PRODUCTIVITY IMPROVEMENT IN THE TEACHING OF PHILOSOPHY:

FINAL REPORT

by

Robert Laddaga, Lee Blaine, and Patrick Suppes

TECHNICAL REPORT NO. 295

May 22, 1978

PSYCHOLOGY AND EDUCATION SERIES

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

STANFORD UNIVERSITY

STANFORD, CALIFORNIA 94305

Table of Contents

Acknowledgments

The authors wish to acknowledge the contributions of the following people to the research described in this report:

The curriculum for the proof theory course was written by Wilfried Sieg under the guidance and direction of Georg Kreisel, and later revised by Ingrid Lindstrom. In addition to these people, Vladimir Lifschitz and Sten Lindstrom contributed to the on-line version of the curriculum. Gary Bower and Inge Larsen contributed to the on-line version of the curriculum for the course in the foundations of probability.

The programming needed specifically for these courses was done principally by Lee Blaine and James McDonald. The programming for the HELP system was done by David Ferris and Teri Pettit. Teri also wrote the curriculum modules for the HELP system and the introductory lesson.

The MISS audio system is the result of work by William Sanders and Gerard Benbassat. The work on automatic generation of intonation was done by Arvin Levine.

In addition, the authors wish to thank Wilfried Sieg for his contributions to this report.

The research described in this report was supported by a grant from the Exxon Education Foundation.

1 Objectives

The major objective of this project has been to examine the use of computer-assisted instruction (CAI) as a method to increase productivity in higher education. We quote from the project proposal:

> "The purpose of this project is to demonstrate that by the use of computer-assisted instruction the productivity of faculty members in philosophy may be increased by increasing their course loads. In particular, the purpose of the project will be met by implementing two upper division, undergraduate courses in philosophy that will be taught wholly at computer-based terminals. These two courses will be added to the regular teaching load of the two faculty members involved. The first course will be a foundational course on proof theory and will be taught by Professor Georg Kreisel. The second course will be an undergraduate course on the foundations of probability and induction and will be taught by Professor Suppes. In addition, each of the courses will ordinarily be offered every term, that is, three times a year, rather than simply one term per academic year. A subsidiary purpose of the project is thereby to increase the flexibility of the course offerings to undergraduate students at Stanford."

In the following sections of this report, we will discuss the implementation of the curriculum (see Section 2) and program (see Section 3) for these courses, and evaluate the courses with respect to the productivity argument in Section 5. In addition, Section 4 will discuss the use of the IMSSS system of computer generated speech to present on-line lecture material for the courses.

2 Curriculum

2.1 Proof Theory

The content of the of the proof theory course consists mainly of Goedel's famous, and now classical, incompleteness theorems, as formulated for the system ZF (Zermelo-Fraenkel) of set theory. The axioms of ZF and their intended models (segments of the cumulative hierarchy) are carefully described in the first part of chapter 1. In the second part we recall how informal mathematical (in particular, number theoretic) notions can be represented in a subsystem ZF* of set theory. ZF* is ZF without the axiom of infinity and of the same strength as arithmetic. The logical form of these definitions is analyzed. In the third part, attention is given to the problem of representing number theoretic functions and predicates given by informal recursion or induction. We show that SIGMA-recursive functions can be introduced in a definitional extension of ZF*.

The informal metamathematical arguments involved in the above considerations serve as the motivation for a more rigorous description of the syntax of ZF. That description is actually given in the first part of chapter 2. In the second part of the chapter, we use a novel approach which simplifies the description and comparison of formal theories. We analyze the syntactic objects as binary trees, instead of sequences. Pairing, rather than concatenation is the basic operation used to build up syntactic objects. A formal theory of elementary inductive definitions (TEM) for these trees provides a proper framework for describing formal theories, like ZF, and analyzing them. TEM, which is analogous to Peano-arithmetic, allows the definition of syntactic notions in a natural way. The associated principle of proof by induction permits a direct formalization of metamathematical arguments.

In Chapter 3 Goedel's First and Second Theorems are proved--assuming basic representability and derivability conditions. Some examples of nonstandard representations are given, which show that the derivability conditions are crucial to the Second Theorem.

