.

Brown, A. L. (1978). Knowing when, where, and how to remember: A problem of metacognition. In R. Glaser (Ed.), Advances in instructional psychology (Vol. 1). Hillsdale, NJ: Erlbaum.

Brown, J. S. (1985). Process versus product: A perspective on tools for communal and informal electronic learning. Journal of Educational Computing Research, 1, 179-201.

Bundy, A. (1983). The computer modelling of mathematical reasoning. London, UK: Academic Press.

Collins, A. (1985). Teaching reading and writing with personal computers. In J. Oransanu (Ed.), A decade of reading research: Implications for practice. Hillsdale, NJ: Erlbaum.

Collins, A., & Smith, E. E. (1982). Teaching the process of reading comprehension. In D. K. Detterman & R. J. Sternberg (Eds.), How much and how can intelligence be increased (pp. 173-185)? Norwood, NJ: Ablex.

Flavell, J. H. (1978). Metacognitive development. In J. M. Scandura & C. J. Brainerd (Eds.), Structural/process theories of complex human behavior. Alphen a.d. Rijn, the Netherlands: Sijthoff and Nordhoff.

Flower, L. S., & Hayes, J. R. (1980). The dynamics of composing: Making plans and juggling constraints. In L. W. Gregg & E. R. Steinberg (Eds.), Cognitive processes in writing. Hillsdale, NJ: Erlbaum.

Palincsar, A. S. & Brown, A. L. (1984). Reciprocal teaching of comprehension-fostering and monitoring activities. Cognition and Instruction, 1, 117-175.

Papert, S. (1980). Mindstorms. New York: Basic Books.

Footnote

[1]Actually there are two kinds of abstractions that need to be considered: the first concerns how to structure and present the problem solving audit trail, the second concerns choosing the right grain size of events that are to be stored on the audit trail so that, metaphorically, the wheat can be easily separated from the chaff. In Algebraland, this latter issue is solved by choosing a set of moderately high level algebraic operators for the student to use in transforming mathematical expressions and to have all the arithmetic simplifications done by just one operator.

Figure Captions

Figure 1. Layout of the screen for Algebraland

Figure 2. (a) The Geometry Tutor's initial representation of the problem; (b) a representation in the middle of the problem; and (c) a representation at the solution of the problem.

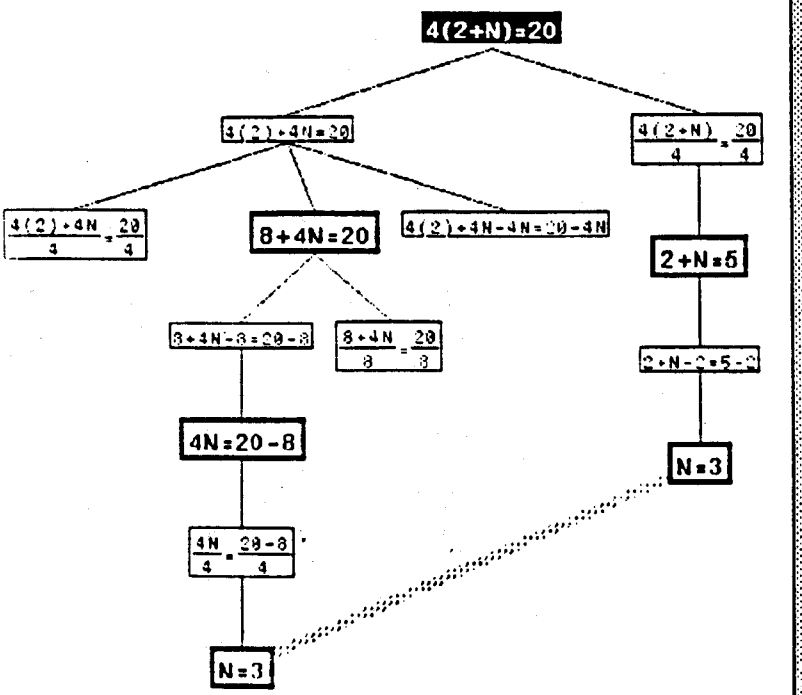
Figure 3. Albebraland reflection window showing the trace of an actual student working on the problem at the top of the screen to solve for V.

Figure 4. Screen from NoteCards showing one of the browsers created by a graduate student working with the system.

Figure 5. Stone Soup by Aesop with inserted questions and expert answers.

