DOCUMENT RESUME

ED 284 402 EC 200 220

AUTHOR Hofmeister, Alan M.; Ferrara, Joseph M.
TITLE Artificial Intelligence Applications in Special Education: How Feasible? Final Report.
INSTITUTION Utah State Univ., Logan.
SPONS AGENCY Special Education Programs (ED/OSERS), Washington, DC.
PUB DATE Jun 86
GRANT G008400650
NOTE 193p.; For related document, see EC 200 221.
PUB TYPE Reports - Research/Technical (143) -- Reports - Evaluative/Feasibility (142)

EDRS PRICE MF01/PC07 Plus Postage.
DESCRIPTORS Articulation Impairments; *Artificial Intelligence; Behavior Disorders; Behavior Problems; Classification; Computer Managed Instruction; Computer Software; *Computer Uses in Education; Decision Making; *Disabilities; Elementary Secondary Education; *Expert Systems; *Handicap Identification; Individualized Education Programs; Instructional Development; Learning Disabilities; *Special Education; Systems Development

ABSTRACT

The research project investigated whether expert system tools have become sophisticated enough to be applied efficiently to problems in special education. (Expert systems are a development of artificial intelligence that combines the computer's capacity for storing specialized knowledge with a general set of rules intended to replicate the decision-making process of a human expert.) To assess the feasibility of the technology, a series of prototypes was developed, in which a range of expert system development software tools and hardware systems was used. These prototype systems, which sampled administrative assessment and instructional problems, addressed: (1) classification of students as learning disabled; (2) classification of students as behaviorally disturbed; (3) classification of students as intellectually handicapped; (4) classification of students as having articulation problems requiring special education; (5) advice to teachers planning specific procedures to deal with behavior problems (Behavior Consultant); and (6) development of a second opinion of the appropriateness of the decision-making process used in the development of Individualized Education Programs (Mandate Consultant). Additional data are presented on the two prototypes that were taken through more extensive development and field testing. It was concluded that a need for the technology exists in special education and that it is possible to develop practical expert systems with the tools and research and development resources presently available. Nine appendixes comprising the bulk of the document are concerned with expert systems in relation to such topics as: (1) individual education program planning; (2) diagnosing, classifying, and treating learning disabled students; and (3) evaluating the development of such systems in education. (KM)

Final Report

Artificial Intelligence Applications in Special Education: How Feasible?

Grant #G008400650

UTAH STATE UNIVERSITY

June 1986

Principal Investigator: Alan M. Hofmeister

Co-principal Investigator: Joseph M. Ferrara

Funded by Special Education Programs

U.S. Department of Education

Table of Contents

Abstract

The question of primary concern in this project was: Have expert system tools become sophisticated enough to be applied efficiently to problems in special education?

The expert system, a technology within the field of artificial intelligence, was developed in response to the need for expertise in areas where field problems are complex and expert help is limited and expensive. Expert systems had been developed in industry, defense projects, and health science. There had been little research done on the application of the technology to education, and special education in particular. This project represented one of the first efforts to assess the potential of expert systems to solve some of the problems in special education.

To assess the feasibility of the technology, a series of prototypes were developed. These prototypes sampled administrative assessment and instructional problems in special education. In the prototype development process, project staff experimented with a range of expert system development software tools and hardware systems.

Two of the prototype systems were then taken through more extensive development and field testing. Extensive data on user reliability, decision validity, and user and administrative acceptability was collected.

It was concluded that there were important problems in special education that could benefit from the application of expert systems technology. It was also concluded that it was

possible to develop practical exportable expert systems with the tools presently available and with the limited research and development resources available to the field of special education. There was every indication that the cost of the software development tools would continue to decrease, and the power and flexibility of the hardware and software would continue to increase.

Introduction

During a presentation to Congress, former Secretary of Education Bell (1983) made the observation that "Too much computer software is simply electronic page turning, and it has little advantage over a well-illustrated book" (p. 4). The Secretary then called for the application of ". . . artificial intelligence to interact with the minds of learners" (p. 4). Unlike industry, medicine, and the defense fields, public education has done very little research and development on the applications of artificial intelligence to the problems faced by educators. There are several reasons why educators have not been active in this area.

First, the technical and personnel resources necessary for the development of artificial intelligence (AI) products have, until recently, been rare and expensive. Second, the long-term efforts necessary for AI product development did not fit the funding patterns for educational research. In referring to examples of effective expert systems (one type of AI product), Winston (1979) noted that no one should look at these systems without understanding the "years of team effort that have gone into translating the basic strategies into working, useful systems" (p. 273).

Expert Systems

Recently, the expert system area of AI has received a great deal of attention. An expert system is a computer program which attempts to replicate the decision-making and problem-solving

skills of knowledgeable and effective human experts.

Waltz (1983) reported that:

> The biggest AI news of the recent past has been the commercial introduction and industrial use of a number of AI systems, especially NL (Natural Language) and expert systems. This news is significant because (1) it has quieted critics who argued that AI would never produce useful results, and (2) the applications themselves have high intrinsic value. (p. 56)

If expert systems technology can be applied to special education's problem domain, a number of benefits are likely. First, the quality of the special education knowledge base might be improved. In addition, diagnostic and treatment resources could be significantly expanded. In discussing the value of AI products, Hayes-Roth, Waterman, and Lenat (1983) state,

> We can anticipate two beneficial effects. The first and most obvious will be the development of knowledge systems that replicate and autonomously apply human expertise. For these systems, knowledge engineering will provide the technology for converting human knowledge into industrial power. The second benefit may be less obvious. As an inevitable side effect, knowledge engineering will catalyze a global effort to collect, codify, exchange, and exploit applicable forms of human knowledge. In this way, knowledge engineering will accelerate the development, clarification, and expansion of human knowledge itself. (p. xii)

A third potential benefit of expert system development and

use is improved training. Where expert systems have been employed, users of these systems have experienced a training effect. Furthermore, specific efforts to modify expert systems for training purposes suggest that such modification can enhance the value of the systems.

The knowledge generation and clarification activities associated with expert systems development have research implications of considerable value. In perhaps the only well-validated AI program with significant implications for special education, DEBUGGY, the knowledge generation aspect was significant. The developers, Brown and Burton (1978), added considerably to our knowledge of student errors in arithmetic. With DEBUGGY, the user is trained to identify error patterns in students' attempts at arithmetic problems. In developing the program's knowledge base, Brown and Burton not only arrived at an estimate of the percentage of errors that were systematic (80 percent), they also documented the different types of systematic errors and their relative frequencies.

Expert System Development Tools

Because of a recent software development (the expert systems development tools), research in artificial intelligence has ceased to be the province of a few basic researchers working in well-endowed laboratories. These tools allow lesser-trained individuals to apply AI technology. This trend has been summed up as follows:

> Training in knowledge engineering usually requires several years of study at one of a handful of universities. A group

of us in the knowledge systems area at Xerox PARC is trying to shorten this training time. Our goal is to increase the impact and scale of knowledge engineering by simplifying the methods of knowledge programming and making them more widely accessible. (Stefik, et al., 1983, p. 4)

When the proposal for this study was written, a number of expert systems development tools were available, and software houses were promising that new tools would be released.

Problem Statement

The problem addressed by this study was that no literature or data existed which described the application of expert systems technology to special education. We didn't know if the application of this technology to special education's problems was feasible.

The question of primary concern in this project was: **Have expert system tools become sophisticated enough to be applied efficiently to problems in special education?**

Implicit in this question was the notion that such educational applications should require fiscal and time resources no greater than those normally available to researchers in special education.

Program Objectives

To answer this question, a series of activities were planned to accomplish three key objectives. The first objective of the AI feasibility project was to evaluate AI expert system development tools. The second project objective involved

designing small prototype systems to evaluate the potential of expert system authoring tools in special education applications. The third objective was to develop and test a practical expert system to help deal with one special education area.

Objective 1: Comparison of expert system tools. During the project year, staff completed a review of a variety of expert system tools. As a result of this work, three articles, which compare expert system tools, have been published. The articles were co-authored by project staff, graduate students and colleagues from other disciplines (Lubke, Ferrara, & Parry, 1985; Ferrara, Parry, & Lubke, 1985; Ludvigsen, Grenney, Dyreson, and Ferrara, 1986). Copies of these articles are appended to this report. Table 1 summarizes the tools reviewed by project staff and their opinions, which are detailed in the appended articles (see Appendices A, B, and C).

Objective 2: Development of prototype systems. Over the course of the two-year feasibility study, project staff have developed prototype systems. A few of these systems were, in retrospect, ill-conceived. The majority are either being fully developed (with other funding) or have the potential to be completed.

CLASS.LD. The earliest prototype system was CLASS.LD. The system was designed to give a second opinion on decisions to classify students as learning disabled. The first version was developed using Expert-Ease as the development tool (Hofmeister, 1983). The system was then revised, and further development continued using the more flexible and powerful M.1 system

(Ferrara & Hofmeister, 1984). CLASS.LD became the primary system to check the flexibility, power, and ease of use of different development tools. Appendix D (Hofmeister & Lubke, 1985) contains more detailed information on CLASS.LD.

CLASS.BD and CLASS.IH. These two prototypes were designed to test the modular structure of interrelated programs. CLASS.BD was designed to give a second opinion on classification decisions related to behavioral disturbance, and CLASS.IH was designed to provide similar information for intellectually handicapped classification decisions.

These are potentially large programs, and project resources did not exist to take them past their prototype forms. Work on CLASS.BD was recently continued as a part of a cooperative effort with the Utah SEA. The SEA is extremely interested in the system for three reasons: (1) the SEA's need to improve field decision making; (2) the need for an inservice training tool, and (3) the expert system's knowledge base is helping the SEA refine state regulations related to the classification of students as behaviorally disturbed.

CLASS.IH has been left as a rather primitive prototype because of a lack of resources to continue development.

CLASS.SP. A prototype system was developed to provide a second opinion regarding a student's articulation problems and the student's eligibility for special education services. This system was left as a prototype for lack of resources and limited field interest.

Mandate Consultant. This expert system was designed to

provide a second opinion on the appropriateness of the decision-making procedures used in the development of an IEP. One major purpose of the system was the reduction in the need for hearings to resolve conflicts between parent and school.

Because of the extensive interest shown in the prototype by SEA and LEA special education administrators, additional resources, in the form of a student initiated research grant, were obtained to take the system through extensive field testing. This field testing has been completed (see Appendix E, Parry & Hofmeister, 1986) for a summary of the field-test results.

Behavior Consultant. The Behavior Consultant was designed to explore the potential of expert systems with a more instructional emphasis. The previous prototypes had stressed the assessment and administrative aspects of special education. The Behavior Consultant prototype was designed to give advice to teachers planning specific procedures to deal with behavioral problems. The expert system was designed for both field consultant and clinical training applications (see Appendix F, Serna, Baer, & Ferrara, 1986). In our initial evaluation of the prototype, it appeared that the size and complexity of the program would require a mainframe rather than a powerful microcomputer. In the last review it appeared that recent developments in software, particularly the move from Lisp to the "C" language, would allow the use of microcomputers for this program. The program is being viewed as (1) an instructional planning and management tool for the teacher, and (2) a training tool for advanced practicum experiences. Development is continuing with this system, using training resources.

Objective 3: System testing and validation. Information regarding the validation of expert systems in education has been limited prior to this study. Hofmeister (in press, see Appendix G) describes a validation and expert system development model created for this project. This model has been employed in the development in a number of the prototype systems described in the previous section.

The validation of an expert system is a complex process. There are several ways in which to evaluate a system like CLASS.LD. First, there are issues of basic interuser reliability. Two users may, for example, look at the same data source, e.g., a student's file, and find different information. How well one knows a student may have an impact on a user's responses to the expert system's questions. Finally, the setting in which a system is used will affect its reliability. If CLASS.LD is used for a "private" consultation, one set of data may be provided. If the computer program is used to provide a second opinion at a staffing, a different set of data describing the same student may be entered. Reliability is then threatened by a variety of factors which have little to do with the internal function of the machine.

A second issue involves the validity of the system. Does its knowledge base accurately reflect the best thinking of content area experts.

Two additional issues address the system's impact on its environment. If a reliable and valid expert system is not used or if its advice is ignored, the value is limited. A system's

values can be measured in a variety of implementation situations. CLASS.LD2 could be used in training, field consultant, and program auditing roles.

This study's validation efforts addressed two issues with regard to CLASS.LD2. First, the degree to which initial data suggests agreement with content area experts was measured. Second, the efficiency of CLASS.LD2 in a training situation was evaluated. Two articles describe this project's findings recognizing these issues. The article by Martindale, Ferrara, and Campbell (submitted for publication) describes the validity study. The article by Ferrara, Prater, and Baer (in press) provides preliminary information regarding CLASS.LD2's ability to function in two different training situations. Both articles are appended to this report.

Summary

Development Software

The introduction of expert system technology to special education is clearly feasible.

Tools. Early PC-based tools were, at best, crude instruments for building simple and costly systems. When this project began, mainframe tools were slow, cumbersome monsters which were known to eat a VAX for breakfast. Other large expert system tools required $150,000 dedicated Lisp machines. In the two-year project period, this somewhat grim state of affairs has changed dramatically. Current PC systems are fast, efficient, and capable of producing professional expert systems which address substantive problems. Mainframe systems are much faster

and require much less RAM than their predecessors. The Lisp version of S.1 had to drag along five megabytes of Franz Lisp baggage. The newest version of the same shell is written in "C." As a result, other users of our VAX are less likely to send us electronic hate mail when we log on.

Every month has seen the introduction of new shells and more powerful versions of existing systems. In addition, costs have gone down. We are paying less for more powerful software.

Special Education Prototypes

The prototype systems developed by this project suggest that there is no reason to delay the development, testing, and implementation of expert systems for special education. Existing tools have all the power we need to do a first-class job in the area of diagnosis, planning, and instruction. As tools become more powerful, we can add more "bells and whistles," but solid, workable systems are certainly possible today at a reasonable cost. It should be kept in mind that most of our prototype systems were developed by special educators with little or no computer programming background. Although, to date, only two of these prototypes (Mandate Consultant and CLASS.LD2) are being fully developed, the success of these prototypes suggests that complete systems are possible in each of the prototype areas.

CLASS.LD2. CLASS.LD2 has been demonstrated to be a valid, usable expert system. If implemented, the system may be the answer to some of the current overclassification problems in the area of LD classification. To suggest, however, that CLASS.LD2 is the solution to anything is premature and goes well beyond the

data. Current SEP-funded research, as well as testing efforts being made by individual state education agencies, is increasing our certainty regarding the efficiency of CLASS.LD2 in both field consultant and training roles.

Potential Problems

There are several potential problems which may delay or restrict the general distribution and use of expert systems. First, schools may view hardware costs as prohibitive. Even our smallest prototypes will NOT run an Apple IIe with 64k of RAM. A minimally configured system requires an IBM-PC compatible computer, with at least 512k of memory, and two disc drives. Such machines are not commonly found in public school classrooms. They are, however, becoming common in administrative settings in public schools.

Second, software licensing fees are very high by educators' standards. TEKNOWLEDGE, for example, charges $250 per disc for distribution copies of programs that include their software tools. This high cost may limit expert systems building efforts to areas where a substantial economic benefit is immediately apparent (as with CLASS.LD2 and Mandate Consultant). Behavior Consultant, and other systems designed to directly benefit teachers and students, may have to wait for costs to go down. There can be little doubt that costs will continue to drop.

Conclusions

Project Objectives

All project objectives have been met or exceeded. We were

fortunate to acquire additional university funds to supplement the federal resources for the project. As a result:

1. We acquired and field tested more development tools than originally planned and budgeted for;
2. More prototypes were developed than planned;
3. More of the prototypes were taken to a field-test level than planned, and
4. More refereed publications were prepared than the project objectives called for.

In addition to meeting the formally stated project objectives, a number of important secondary objectives were met, including:

1. Three doctoral students in special education received extensive internship experiences;
2. The Department of Special Education added important advanced clinical training experiences to their training program through the incorporation of CLASS.LD.
3. The project was largely responsible for three SEA's and two other universities making investments in the development of other expert systems in special education.
4. The project was largely responsible for an increase in university commitment to further research and development in the application of technology to the needs of the handicapped.

Feasibility of Expert Systems in Special Education

We entered the project with considerable uncertainty regarding the practical value of expert systems in special education. We finished the project certain that expert systems can make a contribution.

Our certainty is the result of the following observations.

1. During the project, two major advances in software and one major advance in hardware allowed expert systems of a practical size to be developed and run on powerful, yet modestly priced, microcomputers. If we had been restricted to mainframe applications, the technology would have been too costly and inflexible for many field applications in special education.
2. After experimenting with a range of prototypes which sampled administrative, assessment, and instructional problems in special education, we collected sufficient information to suggest that the technology could be adapted to selected, important problems in special education.
3. After taking two of the prototypes and developing them to a point where we had extensive data on product validity, user acceptance, and administrative support for their use, we became convinced of the potential effectiveness and acceptability of expert systems in special education. The fact that four SEAs are now investing their resources in the further development

and adaption of these expert systems provides substantive evidence of field acceptance.

4. The observation that some of the secondary benefits of expert systems development were greater than originally hypothesized, provided additional data to support the value of the technology. One secondary benefit related to the value associated with the building of a knowledge base for an expert system. To build a knowledge base for an expert system, you are required to collect existing knowledge in a specific area and organize that knowledge so that it can be aplied to the solution of pressing field problems. It is the analysis and synthesis of existing knowledge so that it rationally directs field decision making that allows the development of expert systems to make a substantive contribution to the need to translate our research findings into practice.

Perhaps the major secondary benefit related to the training value of expert systems. The original intent of an expert system was the emulation of the field consultant. An expert system (1) models the decision-making processes of an expert, and (2) makes its decision-making process and the associated rationale overt. This modeling and transparency of reasoning combine to form an effective training function in areas where training is difficult and expensive. Advanced clinical training is one such area, and administrative

decision making is another. Expert systems could very likely be justified for their training value alone.

References

Bell, T. H. (September 28, 1983). Computers and education. Statement made to Subcommittee on Investigations and Oversights, House Committee on Science and Technology, U.S. House of Representatives, Washington, DC.

Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science, 2, 155-192.

Ferrara, J. M., Parry, J. D., & Lubke, M. (1985). M.1 and Expert-Ease: Two expert system authoring tools for the microcomputer (Technical paper). Logan, UT: Utah State University, Artificial Intelligence Research and Development Unit.

Ferrara, J. M., Prater, M. A., & Baer, R. (in press). LD.Trainer: Modification of an expert system for complex conceptual training. Educational Technology.

Ferrara, M. J. & Hofmeister, A. M. (1984). CLASS.LD2: An expert system for classifying learning disabilities. [Computer program for the IBM PC]. Logan, UT: Utah State University, Artificial Intelligence Research & Development Unit.

Hayes-Roth, F., Waterman, D. A., Lenat, D. B. (1983). Building expert systems. Reading, MA: Addison-Wesley.

Hofmeister, A. M. (in press). Formative evaluation in the development and validation of expert systems in education. Computational Intelligence.

Hofmeister, A. M. (1983). CLASS.LD: An expert system for

classifying learning disabilities. [Computer program for the IBM PC]. Logan, UT: Utah State University, Artificial Intelligence Research and Development Unit.

Hofmeister, A. M. & Lubke, M. M. (1985). Expert systems: Implications for the diagnosis and treatment of learning disabilities. Learning Disability Quarterly, 9(2), 124-132.

Lubke, M. M., Ferrara, J. M., & Parry, J. D. (1985). Expert systems in the individual education program process (Technical paper). Logan, UT: Utah State University, Artificial Intelligence Research and Development Unit.

Ludvigsen, P. J., Grenney, W. J., Dyreson, D., & Ferrara, J. M. (1986). Expert system tools for civil engineering applications. Expert Systems in Civil Engineering Symposium Proceedings, Seattle, WA: TCCP Council/ASCE.

Martindale, E. S., Ferrara, J. M., & Campbell, B. W. (Submitted for publication). Accuracy of Class.LD2: An expert system for classifying learning disabled students. Computers in Human Behavior.

Parry, J., & Hofmeister, A. M. (1986). The development & validation of an expert system for special educators. Learning Disability Quarterly, 9(2), 124-132.

Serna, R. W., Baer, R. D., & Ferrara, J. M. (1986, May). An expert system for behavior analysis. Paper presented at the Meeting of the Association for Behavior Analysis, Milwaukee, WI.

Stefik, M., Aikins, J., Balzer, R., Benoit, J., Birnbaum, L., Hayes-Roth, F., & Sacerdoti, E. (1983). The arthitecture of expert systems. In F. Hayes-Roth, D. A., Waterman, & D.

B. Lenat (Eds.), Building expert systems. Reading, MA: Addison-Wesley.

Waltz, D. (1983). Artificial intelligence: An assessment of the state-of-the-art and recommendations for future directions. The AI Magazine, 4(3), 55-67.

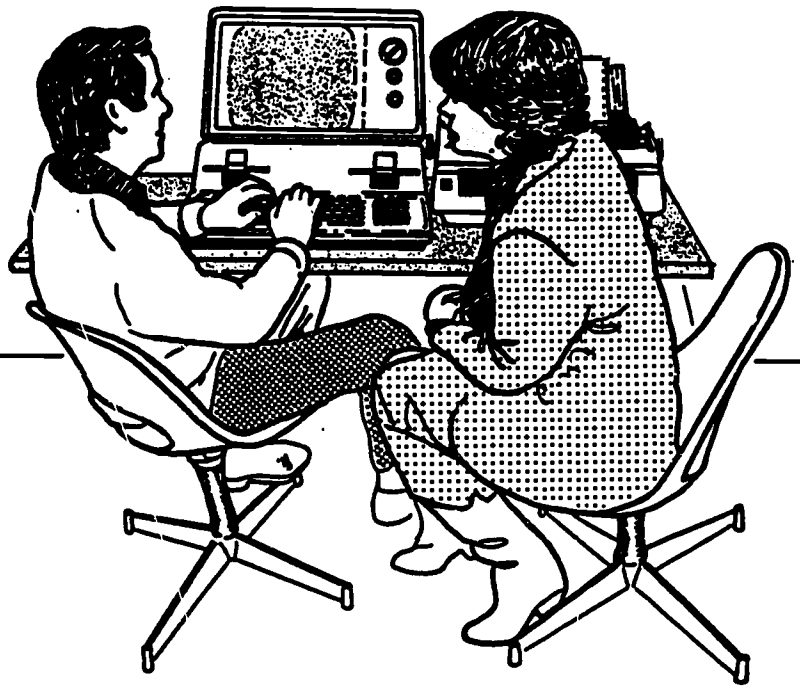
Winston, P. H. (1979). Artificial intelligence. Reading, MA: Addison-Wesley

Appendix A

Expert Systems in the Individual Education Program Process

Technical Paper

Expert Systems in the Individual Education Program Process



Margaret M. Lubke
Joseph M. Ferrara
James D. Parry
May, 1985

Artificial Intelligence Research & Development Unit
Developmental Center for Handicapped Persons
Utah State University
Logan, Utah 84321

Expert Systems in the Individual Education Plan Process

Multidisciplinary teams must develop individual educational plans (IEPs) for handicapped children (Education for All, Sec. 1.[a] [19]). The purpose of an IEP is primarily to guide the delivery of instructional services to a handicapped child (Dudley-Marling, 1985). The process of developing an appropriate instructional plan begins with collecting test and observational data. This information is used to determine each child's current level of performance. A planning team then proceeds to develop goals and objectives, which should match the student's performance. A review of the research has identified several problems which are associated with this element of the IEP process.

Problems in Moving from Data to Objectives

One such problem is related to the quantity and quality of information describing student performance. Thurlow & Ysseldyke (1979) found that a great deal of data describing student performance is collected, but much of it is technically inadequate and irrelevant. For example, student observational data, which is collected before an IEP meeting, often fails to operationalize behavior, appropriately quantify behavior, or list antecedent and consequent events. These limited observational records have little value for program planners. They are not specific enough to direct the development of goals and objectives.

Besides inadequate data, multidisciplinary teams often collect information irrelevant to instructional planning. Norm-

referenced tests, used frequently in public schools to evaluate performance, can be considered instructionally irrelevant. A norm-referenced test produces a score that reflects how an individual's performance compares with the performance of other individuals. For a test to be instructionally relevant, an individual's performance must be assessed in absolute terms.

Criterion-referenced instruments, rather than norm-referenced instruments, assess student performance in specific, precisely defined content areas using absolute terms (Borg & Gall, 1979). Since criterion-referenced instruments can point out specific performance deficits, criterion- referenced tests can, therefore, be more useful to program planners than norm-referenced instruments to program planners.

An additional problem is that many instructional planners have difficulty moving from data collection to writing instructional objectives. Translating criterion-referenced test data into prescriptive objectives is a difficult task. The task, despite its level of difficulty, is critical to appropriate program planning. A student's program plan should directly relate to his current performance.

Authors and publishers of many criterion-referenced tests attempt to make the job of translating test data into prescriptive objectives easier by providing tables which reference specific objectives to test performance. For example, Connolly, Natchman, & Pritchett (1976) provide such reference tables for the Key Math Diagnostic Arithmetic Test.

In spite of the Key Math developers' efforts to make the test prescriptive, Goodstein, Kahn & Cawley (1976) reported that Key Math has utility only as a preliminary screening instrument for assessing areas of strength and weakness in general mathematics achievement. Goodstein, et al. (1976) felt that the usefulness of the Key Math for diagnosis of mathematical disabilities and the prescription of specific intervention tactics remained limited. Furthermore, Goodstein et al. (1976) described Key Math objectives as too broad-based for most teachers to adequately develop a prescriptive program likely to meet individual student's needs.

