Hillman, D.C.A., Willis, D.J., & Gunawardena, C.N. (1994). Learner-interface interaction in distance education: An extension of contemporary models and strategies for practitioners. *The American Journal of Distance Education, 8*(2), 30-42.

Lotus Institute White Paper (1997). *Distributed learning: Approaches, technologies and solutions.* Unpublished manuscript.

Moorehouse, J. (1997). *Teaching strategies for distance learning.* Unpublished manuscript. Denmark: Naestved.

Murphy, K.L. & Collins, M.P. (1997). Development of communication conventions in instructional electronic chats. *Journal of Distance Education, XII*(1/2), 177-200.

Rohfeld, R.W., & Heimstra, R. (1997, August). *Moderating discussions in the electronic classroom.* Paper presented at the 11th Annual Conference on Distance Teaching and Learning. WI: Madison.

Schlechter, T.M., (1990). The relative instructional efficiency of small group computer-based training. *Journal of Educational Computing Research. 6*(3), 329-341.

Shale, D. (1990). Toward a reconceptualization of distance education. In I.M. Moore (Ed.). *Contemporary issues in American distance education.* NY: Pergarnon.

Willis, B. (1992). From a distance: Making distance learning effective: Key roles and responsibilities. *Educational Technology. 35*(6), 35-37.

Note

1. The on-line course was developed by Mercedes Fisher, Heidi Schweicer, and Joan Whipp.

Jl. of Interactive Learning Research (1998) **9**(3/4), 235-248

Can Mathematics Students be Successful Knowledge Engineers?

DJORDJE KADIJEVICH
Mathematical Institute
Kneza Mihaila 35, 11001
Belgrade, p.p. 367, Yugoslavia
djkadij@mi.sanu.ac.yu

This study dealt with learning through instructional design realized through the development of expert system knowledge bases. Its major objective was to determine the empirical values of this kind of learning in respect of knowledge base development. The study used a sample of 18 ninth grade Gymnasium (high school) students whose mathematical and non-verbal intellectual abilities were mostly above average. The students solved problems on rectilinear (piecewise) uniform motion involving one and two objects. Two important findings emerged from this study. First, armed with a text editor and an expert system shell, the students successfully developed, within a few hours, small-scale knowledge bases. These bases comprised around ten (hierarchically organized) rules expressing how several related problems can be solved. Second, the students primarily encountered difficulties in respect of wrong rules utilization—rules with surplus variables, wrongly linked rules, viciously circled rules and wrongly ordered rules. As these difficulties did not significantly affect knowledge base development, the study evidenced that mathematics students of above-average mathematical and intellectual abilities can indeed be successful knowledge engineers.

Much research has been done on using computers in teaching/learning mathematics (e.g., Ruthven, 1989; Kaput, 1992; Keitel & Ruthven, 1993). However, except for Harel and Papert (1990) and Lippert (1990), researchers

have not focused on learning through educational software design (Kadijevich, 1993) although it may be a promising alternative to traditional instructional designs (Wilson & Cole, 1991). According to Lippert (1990), learning through knowledge engineering may be fruitful for teaching mathematics and physics. She speculated that even sixth grade students might be able to create small-scale knowledge bases in these subjects. It is true that several constructivist educators have suggested knowledge engineering and the use of expert system shells as cognitive tools (e.g., Jonassen, 1996), but there is a lack of rigorous research investigating the efficiency of this approach. In order to remedy this gap, the major objective of this study was to determine the empirical values of learning through knowledge engineering in respect of knowledge base development.

The knowledge base development was utilized within LISD (Learning through Intelligent Software Development)—an approach to teaching/learning mathematical problem solving. This approach enables students to develop and test knowledge bases with the aid of a text editor and an expert system shell. LISD uses the heuristic approach of Pólya (1990) and Schoenfeld (1985) within a framework for developing logic programs (Galle & Kovács, 1992). This framework is based upon rapid prototyping (Tripp & Bichelmeyer, 1990) through programming in logic and PROLOG (Sterling & Shapiro, 1986; Bratko, 1990). LISD design, underlying research framework and theoretical values are examined in Kadijevich (1998).