Prompt Window

SEARCH SPACE WINDOW



PLAN MENU

ISOLATE the variable.
COLLECT like terms into a single expression.
GROUP together like terms (transpose terms).
SPLIT apart expressions containing the variable.
SIMPLIFY the expression.

RECORD WINDOW

(8)	4(2+N)=20	(PROBLEM)
(9)	$\frac{4(2+N)}{4}=\frac{20}{4}$	(DIVIDE)
(10)	2+N=5	(Do-Arithmetic)
(11)	2+N-2=5-2	(SUBTRACT)
(12)	N=3	(Do-Arithmetic)
(13)	N=3	(SOLVED)

PROBLEMS (done)

Problem Menu

- 58+17=7-38
- X(17+2)=41
- $\frac{N}{8}=\frac{1}{4}$
- $\frac{1}{U}+\frac{1}{V}=\frac{1}{F}$
- 17(A-2)=10(A+3)
- $A=P(1+\frac{R}{100})$

Solve for N

$4(2+N)=20$

BASIC OPERATIONS

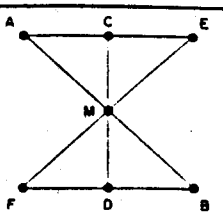
To Both Sides:

ADD	SUBTRACT
MULTIPLY	DIVIDE

- Do-Arithmetic
- Distribute
- Expand
- Combine-Terms
- Cancel-Terms

(undo) (next problem)

a

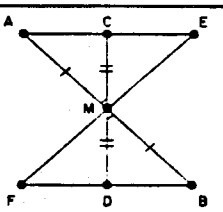


M is midpoint of $\overline{EF}$

M is midpoint of $\overline{AB}$

M is midpoint of $\overline{CD}$

b



M is midpoint of $\overline{EF}$

DEF-MIDPOINT

$\overline{EM} \cong \overline{MF}$

$\triangle MBD \cong \triangle MAC$

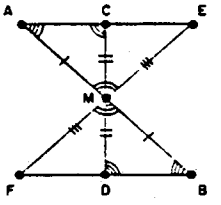
SAS

$\overline{MB} \cong \overline{AM}$ | $\angle AMC \cong \angle BMD$ | $\overline{MD} \cong \overline{CM}$