Often skilled planners require more detailed and time-consuming, criterion-referenced test data as well as additional information to write suitable objectives (Colburn & McLeod, 1983). Many times, unskilled planners don't even know when to ask for additional information.

Although academic objectives are an important part of most IEPs, social skills must also be considered. For an IEP to be appropriate, objectives which relate to social skills must also be tied to a student's performance. This means that planners must translate observational data into objectives for social/emotional behavioral prescriptions. Since acceptable and unacceptable student behavior in the classroom often covers a much greater range of circumstances than those in an academic area, the problems associated with social/emotional elements of program planning can become very intricate.

There are at least two issues that limit the likelihood that planners will write appropriate instructional objectives in both academic social/emotional areas. First, inappropriate data are often collected. Second, planners often lack expertise that allows them to translate good data into prescriptive objectives. These two issues are interrelated because persons unfamiliar with handling data appropriately cannot request adequate information. Planners need adequate information to write appropriate objectives. Without this information, implementation of the IEP is severely hampered. Failing to correctly implement an IEP can be considered the most critical detriment to appropriate programming for a handicapped child (Gerardi et al., 1984).

Artificial Intelligence: A Possible Solution

The field of Artificial Intelligence, specifically expert systems, may hold solutions for the problems identified in the research.

Artificial intelligence is the part of computer science concerned with designing intelligent computer systems; that is, systems that exhibit the characteristics we associate with intelligence in human behavior--understanding, language, learning, reasoning, solving problems, and so on (Barr & Feigenbaum, 1981, p. 3).

Artificial intelligence systems intended to replicate decision-making by knowledgeable and experienced humans are called expert systems. An expert system is typically set up to engage the user in a dialogue. This dialogue, in many ways,

selected, and (c) a set of criterion-referenced test items designed to obtain missing student information.

The MTI asks questions to gather information and then analyzes the user's answers by comparing them to the rules in the knowledge base. When necessary, the system prints out additional criterion referenced test items to gather more specific information about a student's performance.

Behavior Consultant (BC)

The Behavior Consultant (BC) program applies expert system technology to student behavior-problems in the classroom. Ultimately, two videodisc components will be associated with the BC expert system. The overall structure for BC will include: (a) an initial videodisc component designed to teach effective skills for observing student behavior; that is, to teach educators and others to be the "eyes and ears" of the system, (b) an expert system component designed to evaluate data from the user regarding student behavior-problems and suggest strategies for addressing the behavior-problems in the classroom, and (c) a second videodisc component designed to teach effective implementation of the behavior strategies recommended by the expert system.

Currently, the expert system component of BC is in the developmental stages while the videodisc components are in the planning stages. The current version of BC is designed as a microcomputer-based system. Because of the complexity anticipated for later versions of the expert system, it will

parallels the type of conversation a person might have with an expert consultant. The computer is programmed to ask the user questions to detail the problem or situation (Barr & Feigenbaum, 1981). For example, a well-known medical system for physicians is MYCIN (Davis, Buchanan & Shortliffe, 1975). With MYCIN the user inputs data into the computer information on the characteristics of the patient's bacterial cultures and the patient's symptoms. The computer is programmed to match the patient's data with information in the program on the characteristics of bacterial cultures and then, based on programmed logic, present a disease diagnosis.

Expert Systems and IEP Planning

Two prototype expert systems, Math Test Interpreter (Lubke, 1985) and Behavior Consultant (Ferrara & Serna, 1985) have been developed to test the feasibility of applying expert systems technology to the task of translating test and observational data into prescriptive objectives.

Math Test Interpreter (MTI)

The Math Test Interpreter (MTI) is designed to combine student information, results from the Key Math diagnostic Arithmetic Test (Connolly et al., 1976) and additional program generated criterion-referenced test data to produce a prescription for program planning in the area of mathematics.

The knowledge base of the MTI contains several components: (a) a set of rules to guide the consultation, (b) a master-set of objectives from which review and instructional objectives are

ultimately be moved to a mainframe computer. This paper will show examples from the current BC expert system prototype. The basic structure employed in the current version of BC will also be used in later versions transferred to the mainframe computer.

Expert Systems Functioning

Consultations

Both prototype systems described above engage the user in a dialogue. For example, in the case of MTI, the user supplies information about the student such as grade, data of examination, mental ability, past math performance, IQ, chronological age, priority ratings for content areas, as well as item scores on the Key Math test. When consulting BC the user's answers to a series of questions describe the behavior-problems, the condition in which the behavior takes place, and the condition in which the teacher will attempt to modify the problem behavior.

Figures 1a and 1b present examples of a typical consultation with each of the expert systems.

Knowledge Base Rules

Both expert systems were written using a computer language that organizes human knowledge into a series of rules. These rules have two components: an "if component, or antecedent component, and a "then" component, or consequent component. When the conditions in the antecedent component of the rule match the conditions in the user's problem description, a conclusion in the consequent component of the rule is invoked. Figure 2 present an "if-then" rule taken from the MTI.

What was the student's age in months at the time the test was administered?

>> 120.

What was the student's grade level at the time the test was administered? (Enter the score as a real number, for example 3.5 or 6.8.).

>> 5.1.

Based on your information about the student's intellectual functioning (IQ), this student would be considered:

-normally functioning (that is, above 75)

-intellectually handicapped (about 55-75)

-severely-intellectually handicapped (below 60)

>> intellectually handicapped.

The three basic areas covered by the Key Math test are Content, Operations and Applications.

Please rate the CONTENT area in terms of priority, using a 1, 2, 3, with a "1" being the highest priority.

>> 2.

Please rate the OPERATIONS area in terms of priority,

>> 3.

Please rate the APPLICATIONS area in terms of priority,

>> 1.

How much time is devoted to mathematics instruction per day for this student? (Please enter the average amount of time per day in minutes).

>> 40.

Figure 1a. Typical consultation with the Math Test Interpreter

What is the behavior which you wish to stop or retard?
>> talking-in-class.

Is there a good behavior which is incompatible with talking-in-class? (For example, speaking normally is incompatible with yelling. Working on math worksheets, on the other hand, is NOT incompatible with making strange noises).
>> no.

How quickly must the talking-in-class be stopped?
1. RIGHT AWAY! This talking-in-class is an immediate threat to the physical well-being of someone. (e.g., head-banging).
2. Quickly. This talking-in-class is making my life and/or the lives of the other kids miserable. (e.g., screaming).
3. There is no big rush, but I'd like to stop the talking-in-class as soon as I can. (e.g., talking in class).
4. The talking-in-class is only an annoyance. There is no need for a major effort to control it. (eg., nose-picking).

>> 3.

What consequent do you think is maintaining the talking-in-class?
>> teacher-attention.

Can the teacher control the teacher-attention which appears to be maintaining the talking-in-class behavior?
>> yes.

On a scale of 1 to 50, does the student enjoy being in the classroom where the talking-in-class is taking place?

He/she finds this to be an aversive place. — The classroom is among this child's favorite place

1----------50

>>40.

On a scale of 1 to 50, does the student enjoy the activities taking place in the classroom while the talking-in-class is happening?

He/she finds these activities to be aversive — These activities are among this child's favorites

1----------50

>> 40.

Figure 1b. Typical consultation with the Behavior Consultant

If IQ = intellectually handicapped, and

age = AGE

AGE = > 13, and

PAST PERFORMANCE = poor

Then EXPECTED PROGRESS = 7.5 months

Figure 2. An example of a rule from Math Test Interpreter.

are illustrated in Figure 3. When one system seeks a value for an expression within a rule, it will first check to see if it already knows the value. If it has previously asked or inferred the expression's value it will be stored in the system's global memory. If the system looks in the global memory and finds a value for the expression, it will stop looking and use that value to test the rule. If a value for the expression is not found in the global memory, the system will then seek rules which conclude with a value for the expression. The system will then test this next set of rules to identify the value of the expression. Finally, if there are no rules which conclude with a value for the expression, or if all such rules fail, the system will ask the user if he/she knows the expression's value.

Figure 4 shows how BC tests a rule, which concludes that time-out is an appropriate procedure for modifying "throwing objects" behavior. The steps used by BC in this situation are detailed below.

1. BC seeks a value for the expression "bad-behavior" in the global memory (the global memory contains information already acquired by asking the user questions).
2. Since the computer already had a value for bad-behavior stored in the global memory it returns the value "throwing" for the expression "bad-behavior." The expression "b" found in this condition indicates a variable. Thus, the value "throwing" is associated with the variable "b."

MTI and BC programs contain factual and heuristic rules. Factual knowledge consists of information that can be documented, such as state and federal regulations and proven hypotheses (Feigenbaum & McCorduck, 1983). An example of a strictly factual rule would involve the calculation of the student's mental age based on the IQ and the chronological age input by the user.

Heuristic knowledge captures the "rule of thumb" experiences of humans. In special education, such knowledge might come from expert diagnosticians or instructors. Referring to the rule presented in Figure 2, it may be the heuristic opinion of several experts that under the circumstances described in the antecedent parts of the rule, a student would likely make seven and one-half months progress.

Back Chaining

Both MTI and BC expert systems use back chaining. This is a problem-solving technique which works backward from hypothesized conclusions to known facts. Thus, the expert system can determine if rules succeed or fail. For example, when testing the rule stated in Figure 2, MTI first seeks a value for the expression for "IQ." Then values for the expressions "AGE" and "PAST PERFORMANCE" will be sought. Thus, if all the conditions of a rule are confirmed, the conclusion is confirmed and the rule succeeds. Conversely, if any of the conditions in a rule cannot be confirmed, the conclusion cannot be confirmed and the rule fails.

There are three ways in which MTI and BC seek values for the expressions within rules. These systems' value-seeking behaviors

3. Next, BC seeks a value for the expression "speed(B)," that is, a value for the "speed at which throwing must be stopped." BC finds rules concluding with "then speed(B)" and tests the first condition of the first rule.

4. BC seeks a value for the expression "bad-behavior" in the global memory.

5. BC returns the value "throwing" for the expression "bad-behavior."

6. Because quickness does not have a value in the global memory the system seeks a value for the expression "quickness(B)" by asking the user a question.

7. BC returns the user's value "1-real-fast" for the expression "quickness(B)." Because the user's value "1-real-fast" does not match the value "4-real-slow", this rule fails.

8. BC enters "1-real-fast" in the global memory as the value for the expression "quickness(B)."

9. BC considers the next rule concluding "then speed(B)" and seeks a value for the expressions "bad-behavior" and "quickness(B)" in the global memory.

10. BC returns "throwing" and "1-real-fast" as the values for the expressions "bad-behavior" and "quickness(B)" respectively.

11. The rule concluding "then speed(B) = fast" succeeds.

12. The condition of the original rule stating "and speed(B) = fast" is confirmed.

13. BC seeks values for the expressions in the remaining conditions of the rule, that is, "time-out-ratio" and "child-characteristics." It finds these values either in the global memory, by testing rules, or by asking the user.

14. The rule succeeds or fails depending on the outcome of the expressions in the premises.

Possible System Outcomes

Inadequate information. Both BC and MTI can identify situations in which the data provided by the user either is inconsistent, lacks validity, or is incomplete. In situations where this is the case, the system will alert the user and suggest that additional information should be obtained. MTI will, in certain cases, print out specific criterion-referenced test items to be administered to the student. Two options are available at this point; the user may continue with the consultation, or the user may abort the consultations and gather the information needed to make a complete diagnosis and prescription. Figure 5 describes the output of this section of the MTI consultation process.

Objectives. Both MTI and BC are designed to print objectives for IEP development. MTI presents the user with two types of objectives, review objectives and instructional objectives. Review objectives cover those isolated skills a student appears to be lacking. The instructional objectives correspond with the level of the test items that fall at or above

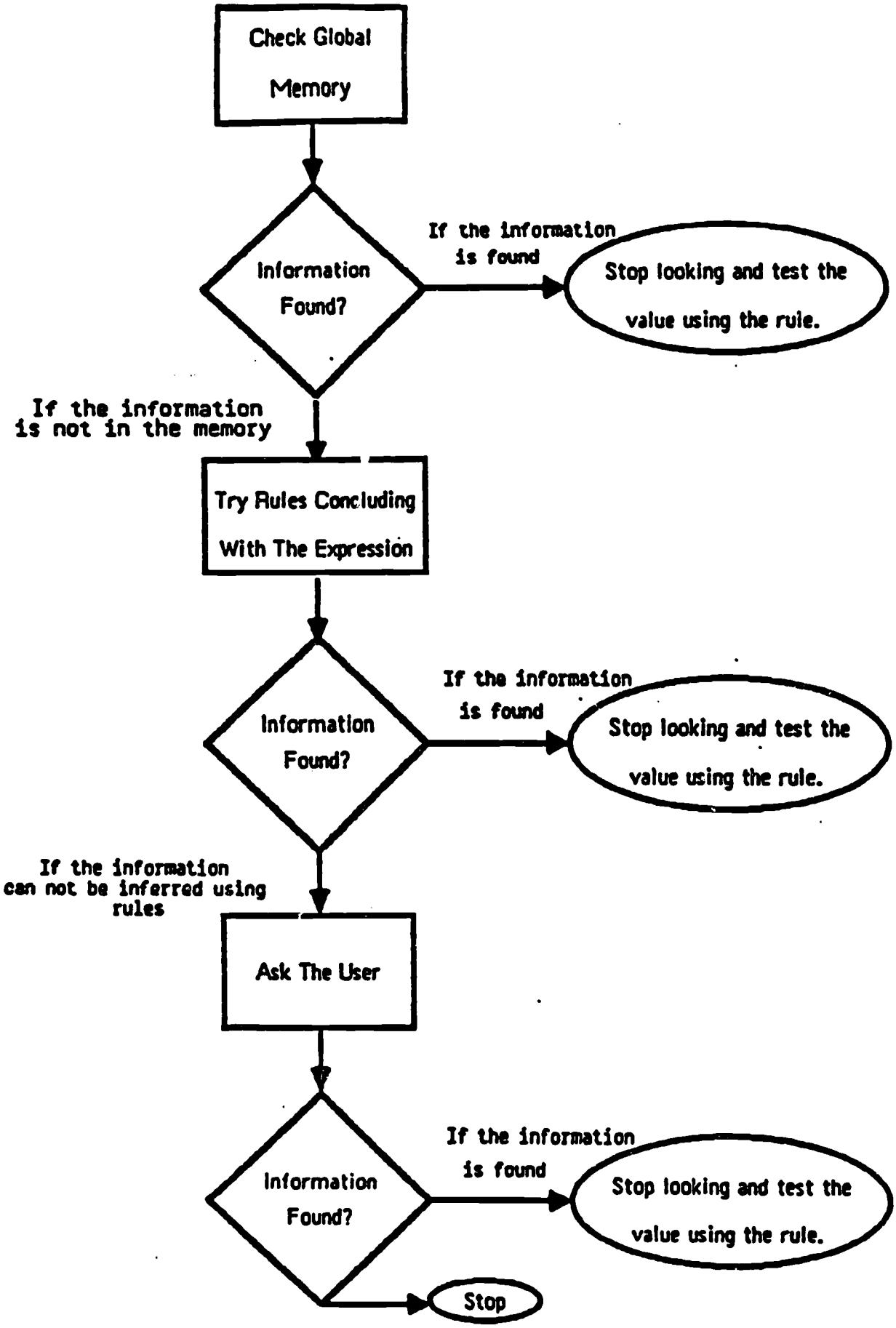


Figure 3. Three ways to obtain a value for an expression.

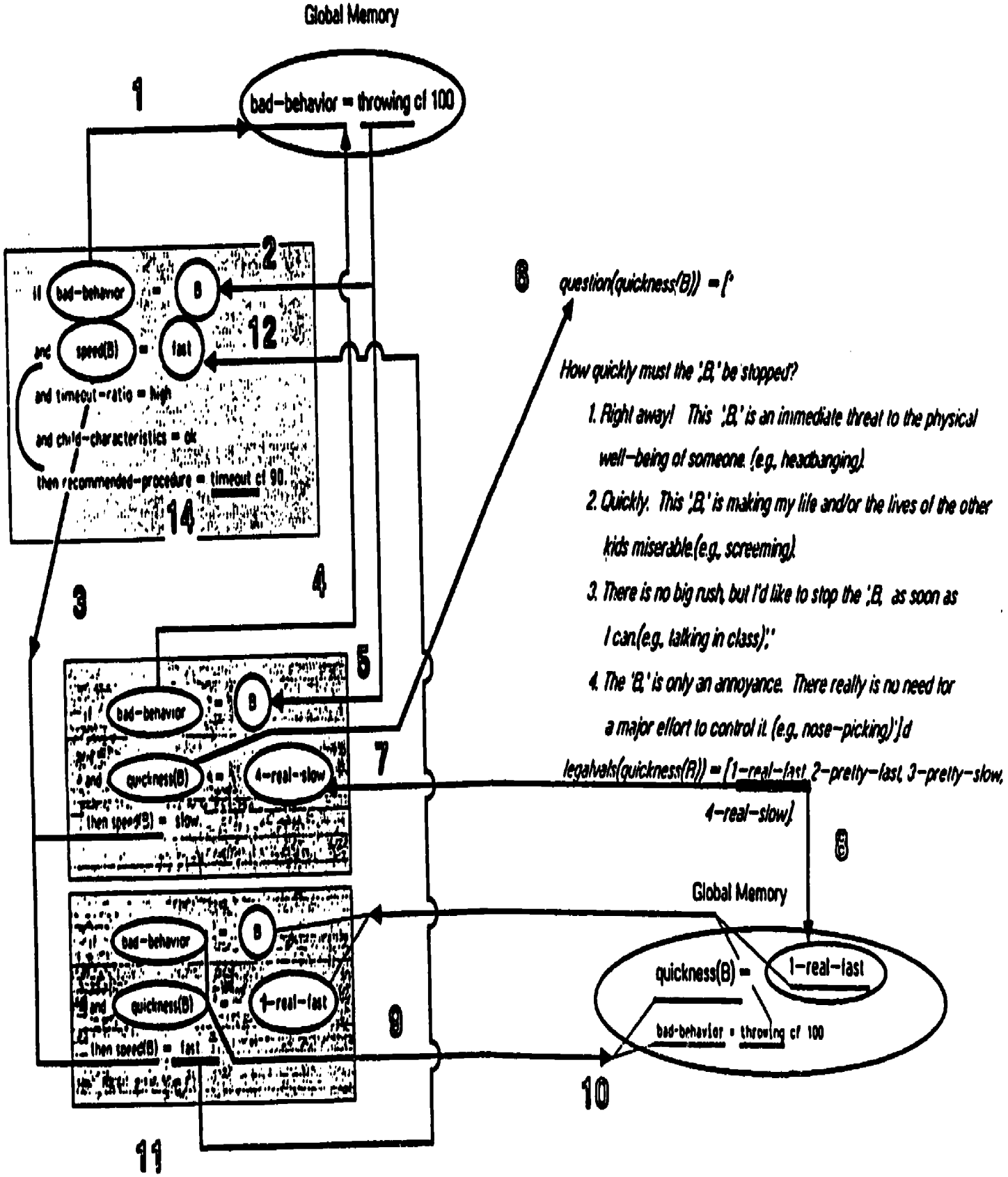


Figure 4. Example illustrating how Behavior Consultant tests a rule.

What is the value of item 4? (Enter a "1" if the student responded correctly to this item and a "Ø" if he/she failed to respond correctly).

>> 1.

What is the value of item 5?

>> 1.

What is the value of item 6?

>> 1.

What is the value of item 7?

>> Ø.

In order to determine the appropriate prescriptive objectives dealing with Identification and Addition of Coins and Currency, I need more information. It would be helpful if you would administer the following short criterion-referenced check with your student.

(Prints out check-test items on the printer).

Would you like to STOP and continue with this consultation at a later time or would you prefer to GO ON with the consultation without using the additional information.

>> STOP.

Figure 5. An example of a request for additional data from Math Test Interpreter.

the student's ceiling level. A student's ceiling level occurs when he/she has made three consecutive errors on the Key Math test. Figure 6 shows the screen display of the type of message presented to the user at the end of the consultation along with the appropriate objectives. These review and instructional objectives would be appropriate to include as short-term objectives in a student's IEP.

BC provides terminal objectives as well as an explanation of step-by-step procedures for achieving those objectives. When the entire BC system is finally completed in 1989, the computer will use an interactive videodisc to teach an instructor how to implement the suggested procedures.

Other General Features of Expert Systems

The M.1 authoring system (Teknowledge, 1984) was used to create both MTI and BC. M.1 has several features which make the system particularly attractive to educators.

1. The "TRACE" facility allows the user to monitor the computer logic as it attempts to provide advice.
2. The "WHY" facility allows the user to question the program about "why" it asked a question. The machine's response can be an M.1 rule, an English translation of an M.1 rule, or a reference to state and/or federal law.
3. The "SHOW" facility allows the user to query the program at any point in the consultation regarding its intermediate conclusions.

The student needs to review the following objectives:

3-A The student will verbally state in "cents" the value of a penny.

3-B The student will count out up to 20 pennies and verbally state the amount as ____ pennies.

4-C Given a nickel and five pennies the student will pcik out any combination of cents up to ten cents upon verbal instruction.

The following objectives are considered appropriate for the student's instructional level:

6-A The student will be able to match each amount with the correct corresponding amount written when using a dollar sign and a decimal point, given a worksheet with amounts written in cent form in one column.

6-B The student will be able to match the numerical values of money word values, such a $.50 with fifty cents.

6-C The student will be able to write the following dictated amounts using a dollar sign and decimal point--$1.20, $.75, $2.68, $.62, and $.05.

7-A The student will be able to select the quarter when directed to do so, given sets of coins containing pennies, nickels, dimes and quarters.

7-B The student will be able to indicate that another name for a quarter is 25 cents when shown a quarter.

7-C The student will be able to identify the one coin which is worth 25 cents when given sets of coins containing pennies, nickels, dimes and quarters.

Figure 6. Output of prescriptive objectives from the Math Test Interpreter

Summary and Conclusions

Expert systems and special education. Recent efforts to apply expert systems to the problems in special education represent a truly different approach. Considerable research is needed before firm conclusions can be reached regarding the value of expert systems for handicapped children. There are, however, some preliminary findings that indicate that this line of research is warranted (Hofmeister & Lubke, in press).

1. Evaluations conducted with prototypes indicate that these systems can perform as well as humans in specific areas.
2. Some of the problems faced by special educators are similar to the problems faced in other disciplines where expert systems have been successful.
3. The process of assembling and organizing knowledge bases for expert systems is a productive activity in its own right. The development of the "if-then" rules of a knowledge base clarifies existing knowledge and identifies areas where knowledge is needed.

Integrating expert systems into the IEP process. Paper compliance is relatively easy, that is, given the time-factor, fulfilling the "letter of the law" in writing IEPs can be accomplished with little effort. But, making a difference in the quality of a handicapped child's education, is a challenge that involves fulfilling the "spirit of the law." It is anticipated that expert systems like MYCIN and BC can upgrade the

quality of the IEPs produced for handicapped children. With appropriate, clearly-stated objectives providers can plan daily instruction lessons that relate directly to the identified needs of their students. Today's handicapped children have "more rights" but the price they pay should not be "less quality education" (Gerardi et al., 1984).

REFERENCES

Barr, A., & Feigenbaum, E. A. (Eds.). (1981). The handbook of artificial intelligence (1st ed.). Los Altos, CA: William Kaufman, Inc.

Borg, W. R., & Gall, M. D. (1979). Educational research: An introduction. (3rd. ed.) NY: Longman, Inc.

Colburn, M. & McLeod, J. (1983). Computer guided educational diagnosis: A prototype expert system. Journal of Special Education Technology, 6(1), 36-39.

Connolly, A. J., Nachtman, W., & Pritchett, E. M. (1976). Key Math Diagnostic Aritnmetic Test, Circle Pines, MN: American Guidance Service, Inc.

Davis, R., Buchanan, B., & Shortliffe, E. (1975). Production rules as a representation for a knowledge-based consultant-program. (Report STAN-CS-75-519 Memo AIUM-226). Palo Alto, CA: Stanford University, Computer Science Department.

Dudley-Marling, C. (1985). Perceptions of the usefulness of the IEP by teachers of learning disabled and emotionally disturbed children. Psychology in the Schools, 22, 65-67.

Education for all handicapped act of 1975, Statutes at large, vol. 89 (1975).

Feigenbaum, E. A., & McCorduck, P. (1983). The fifth generation. Reading, MA: Addison-Wesley.

Ferrara, J. M., & Serna, K. (1985). Behavior Consultant: An expert system for the diagnosis of social/emotional behavioral problems, [Computer program]. Logan, UK: Utah State University.

Gerardi, R.J., Grohe, B., Benedict, G. C., & Coolidge, P. G. (1984). I.E.P.--more paperwork and wasted time. Contemporary Education, 56(1), 39-42.

Goodstein, H. A., Kahn, H., & Cawley, J. F. (1976). The achievement of educable mentally retarded children on the Key Math Diagnostic Arithmetic Test. Journal of Special Education, 10(11), 61-70.

Hofmeister, A. M. (1984). Development of a microcomputer/videodisc aided math instructional management system for mildly handicapped children. Final report to the U.S. Department of Education, Office of Special Education, Project #G008101536. Logan, UT: Utah State University.

Hofmeister, A. M., & Lubke M. M. (in press). Expert systems: Implications for the diagnosis and treatment of learning disabilities. Learning Disability Quarterly.