The curriculum for the course is presented in two forms, a text for students, and an on-line version. For the text, see [1]. The VOCAL (see [2]) language was used to prepare the on-line version in a lecture style format with audio (see Section 4).

2.2 Probability Theory

As stated in the project proposal, the curriculum for the course in foundations of probability is derived largely from Chapter 3 of [3]. The on-line curriculum is being written entirely in VOCAL, (see [2]). The first segment of the course presents the formal Kolmogorov axiomatic theory for probability. We cover finite probability spaces, conditional probability, Bayes' Theorem and Bayes' postulate, independence, and discrete and continuous random variables.

After presenting the formal background, we discuss a series of foundational views on the interpretation of probability. Discussed and criticized are the classical definition of Laplace, confirmation views of Carnap and others, relative frequency interpretations, Bayesian subjective theories, qualitative theory, and propensity theory.

2.3 Student Aids

The proof theory and foundations of probability courses presume no knowledge of computer science, programming or interactive systems on the part of the student. Nor are these subjects part of the traditional curriculums of these courses. The student does, however, need to be instructed in the use of the interactive programs that constitute the courses. In the past, various manuals have been exclusively used to describe procedures for log-on, response input, and administrative matters, as well as definitions of the interactive commands for both the top level of the program and the proof checker.

We believe that the CAI environment itself is the best place to provide instruction in its use. For this reason, and because the instruction on the use of the system is in large part common to all the mathematically oriented courses, we designed and implemented an introductory lesson to cover most of the issues previously explained in the manuals (which are still provided for off-line reference).

The introductory lesson, in particular when dealing with course dependent material, utilizes a HELP system which allows the student to access remedial, enrichment, or administrative information at any point in the course. The HELP system has a program component which creates and manipulates a graph of topics, and VOCAL curriculum modules corresponding to the nodes of the graph.

An additional aid to students is the BROWSE MODE feature which allows students to review past work or look ahead to future work. Between BROWSE MODE and HELP, the curriculum author has available the machinery to provide more individualized assistance to students in need of further explanation, whether for review, or extensions of the current (relative to the student's position in the series of basic lessons) material.

3 Programming Efforts

3.1 Proof Theory

As a first step, TEM (see Section 2.1) was fitted straightforwardly into the the existing proof machinery, and the central results for the course were proved on the computer. Some of these proofs depended on establishing that, under given conditions, certain results are provable in the object theory. One way to do this (and the way we initially chose) is to axiomatize the provability relation of the object theory in the metatheory and to establish the result from these axioms. However, it is often far easier to simply derive the result directly in the object theory and then use this fact in the metatheory.

Such procedures were added to the proof checker system (EXCHECK) for the proof theory course.[1] Two inference procedures are involved: ZFSTART--for starting a derivation in ZF from the metatheory; and ZFFINISH--for finishing the derivation in ZF and returning to the metatheory. After starting a ZF derivation from the metatheory you may reference prior results from the metatheory or the metatheoretical part of the derivation. In the second example below, two lines in the metatheoretic part of the derivation are referenced from the ZF part of the derivation. There is a restriction on the form of the metatheoretic results that may be referenced from inside ZF: they must be atomic

[1] For a description of the EXCHECK system see [4].

formulas of the form ZF |- F or ZF* |- F. Conjunctions of such formulas are also allowed.

In the examples below, underlining indicates what the student actually types ('$' indicates pressing an 'enter' key). All the other text is the output of the proof checker, in response to student input.

Example 1:

```
Derive:
IF Y IS Z THEN ZF* |- |Y| = |Z|

*hyp$
HYP     (1)  Y IS Z
*zfs$TART
                ******************  ZF  ******************

     teq$  (2)  *|Y| = |Y|$
Will you wish to specify? (No) *$
Using *g$O
     zff$INISH
                ******************************************

2 ZFFINISH
        (3)  ZF* |- |Y| = |Y|
*1,3teq$  (4) *ZF* |- |Y| = |Z|$
Will you wish to specify? (No) *$
Using *g$O
*
```

