Computer Environments for Proof Construction

by

Richard Scheines and Wilfried Sieg

April 1994

Report CMU-PHIL-52

Philosophy
Methodology
Logic

Pittsburgh, Pennsylvania 15213-3890

Computer Environments for Proof Construction*

Richard Scheines
Wilfried Sieg

Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA 15213
email: R.Scheines@andrew.cmu.edu
W.Sieg@andrew.cmu.edu

Abstract

Does the presentation of search space matter for complex problem solving tasks? We address this question for the construction of proofs in sentential logic by comparing three computerized environments and measuring their relative pedagogical effectiveness. After being given a pretest for logical aptitude, three matched groups were presented identical course material on logic for approximately five weeks. Problems from this material were completed in three different computerized environments, however. During that time, all students completed over one hundred exercises and then took a midterm. The group using the most informative and flexible interface performed substantially better on the midterm, with nearly all the difference arising in the harder problems. In two follow up experiments we added strategic problem solving help; student performance improved again (entirely on the more difficult problems).

* We thank Ken Koedinger for many helpful discussions and some needed encouragement. We also thank Virginia Hoage, who collected the data, and Chris Walton, who performed the programming miracles that made this study possible.

1. Introduction

Formal logic is our best theory of rigorous reasoning. Because formal logic and computation are so closely related, computer aids to logic teaching and theorem proving have proliferated. At present there are over 40 computer programs that help teach logic, and surely more will follow [Croy 86, Burkholder 89]. During the last few years we have developed the Carnegie Mellon Proof Tutor (CPT) a computerized proof construction environment that combines a sophisticated graphical interface with a system for strategic help based on a complete and cognitively natural algorithm for finding proofs in sentential logic [Scheines and Sieg, 93; Sieg and Scheines 92, Pressler 88].

The design of CPT was based on the belief that students learn more from exercises when their problem solving environment has three features. First, its interface must relieve the student of non-essential cognitive load. That is, it must take over routine calculations, display the problem state informatively, and identify errors instantly.[1] Second, it must allow students maximal flexibility in traversing the problem space. Third, it must provide locally appropriate *strategic guidance.*[2] Students quickly master the syntactical constraints on logic proofs, but it takes them significantly longer to internalize the strategic subtleties of proof construction.

In this study we report on a series of experiments that tested the pedagogical value of three components of CPT's problem solving environment, namely:

1) an informative and manipulable display of the problem state,
2) the flexibility to work forwards and backwards instead of either way alone, and
3) minimal strategic help at any point in a problem.

We gathered data relevant to the first two components in the fall semester of 1989 and data relevant to the third in the spring of 1989 and the fall of 1990. The 1989 data support the utility of the first feature and strongly support the utility of the second. The 1990 data confirm the importance of strategic help.

These results are of interest for the particular task at hand, as they undermine the view that working backwards is the method to be preferred for proof construction. But the significance goes beyond this parochial logical point. The computer-based course, used as a platform for our experiments, allowed us to isolate very clearly the impact of *problem space representation* and

---

[1]John Anderson [Anderson 85a] has shown that immediate error correction reduces the frequency of future errors. Jill Larkin [Larkin 87] has shown that an informative and easy to parse display of the problem state improves performance and reduces training time.

[2]This point motivates most "intelligent" computer tutors, e.g. [Anderson 85b].

space traversal possibility on student performance. Informativeness of the former and flexibility of the latter improved performance quite significantly.

In section 2 we describe the computer environment and the underlying logical system; at the end of section 2 we can then formulate the central issues we wanted to clarify. Section 3 is devoted to a discussion of our experimental methods, whereas section 4 contains the central results. The last section considers these results with reference to our original questions.

2. Constructing Proofs (with CPT)

Most philosophy departments offer a course in formal logic, and most such classes assign dozens of proof construction exercises. Several logical systems exist within which to give proofs, but most share one feature: a proof is a series of lines, each of which is either an assumption or the result of applying a logical "inference rule" to previous lines. As in many other problem solving tasks, one begins a proof construction problem with a gap between what is given (the premises) and what is sought (the conclusion).[3] The gap can be closed by inferring new lines from the premises, or by creating subgoals such that, if they were proved, they would allow one to infer a line desired, i.e., the gap can be closed by forwards and backwards chaining.

Although it is almost never taught as such, finding a proof in a formal framework is a classic search problem. One can work forwards and search for a goal G in the forest of lines that could be produced via the inference rules from a set of assumptions Γ (figure 1),

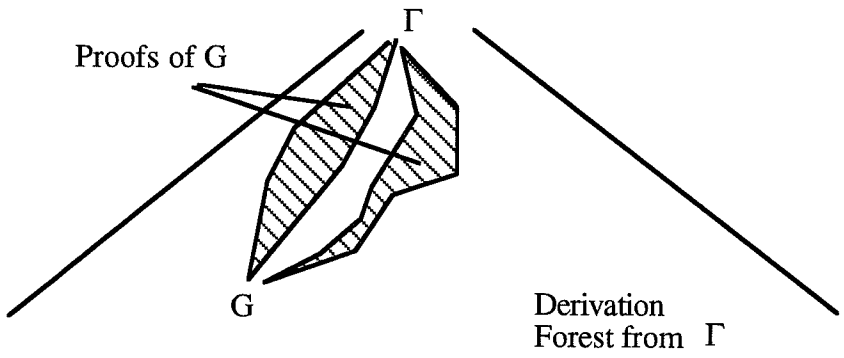

Figure 1

or work backwards and search for the premises Γ in the inverse forest of arguments that *lead to* G (figure 2),

[3]See [Newell 72].