Lubke, M. M. (1985). Math Test Interpreter: An expert system for interpreting the Key Math Diagnostic Arithmetic Test, [Computer program]. Logan, UT: Utah State University.

Teknowledge, Inc. (M.1), (1984). 525 University Avenue, Palo Alto, CA 94301-1982, 415/327-6600.

Thurlow, M., & Ysseldyke, J. E. (1979). Current assessment and decision-making practices in model programs for learning disabled students. Learning Disability Quarterly, 2, 15-24.

Appendix B

Expert Systems Authoring Tools for the Microcomputer: Two Examples

Expert Systems Authoring Tools for the Microcomputer: Two Examples

Joseph M. Ferrara, James D. Parry, and Margaret M. Lubke

Artificial intelligence (AI) is one of the most exciting areas within the field of computer science. Much of the current interest in AI has been a result of the practical success of computer-based expert systems (Buchanan and Shortliffe, 1984; Clancey and Shortliffe, 1984; Feigenbaum and McCorduck, 1983; Weiss and Kulikowski, 1984; Winston and Pendergast, 1984). Expert systems are computer programs which provide users with advice. During a consultation with an expert system, the user answers computer-generated questions. Those answers are used to test a series of knowledge-based rules. When enough information is provided to allow the system's advice-related rules to succeed, a potential solution to the user's problem is provided by the system (Weiss and Kulikowski, 1984).

Although expert systems have been successful in a variety of fields, two key factors have delayed their use in education (Hofmeister and Ferrara, in press). One factor has been a lack of equipment. Most computers owned by public schools do not provide an ideal expert system authoring environment. Expert systems have typically been developed using mainframe computers or dedicated LISP machines like the XEROX 1108 or the Symbolics 3600. Few public schools have access to that type of hardware.

Programming problems have also limited the use of expert systems in education. Until recently, expert systems were usually written in LISP and PROLOG. These languages are flexible and lend themselves to the logical rule representation required by expert systems. They are not, however, easy to learn or to use. The development of a usable expert system often has required years of work by a team of skilled programmers and content-area experts (Sleeman and Brown, 1982). Public education has lacked the resources which the development of expert systems has demanded.

Improvements in microcomputer hardware coupled with the development of microcomputer-based expert system authoring tools have made the use of AI programs within public schools appear more feasible today (Hofmeister and Ferrara, in press). AI authoring tools allow non-LISP programmers to build expert systems. These authoring tools may provide users with a unique language or approach to program generation. In addition, debugging aids, designed to assist the programmer, may be part of the authoring tool.

Microcomputer-based expert system authoring tools can be used effectively to train novice knowledge engineers. In addition, these tools can be used to produce small scale, practical, expert systems which may be useful in solving a variety of educational problems (Hofmeister and Ferrara, 1984a; Hofmeister and Ferrara, 1984b; Parry and Ferrara, in press).

Expert-Ease (Export Software International Ltd.), *M.1* (TEKNOWLEDGE, Inc.), and *EXPERT-2* (Miller Microcomputer Services) are authoring tools designed for use with microcomputers. What follows is a discussion of two of these authoring tools: *Expert-Ease* and *M.1*. This discussion is not a formal review but rather an informal reflection based on the experiences of the Special Education AI project staff at Utah State University.

The goal of the Special Education AI project at Utah State University is to test the feasibility of using expert systems to solve immediate problems in special education. Prototype programs in the areas of diagnosis, classification, program evaluation, classroom management, and videodisc control are currently in various stages of development and testing.

As a result of our project we have had an opportunity to use *Expert-Ease* and *M.1*. Readers should keep in mind that the comments which follow are not those of AI experts, but rather reflect our subjective judgments after several months of work with these two authoring tools.

Computer Hardware

M.1 and *Expert-Ease* are made for IBM-PC and IBM-PC XT microcomputers. Both *M.1* and *Expert-Ease* can be used with either a monochrome or color monitor. *M.1*, however, is designed to take advantage of a color display, while *Expert-Ease* is not. *M.1* operates under PC-DOS, and we have found little or no problem running *M.1* on PC-compatible machines. *Expert-Ease*, on the other hand,

The authors work at Utah State University's Development Center for Handicapped Persons in Logan, Utah. Joseph M. Ferrara is a Research Associate; James D. Parry and Margaret M. Lubke are Research Assistants.

uses UCSD and has caused problems for us on several IBM-compatible machines. Both systems are quicker and easier to use with a hard disk machine (more about this later).

Learning to Use the Tools

Expert-Ease is simple to learn and use. One project staff person lightheartedly contends that it takes about 40 minutes to learn to use *Expert-Ease* and that within five hours you are about as proficient as you are likely to get. While this is clearly an exaggeration, it isn't far from reality. We believe that a person with limited computer experience can learn the system in less than one day. Everything one needs to know about *Expert-Ease* is contained in the manual which accompanies the program. The manual provides information on program installation, a careful tutorial giving step-by-step instruction in system use, and additional information on knowledge organization and advanced (larger and more complex) applications.

M.1 takes longer to learn than *Expert-Ease*. A four-day workshop, designed to teach students to use *M.1*, is provided by TEKNOWLEDGE. In addition, *M.1* novices should plan on additional practice before they use *M.1* for serious expert system development. This is not to say that *M.1* is horribly difficult. Even staff members with limited computer experience were able to become competent *M.1* users within a reasonable amount of time.

During training, TEKNOWLEDGE provides three volumes of well-organized materials: (a) Sample Knowledge Systems; (b) Training Materials (illustrated lecture notes); and (c) Reference Manual and User's Guide. The formal workshop instruction, combined with the *M.1* program materials, was viewed by the project staff as effective training.

In addition to learning the syntax of *M.1* and *Expert-Ease*, a novice user of either system would benefit from training in knowledge representation (Hayes-Roth, Waterman, and Lenat, 1983). We believe that knowledge representation skills facilitate the design and production of improved expert systems. In our judgment, these skills are not covered adequately by either the *M.1* or *Expert-Ease* training packages.

Creating a Program

Users developing programs with *Expert-Ease* employ a four-step process: (1) identify possible answers; (2) identify critical attributes for use in discriminating between differing examples; (3) write questions which are designed to help the system assign values to critical attributes; and (4) enter examples of problems. *Expert-Ease* then induces a logic matrix which determines and controls the presentation of appropriate questions to the user. The user's responses to those questions lead through the logic matrix to the "expert" conclusions. Explanations for the conclusions reached by the system can be identified by reviewing the logic matrix generated by *Expert-Ease*. In short, once appropriate outcomes, attributes, questions, and examples have been entered, the system does the work.

A single *Expert-Ease* program can handle a problem with up to 30 critical attributes and 300 examples. Larger problems can be addressed by chaining together several expert systems and having each system deal with a component of the larger problem. When this is done, however, the system must load each sub-program. Without a hard-disk system, this can be a long and noisy operation.

Our staff found creating *M.1* programs to be more complex. In *M.1* the programmer must identify a system goal and then develop a set of rules which will allow the system to question potential users, make inferences based on user responses, and arrive at conclusions which achieve the system's goal. *M.1* is much more flexible and feature-laden than *Expert-Ease*. As a result, debugging a large *M.1* system is a complex job.

To make this job easier, *M.1* provides programmer-friendly facilities; these include: (a) a series of error messages which identify syntax problems; (b) a command which displays a system's intermediate conclusions; (c) a command which allows the programmer to print out a history of the system's use of rules throughout each consultation; and (d) a command which provides an on-screen collection of useful information throughout the consultation.

Use of a hard-disk system makes authoring *M.1* programs faster. *M.1* programs are not usually written while an authoring tool is running. Instead, a text editor (we use *WordStar*) is used to actually write the program. The program is then submitted to *M.1* for analysis and debugging. When using a floppy disk system, the process of moving large files from *WordStar* to *M.1* and back is time consuming. The PC-XT's hard-disk speeds this process.

Consulting an Expert System

During a consultation with an *Expert-Ease* program, questions are displayed on the screen in the sequence dictated by the logic tree and the user's answers to the program's questions. When the system has collected enough information to reach a conclusion, advice is printed for the user. When *Expert-Ease* is unable to arrive at a conclusion, the

user is alerted and the program can quickly be changed to accommodate the new case.

Since the user may question the program in a number of ways, a consultation with an *M.1* program can be more complex than an *Expert-Ease* consultation. For example, an *M.1* program can provide the user with information about the content-related rules which have caused specific questions to be asked. Furthermore, an *M.1* user can halt a consultation and have the system display its intermediate conclusions. In addition, the *M.1* "trace" command can be used to print a list of the rules used throughout a consultation.

Expert Systems in Education

In the United States, only a few AI programs have been designed for use in schools. Furthermore, the educational market is still limited by hardware costs. PC-XTs are not commonly found in public school classrooms.

Additional information on microcomputer-based expert systems may be obtained from:

- TEKNOWLEDGE Inc. (for *M.1*)
  525 University Ave.
  Palo Alto, CA 94301-1982
  Tel. (415) 327-6600
- Jeffrey Perrone and Assoc., Inc. (for *Expert-Ease*)
  3685 17th Street
  San Francisco, CA 94114
  Tel. (415) 431-9562
- Miller Microcomputer Services (for *EXPERT-2*)
  61 Lake Shore Road
  Natick, MA 01760
  Tel. (617) 653-6136

We believe that current problems limiting the use of microcomputer-based expert systems in schools are not insurmountable and that within the next few years expert systems will play an important role in American education. □

References

Buchanan, B.G., and Shortliffe, E.H. (Eds.) *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.* Reading, MA: Addison-Wesley, 1984.

Clancey, W.J., and Shortliffe, E.H. (Eds.) *Readings in Medical Artifical Inteiligence: The First Decade.* Reading, MA: Addison-Wesley, 1984.

Feigenbaum, E.A., and McCorduck, P. *The Fifth Generation.* Reading, MA: Addison-Wesley, 1983.

Hayes-Roth, F., Waterman, D.A., and Lenat, D.B. (Eds.) *Building Expert Systems.* Reading, MA: Addison-Wesley, 1983.

Hofmeister, A.M., and Ferrara, J.M. Expert Systems and Special Education. *Exceptional Children*, in press.

Hofmeister, A.M., and Ferrara, J.M. (1984a). *Artificial Intelligence Applications in Special Education: How Feasible?* Grant Award to Utah State University No. 023 CH 40219.) Washington, DC: Special Education Programs, Department of Education, 1984a.

Hofmeister, A.M., and Ferrara, J.M. *A Study of Selected Microcomputer-Based Artificial Intelligence Systems in Special Education.* (Grant Award to Utah State University No. 023 HH 40051.) Washington, DC: Special Education Programs, Department of Education, 1984b.

Parry, J.D., and Ferrara, J.M. The Potential of Computer-Based Expert Systems for Special Educators in Rural Settings. *Small School Forum*, in press.

Sleeman, D., and Brown, J.S. (Eds.). *Intelligent Tutoring Systems.* London: Academic Press, 1982.

Weiss, S.M., and Kulikowski, C.A. A *Practical Guide to Designing Expert Systems.* Totowa, NJ: Rowman and Allanheld, 1984.

Winston, P.A., and Pendergast, K.A. (Eds.) *The AI Business: Commercial Uses of Artificial Intelligence.* Cambridge, MA: MIT Press, 1984.

This article was prepared with the assistance of funds from Special Education Programs, Department of Education Grant G008400650; Project Officer: Jane Hauser. Principal Investigator: Alan Hofmeister.

Forthcoming Articles

Among the articles scheduled to be published this spring in *Educational Technology* are the following:

- A Call for Action to Improve the Design of Microcomputer Instructional Courseware.
- Design Considerations for Planning a Computer Classroom.
- An Analysis of Computer Software Preferences of Preschool Children.
- Interactive Video: Fifty-One Places to Start—An Annotated Bibliography.
- Making Computers Work in the Writing Class.
- How Are Today's Elementary Schools Using Computers?
- Individualizing Learning with Computer-Based Instruction.
- Improving the Meaningfulness of Interactive Dialogue in Computer Courseware.

In addition to these and other precedent-setting articles, the magazine will publish the comments of its regular Columnists, plus reviews of new products and professional literature, technology news, notes on new products and services, and numerous other regular features.

Appendix C

Expert System Tools for Civil Engineering Applications

Reprinted from
Expert Systems in Civil Engineering Symposium Proceedings,
TCCP Council/ASCE
Spring Convention, Seattle, WA, April 8-9, 1986

Expert System Tools for Civil Engineering Applications

Phillip J. Ludvigsen, William J. Greaney[1], Del Dyreson[2],
and Joseph M. Ferrara[3]

INTRODUCTION:

In the past, expert system development was a monumental undertaking reserved for major universities and corporate giants. Fortunately advances in microcomputers coupled with a more pragmatic understanding of how expert system technology can be applied have initiated a new era in user developed expert systems. Development time, that took multi-man years, now takes months if not weeks. Programming, which required highly technical computer skills, can now be accomplished by novice programmers with the aid of software tools. Accessibility to AI (artificial intelligence) expertise, that was only available on the university campus, is now available (if only indirectly) through customer service, support, and training. All these changes manifest in one more factor - the cost. Projects that once required major public and/or private funding can now be undertaken by small companies and even individuals.

Expert system software tools have played a major role in expediting program development, however, they do not offer a panacea to all problems which require expertise. It is important to know that some problems should not and possibly can not be solved by current expert system technology. For problems that can be solved with current technology, consideration must be given to the design of the tool and how it relates to your particular problem. Each tool, as with any software, has advantages vs. limitations which must be evaluated before project development begins. The final decision on which tool is "best" (most appropriate) is dictated by various factors, such as flexibility, user support, documentation, and of course cost.

[1] Graduate Research Assistant, Professor, Respectively, Dept. of Civil and Environmental Engineering, Utah State Univ., UMC 41, Logan, UT 84322

[2] Associate Professor, Computer Science Department, Utah State Univ., UMC 42, Logan, UT 84322

[3] Associate Research Professor, Artificial Intelligence Research & Development Unit, Dept. of Special Education, Utah State Univ., UMC 68, Logan, UT 84322

The purpose of this paper is not to endorse any one particular software tool to build civil engineering expert system applications, but rather to emphasize their particular advantages as well as limitations from a civil engineering standpoint. The information presented is vintage 1985 and one should be aware of current changes in program features and price. Caveats aside, the body of this paper is based on hands-on experience and should prove useful.

Brief History:

Expert systems tools, sometimes called authoring tools, or more recently shells, have a relatively short history. Basic research of developing a tool to aid in building an expert system is approximately ten to fifteen years old. Much of the early work was done in the field of medical diagnosis. From this work, a tool named MYCIN emerged. It used "if-then" production rules, certainty factors and backward-chaining inference, thus setting the standard which many current tools follow today (1).

Teknowledge (Palo Alto, CA), probably the largest and oldest "expert systems company" has a corporate history of approximately five years. Experience in developing expert system tools and applying them is somewhat limited, especially in the field of civil engineering. Fenves, Maher, and Sriram in their paper "Expert Systems: C.E. Potential" highlight future uses of expert systems but the lack of current applications is apparent (2).

It is interesting to note the evolution of these tools and how recent advancements affect solving problems within the civil engineering domain. But first, lets take a look at what distinguishes civil engineering problems from other kinds of complex problems.

Problem Domain:

Civil engineering exhibits an extremely wide, as well as deep problem domain. The sheer diversities of disciplines involved and complexities encountered are self-evident. Because of this, civil engineering expert systems and thus the tools to build them must be extremely flexible. The ideal tool for building civil engineering systems would allow for the following:

* Complex mathematical manipulations within the tool (Including scientific functions plus canned algorithms, ie. statistics, integration, etc.).
* Various forms of knowledge representation (Not just "if-then" production rules).
* Various inference strategies (Not just "backward-chaining").
* Simple calls to other programs or expert systems written in ANY programming language.
* Natural language interfaces.
* Unlimited degree of expert system explanation.

* Extensive development environments (eg. intelligent editors, debuggers, graphics, and help facilities).

Unfortunately, no tool available today allows for all of the above. Many optimistic sales and customer service people will tout "we can't do that directly, but we can show you ways to work around it" or "our next version is slated to have that improvement". Upon hearing such statements, beware! Given the time and money one can "work around" or wait for anything, but the time or money might not always be available.

A good example of what appears to be a universal limitation of current tools is the inability to handle complex mathematical manipulations directly within the tool. The standard solution is to call (sometimes referred to as "hook") a module written in some common programming language to return the desired calculated data. However, you might have to call this module or others many times within an iterative solution process. This can slow execution down tremendously. In some cases, you can use knowledge engineering (programming) "tricks" for greater efficiency. The drawback to this approach is that your solution logic is dictated more by the tool's limitations than by the problem.

Even though current tools fall short of the ideal, the future looks bright. Expert systems and the tools to build them are heavily dependant on hardware speed and memory. This is why we see special machines designed just for AI work. Fortunately, advancements in hardware design are bringing tremendous computing power and thus making meaningful expert system tools available to desk top computers.

Tools:

Seven tools are investigated (EXPERT-EASE, INSIGHT, M.1, RuleMaster, EXPERT, ROSIE, and S.1). They represent a diversity of complexity, flexibility and cost. EXPERT-EASE, INSIGHT, and M.1 are suitable for micro computers while RuleMaster is a medium size tool suitable for super mini's. EXPERT, S.1, and ROSIE can be considered large, main frame software.

The following criteria are used to investigate important features of each tool.

* Approximate cost.
* Ability to handle complex mathematics.
* Ability to interface with other software.
* Explanation facilities.
* Overall friendliness.
* Documentation.
* User Support.

As means of a brief summary, Table 1 compares each evaluated tool according to the above criteria. The

EVALUATED EXPERT SYSTEMS TOOLS

	SYSTEM SIZE						
	SMALL			MEDIUM	LARGE		
ATTRIBUTE	EXPERT-EASE	INSIGHT II	M.1	RULEMASTER	EXPERT	ROSIE	S.1
Proprietary Interest	Human Edge Software (Palo Alto, CA)	Level Five Research Inc. (Melbourne Beach, FL)	Teknowledge (Palo Alto, CA)	Radian Corp. (Austin, TX)	Rutgers Univ. (New Brunswick, NJ)	Rand Corp. (Santa Monica CA)	Teknowledge (Palo Alto, CA)
Phone	800-624-5227	305-729-9046	415-327-6640	512-454-4797		213-393-0411	415-327-6640
Approx. Cost	$695 (IBM-PC)	$495 (IBM-PC)	$5000 (IBM-PC)	$10,000 (IBM-PC) $25,000 (VAX)	No Charge	$200 (VAX)	$50,000 (VAX)
Ability to handle Complex Math	NO	YES (directly)	YES (indirectly)	YES (indirectly)	NO	NO	YES (indirectly)
Ability to interface with other software	NO	YES (directly)	YES (indirectly)	YES (directly)	NO	NO	YES (indirectly)
Explanation	NONE	Extensive	Limited	Extensive	Limited	Very Limited	Extensive
Overall Friendliness	Buddies	?	A Friend	A Friend	Fair-Weather Friend	Fair-Weather Friend	Real Buddies
Documentation	Outstanding	?	Sufficient	Meager	Meager	Sufficient	Sufficient
User Support	Limited	Available	Available	Available	None	None	Available

Table 1. Expert system tools comparison.

following offers a bit more detail.

Expert-Ease:

Expert-Ease (Expert Software International Ltd) is probably the smallest of the tools evaluated. It is designed to aid the use in quick development of small prototypes. This Pascal based tool features an automatic induction routine. One sets up a decision table and Expert-Ease translates it to Pascal code which can only be executed from within Expert-Ease. The user really has very little control over the inference strategy (dedicated forward-chaining). If Expert-Ease sees fit to ask a certain question first, the programmer (knowledge engineer) can not get at the Pascal code to over-ride the tool's decision.

This tool does not handle mathematical functions nor will it run on several IBM compatible machines. Expert-Ease does not make allowances for interfacing to other software, plus it does not have any explanation facilities. Due to these limitations, Expert-Ease can not be considered for large complex problems which involve mathematics, this rules out most engineering problems. However, Expert-Ease could be used to develop skeletal logic structures involved in solving larger problems. It has outstanding documentation and it is easy to use. But for large engineering expert systems this tool will not handle the load.

Other descriptions and evaluations of Expert-Ease are available (3, 4, 5).

Insight II:

Insight II (Level Five Research) is the newest of all the tools evaluated. Unfortunately, we were not able to receive an evaluation copy in time for this paper. So, attributes such as overall, friendliness and quality of documentation are not evaluated here. Without hands on experience with Insight II, little can be said about its limitations and shortcommings, however, Insight II appers to have some powerful facilities at an attractive price ($495).

Insight II is a Pascal based program which boasts a menu - driven development environment, links to other programs, confidence factors, tiered explanation, the ability to produce "run-only" end user versions, the ability to address 2000 rules, the speed of a compiler based system, and believe it nor not, complex mathematical functions which are intrinsic to the system. Insight II is the only tool evaluated which directly incorporates transcendental math functions (ie. cos, sin, tan, etc.). For such a low price, this tool might be just the ticket, but just how well everything fits together remains to be seen.

M.1:

M.1 (Teknowledge Inc.) is considered by many to be

the Cadillac of PC-based expert system tools. The reasons for this analogy is 1) its price ($5000 - down from last year's price of $10,000); 2) it has many features that were once found on only mainframe system tools; and 3) Teknowledge offers first class user support (training and consulting). Teknowledge has gone to great lengths to produce a professional piece of software, but just as a Cadillac can be inappropriate for certain jobs, so can M.1.

M.1 (version 1.3) is a Prolog based tool which interprets english-like production rules. The form of inference used is backward chaining. Forward chaining can be simulated but the result is somewhat awkward. Because M.1 acts as an interpreter, rules are acted on much slower than a compiler based system. This can be a limiting factor, especially if you have to constantly "hook" out to other software. M.1 can not handle complex mathematics or interface with other software directly from within itself. One must customize the provided interfaces, write interfacing software in assembly language or "C" (Oh Boy!), and finally link the whole thing together with M.1 to get a new executable version. This is not a trivial task, considering most civil engineers are not fluent in assembly language or "C".

The explanation facilities are somewhat limited. When asked why?, M.1 will either give a programmer supplied explanation of just the rule which caused the query, or it will trace the rules (only by number) which were used to reach that point in the run. Since most expert systems reason within networks, it is impossible for a knowledge engineer to write one single explanation of a particular rule that will be in context. Some tools use a "tiered" explanation in an attempt to establish context, M.1 does not.

M.1 does not have an extensive development environment, in fact it does not have its own editor to make permanent changes or addition to the knowledge base. One must leave M.1 in order to use your own text editor or word processor. But M.1 does have nice tracing and debugging facilities. It also has the ability to produce "run-only" end user systems. This feature is particularly attractive to those who wish to diseminate their work but can not afford to buy numerious copies of the expert system tool.

The documentation is sufficient but not impressive when one considers the cost of the tool. About half of the documentation contains example expert systems. These examples are nice to have, but it would be nicer if each M.1 command was defined along with numerous examples of just how that command might be used. Included is a helpful section on how to develop an expert system from proposal to turn-key delivery. This "how to" section offers some good program development advice, unfortunately most of this rather large section has very little to do with using M.1 directly.

There is some question whether or not M.1 can be used to build a significant expert system (>500 rules). The answer is maybe. M.1 can not address more than 200 rules at one time however, allowances are made to "shuffle" in and out groups of rules as needed. Here again, the interpreter nature of M.1 makes this process painfully slow.

M.1 is a powerful but expensive tool. Complex systems can be built but certain inconveniences (interfacing difficulties and slow execution) must be considered limiting. We are anxiously awaiting version 1.4 due out in early 1986.

Other descriptions and evaluations of M.1 are available (6, 7, 8).

RuleMaster:

RuleMaster (Radian corp.) is somewhat of an enigma. The tool is not just one program but rather three distinct entities: 1) Radial is a highly structured, pascal-like language; 2) Rulemaker is an induction routine which translates "examples" (logic tables) into Radial code (similar to Expert-Ease); and 3) the User interface, a sophisticated menu-driven collection of editors, tools, and various applications which help in the building of RuleMaster programs. What is puzzling about RuleMaster is its apparent lack of a separate control structure (inference engine). It is generally accepted that a separate control structure is one of the things that make an expert system - an expert system (9). If one works with RuleMaster, it becomes clear that the control structure is up to Rulemaker and/or the programmer (Knowledge engineer). This aspect of RuleMaster is truly a double-edge sword. On the positive side, the programmer is forced to structure the expert knowledge into modules that are easily updated by Rulemaker as well as the programmer. On the other side, trying to produce the effect of anything besides forward-chaining (ie. backward chaining) is practically impossible. Fortunately, many engineering problems are well structured and can be solved via forward-chaining inference.

RuleMaster handles complex mathematical functions by means of the usual "hook" to a separate program. What sets RuleMaster apart from the other tools evaluated is its ability to run under Unix or Unix-like operating systems. Since Unix can handle multitasking I/O, information can be easily shared between any number of programs written in any language. This is important if one has a large number of engineering algorithms written in different languages.

The explanation facilities are good. A nice feature is tiered explanation. One can keep querying the computer to get various levels of detail to the question - why?. RuleMaster does a good job at translating example knowledge to understandable explanation in context. The fact that the explanation can be put into context is not a

trivial feature. If the problem is very complex, then a simple explanation of the rule being queried is easily misunderstood. The User interface makes working with this tool very enjoyable, but, the interface was just moving out of development in August 1985. At that time, there were some bugs and some of the applications were not available.

If RuleMaster has a soft-spot, it must be considered the lack of comprehensive documentation. For such a powerful tool, all we received is the training course notes which they call a "system user manual". User support is available through training and contract consulting.