Example 2:

```
Derive:
IF ZF |- x=y AND ZF |- y=z THEN ZF |- x=z

WP$  (1)  *ZF |- x=y \WP$\  (2)  *ZF |- y=z *\zfs$\TART
                ******************  ZF  ******************

     1,2teq$  (3)  *x=z$
Will you wish to specify? (No) *$
Using *g$O
     3zff$INISH
                ******************************************

3,1,2 ZFFINISH
        (4)  ZF |- x=z
*
```

3.2 Probability Theory

The foundations of probability course required more extensive development of input and output grammars than previous mathematically oriented courses developed at the Institute. The main reason for this is the need to use several theories in the derivations for the course, e.g., set theory, real number theory, and the particular probability theory of interest at a given stage in the course. For similar reasons, a new decision procedure for some real number inferences involving equality was needed and developed; it is called the ALGEBRA procedure.

The ALGEBRA command has two forms. In the simpler form the student types in any equation which is a valid formula of rational field theory and the program checks for validity. The second form allows the student to make any algebraically valid manipulation of a previous line which contains an equation. See below for illustrations of the use of the command. At present the ALGEBRA command works only for equalities and not for inequalities, order relations, inside quantified expressions or boolean expressions. Extensions to these cases would be very desirable.

The basis of the ALGEBRA command is the program REDUCE (see [5]). The REDUCE function REVAL takes as argument a symbolic algebraic term (e.g., 2*x + y*z) and 'reduces' it to a simple canonical form. For example the term x + x would be reduced to 2*x. Note that if t1 and t2 are two terms and t1 = t2 is algebraically valid then t1 - t2 = 0. Hence to use reduce to test an equation for validity REVAL is applied to the term t1 - t2 and the equation is valid if zero is returned. The completeness of this procedure is thus relative to the reduction of the term to zero by REDUCE.

When ALGEBRA is used to justify an algebraic manipulation of a previously given equation (which is not necessarily an algebraically valid equation itself)

(A) $s1 = s2$

to derive a new equation

(B) $t1 = t2$

the following use of REVAL is made to test for algebraically valid inference. We distinguish two cases. In the first case the manipulation is based on application of commutative, associative, distributive, inverse or identity laws alone (i.e., no cancellation of or multiplication by factors). In this case the inference is valid if and only if either

$$(s1 - s2) - (t1 - t2) = 0 \quad \text{or}$$

$$(s1 - s2) - (t2 - t1) = 0$$

is valid. In case (B) comes from (A) by multiplying (A) by some factor, m, in addition to applying the commutative, etc laws the inference is valid if and only if either of the following is valid

$$m*(s1 - s2) - (t1 - t2) = 0 \quad \text{or}$$

$$m*(s1 - s2) - (t2 - t1) = 0.$$

Below are some examples of the algebra rule, one which verifies an equation, another which applies algebra to a previous line, and the last an example from the probability course. As in the previous examples, underlining indicates what the student actually types (`$′ indicates pressing an `enter′ key).

Example 3:

```
*alg$EBRA   *P(A)*P(B)=P(B)*P(A)
ALGEBRA
        (1)  P(A)*P(B) = P(B)*P(A)
*
```

Example 4:

```
       (8)  *P(A) + P(B) = P(A U B)

*8alg$EBRA
Schematic form:
r+t = xl

New form *t=xl-r$

Multiplicative factor used to get new form *$
8 ALGEBRA
        (9)  P(B) = P(A U B) - P(A)
```

Example 5:

```
Derive:
P(0) = 0

*boole$   (1)  SↄO = 0
*1 Th. P1.2.3$
        (2)  P(S U O) = P(S) + P(O)
*2,(boole) re$
        (3)  P(S) = P(S) + P(O)
*3 alg$EBRA
Schematic form:
x = x + y

New form *y = 0$

Multiplicative factor used to get new form *$
3 ALGEBRA
        (4)  P(O) = 0
*** QED ***
```

4 Audio Implementation

The Institute's random access speech synthesis system (MISS: Microprogrammed Intoned Speech Synthesizer) is being used in conjunction with the courses in proof theory and probability. A description of the system as used in CAI contexts can be found in [6].