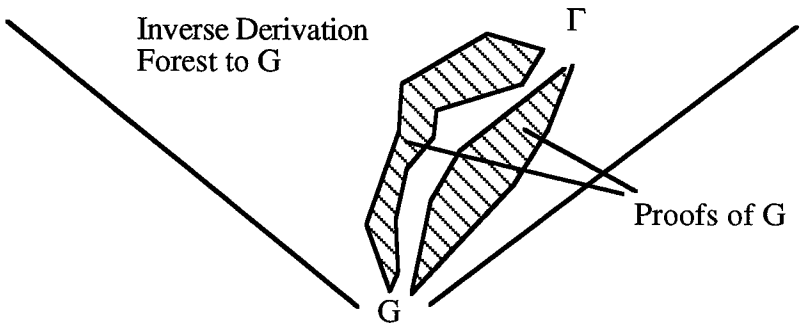

Figure 2

or one can confine the search to the region where the forward and backward searches overlap (figure 3).

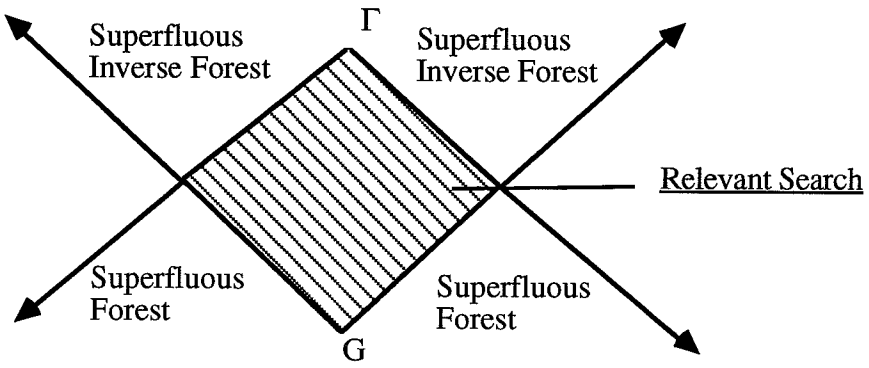

Figure 3

All the standard problems of search arise in proof construction. Students wander down dead ends and cannot recover, they travel in circles, and they move laterally instead of vertically, i.e., they make moves which are legal but which do not bring them closer to a solution. An issue special to proof construction in any of a variety of "natural deduction" systems[4] is the introduction and discharge of temporary, or working assumptions.

How can an appropriate computer environment aid the student who faces these obstacles? CPT provides all the operations that are now standard among logic programs,[5] but it also provides an informative display, the ability to work forwards or backwards, and strategic help. CPT's

[4]Such systems were introduced by Gentzen in the thirties. See [Gentzen 69].
[5]i.e. mistake correction and calculational relief, etc.

display (figure 5) is novel in that it combines information about the search along with information about the structure of temporary assumptions. In the single column of lines used by standard logic programs, associated with each line is a set of temporary assumptions on which that line depends. Thus when a temporary assumption q is introduced, any line that depends on q includes q's line number in its dependency list. Fitch diagrams offer a superior and graphical option for displaying logical dependency;[6] a box is drawn for each new temporary assumption added. Everything inside the box depends on this assumption. When the assumption is discharged, the box is closed. Thus calculating logical dependency in a Fitch diagram is cognitively as natural and easy as deciding box containment. Consider alternative displays of the same proof in figure 4.

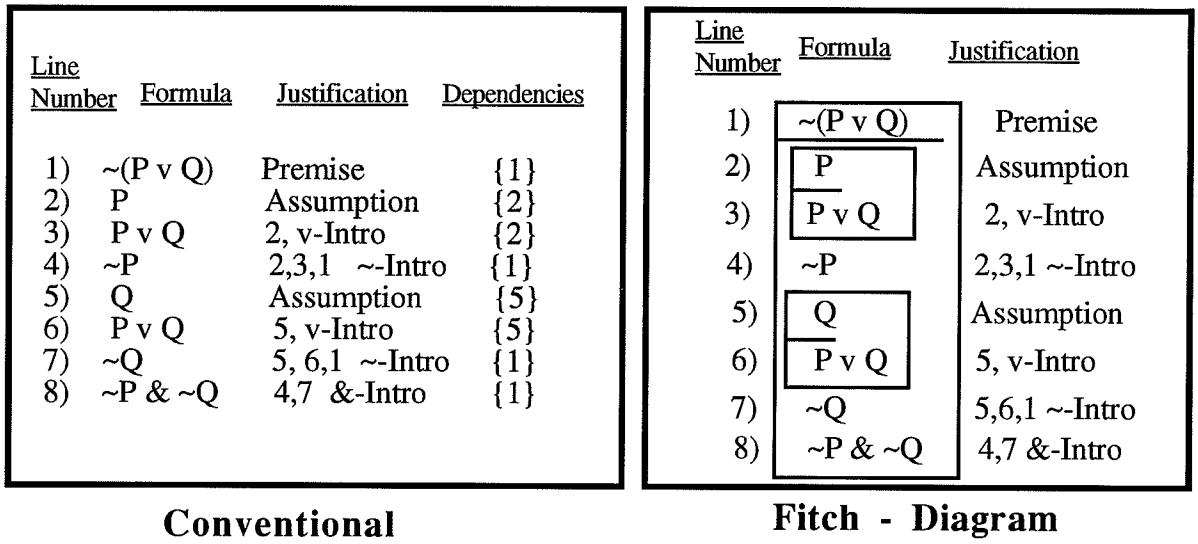

Figure 4

Glancing at the conventional diagram, it might appear strange that lines 3 and 6 contain the same formula. It might also appear that we have derived several contradictions, e.g. lines 5 and 7. Neither "oddity" appears problematic in the Fitch diagram: one can see immediately that lines 3 and 6 occur within different boxes and thus depend on different assumptions; one can also see that line 5 is an assumption and line 7 is the result of an argument by *reductio* (~-Intro) from this assumption.

Fitch diagrams do not represent the search for a proof any better than a column of lines. Thus, we combined a tree-like diagram of subgoals with a Fitch diagram to give the student a "Goal Tree." The standard Fitch diagram is provided in parallel to the Goal Tree as a representation of "The Proof" (Figure 5).