The approximate cost of RuleMaster is $10,000 - $15,000 for IBM - PC and AT computers and $25,000 for supermini's such as a VAX. Educational discounts and trial period arrangements are available.

Important to note is that Radian is primarily a scientific - engineering company and its product RuleMaster is primarily designed for those domains. It is nice to know that if you do have problems, you can talk to a civil engineer who knows knowledge engineering rather than just a knowledge engineer who is not a civil engineer. All in all, RuleMaster is an excellent tool for building large systems that do not require various levels of abstraction.

Additional descriptive information is available from Radian Corporation (10).

EXPERT:

EXPERT was developed at Rutgers University for use in their biomedicine program so it is not surprising to find the tool's design directed at this domain. But, just as the field of medicine relies on expert diagnosis so do many problems in engineering. For example, diagnosing operational problems in a "sick" wastewater treatment system or aiding in structural design problems. Surprisingly, EXPERT is a Fortran based tool. It uses standard productions rules to represent procedural knowledge and in the tradition of MYCIN, EXPERT incorporates certainty factors within a backward-chaining inference.

Even though EXPERT is written in Fortran it only handles the standard mathematical operations of +,-,/,*, and **. It also does not hook out to other software (even Fortran) and unless you are a Fortran Guru with lots of extra time, do not expect to change this. The explanation facilities are somewhat limited. The user can ask why?, unfortunately the explanation only concerns the last question asked by the computer and not with the context of the question. EXPERT does have a "trace" facility which allows the user to follow the program's logic. For a large mainframe system, EXPERT's documentation is a meager 40 pages. This consists of an overview, a simple diagnostic example, and command definitions.

The main advantage of expert is its ability to run on larger computer systems (ours is running on a VAX 11-780). This allows for large number of rules to be incorporated and executed quickly. EXPERT is available for prototyping at little or no cost, however, there is no formal source of user support. EXPERT could be customized into a very powerful engineering tool, if one has access to a relatively large computer and is proficient in Fortran and fundamental expert system programming. If you can not afford customization, then plan on using EXPERT to solve diagnostic problems which do not involve complex algorithms.

Other descriptions and evaluations of EXPERT are available (11, 12).

ROSIE:

ROSIE (Rand Corporation) has been described as a general- purpose AI language as well as an expert system building tool (13). Since ROSIE is written in the INTERLISP programming language, it naturally picked up many of INTERLISP's features. As an expert system tool, ROSIE uses an English-oriented syntax in its knowledge base and input/output facilities. At first glance, it seems ROSIE has a built in natural language interface, however, it makes no attempt to grasp unrestricted English input. One must learn to "talk" to ROSIE in a very structured manner which resembles simple english sentences. It is still impressive to see the user interact with ROSIE by typing in small reports describing certain situations rather than answering one query at a time.

Because ROSIE is a large program which requires a large language (INTERLISP) to run, it commandeers significant memory and run-time. If one is paying for these services, the development costs can be restrictive. A mitigating factor is ROSIE (VAX-VMS version) can be obtained for approximately $200, but be prepared to spend several thousand dollars for INTERLISP.

ROSIE does not incorporate complex mathematical functions as part of the tool nor does it make allowances for interfacing with other software. The explanation facilities are very limited. One must use a "trace" or "scan" command to indirectly find out what is going on, rather than just simply asking why?. The development environment is also limited. There are no build in editors, menus, or graphical aids. The documentation consists of three volumes and is sufficient to get started on small to medium size systems. Little advice is given on building large systems via ROSIE. Finally, no formal support is available for ROSIE. Rand Corporation does consult on ROSIE but does not support it in a marketable way.

Additional descriptive information is available from Rand Corporation (14, 15, 16).

S.1:

Just as M.1 was considered the Cadillac of small system tools, S.1 (Teknowledge) is the Rolls Royce of the large system tools evaluated. The features of S.1 are too numerous to even list in this paper. Ironically, S.1's biggest drawback is its luxury. Just as most people can not afford to use a Rolls Royce as a pickup truck, most knowledge engineers can not afford to use S.1 to develop small systems. Unfortunately, most prototyping falls into this category. S.1 is a huge program that requires alot of computer memory and time to run. It had no trouble eating one of our time-shared VAX11-780's for lunch, in fact, S.1 should really have its own super mini dedicated just for itself (ie. Xerox 1108 workstation).

On one hand, S.1 has a sophisticated development environment consisting of an editor, debugging tools, and graphical aids. It allows for various forms of knowledge representation as well as inference strategies. Also, its explanation facilities are quite extensive. One can ask such question as how, what, and why. On the other hand, complex mathematical algorithms must be written outside of S.1 in the Lisp programming language (What fun!?). Also, interfacing to existing data-bases does not appear to be a simple task.

The documentation consists of five volumes, mostly training material or sample expert systems. Unfortunately there is no master index and just like M.1, no individual examples of each command are given. Here again, one must dissect an entire expert system to understand why and how certain commands were used in order to build ones own system.

Teknowledge is known for their outstanding user support, little of which comes for free, but still it's nice to know its there when needed. These people probably have more experience in building expert systems than any other company. Their products might cost more, but the strong user support might more than compensate for the initial investment.

One gets the feeling S.1 was designed to solve very grandiose but not very technical problems, something like automating the mail room at the pentagon. If price is no object and one is fluent in Lisp, S.1 allows enough flexibility to handle even engineering problems, but remember to ask yourself "do I need a pickup or a Rolls Royce". Needless to say, S.1 is not for everyone.

Case Study:

Utah State University, department of Civil and Environmental Engineering is pioneering the application of expert system technology to the areas of environmental systems modeling and hazardous waste management. One current study deals with the development of a demonstration expert sytem for assessing organic chemical mobility and degradation in order to consult on soil treatment options.

Since this system would be used for demonstration purposes, the portability of a PC-based program appeared attractive. For this same reason of demonstration, the ability to produce "run-only" versions of the expert system was considered an important factor. After some deliberation, M.1 was choosen for system development because the potential was there for building an expert system with the forementioned characteristics. However, one must be aware of the surrounding circumstances. First, since we are an education institution, all of the evaluated tools have been aquired at a greatly reduced cost or no charge at all. Second, Utah State University has available considerable in-house expertise in building expert systems as well as software engineering in general. Lastly, some of the newer PC-based tools, such as Insight II, were not available at the onset of this project. If carried out today, the decision of which tool to use for building this demonstration expert system would probably be different.

Food For Thought:

It is of utmost importance for any civil engineer who wishes to build an expert system to realize that one can learn to be a knowledge engineer rather rapidly, in fact many civil engineers are already knowledge engineers without even knowing it. Most engineers are quite good at extracting complex knowledge and based on scientific assumptions, produce simplified heuristics (Rules-of-thumb). Obviously, one can not be just a knowledge engineer and expect to become a civil engineer overnight. For this reason, civil engineers should seriously consider building their own systems with the help of flexible and user friendly tools before hiring those who are not familiar with civil engineering problems.

Appendix D

Expert Systems:
Implications for the Diagnosis
and Treatment of Learning Disabilities

EXPERT SYSTEMS: IMPLICATIONS FOR THE DIAGNOSIS AND TREATMENT OF LEARNING DISABILITIES

Alan M. Hofmeister and Margaret M. Lubke

Abstract. **Application of artificial intelligence to the problems of education is a relatively recent endeavor. This article will focus on one of the most promising aspects of artificial intelligence — *expert systems* technology — and some of the characteristics that make expert systems "intelligent". Selected present and potential applications of expert systems to the field of learning disabilities are presented along with examples of specific expert systems.**

Application of computer technology to the field of learning disabilities has taken a variety of forms, the most common being computer-assisted instruction (CAI), computer-managed instruction (CMI), and computer-assisted testing (Hofmeister, 1984c). To a large extent these applications represent reasonably well-developed procedures that existed before microcomputers, but had to wait for the widespread availability of this technology to achieve their present popularity. More recently, a new computer technology —the *expert system*— has been developed.

A field within artificial intelligence, expert systems technology is concerned with the use of computers to capture and disseminate human expertise. Typically, expert systems have proven effective in medicine, geology, chemistry, engineering, and business. However, educators have recently begun to show an interest in this technology, particularly as it can be applied to the problems associated with learning disabilities. This article reports on present and potential applications of expert systems technology to diagnosis and treatment of learning disabilities.

EXPERT SYSTEMS

Knowledge Engineering and "Expert Systems"

Knowledge engineering is the term often used to describe the process of capturing human expertise, developing a problem-solving framework, and eventually making the knowledge available to others through a computer-based expert system. The expert system usually gathers information from the user in a dialogue format that simulates a consultation with a human expert. Many expert systems are designed to explain their line of reasoning in everyday English rather than computer code.

Reasoning Procedures

The expert system's reasoning procedures, sometimes referred to as the *inference engine*, acts upon the combination of user-supplied information and information contained in the expert system's knowledge base.

To facilitate the interaction with the inference engine, the knowledge base is organized into rules, consisting of two components. an "if" component and a "then" component. When the conditions in

ALAN HOFMEISTER, Ph.D., is a professor in the Department of Special Education and the Director of the Artificial Intelligence Research and Development Unit, Utah State University.

MARGARET LUBKE is a doctoral student in the Department of Special Education and a Research Associate in the Artificial Intelligence Research and Development Unit.

the "if" component match the conditions in the user's problem description, a conclusion in the "then" component of the rule is invoked. The following is an "if-then" rule taken from MYCIN, a medical expert system.

Rule 27

If (1) the gram stain of the organism is gram negative, and
(2) the morphology of the organism is rod, and
(3) the aerobicity of the organism is anaerobic,

Then There is suggestive evidence (.7) that the identity of the organism is bacteroides.

Knowledge-Based Content

A knowledge base is built on two types of knowledge: factual and heuristic. *Factual knowledge* consists of information that can be documented, such as state and federal regulations and proven hypotheses (Feigenbaum & McCorduck, 1983). *Heuristic knowledge*, in turn, captures the rule-of-thumb experiences of humans. In special education such knowledge might come from expert diagnosticians or instructors.

Developing a knowledge base is a major activity of considerable value. In discussing the need to develop *intelligent* tutoring systems, Sleeman and Brown (1982) noted that

> Much remains to be discovered and made explicit. We hope that educational theorists will find the explicit formulation of tutoring, explanation and diagnostic processes inherent in intelligent tutoring systems a test bed for developing more precise theories of teaching and learning. (p. 9)

Rationale and Conclusions

During an expert-system consultation, the user can ask why the expert system asks a certain question. The following dialogue is from CLASS.LD2 (Ferrara & Hofmeister, 1984), an expert system that will be presented in more detail in following sections. The expert system asks:

> Does the child have a learning deficit in one or more of the following areas:
> listening comprehension
> written expression
> basic reading skills
> reading comprehension
> mathematics?

Rather than respond "yes" or "no", the user could ask, "why". The expert system would then respond: "An answer to this question will aid in determining if the child's deficit(s) are in an area which qualifies the child as 'learning disabled' under federal regulations."

An expert system may also include a "show" feature that provides a list of the information that has been obtained up to that point in the consultation. In addition, a "tracing" function is often available to display information that documents the problem-solving process used in reaching a given conclusion.

Incomplete Information and Certainty Factors

A consultation may continue even when information requested by the expert system is incomplete. When the user responds "unknown" to a specific question, the program may note the response and continue the consultation.

However, if the expert system determines that missing information is valuable, the certainty associated with any conclusions is reduced. Because many expert systems are used in areas which deal with conclusions that are rarely definite, "certainty-computing" procedures become necessary. *Certainty factors* are usually based on a scale of 0-100. Hence a certainty factor of 30 would indicate a relatively low level of confidence in the outcome, whereas a certainty factor of 80 suggests a relatively high confidence level. The following is an example of an outcome and an associated confidence factor from CLASS.LD2: "Based on the information provided, this child can be classified as learning disabled with a certainty factor of 90."

The features described above demonstrate the characteristics of some expert systems. Although terminology and specific features will vary among systems, most contain provisions for explaining the inference process used in reaching a conclusion.

PRESENT APPLICATIONS OF EXPERT SYSTEMS

Intelligent Diagnostic Programs

Some of the earliest applications of artificial intelligence to the field of education focused on diagnosis. Specifically, in the diagnosis of learning problems, the approaches that have been deemed "intelligent" have been concerned with explaining why a student is making a mistake as opposed to merely identifying certain skill deficits.

BUGGY. One of the first and most substantial examples of an intelligent diagnostic program is BUGGY (Brown & Burton, 1980), which diagnoses learning problems in terms of the underlying "bugs" or consistent computational errors. An example of a bug would be, "When borrowing into a column whose top digit is 1, the student gets 10 instead of 11" (Brown & Burton, 1984). Reporting on findings from one of their field tests, Brown and Burton (1984) commented,

> It is interesting to note that 107 of the 1,325 students tested had a bug in their borrow-from-zero subprocedure and missed 6 of the 15 problems on the test because of this one underlying bug. The characterization given by BUGGY is a much fairer evaluation than scoring these students 60 percent correct. (p. 288)

Interactive videodisc program. Developed by Hofmeister (1984b), this program assesses beginning math skills in English or Spanish and is capable of diagnosing 27 common bugs. The microcomputer that is linked to the videodisc player analyzes both the correct and the incorrect answers and provides a listing of mastered skills from a possible total of 335 skills. The program also identifies which of the 27 common computational errors are present (Eastmond, 1984).

Expert Systems and Learning Disabilities

Although the previously mentioned *intelligent* diagnostic programs have applications to the field of learning disabilities, they were not initially designed to replicate the expertise of an LD specialist. Two systems specifically designed as expert systems applicable to the diagnosis and treatment of learning disabilities include a diagnostic and prescriptive program (Colbourn & McLeod, 1983) and a classification program (Hofmeister, 1984a).

Diagnosis and prescription. Colbourn and McLeod (1983) developed an expert system intended to serve as a *consultant* in the process of diagnosis and prescription. The system was designed to guide the user "through the various stages and levels of diagnosis, from the initial suspicion that a reading problem may exist through to the point at which sufficient information had been gathered to plan an appropriate remedial program" (p. 32). The system's effectiveness has been evaluated by comparing its diagnostic reports with those of human diagnosticians. In summarizing the results of this comparison, Colbourn and McLeod reported that,

> In general, the results of the evaluation were encouraging; the expert system's diagnoses were accurate. Furthermore, because of the system's speed at analyzing error patterns, its diagnostic reports included more information than those of the human diagnosticians. This was particularly noticeable with regard to the analysis of phonic skills. (p. 37)

Classification. One of the most perplexing problems facing special education program administrators in the United States is the frequent misclassification of students as learning disabled. Thus, research findings have indicated that more than half the LD student population may be misclassified (Shepard, Smith, & Vojir, 1983; Ysseldyke, 1983). The major problem is one of overclassification.

To provide a second opinion about the accuracy of LD placement decisions Hofmeister (1984b) developed an expert system, CLASS.LD. This program enabled individuals who make diagnoses of "learning disabled" to check their reasoning and conclusions against decision rules programmed into the computer. An updated version of this program, CLASS.LD2 (Ferrara & Hofmeister, 1984) contains over 200 "if-then" rules in its knowledge base and produces conclusions with associated certainty factors. With CLASS.LD2, the user can obtain a printed record of the rules used by the computer program and statements about how they were applied in reaching the conclusion that a student was or was not learning disabled. The record shows what questions the computer program presented, the answers the user provided, and the rules the program applied to make "judgments" based upon those answers.

FUTURE APPLICATIONS OF EXPERT SYSTEMS

Hayes-Roth, Waterman, and Lenat (1983) suggested that in addition to diagnosis, prescription, and classification, expert systems may be developed in the areas of prediction, interpretation, remediation, planning, monitoring, and instruction. Already, several prototype programs of this type are being designed by staff of the Artificial Intelligence Research and Development Unit at Utah State University. These prototypes, in turn, are used to test the feasibility of applying expert systems to solve problems in special education.

Intelligent Test Interpretation

One prototype expert system is the Intelligent

Test Interpretation which yields an individual prescription in mathematics. Results from the *Key Math Diagnostic Arithmetic Test* (Connolly, Nachtman, & Pritchett, 1976), along with demographic data, will constitute most of the data the user enters. Based on this information the computer program will produce a prescription for program planning.

In a study that provided an information base for the expert system, Hofmeister (1984a) found that *Key Math* scores correlated .82 with another much more comprehensive criterion-referenced instrument. Consequently, the knowledge base built into the proposed expert system makes use of rules based on correlations between the *Key Math* instrument and the more prescriptive but time-consuming criterion-referenced instrument.

Intelligent Monitoring of Pupil Performance

Mandate Consultant is a third knowledge-based expert system prototype being developed. This system emulates the decision-making processes of a human expert familiar with federal and state regulations pertaining to the Education of All Handicapped Children Act. Thus, the expert system is capable of providing school officials and parent advocates with expert advice on how to plan and implement instructional programs. The advice identifies the extent to which planning and instructional procedures are consistent with federal and state regulations.

Mandate Consultant holds potential for addressing many of the issues currently resolved at a due-process hearing level, such as categorization, extent of services, and placement decisions for handicapped children. At this time, the primary application of this expert system is in the training of administrators and hearing officers rather than as a field consultant.

Classroom Behavior Consultant

This prototype expert system was designed to generate behavior-management advice to teachers. The user provides information about the type of problem encountered and the conditions under which it usually occurs. The knowledge-base rules are organized into three sets. The first elicits information from the user and clarifies the type of behavior problem. The second set of rules determines the cause of the problem or the factors associated with maintenance of the problem. Finally, the third group of rules generates recommendations about the intervention procedures deemed capable of successfully treating the problem.

In its prototype form the Classroom Behavior Consultant contains approximately 600 rules and runs on a powerful microcomputer. It is anticipated that a larger final version will include more than 1,000 rules and require a minicomputer.

SUMMARY AND CONCLUSIONS

Recent efforts at applying expert systems to the problems encountered in the field of learning disabilities differ greatly from traditional computer applications such as CAI and CMI. Considerable research is needed before any firm conclusions can be reached regarding the value of expert systems for identification and treatment of learning disabilities. However, some preliminary findings indicate that this line of research is warranted.

1. Evaluations conducted with prototypes indicate that expert systems can perform as well as humans in specific areas.

2. Some of the problems faced by special educators are similar to those encountered in other disciplines where expert systems have proven successful.

3. The process of assembling and organizing knowledge bases for expert systems is a productive activity in its own right. The development of the "if-then" rules of a knowledge base clarifies existing knowledge and identifies areas where knowledge is needed.

REFERENCES

Brown, J.S., & Burton, R.R. (1980). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science, 2*, 155-192.

Brown, J.S., & Burton, R.R. (1984). Diagnostic models for procedural bugs in basic mathematical skills. In D.F. Walker & R.R. Brown (Eds.), *Instructional software* (pp. 269-298). Belmont, CA: Wadsworth Publishing Company.

Colbourn, M., & McLeod, J. (1983). Computer guided educational diagnosis: A prototype expert system. *Journal of Special Education Technology, 6*(1), 30-39.

Connolly, A.J., Nachtman, W., & Pritchett, E.M. (1976). *Key Math Diagnostic Arithmetic Test.* Circle Pines, MN: American Guidance Service, Inc.

Eastmond, D.V. (1984, November). Math assessment videodisc. *The Computing Teacher,* 57-59.

Feigenbaum, E.A., & McCorduck, P. (1983). *The fifth generation.* Reading, MA: Addison-Wesley.

Ferrara, J.M., & Hofmeister, A.M. (1984). *CLASS.LD2: An expert system for classifying learning disabilities* (Computer program). Logan: Utah State University.

Hayes-Roth, F., Waterman, D.A., & Lenat, D.B. (1983). An overview of expert systems. In F. Hayes-Roth, D.A. Waterman, & D.B. Lenat (Eds.), *Building expert systems* (pp. 3-9). Reading, MA: Addison-Wesley.

Hofmeister, A.M. (1984a). CLASS.LD: An expert system for classifying learning disabilities (Computer program). Logan: Utah State University.

Hofmeister, A.M. (1984b). *Development of a microcomputer/videodisc aided math instructional management system for mildly handicapped children.* Final report to the U.S. Department of Education, Office of Special Education, Project #G008101536. Logan: Utah State University.

Hofmeister, A.M. (1984c). *Microcomputer applications in the classroom.* NY: Holt, Rinehart & Winston.

Shepard, L.A., Smith, M.L., & Vojir, C.P. (1983). Characteristics of pupils identified as learning disabled. *American Educational Research Journal, 20*(3), 309-331.

Sleeman, D., & Brown, J.S. (1982). *Intelligent tutoring systems.* London: Academic Press.

Ysseldyke, J.E. (1983). Current practices in making psychoeducational decisions about learning disabled students. *Annual Review of Learning Disabilities,* 31-38.

FOOTNOTES

Preparation of this article was supported by funds from the U.S. Department of Education, Office of Special Education Programs.

Requests for reprints should be addressed to: Alan Hofmeister, UMC 6800, Systems Impact, Inc., Utah State University, Logan, UT 84322.

Appendix E

Development and Validation of an Expert System for Special Educators

DEVELOPMENT AND VALIDATION OF AN EXPERT SYSTEM FOR SPECIAL EDUCATORS

James D. Parry and Alan M. Hofmeister

Abstract. **The authors describe the development and initial validation of a computer-based expert system, Mandate Consultant (Parry, 1985), designed to review the regulatory procedures for developing Individual Education Programs (IEP). The formative process involved three phases: (a) definition of need and proposed solution, (b) design of a prototype, and (c) progressive refinement through field-testing and revision cycles. The summative component included a two-phase experimental design for validating the accuracy of expert system output through comparisons with human experts. The findings indicated that the expert system-generated conclusions matched the conclusions of the "better" human experts, and were considered more appropriate than the conclusions of the majority of experts. Furthermore, "blinded" evaluators judged the expert system-generated conclusions as being equally acceptable as those produced by the "better" human experts, and more acceptable than those of the majority of experts.**

The Individual Education Program (IEP) process serves as a forum during which parents and school officials should reach agreement on the content and delivery of a handicapped child's education. When this process fails, other parties intervene to mediate the disagreement. If such mediation is unsuccessful, the parties involved proceed to a hearing to resolve the issue based on the intent of the law.

Unfortunately, several problems are associated with such hearings. First, hearings cause delays and interruptions in appropriate services for handicapped students (Budoff, Orenstein, & Abramson, 1981). Second, hearings are costly in terms of money (Henderson & Hage 1979) and stress (Fiedler, 1985). Finally, hearings do not ensure equitable and effective educational decisions (Salend & Zirkel, 1984).

For these reasons, disagreements between school officials and parents should, whenever possible, be resolved prior to formal hearings (Belsches-Simmons & Lines, 1984). Towards this end, school officials need an unbiased, knowledgeable consultant to objectively review the development of an educational program for a handicapped student.

A COMPUTER-BASED EXPERT SYSTEM

An application from a relatively new field within artificial intelligence (AI) holds promise for accomplishing this task (Waterman, 1986). In recent years in the fields of medicine, geology, and engineering, specific domains of knowledge possessed by humans have been cloned in computer-based expert systems. Typically, these

JAMES D. PARRY, M.A., is a doctoral candidate in Special Education Technology at Utah State University, and a research associate at the USU Artificial Intelligence Research and Development Unit.

ALAN M. HOFMEISTER, Ph.D., is Director of the USU Artificial Intelligence Research and Development Unit and Professor of Special Education, USU, Logan.

systems were designed to engage the user in a dialogue, which in many ways paralleled the type of conversation a person might have with an expert consultant. The computer was programmed to present the user with questions, accept the user's responses, and match the responses with information in the program. Finally, based on the programmed logic, conclusions were displayed to the user.

Until recently, little application of expert system technology has taken place in the field of education (Hofmeister & Ferrara, in press). However, the increased power and availability of computer hardware and the gains in artificial intelligence technology make development of expert systems for educators feasible. For example, an AI Research and Development Unit was established in 1984 at Utah State University for the purpose of exploring AI applications to special education. Subsequently, Ferrara and Hofmeister (1984) developed an expert system, CLASS.LD, designed to provide a second opinion about the accuracy of placement decisions for learning disabled students. The expert system consists of two components, a knowledge base and an inference "engine". The knowledge base is made up of rules based on research findings as well as state and federal regulations. The inference engine, in turn, guides the process of bringing the knowledge base to bear on the specific case being reviewed. CLASS.LD is currently being validated; preliminary evaluation results suggest that the system can perform as well as humans in classifying handicapped students as learning disabled (Hofmeister & Lubke, elsewhere in this issue).

Project Goals

The purpose of this project was to develop and initially validate a computer-based expert system designed to help special educators review the procedures followed in developing IEPs. The project included both a formative and a summative evaluation. The former involved a three-phase model proposed by Hofmeister (in press) for (a) definition of the need and proposed solution, (b) design of a prototype in response to the definition, and (c) progressive refinement through field-testing and revision. The summative component involved a two-phase experimental design for validating the accuracy of the expert system output through comparisons with human experts.

Development Phase

The formative stage of the expert system development was concerned with identifying alternative procedures to replace weaknesses. The direction for such activities came from a diverse group of consultants, including local education agency staff, state education agency personnel, and university-affiliated educators. These special educators systematically evaluated each formative activity, and their input subsequently guided revisions of product definition, design, and development. Consultant input also served as a basis for judging successful completion of each phase.