Research in prosody has been carried out with the aim of synthesizing sentences with an acceptable-sounding prosodic contour. We use only the phonetic information contained in individually recorded words together with certain syntactic facts that can be derived from the structure of the sentence or read out of the lexicon. See [7] for a description of the research involved in this approach to generated prosody.

5 Evaluation

The on-line version of the proof theory course has been offered through the Philosophy Department at Stanford each quarter since the spring of 1977. The course has been taken by staff members of the Institute, and begun by several students, with 2 students officially enrolled. As is evident from the table below, these enrollment figures do not appreciably differ from that of previous years. In addition, the course is offered three times per year instead of at most once per year.

Both proof theory students received incompletes, one of which is currently being discharged. We experienced the same phenomenon of maintaining enrollments with an increased drop rate in the early stages of offering set theory by computer. It appears that these courses require a breaking-in phase, during which the user interface is improved and program bugs are repaired. As can be seen in the table, the drop rate and proportion of A grades (for set theory) improves with time.

Another factor influencing the drop rate may be that the current computer versions of the courses differ from previous versions in that students are offered much more opportunity to practice proof techniques. Consequently, students do more 'graded' or checked work in the computer-based courses. In the set theory course, more informalized proof procedures and better student preparation may help account for the recent improvements in the drop rate.

A further note on productivity in connection with these computer-based courses is that they have the potential to increase teaching assistant productivity also. Formerly, the main responsibility of teaching assistants was to grade papers. Providing tutorial assistance to students was relegated to a secondary role. Since the computer routinely checks the validity of the steps in student proofs in these courses, teaching assistants can devote full time to helping students with individual problems. Thus they can give help of higher value to students in more classes, with the same total time spent per teaching assistant. The set theory and proof theory classes, for example, are served by a single teaching assistant, where two would formerly have been required.

The foundations of probability course is being offered for the first time this quarter, to one registered student. Given past experience, we expect enrollment and dropout statistics for this course to be roughly the same as those for set theory and proof theory.

TABLE I

Distribution of Grades

Year	A	B	Pass (or C)	No credit	Incom-plete	Total enrollment
			Set Theory Course, pre-computer:			
71-72	7	6	6	3	0	22
72-73	8	4	4	2	1	19
73-74	6	0	0	0	2	8
			Set Theory Course, computer-based:			
74-75	9	1	3	4	6	23
75-76	7	0	5	12	1	24
76-77	4	1	7	5	2	19
77-78*	9(15)	1(2)	2(5)	3(5)	0(1)	15(28)
			Proof Theory Course, pre-computer:			
71-72	0	0	0	0	0	0
72-73	3	5	0	0	0	8
73-74	2	0	0	0	0	2
74-75			not offered			
75-76	4	0	1	0	0	5
			Proof Theory Course, computer-based:			
76-77	0	0	0	0	1	1
77-78	0	0	0	0	1	1

* figures for 1977-78 are for autumn and winter quarters; figures in parentheses are projected to include spring quarter.

<u>References</u>

1. Wilfried Sieg and Ingrid Lindstrom, <u>Elementary Proof Theory</u>, in preparation

2. M. Hinckley, R. Laddaga, J. Prebus, R. Smith, and D. Ferris. <u>VOCAL: Voice oriented curriculum author language</u>. Stanford Calif.: Stanford University, Institute for Mathematical Studies in the Social Sciences, 1977.

3. Suppes, P. <u>Set Theoretical Structures in Science</u>; in preparation.

4. R. L. Smith and L. H. Blaine. A generalized system for university mathematics instruction. (SIGCUE Bulletin, 1976, 8(1), 280-288.)

5. Hearn, A. C. <u>REDUCE User's Manual</u>. Salt Lake City, Utah: University of Utah, Department of Computer Science, 1973

6. Sanders, W., Banbassat, G., and Smith, R.L., Speech synthesis for computer assisted instruction: the MISS system and its applications. <u>SIGCSE Bulletin</u>, 1976, <u>8</u>(1), 200-211.

7. Levine, A.; <u>Automatic Generation of Intonation in English</u>. 1977, Doctoral Dissertation, Stanford University, Stanford Ca.