DEF-MIDPOINT | VERT | DEF-MIDPOINT

M is midpoint of $\overline{AB}$ | M is midpoint of $\overline{CD}$

c



M is midpoint of $\overline{EF}$

DEF-MIDPOINT

$\overline{EM} \cong \overline{MF}$

CORRES-PARTS

$\triangle AME \cong \triangle BMF$

ASA

$\angle MDB \cong \angle MCA$ | $\angle MBD \cong \angle MAC$

CORRES-PARTS | CORRES-PARTS

$\triangle MBD \cong \triangle MAC$

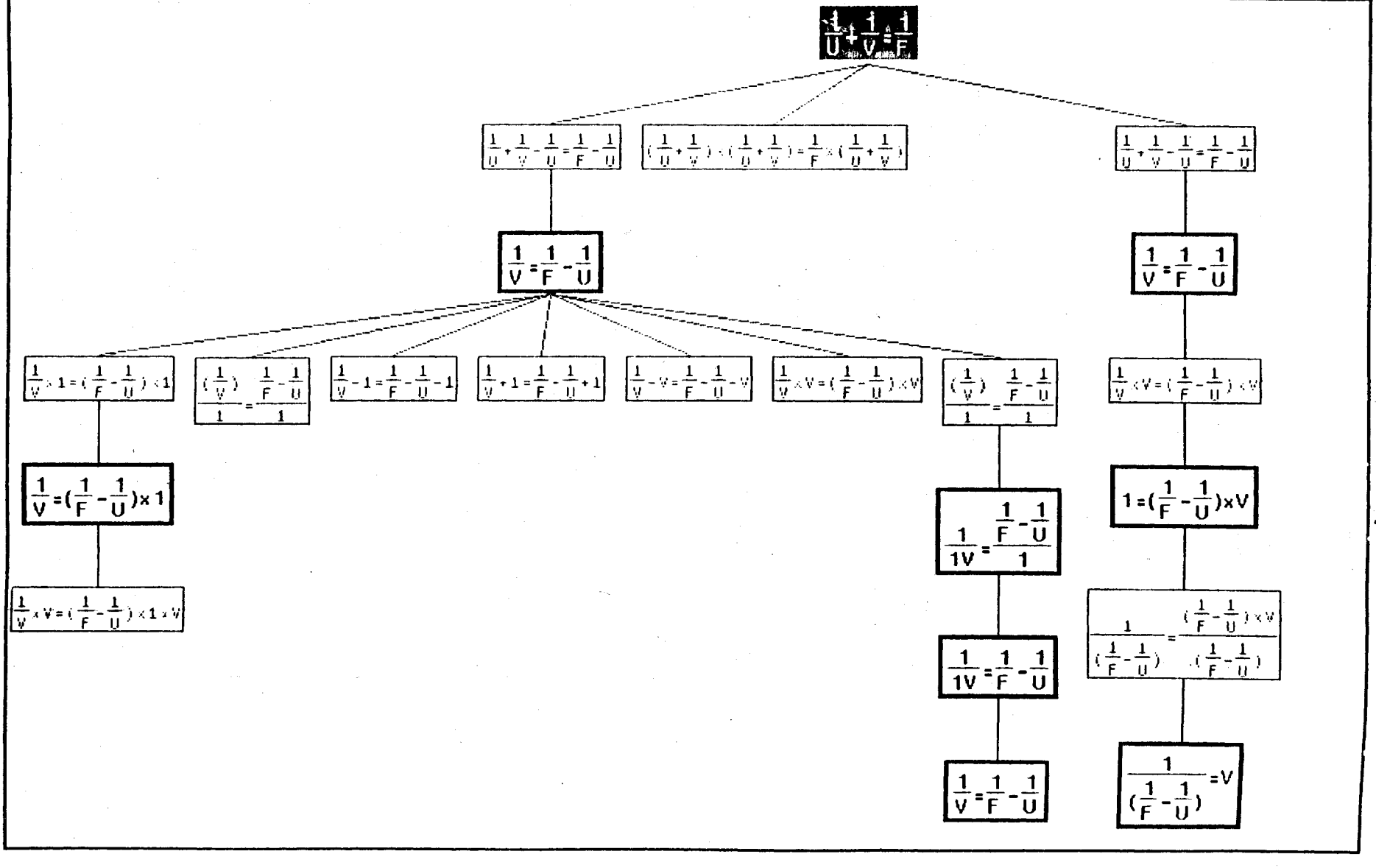
SAS

$\angle MCA$ supp $\angle MCE$ | $\overline{MB} \cong \overline{AM}$ | $\angle AMC \cong \angle BMD$ | $\overline{MD} \cong \overline{CM}$ | $\angle AME \cong \angle BMF$