Much of the formative portion of expert system development and evaluation may be characterized as a recycling process (Hofmeister, in press). Markle (1967), for example, described the formative process as development through successive approximations, that is, a process of revision and trial and revision and trial with much dependence on art and insight by the developer (p. 137). In the case of expert systems, the successive approximations involve testing and revising the system based on feedback from a small set of test cases designed to encompass a range of problems. This stage is followed by further testing and revision based on a set of actual cases representative of the field (Hofmeister, in press). When this recycling process results in no substantive changes to the system, it is judged as being stable. Hence the major formative procedures may be considered near completion.

With Mandate Consultant, the formative evaluation of the prototype was extended to a group of special education professionals representing a state education agency, local education agencies, and higher education. Using multiple combinations of diverse test cases, created by the developer from actual cumulative student files, reviewers of the prototype (a) read the documentation written to accompany Mandate Consultant, (b) reviewed the test-case cumulative file data, and (c) ran consultations on the test cases using the prototype. The feedback from these individuals provided the data necessary for the cyclic process of testing and revising subsequent versions of the prototype. This cycle continued until no substantive modifications were suggested by the reviewers' feedback.

Validation Procedures

During validation the focus shifted from product improvement to formal assessment of the accuracy of the expert system's output. The experimental design involved two formal evaluation phases. In

the first phase, six human experts reviewed the data of 10 representative cumulative case files and provided conclusions regarding failures to implement state and federal regulatory procedures for IEP development. In addition, the expert system generated conclusions about the IEP procedures based on the same 10 cases. In the second phase, three additional human experts reviewed all conclusions and judged their acceptability using a rating scale. These reviewers were "double-blinded", that is, they did not know the source of the conclusions, including not knowing that one of the sources was a computer program. The evaluators' ratings served as a basis for comparing the conclusions drawn by human experts and the expert system.

This type of blinded evaluation of expert-system knowledge-base performance was originally conducted in the medical field to evaluate the expert systems, MYCIN and ONCOCIN (Yu, Fagan, Wraith, Clancey, Scott, Hannigan, Blum, Buchanan, & Cohen. 1979; Hickam, Shortliffe, Bischoff, Schott, & Jacobs, in press). This two-phase design evolved from earlier evaluations of MYCIN. The evaluation compared experts' decisions, in which the answer is not clearly "right" or "wrong", with the expert system's conclusions (Yu et al., 1979).

Sample. A total of nine human experts participated in the study. These "experts" were selected from a list provided by the state education agency in Utah of special education administrators and other leaders (e.g., advocates, attorneys, and university-affiliated staff) actively involved in special education in Utah. In addition, staff of the state agency identified those leaders and administrators on the list who, in the staff's judgment, were the most qualified "experts". When identifying the "experts", state agency staff were instructed to consider potential subjects' amount and diversity of experiences as well as specialized training. The nine most "qualified" human experts were selected for the study. Of these, the three most "qualified", according to the criteria described earlier, were selected for the second evaluation phase.

The six subjects involved in the first phase of the study were local education agency directors of special education. All six had participated as trainers or trainees in inservice on implementation of procedures governing IEP development. In addition, two of the six were qualified as due-process hearing officers. The experts reported from 7 to 22 years of special education experience with a mean of 15.3 years.

The three additional human experts, serving as evaluators for the second phase of the study, included a special education director, an assistant superintendent, and a private special education consultant/advocate. All three had participated as trainers or trainees in inservice on IEP development. Furthermore, they were qualified as due-process hearing officers. Their years of experience ranged from 15 to 34 years with a mean of 24.7 years.

Measurement and instrumentation. To conduct the product validation study, it was necessary to gather a set of representative special education test cases from a local education agency. Special education administrators identified 10 special educators who were representative of the service-delivery continuum provided locally. These special educators were asked to randomly choose one student from their respective student populations and photocopy selected documents from the student's cumulative file. They were also asked to remove any personally identifiable information from those documents.

The test-case data included 7 male and 3 female special education students ranging in age from 5 to 17 years. Although the majority of students were classified as learning disabled, the handicapped conditions they presented ranged from mild specific learning disabled (SLD) to severely intellectually handicapped (SIH).

The human experts of the first phase of the comparison study read the 10 cumulative files and noted discrepancies between the procedures implemented for a given case (as evidenced in the cumulative file documentation) and the procedures governing IEP development. The experts recorded their conclusions for each of the 10 cases on a cumulative file report form.

Simultaneously with this activity, three special educators independently completed consultations with the expert system using the same 10 student files. The authors selected to use three special educators so that if any of the conclusions were discrepant, the conclusions generated by two of the three consultations would serve as the expert system data. The conclusions generated by the system were transferred to cumulative file report forms like those completed by the human experts. All the reports, both human and computer, were

typed and printed using word processors.

For the second phase of the comparison study, three additional human experts independently reviewed and rated the cumulative file reports produced by the experts. The evaluators read the same 10 cumulative files as the experts in the first phase. They reviewed the cumulative file reports written by the experts and rated each according to acceptability, based on a four-category rating scale. This scale was modeled after one validated in the ONCOCIN evaluation (Hickam et al., in press) as appropriately representing experts' opinions. The scale implemented for the present study included: 1 - *Ideal:* The information summarized in the report is synonymous with what I would have written; 2 - *Acceptable:* The information summarized in the report differs from what I would have written, but it is acceptable; 3 - *Less than Acceptable:* The information summarized in the report is inaccurate and/or inadequate; however, I would consider these deficiencies minor; and 4 - *Unacceptable:* The information summarized in the report is inaccurate and/or inadequate, and I would consider these deficiencies major. In addition, if rating a report as less than acceptable or as unacceptable, evaluators identified specific deficiencies of the report.

Data analysis. The data from the cumulative file reports produced by the experts were coded by a special education graduate student who was unfamiliar with the study. The coded data were subsequently analyzed and tabulated to determine: (a) the total number of conclusions produced by the experts, (b) the total number of interexpert agreements and disagreements, and (c) the total number of conclusions most frequently noted by the experts. These tabulations and comparisons were used to evaluate the degree to which the experts' conclusions matched in terms of implementing regulations governing IEP development.

The rating data from the second phase of the comparative study were compiled to determine: (a) percentage of expert case reports judged to be acceptable or unacceptable based on the evaluators' ratings, (b) sets of rankings of the experts based on the ratings by the three evaluators, and (c) association between the sets of rankings (using Kendall's coefficient of concordance). These tabulations and comparisons were used to evaluate the degree to which human expert evaluators judged expert system-generated conclusions as being equally acceptable as the conclusions of human experts.

As a part of the validation study, the reliability of consultation outcomes between and within users of Mandate Consultant was formally assessed. Percentages of agreement for the conclusions generated by the expert system were computed. Data resulted from three special educators independently running the same cases using the expert system, as well as three special educators each running the same cases at different times. The percentages of agreement provided measures of interuser and intrauser reliability, respectively.

RESULTS

Phase I

The expert conclusions in the first phase of the comparative study were tabulated based on the number of conclusions generated, the number of interexpert agreements and disagreements, and the number of most frequently noted conclusions. This information, summarized in Figure 1, shows the variation in the number of conclusions produced by the experts.

Phase II

The ratings by the human expert evaluators in the second phase of the comparative study were used to compute the percentage of acceptable and unacceptable expert reports. This information appears graphically in Figure 2. Furthermore, these ratings provided a basis for rank ordering the experts according to the number of expert case reports each judged to be ideal or acceptable. These rankings were used to compute Kendall's coefficient of concordance correlation (W) to describe the association between the three sets of expert rankings. The results of the computations appear in Table 1.

Reliability Assessment

A subcomponent of the comparative study involved formal assessment of reliability. Specifically, to assess interuser reliability three special educators independently completed consultations for each case. In addition, the special educators reran a sampling of the cases at a later time to assess intrauser reliability. The number of agreements on conclusions was divided by the total number of conclusions to produce measures of interuser and intrauser reliability. The interuser reliability coefficients ranged from .67 to 1.00 with a mean of .94. The intrauser reliability coefficients ranged from .75 to 1.00 with a mean of .95. Thus, a relatively strong agreement was found between the different

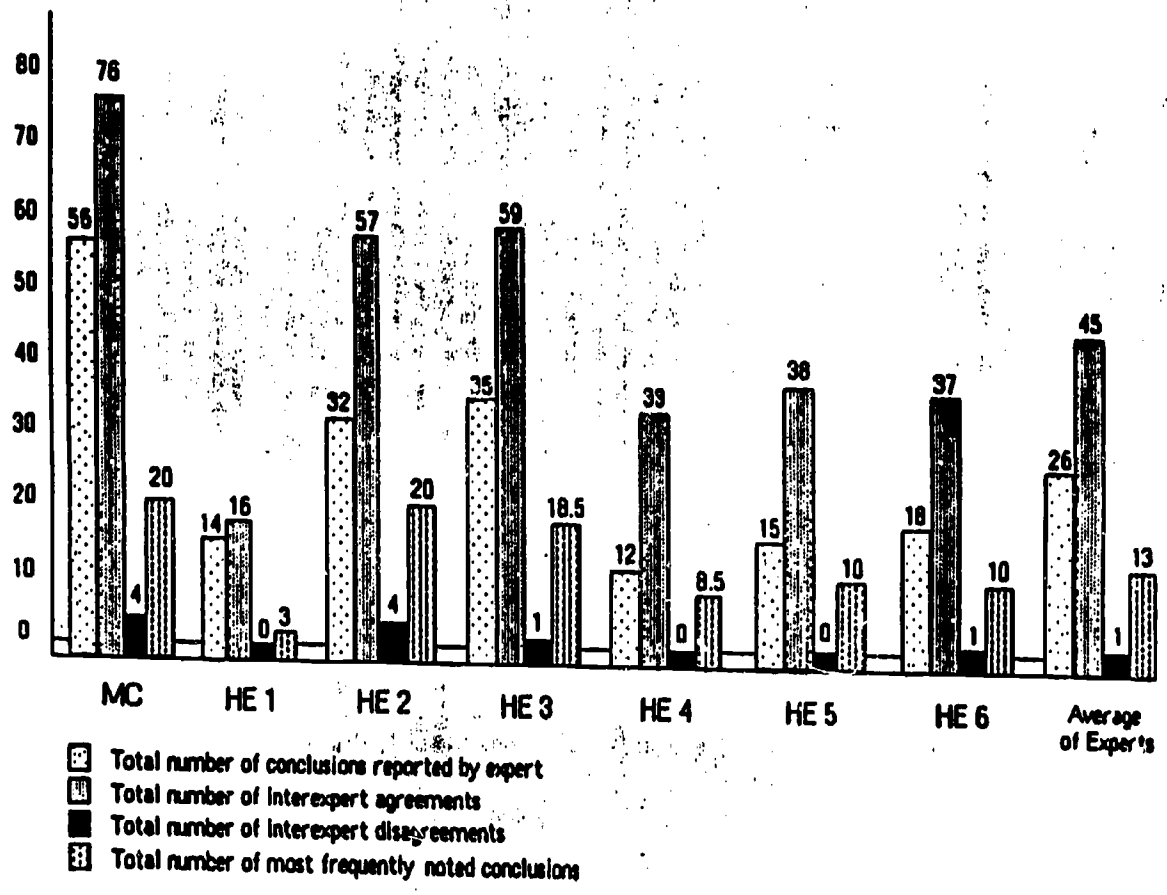


Figure 1. Totals by expert across the 10 cases.

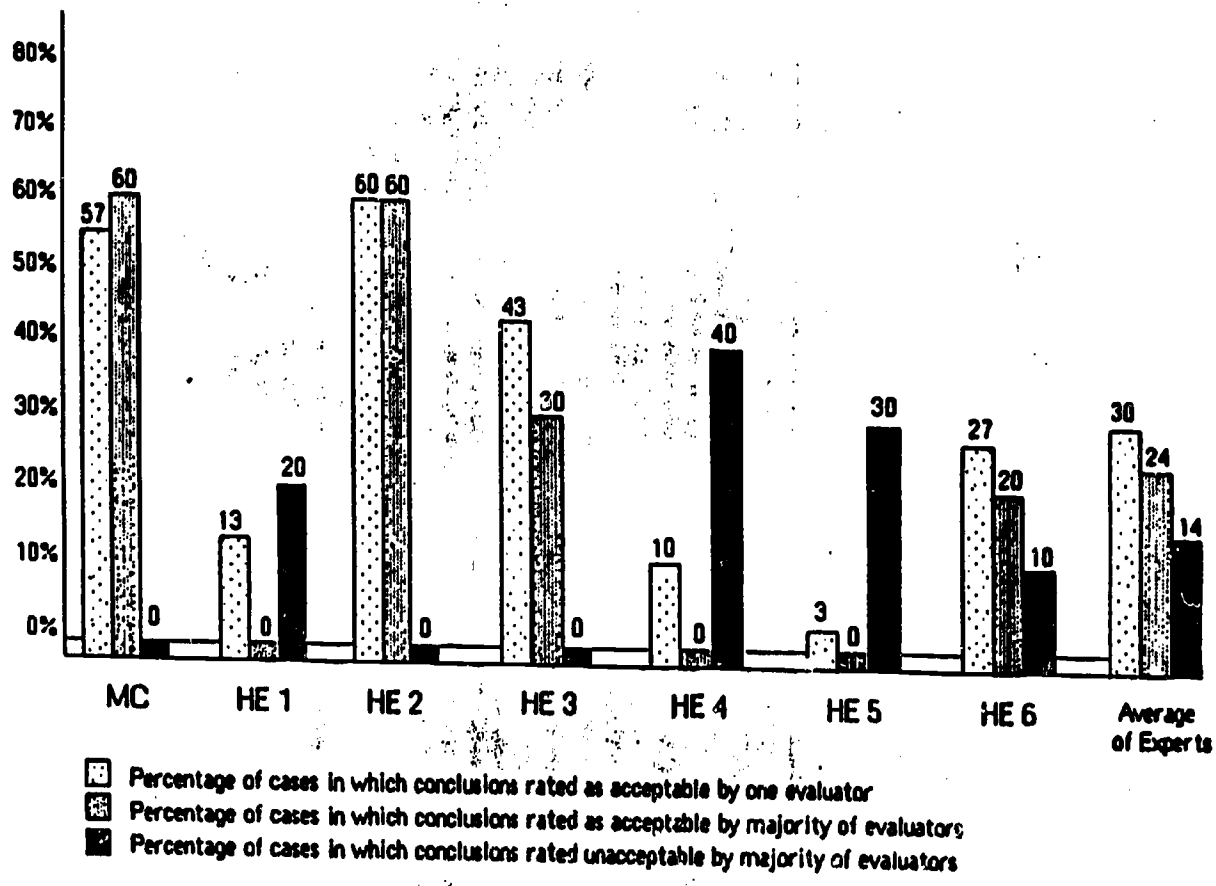


Figure 2. Percentage of expert case reports judged acceptable or unacceptable.

Table 1

Kendall's Coefficient of Concordance Correlation (W)
Results: Three Sets of Evaluator Rankings for Seven Experts

	*MC	**HE 1	HE 2	HE 3	HE 4	HE 5	HE 6
Evaluator 1	2	7	1	3	4.5	6	4.5
Evaluator 2	1.5	4	1.5	5	6.5	6.5	3
Evaluator 3	2.5	5.5	2.5	1	6.5	5.5	5.5

Measure of Disarray (S) = 178.4***
Kendall's Coefficient of Concordance (W) = .77

*Mandate Consultant
**Human Expert
***Critical value of S (M = 3; N =7) = 157.3, $p < .05$ (Siegel, 1956).

users, as well as for the same user over time.

DISCUSSION

Phase I

The results from the first phase of the comparative study demonstrated that conclusions generated by the expert system matched substantially the conclusions of human experts. However, the human expert behavior varied greatly. In some instances, human experts comprehensively noted the failures to implement special education regulatory procedures for IEP development while in other instances, they noted few, if any. Thus, this phase revealed that the conclusions generated by Mandate Consultant generally matched those of the "better" human experts, while exceeding the conclusions drawn by the majority of the human experts. "Better" experts were defined as those producing the most conclusions that agreed with those of other human experts.

Most notable among the findings related to the number of conclusions generated by the experts was the limited number of conclusions produced by most human experts. Specifically, four human experts produced fewer than two conclusions per case. Although no standard existed regarding the number of conclusions for cases, the evaluators in the second phase of the study rated numerous case reports as inadequate because of the few conclusions reported by the expert.

Although two of the human experts (i.e., Human Expert 2 and Human Expert 3) identified substantially more conclusions than their colleagues, none made as many as Mandate Consultant. This finding supports the outcome of a study by Colbourn (1982) who developed and validated an expert system to assist educators in diagnosing reading problems. Her comparison between expert system-generated diagnosis and human diagnosis revealed that the expert system provided more detailed information than human diagnostic reports. Such was the case with Mandate Consultant in this study. It appeared that the extensive knowledge base contained in the structure of an expert system allowed it to generate more specific information than many human experts typically generate.

In addition to generating the greatest number of conclusions, Mandate Consultant also achieved the greatest number of agreements with other experts. The authors expected the number of conclusions to be related to the number of interexpert agreements; the significant frequency of other experts agreeing with the conclusions of Mandate Consultant strengthened the case that Mandate Consultant provided substantial amounts of valid information.

The data also showed occasional disagreements among the expert conclusions. These disagreements appeared to result from different interpreta-

Reprinted from
Expert Systems in Civil Engineering Symposium Proceedings,
TCCP Council/ASCE
Spring Convention, Seattle, WA, April 8-9, 1986

Expert System Tools for Civil Engineering Applications

Phillip J. Ludvigsen, William J. Greeney[1], Del Dyreson[2],
and Joseph M. Ferrara[3]

INTRODUCTION:

In the past, expert system development was a monumental undertaking reserved for major universities and corporate giants. Fortunately advances in microcomputers coupled with a more pragmatic understanding of how expert system technology can be applied have initiated a new era in user developed expert systems. Development time, that took multi-man years, now takes months if not weeks. Programming, which required highly technical computer skills, can now be accomplished by novice programmers with the aid of software tools. Accessibility to AI (artificial intelligence) expertise, that was only available on the university campus, is now available (if only indirectly) through customer service, support, and training. All these changes manifest in one more factor - the cost. Projects that once required major public and/or private funding can now be undertaken by small companies and even individuals.

Expert system software tools have played a major role in expediting program development, however, they do not offer a panacea to all problems which require expertise. It is important to know that some problems should not and possibly can not be solved by current expert system technology. For problems that can be solved with current technology, consideration must be given to the design of the tool and how it relates to your particular problem. Each tool, as with any software, has advantages vs. limitations which must be evaluated before project development begins. The final decision on which tool is "best" (most appropriate) is dictated by various factors, such as flexibility, user support, documentation, and of course cost.

[1] Graduate Research Assistant, Professor, Respectively, Dept. of Civil and Environmental Engineering, Utah State Univ., UMC 41, Logan, UT 84322

[2] Associate Professor, Computer Science Department, Utah State Univ., UMC 42, Logan, UT 84322

[3] Associate Research Professor, Artificial Intelligence Research & Development Unit, Dept. of Special Education, Utah State Univ., UMC 68, Logan, UT 84322

The purpose of this paper is not to endorse any one particular software tool to build civil engineering expert system applications, but rather to emphasize their particular advantages as well as limitations from a civil engineering standpoint. The information presented is vintage 1985 and one should be aware of current changes in program features and price. Caveats aside, the body of this paper is based on hands-on experience and should prove useful.

Brief History:

Expert systems tools, sometimes called authoring tools, or more recently shells, have a relatively short history. Basic research of developing a tool to aid in building an expert system is approximately ten to fifteen years old. Much of the early work was done in the field of medical diagnosis. From this work, a tool named MYCIN emerged. It used "if-then" production rules, certainty factors and backward-chaining inference, thus setting the standard which many current tools follow today (1).

Teknowledge (Palo Alto, CA), probably the largest and oldest "expert systems company" has a corporate history of approximately five years. Experience in developing expert system tools and applying them is somewhat limited, especially in the field of civil engineering. Fenves, Maher, and Sriram in their paper "Expert Systems: C.E. Potential" highlight future uses of expert systems but the lack of current applications is apparent (2).

It is interesting to note the evolution of these tools and how recent advancements affect solving problems within the civil engineering domain. But first, lets take a look at what distinguishes civil engineering problems from other kinds of complex problems.

Problem Domain:

Civil engineering exhibits an extremely wide, as well as deep problem domain. The sheer diversities of disciplines involved and complexities encountered are self-evident. Because of this, civil engineering expert systems and thus the tools to build them must be extremely flexible. The ideal tool for building civil engineering systems would allow for the following:

* Complex mathematical manipulations within the tool (Including scientific functions plus canned algorithms, ie. statistics, integration, etc.).
* Various forms of knowledge representation (Not just "if-then" production rules).
* Various inference strategies (Not just "backward-chaining").
* Simple calls to other programs or expert systems written in ANY programming language.
* Natural language interfaces.
* Unlimited degree of expert system explanation.

* Extensive development environments (eg. intelligent editors, debuggers, graphics, and help facilities).

Unfortunately, no tool available today allows for all of the above. Many optimistic sales and customer service people will tout "we can't do that directly, but we can show you ways to work around it" or "our next version is slated to have that improvement". Upon hearing such statements, beware! Given the time and money one can "work around" or wait for anything, but the time or money might not always be available.

A good example of what appears to be a universal limitation of current tools is the inability to handle complex mathematical manipulations directly within the tool. The standard solution is to call (sometimes referred to as "hook") a module written in some common programming language to return the desired calculated data. However, you might have to call this module or others many times within an iterative solution process. This can slow execution down tremendously. In some cases, you can use knowledge engineering (programming) "tricks" for greater efficiency. The drawback to this approach is that your solution logic is dictated more by the tool's limitations than by the problem.

Even though current tools fall short of the ideal, the future looks bright. Expert systems and the tools to build them are heavily dependant on hardware speed and memory. This is why we see special machines designed just for AI work. Fortunately, advancements in hardware design are bringing tremendous computing power and thus making meaningful expert system tools available to desk top computers.

Tools:

Seven tools are investigated (EXPERT-EASE, INSIGHT, M.1, RuleMaster, EXPERT, ROSIE, and S.1). They represent a diversity of complexity, flexibility and cost. EXPERT-EASE, INSIGHT, and M.1 are suitable for micro computers while RuleMaster is a medium size tool suitable for super mini's. EXPERT, S.1, and ROSIE can be considered large, main frame software.

The following criteria are used to investigate important features of each tool.

* Approximate cost.
* Ability to handle complex mathematics.
* Ability to interface with other software.
* Explanation facilities.
* Overall friendliness.
* Documentation.
* User Support.

As means of a brief summary, Table 1 compares each evaluated tool according to the above criteria. The

EVALUATED EXPERT SYSTEMS TOOLS

ATTRIBUTE	SMALL			SYSTEM SIZE MEDIUM	LARGE		
	EXPERT-EASE	INSIGHT II	M.1	RULEMASTER	EXPERT	ROSIE	S.1
Proprietary Interest	Human Edge Software (Palo Alto, CA)	Level Five Research Inc. (Melbourne Beach, FL)	Teknowledge (Palo Alto, CA)	Radian Corp. (Austin, TX)	Rutgers Univ. (New Brunswick, NJ)	Rand Corp. (Santa Monica CA)	Teknowledge (Palo Alto, CA)
Phone	800-824-5227	305-729-9048	415-327-6640	512-454-4797		213-393-0411	415-327-6640
Approx. Cost	$895 (IBM-PC)	$495 (IBM-PC)	$5000 (IBM-PC)	$10,000 (IBM-PC) $25,000 (VAX)	No Charge	$200 (VAX)	$50,000 (VAX)
Ability to handle Complex Math	NO	YES (directly)	YES (indirectly)	YES (indirectly)	NO	NO	YES (indirectly)
Ability to interface with other software	NO	YES (directly)	YES (indirectly)	YES (directly)	NO	NO	YES (indirectly)
Explanation	NONE	Extensive	Limited	Extensive	Limited	Very Limited	Extensive
Overall Friendliness	Buddies	?	A Friend	A Friend	Fair-Weather Friend	Fair-Weather Friend	Real Buddies
Documentation	Outstanding	?	Sufficient	Meager	Meager	Sufficient	Sufficient
User Support	Limited	Available	Available	Available	None	None	Available

Table 1. Expert system tools comparison.

following offers a bit more detail.

Expert-Ease:

Expert-Ease (Expert Software International Ltd) is probably the smallest of the tools evaluated. It is designed to aid the use in quick development of small prototypes. This Pascal based tool features an automatic induction routine. One sets up a decision table and Expert-Ease translates it to Pascal code which can only be executed from within Expert-Ease. The user really has very little control over the inference strategy (dedicated forward-chaining). If Expert-Ease sees fit to ask a certain question first, the programmer (knowledge engineer) can not get at the Pascal code to over-ride the tool's decision.

This tool does not handle mathematical functions nor will it run on several IBM compatible machines. Expert-Ease does not make allowances for interfacing to other software, plus it does not have any explanation facilities. Due to these limitations, Expert-Ease can not be considered for large complex problems which involve mathematics, this rules out most engineering problems. However, Expert-Ease could be used to develop skeletal logic structures involved in solving larger problems. It has outstanding documentation and it is easy to use. But for large engineering expert systems this tool will not handle the load.

Other descriptions and evaluations of Expert-Ease are available (3, 4, 5).

Insight II:

Insight II (Level Five Research) is the newest of all the tools evaluated. Unfortunately, we were not able to receive an evaluation copy in time for this paper. So, attributes such as overall, friendliness and quality of documentation are not evaluated here. Without hands on experience with Insight II, little can be said about its limitations and shortcommings, however, Insight II appers to have some powerful facilities at an attractive price ($495).