---

[6]Many standard logic texts use a Fitch style representation. See [Kalish 80], for example.

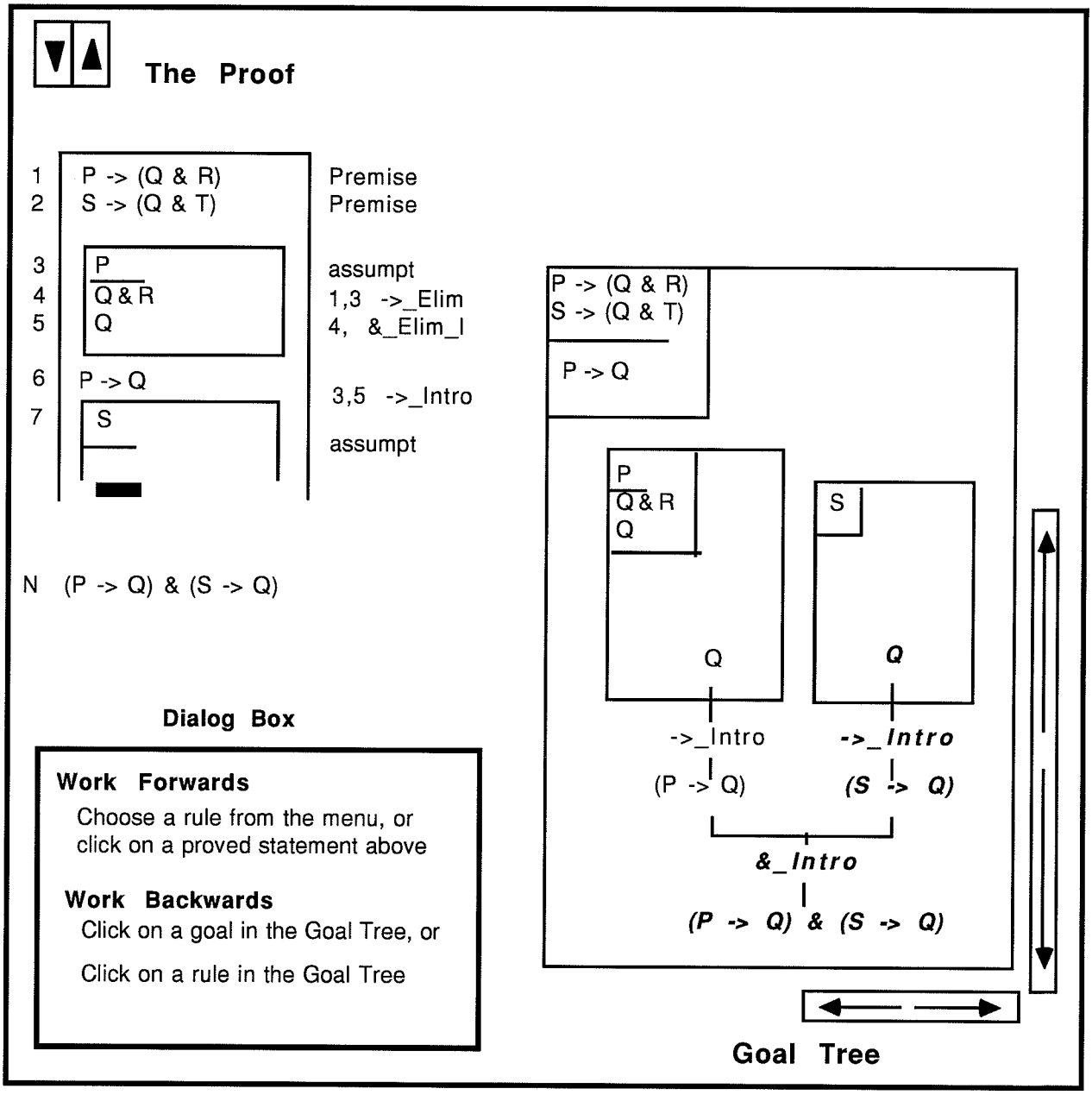

Figure 5

Students can work forwards or backwards in CPT, and they can determine at a glance the state of their search and the logical structure of their temporary assumptions. Students can also request strategic help at any point in a problem. CPT solves the problem from the point the student has left it, and then responds to the student in one of three ways. It will either 1) display the

solution and the search it conducted to find that solution, or 2) allow the student to step through its search for the solution, or 3) conduct a dialogue with the student on how to proceed towards a solution. Having designed such a system and tested it informally on dozens of students, our goal was to test it experimentally. The crucial question was: *Do the goal-tree display, the ability to work backwards and forwards, and the strategic help objectively improve performance.*

3. Methods

Every semester, 30-60 students take *Introduction to Logic* at Carnegie Mellon from a computer program called VALID that was developed at the Institute for Mathematical Studies in the Social Sciences (IMSSS) at Stanford University in the 1970s [Suppes 81]. VALID thoughtfully covers an extensive introductory curriculum and is now used at perhaps half a dozen universities around the country.

VALID alternates between introducing new logical ideas and assigning exercises to build the skills that correspond to these ideas. After a new idea is introduced via several screens of text, students are presented with a menu of proof construction exercises. Upon selecting an exercise, they shift to VALID's interactive proof construction environment and complete the exercise (figure 6). After finishing an exercise they return to the menu, or, if the menu is completed, to the main curriculum. The course is entirely self-paced, i.e., there is no time limit within which a problem must be completed. There is also no penalty for mistakes: students simply keep trying until they solve the required number of problems and then move on to the next concept. The great bulk of a student's time is spent actively working on exercises, not passively reading text.