ADJ-SUP-ANGS | DEF-MIDPOINT | VERT | DEF-MIDPOINT | VERT

M is midpoint of $\overline{AB}$ | M is midpoint of $\overline{CD}$

Reflection Window



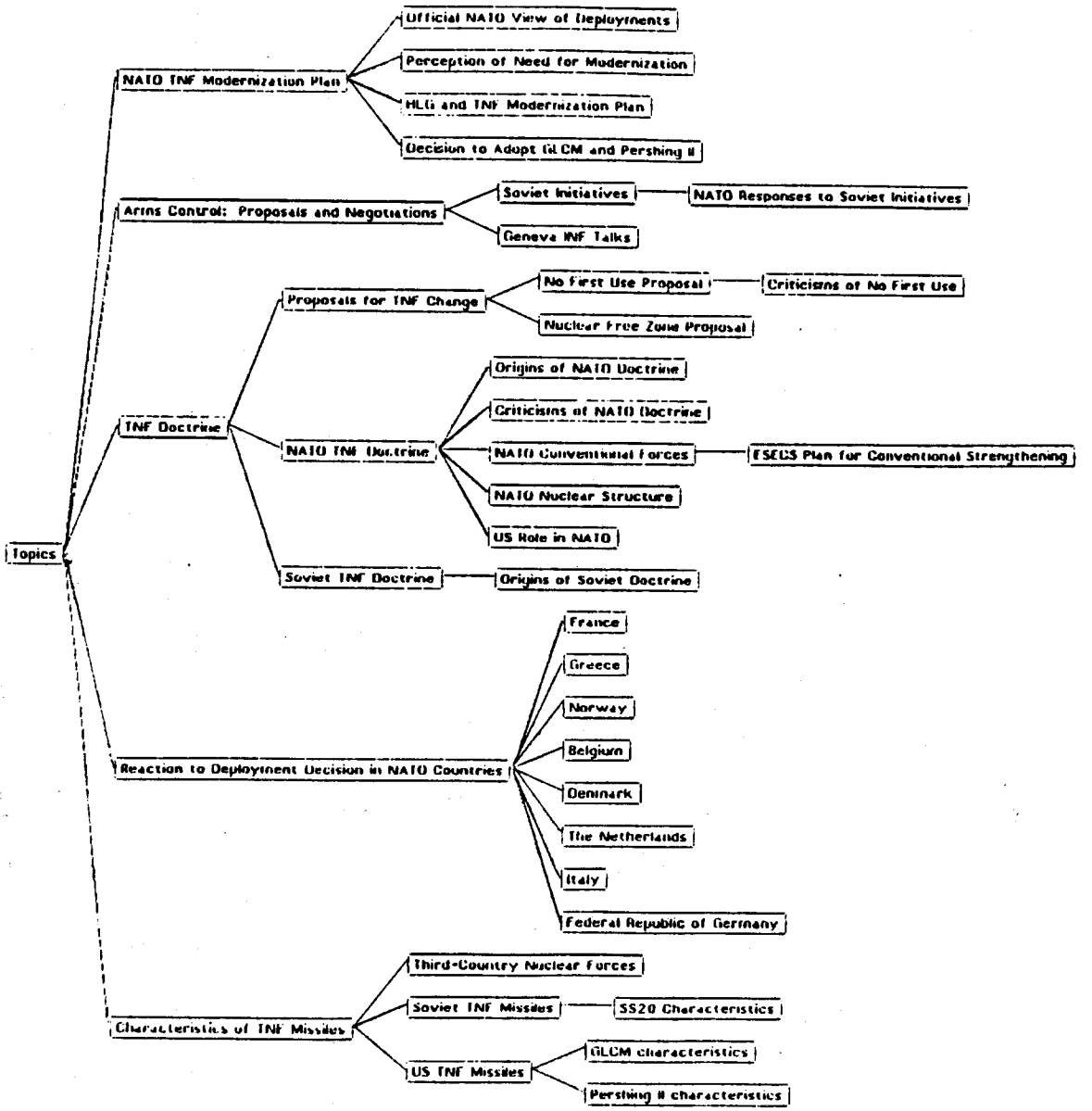

Topics
NATO INF Modernization Plan
Official NATO View of Deployments
Perception of Need for Modernization
HLG and INF Modernization Plan
Decision to Adopt GLCM and Pershing II
Arms Control: Proposals and Negotiations
Soviet Initiatives
NATO Responses to Soviet Initiatives
Geneva INF Talks
INF Doctrine
Proposals for INF Change
No First Use Proposal
Criticisms of No First Use
Nuclear Free Zone Proposal
NATO INF Doctrine
Origins of NATO Doctrine
Criticisms of NATO Doctrine
NATO Conventional Forces
ESECS Plan for Conventional Strengthening
NATO Nuclear Structure
US Role in NATO
Soviet INF Doctrine
Origins of Soviet Doctrine
Reaction to Deployment Decision in NATO Countries
France
Greece
Norway
Belgium
Denmark
The Netherlands
Italy
Federal Republic of Germany
Characteristics of INF Missiles
Third-Country Nuclear Forces
Soviet INF Missiles
SS20 Characteristics
US INF Missiles
GLCM characteristics
Pershing II characteristics

Figure 5

Stone Soup

A poor man came to a large house during a storm to beg for food. He was sent away with angry words. (Q. Who do you think sent him away and why? A. The owner because he didn't care about beggars.) But he went back and asked, "May I at least dry my clothes by the fire, because I am wet from the rain?" The maid thought this would not cost anything, so she let him come in. (Q. Now who do you think sent him away at first and why? A. The maid, because she didn't want to give away her master's property.) (Q. What do you think will happen when he gets inside? A. He will dry his clothes and maybe make friends with the maid.)

Inside he told the cook that if she would give him a pan, and let him fill it with water, he would make some stone soup. This was a new dish to the cook, so she agreed to let him make it. The man got a stone from the road and put it in the pan. (Q. What good is a stone for making soup? A. It is of no use.) The cook gave him some salt, peas, mint, and all the scraps of meat she could spare to throw in. (Q. Why do you think he offered to make stone soup? A. So he could get to eat all the scraps the cook threw in.) Thus, the poor man made a delicious stone soup and the cook said, "Well done! You have made a wonderful soup out of practically nothing." (Q. Why do you think that the man asked to dry himself inside? A. So he could get inside in order to fool the cook into giving him food.)