Insight II is a Pascal based program which boasts a menu - driven development environment, links to other programs, confidence factors, tiered explanation, the ability to produce "run-only" end user versions, the ability to address 2000 rules, the speed of a compiler based system, and believe it nor not, complex mathematical functions which are intrinsic to the system. Insight II is the only tool evaluated which directly incorporates transcendental math functions (ie. cos, sin, tan, etc.). For such a low price, this tool might be just the ticket, but just how well everything fits together remains to be seen.

M.1:

M.1 (Teknowledge Inc.) is considered by many to be

the Cadillac of PC-based expert system tools. The reasons for this analogy is 1) its price ($5000 - down from last year's price of $10,000); 2) it has many features that were once found on only mainframe system tools; and 3) Teknowledge offers first class user support (training and consulting). Teknowledge has gone to great lengths to produce a professional piece of software, but just as a Cadillac can be inappropriate for certain jobs, so can M.1.

M.1 (version 1.3) is a Prolog based tool which interprets english-like production rules. The form of inference used is backward chaining. Forward chaining can be simulated but the result is somewhat awkward. Because M.1 acts as an interpreter, rules are acted on much slower than a compiler based system. This can be a limiting factor, especially if you have to constantly "hook" out to other software. M.1 can not handle complex mathematics or interface with other software directly from within itself. One must customize the provided interfaces, write interfacing software in assembly language or "C" (Oh Boy!), and finally link the whole thing together with M.1 to get a new executable version. This is not a trivial task, considering most civil engineers are not fluent in assembly language or "C".

The explanation facilities are somewhat limited. When asked why?, M.1 will either give a programmer supplied explanation of just the rule which caused the query, or it will trace the rules (only by number) which were used to reach that point in the run. Since most expert systems reason within networks, it is impossible for a knowledge engineer to write one single explanation of a particular rule that will be in context. Some tools use a "tiered" explanation in an attempt to establish context, M.1 does not.

M.1 does not have an extensive development environment, in fact it does not have its own editor to make permanent changes or addition to the knowledge base. One must leave M.1 in order to use your own text editor or word processor. But M.1 does have nice tracing and debugging facilities. It also has the ability to produce "run-only" end user systems. This feature is particularly attractive to those who wish to diseminate their work but can not afford to buy numerious copies of the expert system tool.

The documentation is sufficient but not impressive when one considers the cost of the tool. About half of the documentation contains example expert systems. These examples are nice to have, but it would be nicer if each M.1 command was defined along with numerous examples of just how that command might be used. Included is a helpful section on how to develop an expert system from proposal to turn-key delivery. This "how to" section offers some good program development advice, unfortunately most of this rather large section has very little to do with using M.1 directly.

There is some question whether or not M.1 can be used to build a significant expert system (>500 rules). The answer is maybe. M.1 can not address more than 200 rules at one time however, allowances are made to "shuffle" in and out groups of rules as needed. Here again, the interpreter nature of M.1 makes this process painfully slow.

M.1 is a powerful but expensive tool. Complex systems can be built but certain inconveniences (interfacing difficulties and slow execution) must be considered limiting. We are anxiously awaiting version 1.4 due out in early 1986.

Other descriptions and evaluations of M.1 are available (6, 7, 8).

RuleMaster:

RuleMaster (Radian corp.) is somewhat of an enigma. The tool is not just one program but rather three distinct entities: 1) Radial is a highly structured, pascal-like language; 2) Rulemaker is an induction routine which translates "examples" (logic tables) into Radial code (similar to Expert-Ease); and 3) the User interface, a sophisticated menu-driven collection of editors, tools, and various applications which help in the building of RuleMaster programs. What is puzzling about RuleMaster is its apparent lack of a separate control structure (inference engine). It is generally accepted that a separate control structure is one of the things that make an expert system - an expert system (9). If one works with RuleMaster, it becomes clear that the control structure is up to Rulemaker and/or the programmer (Knowledge engineer). This aspect of RuleMaster is truly a double-edge sword. On the positive side, the programmer is forced to structure the expert knowledge into modules that are easily updated by Rulemaker as well as the programmer. On the other side, trying to produce the effect of anything besides forward-chaining (ie. backward chaining) is practically impossible. Fortunately, many engineering problems are well structured and can be solved via forward-chaining inference.

RuleMaster handles complex mathematical functions by means of the usual "hook" to a separate program. What sets RuleMaster apart from the other tools evaluated is its ability to run under Unix or Unix-like operating systems. Since Unix can handle multitasking I/O, information can be easily shared between any number of programs written in any language. This is important if one has a large number of engineering algorithms written in different languages.

The explanation facilities are good. A nice feature is tiered explanation. One can keep querying the computer to get various levels of detail to the question - why?. RuleMaster does a good job at translating example knowledge to understandable explanation in context. The fact that the explanation can be put into context is not a

trivial feature. If the problem is very complex, then a simple explanation of the rule being queried is easily misunderstood. The User interface makes working with this tool very enjoyable, but, the interface was just moving out of development in August 1985. At that time, there were some bugs and some of the applications were not available.

If RuleMaster has a soft-spot, it must be considered the lack of comprehensive documentation. For such a powerful tool, all we received is the training course notes which they call a "system user manual". User support is available through training and contract consulting.

The approximate cost of RuleMaster is $10,000 - $15,000 for IBM - PC and AT computers and $25,000 for supermini's such as a VAX. Educational discounts and trial period arrangements are available.

Important to note is that Radian is primarily a scientific - engineering company and its product RuleMaster is primarily designed for those domains. It is nice to know that if you do have problems, you can talk to a civil engineer who knows knowledge engineering rather than just a knowledge engineer who is not a civil engineer. All in all, RuleMaster is an excellent tool for building large systems that do not require various levels of abstraction.

Additional descriptive information is available from Radian Corporation (10).

EXPERT:

EXPERT was developed at Rutgers University for use in their biomedicine program so it is not surprising to find the tool's design directed at this domain. But, just as the field of medicine relies on expert diagnosis so do many problems in engineering. For example, diagnosing operational problems in a "sick" wastewater treatment system or aiding in structural design problems. Surprisingly, EXPERT is a Fortran based tool. It uses standard productions rules to represent procedural knowledge and in the tradition of MYCIN, EXPERT incorporates certainty factors within a backward-chaining inference.

Even though EXPERT is written in Fortran it only handles the standard mathematical operations of +,-,/,*, and **. It also does not hook out to other software (even Fortran) and unless you are a Fortran Guru with lots of extra time, do not expect to change this. The explanation facilities are somewhat limited. The user can ask why?, unfortunately the explanation only concerns the last question asked by the computer and not with the context of the question. EXPERT does have a "trace" facility which allows the user to follow the program's logic. For a large mainframe system, EXPERT's documentation is a meager 40 pages. This consists of an overview, a simple diagnostic example, and command definitions.

S.1:

Just as M.1 was considered the Cadillac of small system tools, S.1 (Teknowledge) is the Rolls Royce of the large system tools evaluated. The features of S.1 are too numerous to even list in this paper. Ironically, S.1's biggest drawback is its luxury. Just as most people can not afford to use a Rolls Royce as a pickup truck, most knowledge engineers can not afford to use S.1 to develop small systems. Unfortunately, most prototyping falls into this category. S.1 is a huge program that requires alot of computer memory and time to run. It had no trouble eating one of our time-shared VAX11-780's for lunch, in fact, S.1 should really have its own super mini dedicated just for itself (ie. Xerox 1108 workstation).

On one hand, S.1 has a sophisticated development environment consisting of an editor, debugging tools, and graphical aids. It allows for various forms of knowledge representation as well as inference strategies. Also, its explanation facilities are quite extensive. One can ask such question as how, what, and why. On the other hand, complex mathematical algorithms must be written outside of S.1 in the Lisp programming language (What fun!?). Also, interfacing to existing data-bases does not appear to be a simple task.

The documentation consists of five volumes, mostly training material or sample expert systems. Unfortunately there is no master index and just like M.1, no individual examples of each command are given. Here again, one must dissect an entire expert system to understand why and how certain commands were used in order to build ones own system.

Teknowledge is known for their outstanding user support, little of which comes for free, but still it's nice to know its there when needed. These people probably have more experience in building expert systems than any other company. Their products might cost more, but the strong user support might more than compensate for the initial investment.

One gets the feeling S.1 was designed to solve very grandiose but not very technical problems, something like automating the mail room at the pentagon. If price is no object and one is fluent in Lisp, S.1 allows enough flexibility to handle even engineering problems, but remember to ask yourself "do I need a pickup or a Rolls Royce". Needless to say, S.1 is not for everyone.

Case Study:

Utah State University, department of Civil and Environmental Engineering is pioneering the application of expert system technology to the areas of environmental systems modeling and hazardous waste management. One current study deals with the development of a demonstration expert sytem for assessing organic chemical mobility and degradation in order to consult on soil treatment options.

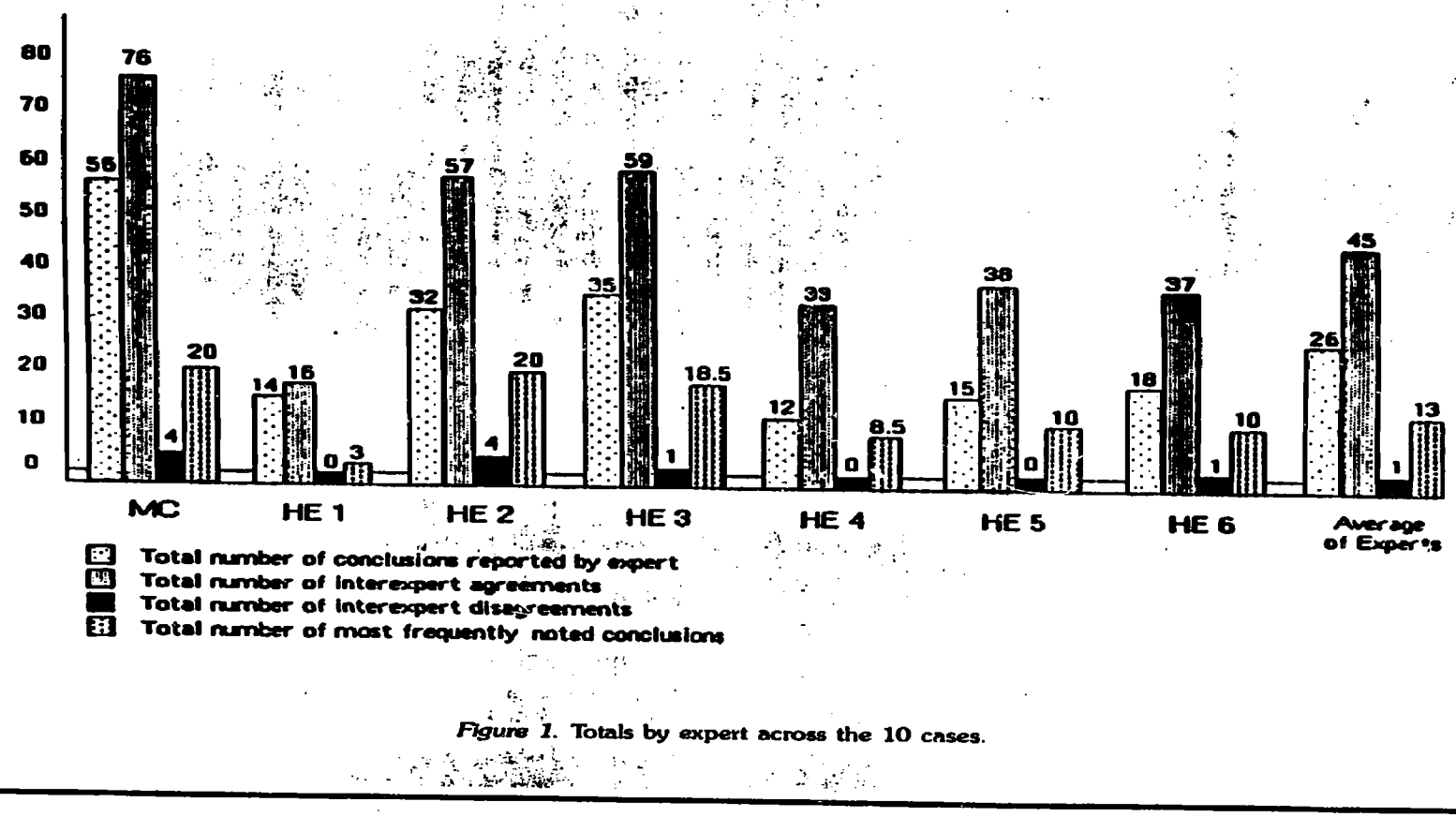


Figure 1. Totals by expert across the 10 cases.

The main advantage of expert is its ability to run on larger computer systems (ours is running on a VAX 11-780). This allows for large number of rules to be incorporated and executed quickly. EXPERT is available for prototyping at little or no cost, however, there is no formal source of user support. EXPERT could be customized into a very powerful engineering tool, if one has access to a relatively large computer and is proficient in Fortran and fundamental expert system programming. If you can not afford customization, then plan on using EXPERT to solve diagnostic problems which do not involve complex algorithms.

Other descriptions and evaluations of EXPERT are available (11, 12).

ROSIE:

ROSIE (Rand Corporation) has been described ar a general- purpose AI language as well as an expert system building tool (13). Since ROSIE is written in the INTERLISP programming language, it naturally picked up many of INTERLISP's features. As an expert system tool, ROSIE uses an English-oriented syntax in its knowledge base and input/output facilities. At first glance, it seems ROSIE has a built in natural language interface, however, it makes no attempt to grasp unrestricted English input. One must learn to "talk" to ROSIE in a very structured manner which resembles simple english sentences. It is still impressive to see the user interact with ROSIE by typing in small reports describing certain situations rather than answering one query at a time.

Because ROSIE is a large program which requires a large language (INTERLISP) to run, it commandeers significant memory and run-time. If one is paying for these services, the development costs can be restrictive. A mitigating factor is ROSIE (VAX-VMS version) can be obtained for approximately $200, but be prepared to spend several thousand dollars for INTERLISP.

ROSIE does not incorporate complex mathematical functions as part of the tool nor does it make allowances for interfacing with other software. The explanation facilities are very limited. One must use a "trace" or "scan" command to indirectly find out what is going on, rather than just simply asking why?. The development environment is also limited. There are no build in editors, menus, or graphical aids. The documentation consists of three volumes and is sufficient to get started on small to medium size systems. Little advice is given on building large systems via ROSIE. Finally, no formal support is available fc. ROSIE. Rand Corporation does consult on ROSIE but does not support it in a marketable way.

Additional descriptive information is available from Rand Corporation (14, 15, 16).

Appendix D

Expert Systems:
Implications for the Diagnosis and Treatment of Learning Disabilities

Since this system would be used for demonstration purposes, the portability of a PC-based program appeared attractive. For this same reason of demonstration, the ability to produce "run-only" versions of the expert system was considered an important factor. After some deliberation, M.1 was choosen for system development because the potential was there for building an expert system with the forementioned characteristics. However, one must be aware of the surrounding circumstances. First, since we are an education institution, all of the evaluated tools have been aquired at a greatly reduced cost or no charge at all. Second, Utah State University has available considerable in-house expertise in building expert systems as well as software engineering in general. Lastly, some of the newer PC-based tools, such as Insight II, were not available at the onset of this project. If carried out today, the decision of which tool to use for building this demonstration expert system would probably be different.

Food For Thought:

It is of utmost importance for any civil engineer who wishes to build an expert system to realize that one can learn to be a knowledge engineer rather rapidly, in fact many civil engineers are already knowledge engineers without even knowing it. Most engineers are quite good at extracting complex knowledge and based on scientific assumptions, produce simplified heuristics (Rules-of-thumb). Obviously, one can not be just a knowledge engineer and expect to become a civil engineer overnight. For this reason, civil engineers should seriously consider building their own systems with the help of flexible and user friendly tools before hiring those who are not familiar with civil engineering problems.

Since this system would be used for demonstration purposes, the portability of a PC-based program appeared attractive. For this same reason of demonstration, the ability to produce "run-only" versions of the expert system was considered an important factor. After some deliberation, M.1 was choosen for system development because the potential was there for building an expert system with the forementioned characteristics. However, one must be aware of the surrounding circumstances. First, since we are an education institution, all of the evaluated tools have been aquired at a greatly reduced cost or no charge at all. Second, Utah State University has available considerable in-house expertise in building expert systems as well as software engineering in general. Lastly, some of the newer PC-based tools, such as Insight II, were not available at the onset of this project. If carried out today, the decision of which tool to use for building this demonstration expert system would probably be different.

Food For Thought:

It is of utmost importance for any civil engineer who wishes to build an expert system to realize that one can learn to be a knowledge engineer rather rapidly, in fact many civil engineers are already knowledge engineers without even knowing it. Most engineers are quite good at extracting complex knowledge and based on scientific assumptions, produce simplified heuristics (Rules-of-thumb). Obviously, one can not be just a knowledge engineer and expect to become a civil engineer overnight. For this reason, civil engineers should seriously consider building their own systems with the help of flexible and user friendly tools before hiring those who are not familiar with civil engineering problems.

Appendix D

Expert Systems:

Implications for the Diagnosis

and Treatment of Learning Disabilities

EXPERT SYSTEMS: IMPLICATIONS FOR THE DIAGNOSIS AND TREATMENT OF LEARNING DISABILITIES

Alan M. Hofmeister and Margaret M. Lubke

Abstract. **Application of artificial intelligence to the problems of education is a relatively recent endeavor. This article will focus on one of the most promising aspects of artificial intelligence — *expert systems* technology — and some of the characteristics that make expert systems "intelligent". Selected present and potential applications of expert systems to the field of learning disabilities are presented along with examples of specific expert systems.**

Application of computer technology to the field of learning disabilities has taken a variety of forms, the most common being computer-assisted instruction (CAI), computer-managed instruction (CMI), and computer-assisted testing (Hofmeister, 1984c). To a large extent these applications represent reasonably well-developed procedures that existed before microcomputers, but had to wait for the widespread availability of this technology to achieve their present popularity. More recently, a new computer technology —the *expert system*— has been developed.

A field within artificial intelligence, expert systems technology is concerned with the use of computers to capture and disseminate human expertise. Typically, expert systems have proven effective in medicine, geology, chemistry, engineering, and business. However, educators have recently begun to show an interest in this technology, particularly as it can be applied to the problems associated with learning disabilities. This article reports on present and potential applications of expert systems technology to diagnosis and treatment of learning disabilities.

EXPERT SYSTEMS

Knowledge Engineering and "Expert Systems"

Knowledge engineering is the term often used to describe the process of capturing human expertise, developing a problem-solving framework, and eventually making the knowledge available to others through a computer-based expert system. The expert system usually gathers information from the user in a dialogue format that simulates a consultation with a human expert. Many expert systems are designed to explain their line of reasoning in everyday English rather than computer code.

Reasoning Procedures

The expert system's reasoning procedures, sometimes referred to as the *inference engine*, acts upon the combination of user-supplied information and information contained in the expert system's knowledge base.

To facilitate the interaction with the inference engine, the knowledge base is organized into rules, consisting of two components. an "if" component and a "then" component. When the conditions in

ALAN HOFMEISTER, Ph.D., is a professor in the Department of Special Education and the Director of the Artificial Intelligence Research and Development Unit, Utah State University.

MARGARET LUBKE is a doctoral student in the Department of Special Education and a Research Associate in the Artificial Intelligence Research and Development Unit.

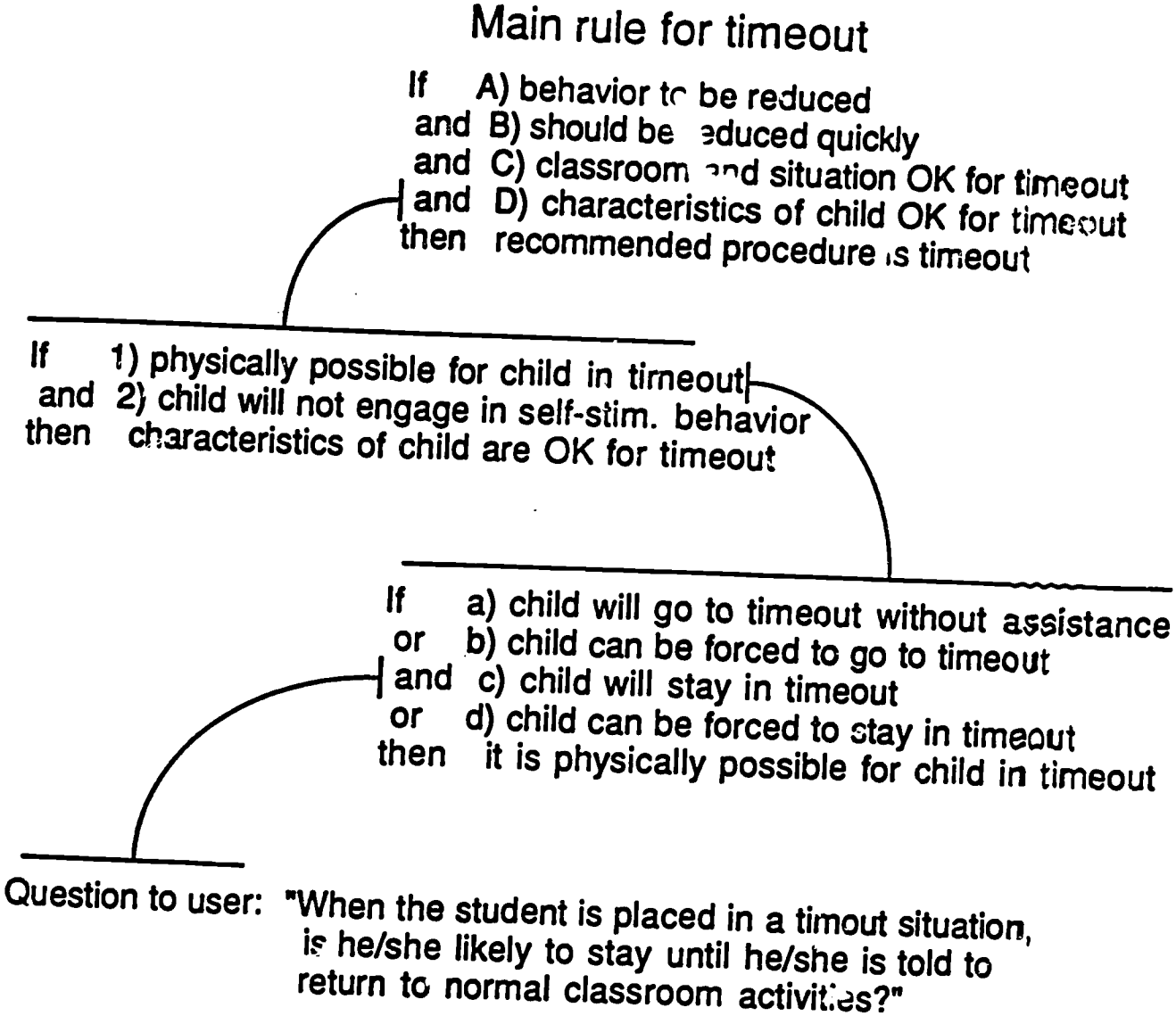


Figure 1. Figure 1 illustrates that intermediates within if-then rules (A,B,C,D within the main rule for timeout) can be made up of other rules which themselves can have intermediates. At the bottom of the path is a question for the user that would begin this segment of information gathering by the system. Only a single path of rules and intermediates is shown.

Appendix G

Formative Evaluation in the Development

and Validation of Expert Systems in Education

Comput. Intell. 2, 000-000 (1986)

Formative evaluation in the development and validation of expert systems in education

ALAN M. HOFMEISTER
Artificial Intelligence Research and Development Unit, Utah State University, Developmental Center for Handicapped Persons, Logan, UT 84322, U.S.A.

Received November 14, 1985
Revision accepted January 16, 1986

Researchers developing and validating educational products often expect the same field-test activities to provide information on product improvement and product effectiveness. For effective and economical use of resources, these two goals, product improvement and product validation, must be stressed at different times and with different tools and strategies. This article identifies the difference in procedures and outcome between formative and summative evaluation practices and relates these practices to the development and validation of expert systems in education.

Key words: expert systems, product development, product validation, research models.

French Here

Introduction

Educators have been comparatively slow in applying expert systems technology to instructional and management problems in the schools. There have been some very good reasons for this delay, not the least of which is education's limited research and development budget. If the limited resources of educators are to be used effectively in the development of expert systems, their efforts must be rationally guided and progressively refined. One way to do this is to use research and development models. These models should provide a theoretical base, serve to crystallize past successful efforts, and ensure that future models and their product outcomes are built on tested practices.

General-purpose models exist to guide the development and validation of expert systems. One of the better-known models lists five major stages in the development of an expert system, namely, identification, conceptualization, formalization, implementation, and testing (Hayes-Roth *et al.*, 1983).

These general-purpose models have considerable value for guiding initial planning. The value of these models diminishes as implementation approaches. The need for more problem-specific models represents a natural evolution. The differentiation in models occurs as we attempt to guide the development of different types of expert systems with different theoretical bases. Constraint-based and structured-selection approaches to expert systems may call for differences in specific product development procedures. Differentiation in development and validation procedures may also occur as we create expert systems in different disciplines. This differentiation should not be construed as a move towards partisanship among the disciplines. The quality of all expert systems product development efforts can only be enhanced by experimentation with a range of product development constructs and their supporting tools.