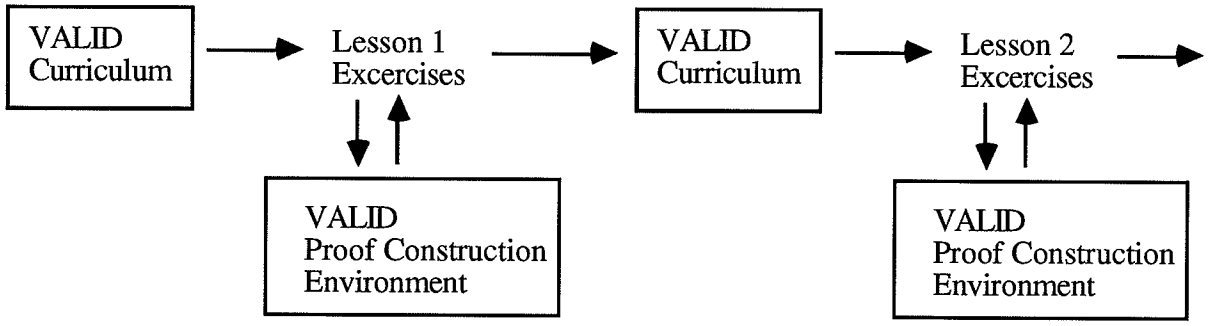

Figure 6

VALID's proof construction environment uses a line presentation of proofs and a standard proof checker. We modified the course by replacing VALID's proof construction environment with CPT. Students taking *Introduction to Logic* use CPT to complete exercises prior to the

midterm, as well as to take the midterm.[7] They are exposed to CPT for approximately five weeks. In the fall semester of 1989, we conducted an experiment to compare the pedagogical effectiveness of three different versions of the proof construction environment (pce).

On the first day of classes we gave all students a pre-test for logical aptitude designed and validated at Stanford. We used the results of this test to split the class into three matched groups of 11 students each. We maximized the range of ability within each group, and made sure that each group showed a similar distribution of abilities. Each group proceeded through the VALID curriculum, but used a different pce for the exercises and midterm. The groups were trained for half an hour by a TA to use their particular pce.

To minimize the differences between treatments, each pce was implemented inside the CPT shell. So all operations except those actually involving proof construction were identical for the three groups. The first pce (for the Forwards group) was a standard proof construction environment. Students using the first pce could *only work forwards* and were given the standard column representation of proofs, i.e., their display resembled the left side of figure 4 and did not include a Fitch diagram or a Goal Tree. The second pce (for the Backwards group) was a version of CPT in which the problem was displayed as we show in figure 6, with both a Fitch diagram and a Goal Tree. However, in this pce the students could *only work backwards* from the conclusion toward the premises by subgoaling in the Goal Tree. The third pce (for the full CPT group) was the same as the second, but students were allowed to work backward in the Goal Tree and forward in either the Goal Tree or Fitch diagram. None of the pces provided strategic help in the fall of 1989.

The midterm exam consisted of eight proof construction problems similar to those the students had faced prior to the midterm. From prior tests we chose two problems that were easy, three that were of medium difficulty, and three that were hard. Students were given three hours to complete the test. Besides recording whether they succeeded or failed to complete a given problem, we measured how much time it took to finish a problem, how many steps were in the completed proof, and how many errors of logical dependency students made. To be fair to the students, we adjusted their grades by an amount equal to the difference between their group's mean and the best group's mean.

We collected data for classes during the next two semesters as well. In the spring of 1990 we had 35 students, and in the fall of 1990 we had 42, but neither of these classes were split into groups. We added strategic help to CPT in the spring of 1990, but the help system was quite buggy; in fact it was hardly used once the word got out that asking for help would often cause the

[7]They receive no strategic help while completing the midterm, of course.

system to crash.[8] We therefore consider the spring 1990 data more relevant to the question of whether the interface effects we had noticed in the fall of 1989 are repeatable. The help system was fixed for the fall semester of 1990. During that semester students were encouraged to use it often and did so.[9] Each of the 1990 classes was given a substantially harder midterm than the 1989 class. The first eight problems were identical (up to notational variations) on all midterms, but for the 1990 classes we added two new and extremely hard problems. We also increased the time pressure on the 1990 classes. Instead of three hours for 8 problems, we gave the 1990 groups only two and a half hours for 10.

Figure 7 shows the pretest means, in percents.[10] The Backwards group is slightly high because three of the poorer students dropped out after we had matched the groups. One student dropped out of the Forwards group.

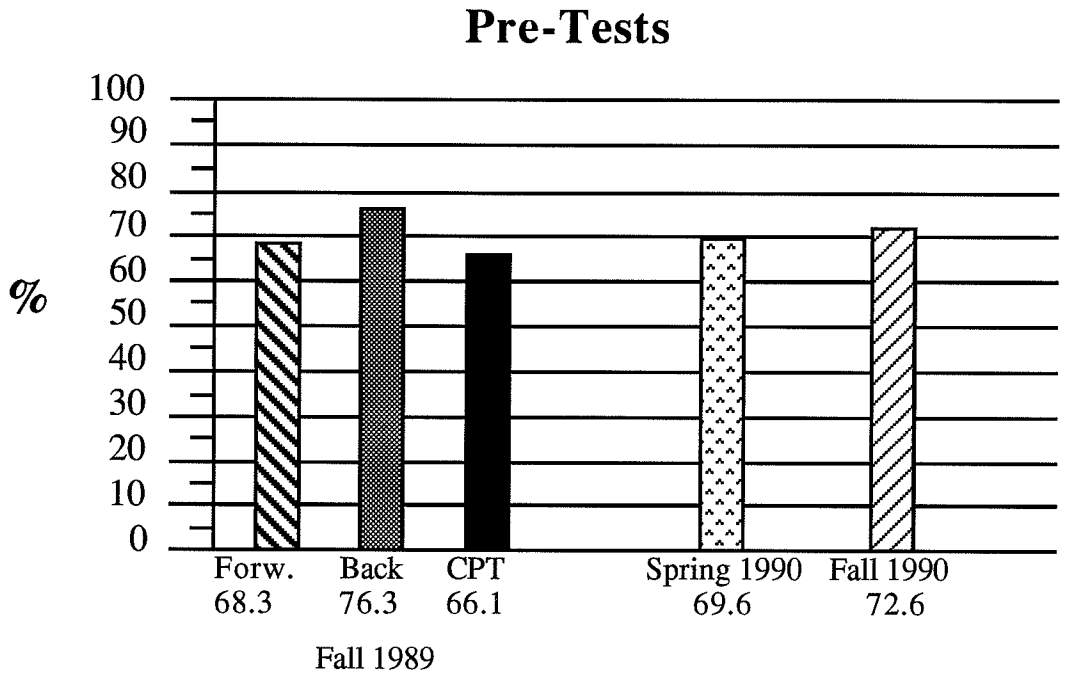

Figure 7: Pre-test Scores

Students in the CPT group, who were allowed to work forwards and/or backwards, chose to work backwards 46% of the time and thus used those parts of their environment that alternative groups could not. Since each group completed over a hundred proof construction problems over

[8]Only 6 students asked for help more than 5 times in that semester.