This article presents and discusses formative evaluation, a key component of most educational research and development models. As educators search for discipline-specific models to provide more direction to the development of expert systems in education, formative evaluation must receive more serious consideration. While the data-based outcome or validation testing is often approached systematically, the preceding

formative evaluations often receive less systematic attention.

Formative and summative stages: an overview

The roles of formative and summative evaluation

In a landmark article on program evaluation practices in education, Scriven (1967) used the terms "formative" and "summative" to differentiate two stages in the development and evaluation of an educational program or product. In formative evaluation, the primary question is "What do I have to do to improve the product?" In summative evaluation, the primary question is "How well does this product work?"

Where a developer stresses the differences between summative and formative evaluation practices, concern for formative procedures increases. Historically, the importance of such formative procedures had been largely ignored except for a few observations, such as the following by Cronbach (1963):

> Evaluation used to improve the course while it is still fluid contributes more to the improvement of education than evaluation used to appraise a product already on the market. [p. 675]

The concepts of formative and summative evaluation are now important components of most models used in educational product development and validation (Borg and Gall 1979; Markle 1967; Hood 1973; Hofmeister 1976; Sanders and Cunningham 1974).

Examples of formative and summative evaluation practices

In the formative stage, investments in "polishing" the product are minimal. Field testing is restricted to the intense observation of small samples of the target population to determine reasons for product malfunction. Consultant critiques and other similar evaluation practices occur as early as possible, before there is a major fiscal and ego investment in the preliminary product characteristics and content.

The summative stage follows the formative stage. Only when the formative stage is largely completed are major investments made in the appearance and "packaging" of the product. Summative field testing usually includes much larger samples of the target population, and the associated evaluation procedures are concerned with assessment of the final outcomes rather than studying product functioning through the more intensive and intrusive monitoring practices of formative evaluation. Generalization and experimental control are far more important concerns in summative evaluations than in formative evaluations.

To discriminate between formative and summative evaluation activities, the roles of the evaluation information, not the specific data-gathering tools, are important. The respective roles have been summarized by Borich (1974) as follows:

> Information for program revision is perhaps the single most important characteristic of formative evaluation, while information for program adoption is the single most important characteristic of summative evaluation. [p. 272]

The formative stage

Major components of the formative stage

Two of the major activities of the formative stage are the design of the system and the development and revision of a series of prototypes. In designing the system, consideration is given to such questions as

(a) What type of problems should the expert system address?
(b) What type of information output should the expert systems provide?
(c) Who will use the system?
(d) Under what field conditions will the system be used?
(e) What software and hardware tools hold the most promise?

In the development and revision of a series of prototypes, the initial emphasis is on individual components or modules. These individual modules must be progressively revised to ensure that performance is consistent with specifications established in the design stage. If opportunities arise that allow the developers to improve upon design specifications, such opportunities should be exercised. Such is the nature of formative processes. In the latter stages of prototype development and revision activities, the emphasis shifts to the relationship among the different modules, and then to the performance of the total system.

Design specifications and knowledge engineering

The relationship between design specification activities and prototype development and revision is complex and interactive. There is certainly not a simple linear lock step relationship. It is tempting, from a management viewpoint, to complete the design specifications and then not look back as the prototypes are developed. Such thinking ignores the very nature of knowledge engineering. Design specifications must be based on

our knowledge of the problem at a point in time. Often, the very process of developing the knowledge base increases our understanding of the problem. This increased understanding could lead to an enhancement of the design specifications of the product. This interaction between design and development has been advocated by Nocentini (1985) who noted, "I know of no case where a software project has been successfully completed by following the original plans laid for it"(p. 122)."

Predevelopmental activities

The previously mentioned formative activities are all premised on the assumption that an expert system is an appropriate approach to handling the problem. Sanders and Cunningham (1974) suggest that "predevelopmental" activities should precede the initial product design activities. These predevelopmental activities should seek to verify the need for the expert system through logical and empirical analyses. For example, if the problem is a lack of expert assistance to interpret complex regulations, it might be more appropriate to simplify the regulations so that expert assistance is not needed.

Formative evaluation procedures

Formative evaluation practices stress the collection of evaluation data as early as possible during the processes of designing, developing, and modifying prototypes. The following are examples of some of the evaluation practices that will yield data to drive the product development processes.

Consultant review of tool selection and general systems characteristics. This is done to ensure that the systems goals are appropriate and that the major systems characteristics are consistent with the goals.

Consultant validation of the primary rules of the knowledge base. As the knowledge base is developed, a range of evaluation processes is available to ensure that the knowledge is accurate and important for the problems addressed by the system. Factual knowledge can be validated by double-checking the sources and searching for supportive evidence in syntheses of the research. Heuristic knowledge can be cross-checked by seeking confirming opinions from additional experts.

Assessment of selected reasoning processes. As the knowledge base and associated reasoning procedures are developed, the accuracy of selected system decision-making procedures can be checked through the use of a carefully designed set of test problems. Each of the test problems consists of user input information and the associated outcome that should be obtained if the rules and the associated reasoning processes are accurate. These test examples can be developed jointly by the experts and the knowledge engineers developing the expert system. Sample test problems can be added as the expert system is refined and expanded during the formative stage.

These test problems and their associated outcomes also provide a vehicle for obtaining input from additional experts. The user input information of test problems is given to these experts. They are asked to provide outcome information for each of the test problems. By comparing this outcome information with that provided by the development team, additional information on the validity of the system can be obtained.

Preliminary assessment of user reliability. Using a small bank of actual field problems, preliminary information on user reliability can be obtained. This bank of problems should be representative of the actual target problems the system is being designed to solve.

Two aspects of user reliability should be evaluated: (a) consistency of outcome by the same user over time and (b) consistency of outcome when different users attempt to solve the same problem. These assessment procedures should provide information on such factors as ambiguities in consultation questions by the system and problems in extracting information from the user. The educational research procedures associated with the coefficient of stability and interobserver agreement measures (Borg and Gall 1979) should provide a range of data collection and data analysis procedures for conducting evaluation of user reliability issues.

Progressive refinement using field problems. Once the reliability of data entry and other user procedures has been established, the system is progressively refined. Field problems are systematically sampled in small groups and the expert system revised, based on the system's response to each group of problems. To ensure a systematic approach to the sampling, the target population of field problems is subdivided into different problem types and a small group of field problems sampled from each type. The results of the consultation for each sample group are examined by knowledge engineers and knowledge base consultants to ensure that the outcome of the consultation is appropriate and the interaction between system and user efficient.

At the end of each major revision, the bank of test problems is rerun. If no problems are encountered, the supporting documentation is revised to reflect all changes made. It is necessary to rerun the full bank of test problems to help ensure that modifications made in one part of the system did not inadvertently affect the decision-making processes in other areas.

If a problem type is encountered that is not adequately represented in the bank of test problems, the bank of test problems is supplemented or revised to ensure that the bank does test all the important decision-making processes.

The sampling and revision process is continued until all problem areas in the target population of problems have been sampled and the implementation of new field problems results in no system changes. At this point the major formative evaluation procedures are completed.

Psychological climate

When conducting formative evaluation activities, the researcher's disciplined objectivity is constantly challenged. In summative evaluation, a comparatively rigid set of statistical and research design procedures exists to ensure that appropriate levels of objectivity are maintained. In formative evaluation, success depends less on adherence to a highly prescriptive set of procedures and more on creativity and the level of objectivity and self-criticism displayed by the individual researcher. These important psychological attributes must be constantly fostered until they pervade and supersede the more technical and prescriptive procedures associated with formative evaluation. The reality is that we are in the business of problem solving, and investments in knowledge engineering are not usually made in areas where the answers are obvious and the best solutions, the result of mindless adherence to a "systems" recipe.

Summary

In summary, the major product development activities of the formative stage are (a) a predevelopment review of the need for the expert system, (b) the design of the system, and (c) the development and revision of a series of prototypes. In their rush to gather summative evaluation data, educational researchers often neglect these formative processes or erroneously assume that field testing designed for generalization and experimental control will also be appropriate for answering the question, "How can I improve this product?" An even greater error is committed when the summative evaluation emphasizes comparisons among partially developed approaches. Such premature comparisons of partially developed procedures may condemn a potentially powerful procedure to obscurity. Too often the consumer of the research focuses on the differences in outcomes without reference to the levels of development of the procedures involved.

Borg, W. R., and Gall, M. D. 1979. Educational research: An introduction. 3rd ed. Longman, Inc. New York, NY, pp. 782.

Borich, G. D. (*Editor*). 1974. Evaluating educational programs and products. Educational Technology Publications, Englewood Cliffs, NJ, pp. 491.

Cronbach, L. J. 1963. Course improvement through evaluation. Teachers College Record, 64, pp. 672–683.

Hayes-Roth, R., Waterman, D. A., and Lenat, D. B. 1983. An overview of expert systems. *In* Building expert systems. *Edited by* F. Hayes-Roth, D. A. Waterman, and D. B. Lenat. Addison-Wesley Publishing Company, Inc. Reading, MA, pp. 3–29.

Hofmeister, A. M. 1976. Educational research and development. *In* Leisure activity participation and handicapped populations: An assessment of research needs. *Edited by* P. J. Verhoven and J. E. Goldstein. National Recreation and Park Association, Arlington, VA.

Hood, P. D. 1973. Evaluation stages for major laboratory products. *In* Educational development: A new discipline for self renewal. *Edited by* J. K. Hemphill. University of Oregon, Eugene, OR.

Markle, S. M. 1967. Empirical testing of programs. *In* Programmed instruction. Sixty-sixth yearbook of the National Society for the Study of Education. *Edited by* P. C. Lange. University of Chicago Press, Chicago, IL.

Nocentini, S. 1985. The planning ritual. Datamation, 31(8), pp. 122–128.

Sanders, J. R., and Cunningham, D. J. 1974. Formative evaluation: Selecting techniques and procedures. *In* Evaluating educational programs and products. *Edited by* G. D. Borich. Educational Technology Publications, Englewood Cliffs, NJ, pp. 279–311.

Scriven, M. 1967. The methodology of evaluation. Perspectives of Curriculum Evaluation. AERA Monograph Series, Rand McNally, Chicago, IL, pp. 39–83.

END OF TAPE 1

Appendix H

Accuracy of CLASS.LD2:

An Expert System for Classifying Learning

Disabled Students

Accuracy of Class.LD2: An Expert System for Classifying Learning Disabled Students

Elizabeth S. Martindale, Joseph M. Ferrara, and Brian W. Campbell

Utah State University

Department of Special Education

Running head: ACCURACY OF CLASS.LD2

Accepted for publication in Computers in Human Behavior (in press)

Abstract

This study determined the accuracy of Class.LD2, an expert system for classifying learning disabled students. Of 264 student files, 78 files were chosen based on disagreement between multidisciplinary team and Class.LD2 decisions regarding eligibility for LD placement. These 78 cases were evaluated by three individuals expert in LD classification, who made an eligibility decision for each case. Their decisions were compared to those of the expert system2. Analysis of the results indicated that 1) Class.LD2 was in agreement with the experts more often than were the placement teams, 2) the expert system's decisions were significantly correlated with those of the experts, and 3) in those cases in which the experts were in unanimous disagreement with Class.LD2, it was shown that Class.LD2 conformed strictly to state rules and regulations in making its eligibility decisions. The use of Class.LD2 by placement teams in order to encourage more data based decisions and to limit overclassification errors is also discussed.

Accuracy of Class.LD2: an Expert System for Classifying Learning Disabled Students

Special education placement teams must make decisions regarding the identification of handicapped children and their eligibility for special education services. Current data suggests the judgments made by placement teams may not be accurate. Ysseldyke (1983) contends that half of the number of identifications in the area of learning disabilities may be inaccurate. Hofmeister (1983) reported an 84% increase in the number of pupils identified as learning disabled during the past few years. Placement team inaccuracy may be related to this dramatic increase in the number of pupils identified as learning disabled (Algozzine & Korinek, 1985; Algozzine, Ysseldyke & Christenson, 1983).

If placement teams are doing a poor job of applying eligibility criteria, handicapped and nonhandicapped children are not being well served. First, nonhandicapped children may be receiving inappropriate services. Second, if nonhandicapped students are qualifying for special education, there is less money available to meet the needs of the

handicapped, and essential services for children with other handicaps are being reduced to pay for the new learning disabled students (Sabatino, 1981).

In some states the problem of overclassification has reached crisis proportions. State education agencies have responded to this problem by placing limits on the number of students who can qualify and receive funding for services in the learning disabled category (Boyan, 1985; Reynolds, 1983). Such limits may solve the immediate fiscal crisis. Unfortunately, they subvert the intent of P.L. 94-142 in that they discriminate against handicapped children who happen to live in schools with an unusually high handicapped population. At the same time, they encourage aggressive and "imaginative" administrators in districts with a low percentage of handicapped students to find pseudo-learning disabled students to fill their "quota" and obtain additional funding.

An alternative to state imposed limits is to improve the accuracy of the decisions made by placement teams. Research on the current functioning of placement teams indicates that their inaccuracy may be the result of a flawed approach to placement

decision making. Few school placement teams employ a systematic approach to determining eligibility (Ysseldyke, Algozzine & Mitchell, 1982). As a result, a great deal of data describing student performance is collected, but much of it is technically irrelevant (Thurlow & Ysseldyke, 1979). Teams spend about 30% of their time discussing these questionable data (Ysseldyke, Algozzine, Rostollan, & Shinn, 1981). Furthermore, individuals discussing these data are likely to use language which is unfamiliar to noneducators (Ysseldyke, 1983). Considering this unfortunate state of affairs, it is not surprising that Ysseldyke, Algozzine, Richey, and Graden (1982) reported that there was little relationship between the psychometric data presented to placement teams and the eligibility decisions made by those teams.

Two changes in the decision making process might reduce the frequency of errors. First, teams should follow a rational series of steps which encourage the examination of relevant data. Second, teams should use a set of regulation-based rules to make judgments about placement data.

One way to bring about these changes involves the application of expert system technology. An expert system is a computer program which attempts to replicate the decision making and problem solving skills of knowledgeable and effective human experts (Hofmeister & Ferrara, in press). If an expert system which behaved like a knowledgeable, systematic expert in the area of eligibility decision making could be developed, it might serve as a useful tool for placement teams. An expert system might force a rational step-by-step approach which would result in judgments which were directly related to data and regulations (Hofmeister & Lubke, in press).

Before such a system can be used by placement teams its judgement must be shown to be competent. An expert system, Class.LD2 (Ferrara & Hofmeister, 1984), has been developed at Utah State University. The purpose of Class.LD2 is to evaluate student eligibility for special education placement (Ferrara, Parry & Lubke, 1985). The degree to which Class.LD2 outperforms placement teams should be related to its potential value as a tool for increasing the accuracy of placement team decisions. One way to evaluate

Class.LD2's performance is to compare its judgments to those of placement teams as well as to the judgments of acknowledged experts in the area of LD classification and placement. The purpose of this study was to make these comparisons.

This study addressed the following research questions:

1. Were the decisions made by Class.LD2 and the placement teams different?

2. When placement teams and Class.LD2 did not agree, what was the nature of the disagreement?

3. In cases where placement teams and Class.LD2 disagreed, whose decisions did experts judge to be correct?

4. What were the characteristics of cases where experts agreed with the placement team instead of Class.LD2?

METHOD

Class.LD2

Class.LD2 (Ferrara & Hofmeister, 1984) is an expert system designed to provide a second opinion regarding the classification of students as learning disabled (Ferrara et al., 1985; Hofmeister &

Lubke, in press). Class.LD2 uses a dialogue or consultation format to obtain information from the user. That information is compared against the rules in a preprogrammed knowledge base and is used to make a classification decision.

CLASS.LD2's knowledge base contains approximately 600 "if-then" rules. The rules are based on P.L. 94-142, Utah state regulations, and current literature describing best practices in the area of student assessment and placement. As questions are generated by the program and answered by the user, decisions are made based on a comparison of the answers to the information in the knowledge base.

Sample cases

Data from 264 files which described potential special education students were used to evaluate the performance of Class.LD2. These files were selected from three school districts. District 1, located in Idaho, provided 150 files. District 2, located in Utah, contributes 54 files, and District 3, located in Wyoming, contributed 60 files. Of the total number of files, 110 of the students were judged by placement teams to be eligible for placement as learning

disabled. The remaining 154 cases were either judged to be nonhandicapped or eligible for services in another disability area.

Data Input

The data from each student file was provided to Class.LD2 in a consultation format. Special education teachers familiar with each case entered the data from Districts 2 and 3. District 1 data were entered by graduate students pursuing a PhD in psychology. The graduate students worked as consultants to District 1 and were familiar with each case. Neither the graduate students nor the teachers were involved in the development of Class.LD2. They were provided instruction on Class.LD2 prior to their use of the system. Records of each data entry session were kept. These records contained the questions asked by Class.LD2 and each response made by the users. Data entry sessions took about 8 minutes per case.

Disagreeing Cases

When Class.LD2 had evaluated each student's data, its judgment was compared with the judgment made by the placement team. Cases where disagreement occurred were identified.

Class.LD2 does not normally provide a dichotomous judgment (i.e., LD or not LD). Rather, Class.LD2 will provide as many as eight positive hypotheses regarding the student's area of disability. Certainty factors of -100 through +100 are then assigned to these hypotheses. In order to judge each case as an agreement or disagreement, Class.LD2's judgments had to be viewed as dichotomous rather than continuous data. If Class.LD2 assigned a certainty factor of +50 or greater to the LD hypothesis, the student was viewed as being judged by the system to be eligible for LD services. Therefore, if a certainty level of +49 or lower was assigned, the student was considered to be not LD.

Expert Evaluator

Cases where Class.LD2 and the placement team disagreed were submitted to three experts. Expert 2 is a school psychologist in a large Utah school district. Expert 1 and Expert 3 are employed by the Utah State Office of Education. A principle component of all three experts' work involves assisting local school districts in making placement decisions.

The experts were given data sheets which

contained the same information that was provided to Class.LD2. Based on this information, the experts were asked to judge students as either eligible or ineligible for LD placement.

In cases where the experts unanimously disagreed with Class.LD2's judgement, the intermediate conclusions of Class.LD2 were examined. Specifically, three critical attributes were examined: (a) IQ level, (b) a discrepancy score based on the student's actual grade level and, (c) a discrepancy score based on an estimate of expected performance.

Results

Placement Team Decisions

Of the 264 files checked with Class.LD2, there were 78 cases where placement team decisions did not match Class.LD2's decisions. In other words, Class.LD2 and the placement teams were in agreement 67% of the time. Table 1 shows a 2x2 table illustrating the relationship between Class.LD2's decisions and the placement teams' decisions. A phi coefficient of phi=.46 was calculated using these data.

Insert Table 1. about here

Tables 2, 3, and 4 show the relationship between the placement decisions of Districts 1, 2, and 3 and those of Class.LD2. A phi coefficient of phi=.27 was calculated for District 1. A phi coefficient of phi=.56 was obtained for District 2. Finally, the relationship between Class.LD2's judgments and those of District 3's placement team is described by a phi=.24.

Insert Table 2. about here

Insert Table 3. about here

Insert Table 4. about here

Nature of Disagreements

There were 78 cases where Class.LD2 and the placement team did not agree. These 78 cases can be divided into two categories: (a) Type A cases, where the placement team said the student was learning disabled and Class.LD2 said the student was not, and (b) Type B cases, where the placement team said the student was not learning disabled and Class.LD2 said

the student was learning disabled. There were 68 Type A cases and 10 Type B Cases. When disagreement occurred, then, 87% of the disagreements were a result of either team overclassification or machine underclassification.

Accuracy of Decisions

The 78 cases where disagreement occurred were submitted to three experts. When taken together, the experts made 234 (3X78) judgments. Figure 1 shows the experts' total agreement and disagreement with Class.LD2.

Insert Figure 1. about here

Table 5 shows the relationship between the experts' placement decisions and those of Class.LD2. Using the data shown in Table 5, a phi coefficient of phi=.40 was calculated. This phi is associated with a chi-square of 36.62, which is significant at the level required to reject the null hypothesis that there is not a relationship between the experts' judgments and Class.LD2's at the $p<.001$ level of confidence.

Insert Table 5. about here

Table 6 shows the relationship between Class.LD2's decisions and all cases where the three experts agreed. There were 58 such cases. Phi=.63 and a chi-square of 23.38 were calculated to describe this relationship. Once again we can reject the null hypothesis that there was no relationship between Class.LD2's judgments and the experts' judgments at the $p<.001$ level of confidence.

Insert Table 6. about here

Table 7 shows a correlation matrix describing the relationship between all five judgments of student eligibility (Class.LD2, placement teams, and the experts). The decisions of Class.LD2 and of all three experts are inversely related to placement team decisions. It should be noted that only cases where placement teams and Class.LD2 disagreed were used to calculate these correlations.

Insert Table 7. about here

Characteristics of Cases

The human experts' judgments were unanimous in only 6 of the 78 cases of Class.LD2/placement team disagreements. Table 8 shows Class.LD2's intermediate conclusions for these cases on their critical variables: IQ, estimated discrepancy and actual discrepancy. In all six of the cases where Class.LD2 and the experts disagreed, Class.LD2 made an underclassification error.

Insert Table 8. about here

In each case, the student's data failed to satisfy one of the LD cutoff levels established in Class.LD2's knowledge base. These cutoff levels include a minimum IQ of 83 or 84 (depending on the standard deviation of the IQ test used) and a 40% discrepancy between expected and observed academic performance.

Discussion

Overall Agreement

The results of this study suggest that Class.LD2 and placement teams tended to agree on most (74%) cases. There was, however, enough disagreement to

justify an evaluation of the 78 disagreeing cases.

Evaluation of Disagreeing Cases

The 78 cases in which Class.LD2 and the placement teams disagreed can be viewed as difficult cases. In these situations Class.LD2 clearly and consistently outperformed the placement teams. The few cases in which the experts all disagreed with Class.LD2 were marginal. Furthermore, Class.LD2's decisions were defensible and its reasons for making those decisions were clear.

Cautions for Interpreting the Data

This study uses a portion of the data which were collected during the formative evaluation of Class.LD2. As a result, both the data and the analysis are geared toward product development rather than a summative report on the performance of the product. The reader should be alerted to a number of conditions which limit the degree to which the conclusions of this study can be assumed to generalize to a larger population.

Utah Guidelines for Idaho and Wyoming Students

The reader should recognize that Utah's LD classification criteria were used to evaluate files

from two other states. Class.LD2 was more likely to agree with placement teams from District 2 (the Utah District) than with teams from the Idaho and Wyoming school districts. The placement teams' degree of inaccuracy which might be inferred from these data is probably inflated.

Fence Sitting was not Allowed

In this analysis, Class.LD2's advice was transformed from continuous data to dichotomous data. As a result, some error was introduced. For example, Class.LD2's consideration of one student's data resulted in an advice statement which suggested that the student could be classified LD at a +48 level of confidence. Using a +50 cutoff, Class.LD2's advice in that case was considered to be not LD. In the field, a +48 certainty level would probably not be interpreted as an absolutely negative response. Most professionals would recognize that Class.LD2 was suggesting that, in this case, an LD placement would be marginally defensible. The continuous nature of Class.LD2's advice would make its judgments useful even in those cases where it disagreed with one or more of the experts. But in this study near misses

did not count.

Controversial Issues: Class.LD2 is Smarter than it Looks

Several elements of the currently accepted criteria for LD placement are open to professional debate. For example, if a student's learning problems can be primarily attributed to environmental, economic, or cultural deprivation, that student cannot be classified as learning disabled.

Not everyone agrees (Sabatino, 1983) that students with learning problems attributable to environmental or economic factors should be excluded from LD placement. Conversely, concern has also been expressed that environmentally and economically disadvantaged students have been placed in special education programs and that this practice could place learning disability programs in danger of becoming a dumping ground for all educational problems (Kirk & Kirk, 1983).

The experts evaluating the placement team/Class.LD2 disagreement cases were not in agreement on the environmental or economic issue. In nine cases where there was clear environmental and/or economic deprivation, the experts disagreed. Two of

the experts chose to call all nine of these children learning disabled. The other expert chose not to classify any of them as learning disabled.

In all of these cases, Class.LD2's advice took a middle ground. The program alerted its users to the controversial issue and then pointed out that a strict interpretation of current state and federal guidelines would not allow an LD placement. For the purpose of analysis, Class.LD2's advice in all such cases was judged as not LD. This interpretation accounted for a large portion of the disagreements between Class.LD2 and the experts. Clearly, the analysis conducted in this study was not sensitive to Class.LD2's appropriately moderate advice in controversial areas.

Use of An Expert System to Limit Overclassification

If Class.LD2 had been used by placement teams to generate a second opinion regarding LD placement, the number of overclassification errors might have been reduced. The decisions made by placement teams might then have reflected a more accurate application of state and federal rules and regulations.

If the use of Class.LD2 does indeed reduce placement team error, the need for state imposed LD

limits will be reduced. Current research is attempting to measure the effect of Class.LD2 on placement team performance.

References

Algozzine, B. & Korinek, L. (1985). Where is special education going? Exceptional Children, 51 (5), 388-394.

Algozzine, B., Ysseldyke, J. E., & Christenson, S. (1983). An analysis of the incidence of special class placement: The Masses are burgeoning. The Journal of Special Education, 17 (2), 141-147.

Boyan, C. (1985). California's new eligibility criteria: Legal and program implications. Exceptional Children, 52 (2), 131-141.

Ferrara, J. M., & Hofmeister, A. M. (1984). CLASS.LD2: An expert system for classifying learning disabilities, [Computer program]. Logan, UT: Utah State University.