[9]In the fall semester students asked for proof construction help over 1600 times, an average of nearly 40 requests per student.

[10]The overall distribution of pre-test scores for the spring and fall classes was approximately Gaussian.

the course of five weeks with their own pce before taking the midterm, training effects were negligible.

4. Results

4.1 Overall Performance

We compare groups by comparing their means. Figure 8 shows the mean percentage of proofs correctly completed fo those problems that were presented to all groups. To compare group scores as they relate to the group's aptitude, we display each group's midterm score next to its pre-test score. Comparing the midterm means to the pre-test means is in itself meaningless, but the relative difference between pre-test and midterm score gives a basis for comparing the various treatments. While the forwards and backwards groups scored (by coincidence) nearly the same percentage as their pretest, every other group improved on their pre-test score by 12-18 percentage points.

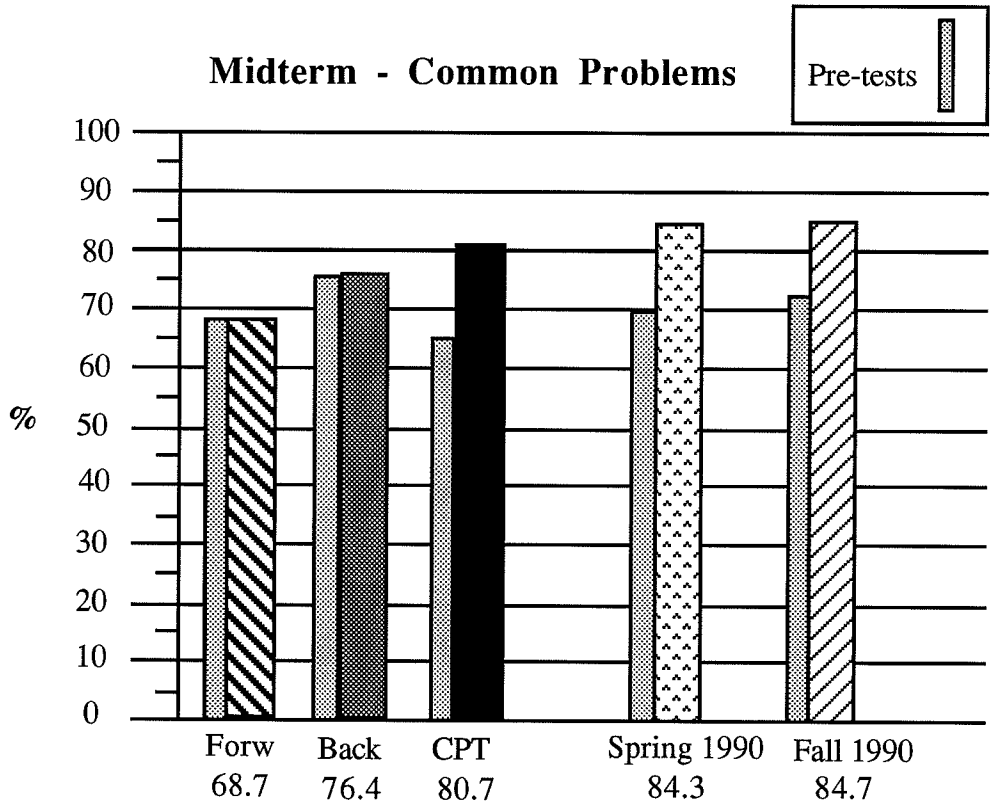

Figure 8

The groups were nearly indistinguishable on the easy and medium problems, but the difference in performance on the hard problems common to all groups was striking (Figure 9).

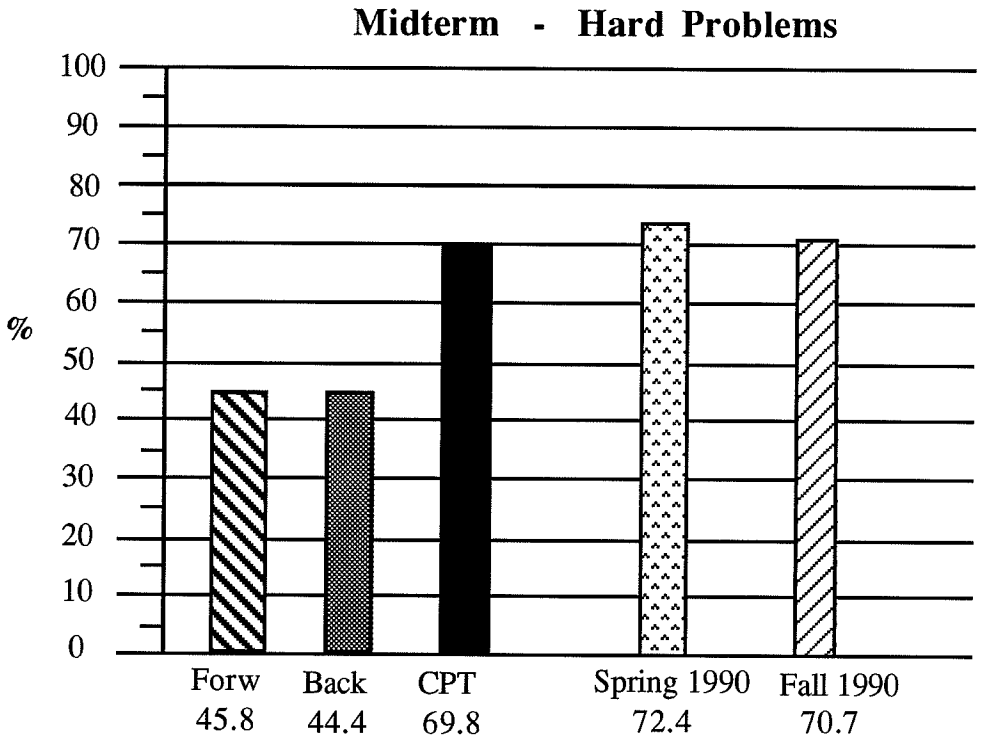

Figure 9

4.2 Interface Effects

Before we consider the effect of strategic help, let us consider the effect of the differences in the interface only. Prior to performing this experiment, we were convinced that working backwards was a superior method for finding proofs in formal logic. We thus expected to see a marked difference in overall performance between the group that could only work forwards and the group that could only work backwards, with the latter having the advantage. As is obvious from the data this difference did not materialize, but one related to it did. Groups that were restricted to work in one direction did dramatically worse than those allowed to work in both directions. To make this difference vivid, we aggregated the forwards only and backwards only groups into a "1-way" group, and compared it to a "2-way" group that is an aggregation of the CPT and Spring 1990 groups.

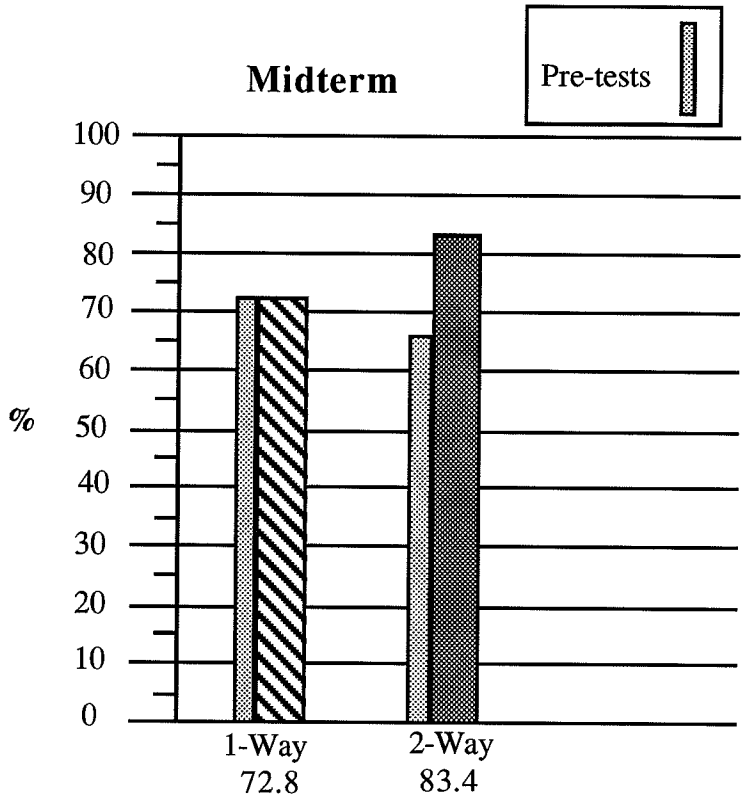

Figure 10

The 2-way group outperforms the 1-way group on the overall midterm (figure 10), and the difference between their scores on the hard problems is dramatic (figure 11). The latter difference is significant at the .026 level on a T-test. That is, if we assume that the two groups went into the midterm with equal ability to solve the hard problems,[11] there is a 2.6% chance that we would observe as large a difference in their means as we did or a larger one. Thus, we reject the hypothesis that the groups were equal when they took the midterm, and since the only apparent difference was the pce each group used, we assume that differences in pces caused differences in proof construction performance.

[11] Actually it appears that the 1-way group had slightly more ability. Their mean pre-test score was 72.5 compared with 68.7 for the 2-way group. This makes the difference in their performance on hard problems seem even more dramatic.

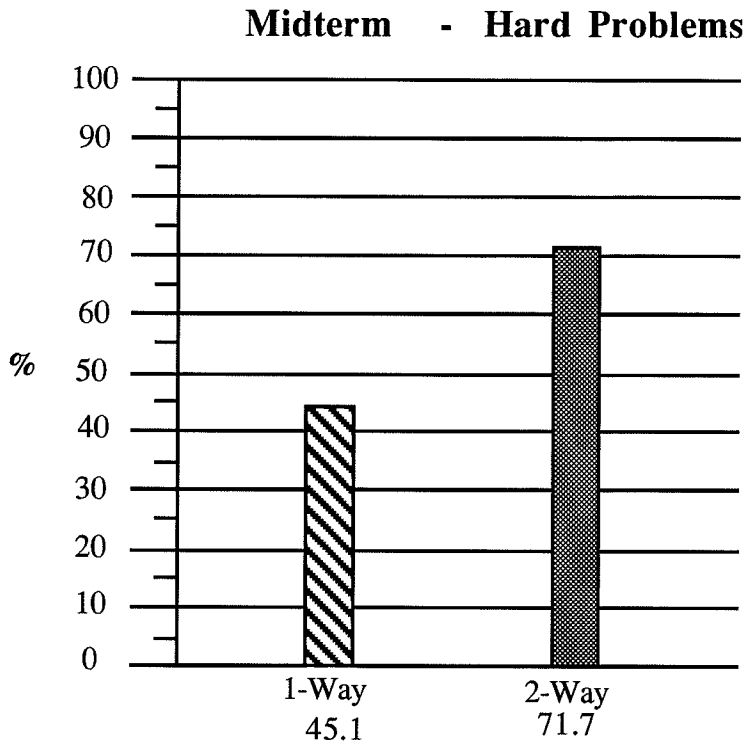

Figure 11

Even though our preconceptions about working backwards were not borne out in overall performance, other measures suggest that they are at least partially correct. Students who worked backwards only seemed to find more elegant proofs than those who worked forwards only (figure 12), and they took less time to find them (figure 13).There is a much greater difference on both of these measures between the forwards only group and the backwards only group then there is between the backwards only and CPT group.