Ferrara, J. M., Parry, J. D., & Lubke, M. M. (1985). CLASS.2: An expert system for student classification. Technical Paper. Logan, UT: Utah State University. (ERIC Document Reproduction Service No. ED 263 734)

Hofmeister, A. M. (March 21, 1983). Comments on part B, Section 618: Evaluation and Part E, research and education of the handicapped, Public Law 94-142.

Testimony before the Senate Committee on Labor and Human Resources, Subcommittees on the Handicapped, Congressional Record. Washington, DC: Government Printing Office.

Hofmeister, A. M., & Ferrara, J. M. (in press). Expert systems and special education. Exceptional Children.

Hofmeister, A. M., & Lubke, M. M. (in press). Expert systems: Implications for the diagnosis and treatment of learning disabilities. Learning Disabilities Quarterly.

Kirk, S. A., & Kirk, W. D. (1983). On defining learning disabilities. Annual Review of Learning Disabilities 1, 17-18.

Reynolds, M. C. (1983). Classification of students with handicaps. Unpublished manuscript, University of Minnesota, Minneapolis.

Sabatino, D. A. (1981). Are appropriate educational programs operationally achievable under mandated promises of P.L. 94-142? The Journal of Special Education, 15, 9-22.

Sabatino, D. A. (1983). The house that Jack built. Annual Review of Learning Disabilities 1, 23-24.

Thurlow, M., & Ysseldyke, J. E. (1979). Current assessment and decision-making practices in model programs for learning disabled students. Learning Disability Quarterly, 2, 15-24.

Ysseldyke, J. E. (1983). Current practices in making psychoeducational decisions about learning disabled students. Annual Review of Learning Disabilities, 1, 31-38.

Ysseldyke, J. E., Algozzine, B., & Mitchell, J. (1982). Regular education teacher participation in special education team decision making. Elementary School Journal, 60, 308-313.

Ysseldyke, J. E., & Algozzine, B., Richey, L., & Graden, J. (1982). Declaring students eligible for learning disability services: Why bother with the data? Learning Disability Quarterly, 5, 37-44.

Ysseldyke, J. E., Algozzine, B., Rostollan, D., & Shinn, M. (1981). A content analysis of the data presented at special education team meetings. Journal of Clinical Psychology, 37, 665-662.

Table 1

Class.LD2 and overall Placement Team Decisions

		PLACEMENT TEAMS: LD	PLACEMENT TEAMS: NOT LD
CLASS.LD2	LD	64	10
	NOT LD	68	122

N = 264

Total Agreement = 64 + 122 = 186

Total Disagreement = 10 + 68 = 78

Phi Coefficient = .46

Table 2

Class.LD2 and District 1 Placement Team Decisions

PLACEMENT TEAMS

CLASS.LD2	LD	NOT LD
LD	32	6
NOT LD	31	81

n = 150

Total Agreement = 32 + 81 = 113

Total Disagreement = 6 + 31 = 37

Phi Coefficient = .27

Table 3

Class.LD2 and District 2 Placement Team Decisions

		PLACEMENT TEAMS: LD	PLACEMENT TEAMS: NOT LD
CLASS.LD2	LD	17	0
	NOT LD	15	22

N = 54

Total Agreement = 17 + 22 = 39

Total Disagreement = 0 + 15 = 15

Phi Coefficient = .56

Table 4

Class.LD2 and District 3 Placement Team Decisions

		PLACEMENT TEAMS: LD	PLACEMENT TEAMS: NOT LD
CLASS.LD2	LD	15	4
	NOT LD	22	19

N = 60

Total Agreement = 15 + 19 = 34

Total Disagreement = 4 + 22 = 26

Phi Coefficient = .24

Table 5

Class.LD2 and overall Expert Decisions in cases of Team/Class.LD2 Disagreement

CLASS.LD2 \ EXPERTS	LD	NOT LD
LD	22	8
NOT LD	42	162

N = 78 Cases X 3 Experts = 234 Judgments

Total Agreement = 22 + 162 = 184

Total Disagreement = 8 + 42 = 50

Phi Coefficient = .46

Chi Square = 36.36

Table 6

Class.LD2 and Unanimous Expert Decisions in cases of Team/Class.LD2 Disagreement

		EXPERTS: LD	EXPERTS: NOT LD
CLASS.LD2	LD	5	0
	NOT LD	6	47

n = 58

Total Agreement = 5 + 47 = 52

Total Disagreement = 0 + 6 = 6

Phi Coefficient = .63

Chi Square = 23.38

Table 7

Correlation matrix: Phi coefficients describing the relationships between the judgements of teams, Class.LD2, and experts in cases where there is team/Class.LD2 disagreement

	CL	E 1	E 2	E 3
*PT	−1	−.39	−.59	−.24
**CL	—	.39	.59	.24
***E 1		—	.50	.79
E 2			—	.45
E 3				—

*Placement Teams

**Class. LD2

***Expert

N = 78

Table 8

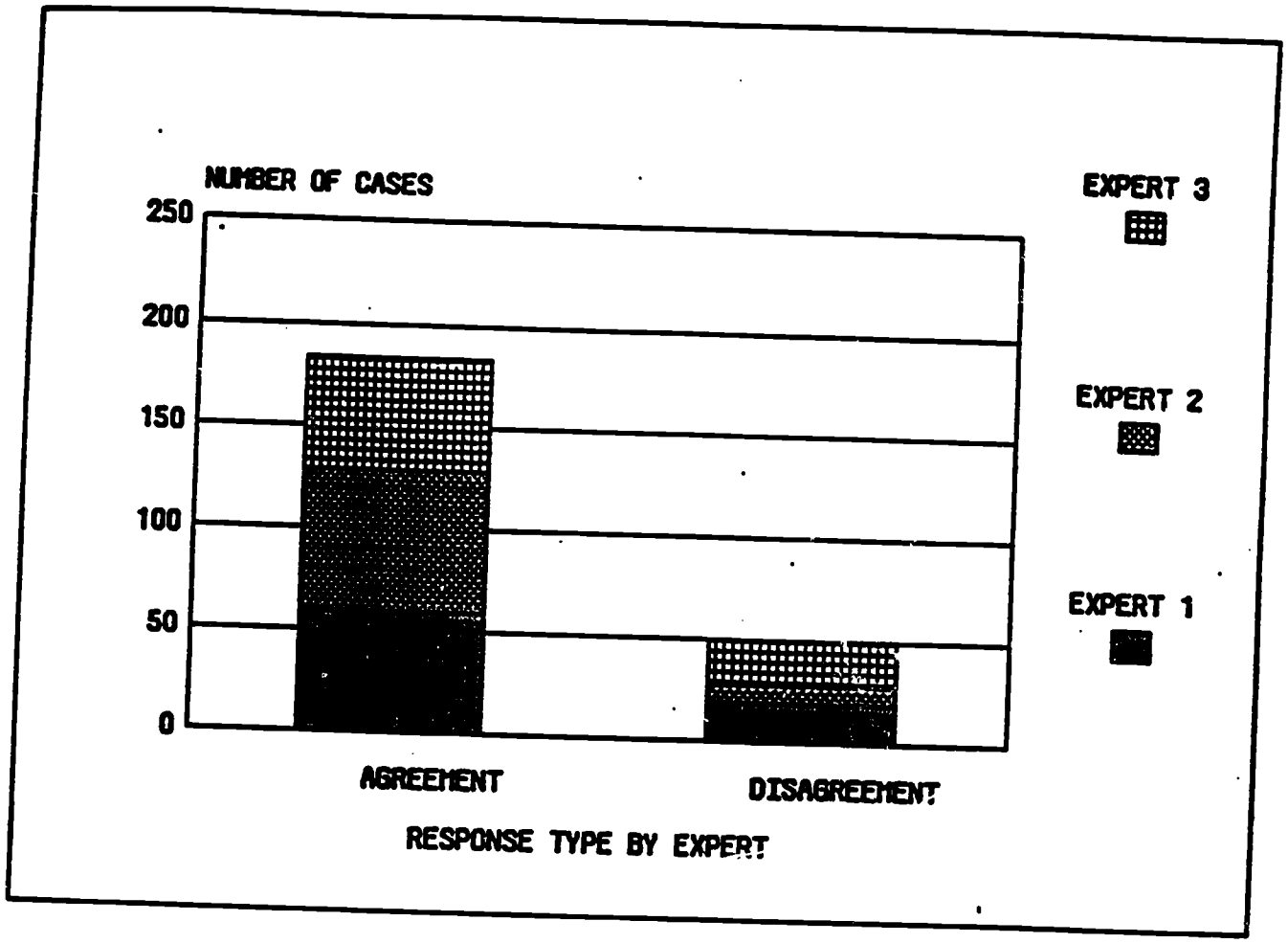
Intermediate Values of Critical Variables in Cases of Unanimous Expert/Class.LD2 Disagreement

Variable	IQ	Est. Discrepancy	Actual Discrepancy
Cutoff Point	85/84	40%	40%
Case # 1	100	49.3%	*39.6%
Case # 2	100	*39.5%	48.9%
Case # 3	*84	54.0%	55.8%
Case # 4	*83	50.0%	54.8%
Case # 5	104	40.0%	*38.5%
Case # 6	95	40.5%	*38.1%

* The critical score which failed to make Class.LD2's cutoff level.

Figure Caption

Figure 1. Number of agreements and disagreements of experts with Class.LD2. N = 78 X 3 = 234 judgments.


NUMBER OF CASES
250
200
150
100
50
0
AGREEMENT
DISAGREEMENT
EXPERT 3
EXPERT 2
EXPERT 1
RESPONSE TYPE BY EXPERT

Appendix I

LD.Trainer:
Modification of an Expert System
for Complex Conceptual Training

LD.Trainer: Modification of an Expert System for Complex Conceptual Training

Joseph M. Ferrara, Mary Anne Prater and Richard Baer
Artificial Intelligence Research and Development Unit
Utah State University

Running head: MODIFICATION OF AN EXPERT SYSTEM

Accepted for publication in Educational Technology (in press)

LD.Trainer: Modification of a Expert System for Complex Conceptual Training

Suppose your two-year-old son, Tommy, has suddenly become infatuated with the word, "blue." Every time you point to something and ask, "What color is this?" Tommy says, "blue." You're tired of him calling everything blue so you decide to teach him that some things are blue and other things are not blue. Intuitively, you point to Tommy's blue shirt and say, "Tommy's shirt is blue." Next, pointing to Daddy's shirt you say, "Daddy's shirt is not blue, it's green." You are beginning the process of teaching Tommy the basic concept, "blue."

Basic Concepts

Basic concepts, like blue, comprise a good portion of any language. They are ideas or discriminations that cannot be easily described using words. Thus, one must employ examples and nonexamples to teach them (Englemann & Carnine, 1982). To illustrate, consider that Tommy's parent, as described above, used his own shirt as an example of something that was not blue, a nonexample. By continually presenting Tommy with examples and nonexamples of blue Tommy will eventually learn to discriminate between blue and other colors. When he has this skill, it can be said that Tommy has learned the concept blue.

In recent years, educators and computer scientists have joined in efforts to develop computer programs that can assist in teaching basic concepts. Most of these programs generate

examples and nonexamples of the basic concept they are designed to teach.

Complex Concepts

Basic concepts may vary along only one dimension. For example, color -- something is or is not blue; temperature -- something is or is not cold; or shape -- something is or is not round. In contrast, complex concepts are a) multifaceted, they vary along two or more dimensions, and b) dynamic, the way they vary along each dimension is defined by a number of variables. Furthermore, several of the concept's dimensions may interact. Thus, a change in one dimension could impact our judgment regarding another dimension.

The concept, "learning disabled student," provides an example of a complex concept. Before students can be classified as learning disabled, they must meet a variety of criteria. In Utah these criteria define three discrete dimensions:

1) A discrepancy between students' expected academic performances and their actual academic performances must exist.
2) The students' learning problems must not be the result of some other handicapping condition (i.e. mental retardation, behavior disorder).
3) The students' learning problems must not be the results of cultural, economic, or environmental factors.

Thus, as is illustrated in Figure 1, the complex concept, "learning disabled student," is multifaceted.

Insert Figure 1 about here

The complex concept, "learning disabled student" is also dynamic. To illustrate, consider that in the State of Utah, the discrepancy between expected and actual academic performance is determined by administering an intelligence test and one or more achievement tests. A discrepancy formula is then used to calculate a score which describes the degree to which the students' academic performances falls below their expected performances. If students' discrepancy scores are greater than 40, they are eligible for a learning disabilities(LD) classification. This criterion defines one of the dimensions of the complex concept "learning disabled student." But suppose a student who comes from a home where only Spanish is spoken has a discrepancy score of 45. If tests standardized only on Anglo students were administered to this student, our confidence in the discrepancy score is altered. We are less than 100 percent confident that the discrepancy is truly 45 because scores on tests not standardized on Spanish-speaking students were used in the calculation. The actual degree to which this fact decreases our confidence is unknown and requires a best estimate or professional judgment. That is, a degree of uncertainty has been introduced. Additionally, the information that the student comes from a Spanish-speak home raises another issue. The learning problems of Spanish-speaking students may be due to cultural factors. The example above illustrates the dynamic nature of complex concepts. The fact that the student comes from a

Spanish-speaking home alters the confidence we can have in two of the three dimensions that define "learning disabled students." This is illustrated in Figure 2.

Insert Figure 2 about here

When the best professionals identify learning disabled students, they consider the student's characteristics in all three dimensions that define this complex concept. They understand that the confidence with which one can conclude a student is learning disabled varies with specific circumstances. The student's age, IQ, and cultural background, as well as the quality of information provided by tests and other sources, are a few of the specific circumstances which modify judgments regarding learning disabled students. Skilled professionals understand that the criteria for a learning disabilities classification is dynamic rather than absolute.

Since specific facets and dynamic characteristics of complex concepts are not always obvious, they are often difficult to teach and learn. For example, research suggests that there are serious problems associated with learning disabilities classification decision making (Hofmeister, 1983; Sabatino, 1983; Thurlow & Ysseldyke, 1979; Ysseldyke, 1983). It appears that many of those charged with the responsibility of qualifying students as learning disabled have failed to accurately do so.

Class LD.2: An Expert System

In response to the need for better systems for classifying

LD students Ferrara and Hofmeister (1984) developed an expert system, Class.LD2. Expert systems are programs which are designed to ask the computer user regarding a problem and provide the same advice one might receive from an expert consultant (Harmon & King, 1984; Weiss & Kulikowski, 1984). Class.LD2 is designed to provide the user with a second opinion regarding the probability that a student can be classified as LD. The class.LD2 knowledge base contains information and decision rules obtained from experts in the area of LD classification, federal law, state regulations, and current literature describing best practices in the area of student assessment (Ferrara,Parry, & Lubke, 1985).

The knowledge base of an effective expert system must be designed to explicitly define each dimension and dynamic characteristic of the complex concept or concepts assocoated with its knowledge area. CLASS.LD2's rules define the concept "learning disabled student" (Martindale, Ferrara, & Campbell, 1986).

LD.Trainer

LD.Trainer (Prater & Althouse, 1986), a computer-assisted instruction (CAI) program, was developed to teach preservice and inservice educators to make appropriate learning disabilities classification decisions. The program was designed to utilize the rules of CLASS.LD2 in a simple and cost-effective instructional format. Individuals who completed training with this system should be able to accurately discriminate between

students who should and should not be classified as learning disabled in the State of Utah.

LD.Trainer consists of a series of 13 lessons, each of which present a portion of the concept "learning disabled student." The lessons teach preservice and inservice teachers to determining that a) the student manifests a discrepancy between expected and actual academic achievement; b) the student's learning difficulties are not due to other handicapping conditions (i.e., vision, hearing, health problems); or c)the student's learning difficulties are not due to cultural, economic, or environmental factors.

In developing the lessons for LD.Trainer, a modified version of the concept instruction model suggested by Merrill and Tennyson (1977) was employed. Each lesson was divided into two parts, instruction and practice.

During instruction, trainees are given a definition which describes the important attributes in distinguishing an LD student and are then given matched examples and nonexamples of this concept. The trainees read a brief case study describing an examples of an LD student. Data from the case study are then fed into the computer and the trainees examine the system's conclusions regarding whether the student can qualify as LD. Following this, they read an explanation of how the system arrived at its conclusions. The explanation details how certain attributes (i.e., those listed in the definition), were used to arrive at this conclusion. The trainees then read a second case study, a nonexample, and enter a second set of data. This time the value of the attributes of interest are changed and the

trainees are provided an opportunity to view how changes in these attributes effect the system's conclusions. This process is repeated with different case studies but focusing on the same attribute.

During the practice portion of each lesson the trainees again read a brief case study. After doing so, they make a decision regarding whether the student is LD and write the decision and justification for their decision in the printed material provided. The trainees then enter the data from the case study into the computer and compare their written conclusions with the system's. The printed materials provide justification for the system's conclusions by focusing on the same attribute as was presented and taught during the instructional part of the lesson. This process is repeated with a second case study. Finally, the trainees are given an opportunity to make LD classification decisions on case study information and then, varying the attribute of interest and holding all the other student data constant, the trainees can use the computer to check the accuracy of their decisions.

Research

As part of a week-long inservice program, twenty-one practicing teachers and administrators participated in one of two training conditions. Eleven completed LD.Trainer materials and ten were given representative special education student files and ran consultations with the expert system, CLASS.LD2.

On a test which served as both the pre and the posttest, trainees were given 12 case studies on which they identified the student as qualifying or not qualifying as learning disabled. Preliminary results indicate that the trainees in both groups improved their performance following participation in the training activities. However, those trainees in the LD.Trainer group scored significantly ($p < .01$) higher than the CLASS.LD2 group.

Conclusions

Preliminary research comparing CLASS.LD2 and LD.Trainer suggests that both systems are effective in teaching the complex concept, "learning disabled student." Trainees who ran data from files on learning disabled students through the CLASS.LD2 expert system improved in their ability to accurately identify students who could and could not be classified learning disabled. However, trainees who systematically completed the LD.Trainer materials showed even greater improvement in their ability to accurately identify learning disabled students.

The success of the LD.Trainer program demonstrates the potential of modified expert systems as tools for teaching complex concepts. LD.Trainer systematically varies each attribute which effect the three dimensions of the complex concept, "learning disabled student" and allows trainees to observe how changes in the attributes change the conclusions drawn. The essence of the system is that relative to each attribute, trainees are presented with examples and nonexamples of the concept "learning disabled student" that are wholly

dependent upon changes in the value of that attribute. This is essentially the procedure used in teaching simple concept such as a color. However, LD.Trainer systematically applies the procedure to each attribute in each dimension of the complex concept and provides trainees with an opportunity to learn how the attributes and dimensions interact.

LD.Trainer is one of the first CAI programs that has attempted to teach a complex concept. It has demonstrated that effective instructional programs can be developed by combining expert system technology with what is known about effective concept instruction. In so doing it opens the door for the development of other programs that could efficiently and cost-effectively teach other complex concepts.

References

Engelmann, S., & Carnine, D. 1982. Theory of instruction: Principles and applications. New York: Irvington Publishers.

Ferrara, J. M., & Hofmeister, A. M. 1984. CLASS.LD2: An expert system for classifying learning disabilities. [Computer program]. Logan, Utah: Utah State University.

Ferrara, J. M., Parry, J. D., & Lubke, M. M. (1985). Class.LD2: An expert system for student classification. Technical Paper. Logan, UT: Utah State University. (ERIC Document Reproduction Service No. ED 263 734).

Harmon, P. & King, D. 1984. Expert Systems: Artificial intelligence in business. New York: John Wiley & Sons.

Hofmeister, A. M. 1983. Comments on Part B, Section 618: Evaluation and part E, research and education of the handicapped, Public Law 94-142. Testimony before the Senate Committee on Labor and Human Resources, Subcommittee on the Handicapped, Congressional Record. Washington, D.C.: Government Printing Office.

Martindale, E. S., Ferrara, J. M., & Campbell, B.W. 1986. Accuracy of Class.LD2: An Expert System for Classifying Learning Disabled Students. Computers in Human Behavior, In Press.

Prater, M., & Althouse B.,LD.TRAINER. [Computer program]. Logan, Utah: Utah State University.

Sabatino, D. A. 1983. The house that Jack built. Annual Review of Learning Disabilities, 1:23-24.

Thurlow, M., & Ysseldyke, J. W. 1979. Current assessment and decision making practices in model programs forlearning disabled students. Learning Disability Quarterly, 2:15-24.

Weiss, S. M., & Kulikowski, C. A. 1984. A practical guide to designing expert systems. Totowa, NJ: Rowman & Allanheld.

Ysseldyke, J. E. 1983. Current practices in making psychoeducational decisions about learning disabled students. Annual Review of Learning Disabilities, 31-38.

Figure Caption

Figure 1. Concept of learning disabilities.

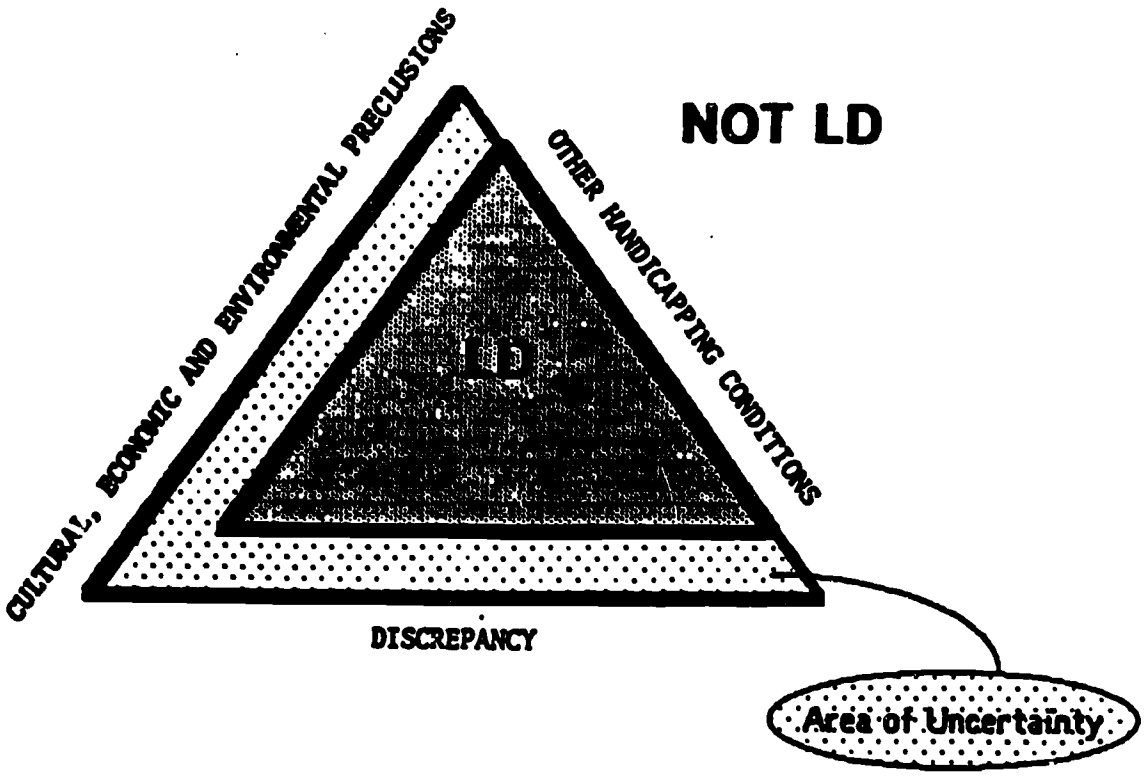
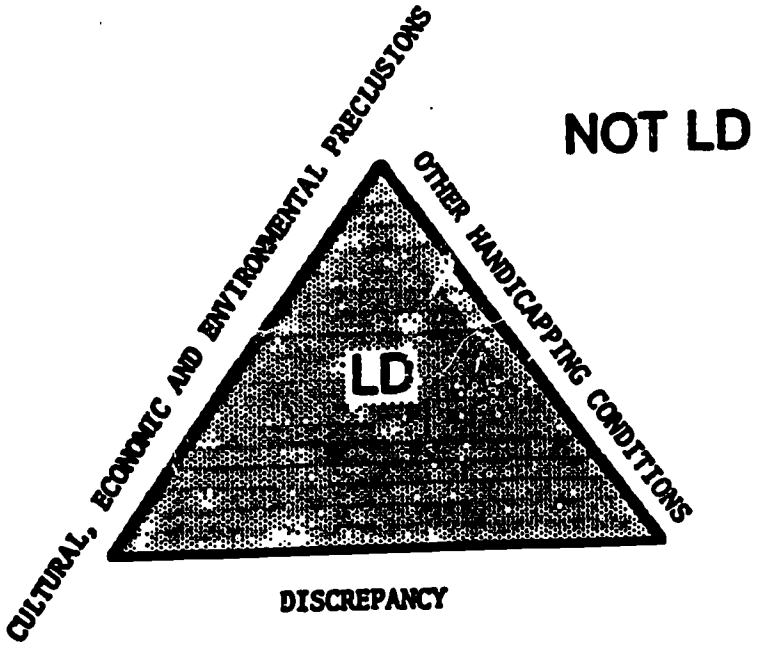

NOT LD
CULTURAL, ECONOMIC AND ENVIRONMENTAL PRECLUSIONS
OTHER HANDICAPPING CONDITIONS
LD
DISCREPANCY
Area of Uncertainty

Figure Caption

Figure 2. LD with two uncertain dimensions.


NOT LD
CULTURAL, ECONOMIC AND ENVIRONMENTAL PRECLUSIONS
OTHER HANDICAPPING CONDITIONS
LD
DISCREPANCY