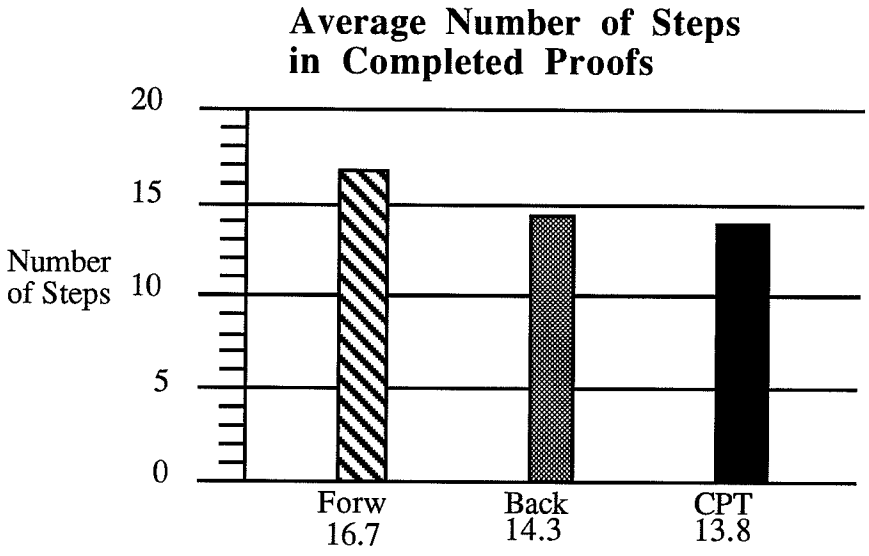

Figure 12

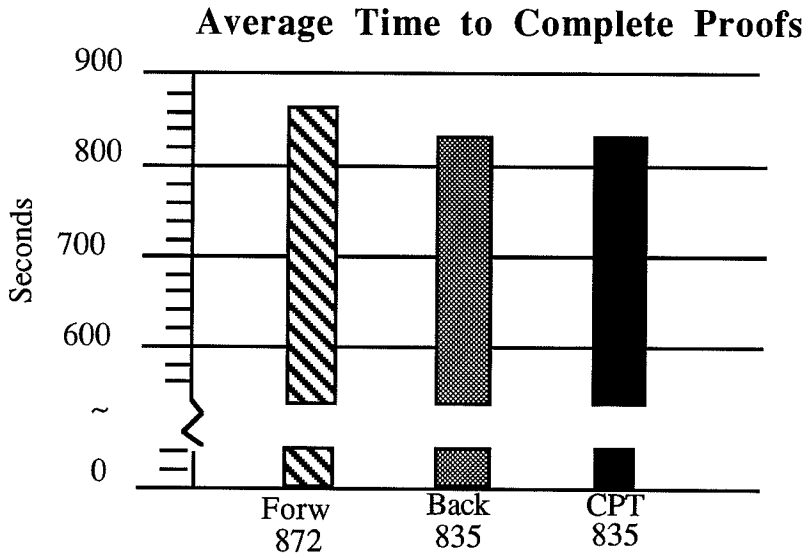

Figure 13

The Fitch diagram, and its corresponding boxes in the Goal Tree, are intended to help students master the relatively difficult topic of logical dependency. To test the effectiveness of these representational aids, we kept track of each time a student attempted to make an inference that was illegal because of inappropriate logical dependencies. In fact one cannot make such an error while working backwards, so for this hypothesis we compare the forwards only group to the groups using the full interface. Including the Spring group, 43 students in the full CPT groups made 3

errors of logical dependency on the midterm while 11 students in the Forwards group made 11 such errors.

4.3 Strategic Help

The interface effects were most striking on the hard problems, and so far the effect of strategic help seems to be confined to the extremely hard problems (figure 14).

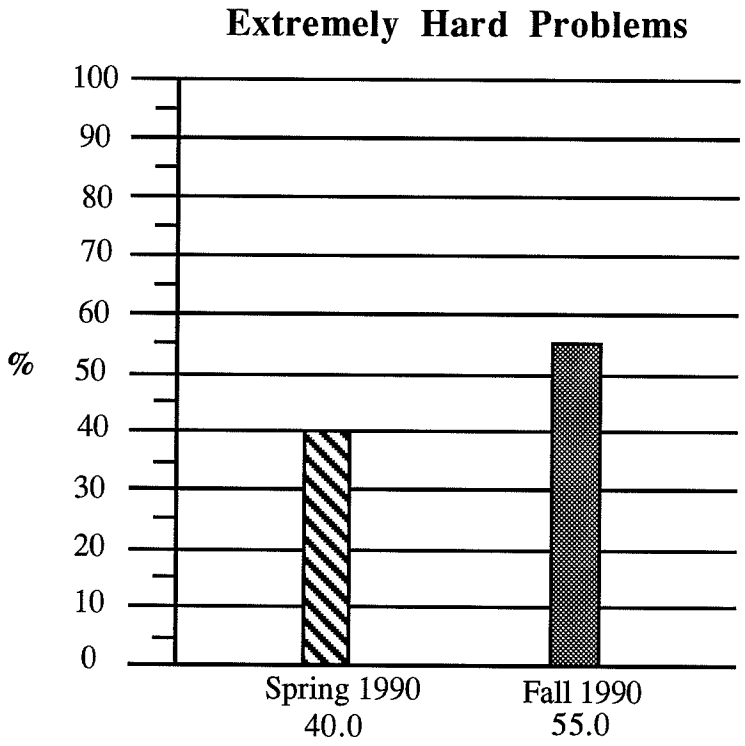

Figure 14

5. Analysis

We set out to test three features of CPT's problem solving environment, i.e.:

1) an informative and manipulable display of the problem state,
2) the flexibility to work forwards and backwards instead of either way alone, and
3) strategic help at any point in any problem.

The data strongly support the utility of the second feature. Working backwards is clearly helpful in certain contexts of proof construction, but it is unnatural in others. The same holds for working forwards. Students forced to work exclusively in either direction were at a severe disadvantage, as is evidenced by the large differences between the 1-way and 2-way groups. The

CPT group differed from the Backwards group only in the ability to work forwards, so being able to work forwards clearly matters. The CPT group differed in two ways from the Forwards group, first in its ability to work backwards and second in the use of an informative display. As a result the utility of working backwards is more difficult to assess. It seems unlikely that the difference between these two groups was entirely due to the display, however. The Forwards group and Backwards group differed in their display, but did not exhibit nearly the difference that the CPT and Forwards group showed. Our interpretation is that only the students in the CPT and Spring groups learned to separate those proof contexts in which working backwards was advantageous from those in which working forwards was better. We hypothesize that *the ability to take alternative routes through the search space helps the student form a deeper appreciation of the structure of the space.* In logic such a distinction clearly exists. Some inference rules reduce the complexity of the problem when applied backwards but increase it when applied forwards, and vice versa. We suspect that similar results would hold in any domain in which reducing the complexity of the problem depends on whether one uses a rule to work backwards or to work forwards.

The Goal Tree and Fitch diagram embody the informative display. Every group had such a display except the Forwards group. That it mattered is evidenced first by the difference between the number of errors in logical dependency that were made by the Forwards and CPT groups. Errors in logical dependency can be made in either group's pce, but the Forwards group actually committed them far more often. Second, although the difference in midterm success rate between the Forwards and Backwards groups matches their pretest difference almost exactly, the Backwards group outperformed the Forwards group in time on search and in the size of the proof found.[12] Our preliminary conclusion is that the Backwards group searched more efficiently and with fewer errors because of the Goal Tree diagram. This is supported to some degree by (subjective) surveys administered to each of the groups we discussed, and to several groups who took Introduction to Logic after 1990. Students were asked to rate features of the tutor on a scale of:

Useless 1 2 3 4 5 Very Helpful

The mean ratings for 1) the Goal Tree and 2) the ability to work backwards have been above 4 in each survey and approached 5 on more than one occasion.

Assessing the effect of strategic help is more difficult and in this paper has to be considered preliminary. Because it would have required changing the VALID curriculum, we did not

[12]Neither of the differences between groups on time of search or on proof length were statistically significant, but at sample sizes of 8 and 10 they would have to be alarmingly large to be so.

emphasize the strategic approach to proof construction that underlies CPT's help facility. In a future study we will teach one group with a new text emphasizing CPT's method of proof construction. Then we will compare its performance to 1) a group using the standard VALID curriculum and CPT without help, and 2) a group using the standard VALID curriculum and a CPT with help.

If it is somewhat difficult to separate the effects of the three features on overall performance, it is not hard to see how well they work in combination. The 1990 groups used the full CPT environment, while the Forwards group used a standard proof checker. The 1990 groups took a substantially harder midterm than the Forwards group, but with only 83% of the time. Yet their average score on the common part of the midterm was over 15 points higher, an improvement of almost 22%.

References

Anderson, J.R.. (1987). Production systems, learning, and tutoring. In D. Klahr, P. Langley, & R. Neches (Eds.), *Production System Models of Learning and Development.* Cambridge, MA: MIT Press, pp. 437-458.

Anderson, J.R. (1980). *Cognitive psychology and its implications.* San Francisco: W. H. Freeman.

Anderson, J.R., Jeffries, R. (1985). Novice LISP errors: Undetected losses of information from working memory. *Human-Computer Interaction*, 1, 107-131.

Anderson, J. R., Boyle, C.F. (1985). The geometry tutor. *Proceedings of IJCAI-85*, Los Angeles, CA.

Burkholder, L. (1989). Cumulative new software. *The Computers and Philosophy Newsletter.* Center for Design of Educational Computing, Carnegie Mellon University, 4.1, 48-89.

Croy, M. J. (1986). The current state of computer-assisted instruction for logic. *Teaching Philosophy* , 9, 333-350.

Gentzen, G. (1969). Investigations into logical deduction. In M.E. Szabo (Ed.), *The collected papers of Gerhard Gentzen.* Amsterdam: North Holland, 68-131.

Kalish, D., Montague, R., and Mar, G. (1980). *Logic: techniques of formal reasoning.* New York: Harcourt Brace Jovanovich, Inc.

Larkin, J. (1987). Display-based problem solving. In *The 21st Carnegie Symposium*, Hillsdale, N.J.: Erlbaum.

Newell, A., & Simon, H. (1972). *Human problem solving.* Englewood Cliffs, N.J.: Prentice-Hall.

Pressler, J. (1989). Computers in logic courses. In Ian Lancashire (Ed.), *Methodologies in humanities computing.* Philadelphia: University of Pennsylvania Press.

Pressler, J., and Scheines, R. (1988). An intelligent natural deduction proof tutor. *Computerised Logic Teaching Bulletin*. 1.

Scheines, R., and Sieg, W. (1993). The Carnegie Mellon Proof Tutor. In Judith V. Boettcher (Ed.), *101 Success Stories of Information Technology in Higher Education: The Joe Wyatt Challenge*. McGraw-Hill.

Sieg, W., and Scheines, R. (1991). Searching for proofs (in sentential logic. In Leslie Burkholder (Ed.),*Philosophy and the Computer*. Boulder: Westview Press.

Suppes, P. (Ed.), (1981).*University-level Computer-assisted Instruction at Stanford: 1968-1980*, Institute for Mathematical Studies in the Social Sciences, Stanford University.