The LISD empirical values regarding knowledge base development were analyzed in respect of the following questions:

1. What kind of knowledge bases can mathematics students develop?
2. Which sort of difficulties may appear in knowledge base development?

The questions were examined for problems on rectilinear (piecewise) uniform motion involving one and two objects (see Figure 1). These problems, which traditionally form a part of mathematics curriculum (e.g., Prilepko, 1985), are invariably beyond the competence of most students. This is because their solutions call for skillful coordinations of required procedural and conceptual knowledge. While procedural knowledge ("knowing how") is concerned with utilizing specific procedures, conceptual knowledge ("knowing that") deals with objects, their properties and the relations among them (e.g., Hiebert & Carpenter, 1992). In AI community conceptual knowledge is usually denoted by declarative knowledge (e.g., Hofstadter, 1980).

Research on the formation and mastery of motion concepts has a long tradition (e.g., Arons, 1990). However, problems on rectilinear (piecewise) uniform motion involving one and two objects (e.g., problems on motion in respect of a reference point, meeting and overtaking problems, timetable problems and average speed problems) have not been extensively researched so far. As regards problems on meeting and overtaking, Harel and Hoz (1990) evidenced their high structural complexity, whereas Nathan and Young (1990) experimented with making the structure of these problems transparent to the solver.

1. A car and a truck started simultaneously from towns that are 150 km apart and met each other after 1 hour 30 minutes. Find the speed of each vehicle if the speed of the car is 20 km/h greater than that of the truck.
2. A student covered half the path on foot moving at the speed of 4 km/h and half on a bike moving at the speed of 12 km/h. Find his average speed for the whole trip.
3. A bus commonly goes between towns A and B moving at the speed of 70 km/h. Due to motor damage, the bus started from town A with a 15 minute delay, but it arrived at town B on time (according to the timetable) since it moved 10 km/h faster than usual. Find the distance between the towns.

Figure 1. Sample problems on motion

METHODOLOGY

Subjects

The study used a sample of 18 volunteers from seven ninth grade classes of a Gymnasium (the average age was 15.7 years; 6 were male) whose mathematical and non-verbal intellectual abilities[1] were mostly above average. The subjects had experience in writing simple BASIC programs, but none of them had any experience with LISD and expert systems. The subjects were taught by the author of this study.

Software

Three pieces of software were used: a simple DOS text editor, an expert system shell and a tracer. The applied shell supports an if-then formalism for representing knowledge. It allows the use of the following commands:

- solve (solve the problem);
- how (show how the problem has been solved);
- why (show the underlying rule);
- help (show the procedural and/or conceptual background of the underlying rule);

- tron (show the way in which the rules are fired);
- troff (turn the tracing facility off); and
- list (list the content of the working storage—a temporary knowledge base).

The shell was developed in logic and implemented in PROLOG by the author of this study on the basis of some shell samples (Sterling & Shapiro, 1986; Merritt, 1989; Bratko, 1990). As PROLOG is used, the shell enables its user to modify and extend it quite easily.

The process of knowledge base development was traced by using a simple PROLOG program. This program, which was also made by the author of this study, permits the use of a text editor within PROLOG system in such a way that every exit from the editor is followed by making a hard-disk copy of the current document (knowledge base in our case).

Treatment

The treatment, the content of which is summarized in Figure 2, lasted for two weeks. The treatment time comprised almost 30 hours of study—18 hours of schoolwork and about 10 hours of homework—during which the subjects engaged in genuine problem solving.

During the first week the subjects were introduced to the following three topics:

- problem solving according to Pólya and Schoenfeld;
- solving typical problems on motion; and
- expert systems and knowledge representation by if-then rules.

The subjects solved problems in pairs which they formed themselves. During the second week the pairs developed knowledge bases encapsulating how the following six types of problems can be solved:

1. basic problems on one object motion—the basic relations regarding speed, covered distance and elapsed time (starting time and finishing time were included)[2];
2. problems on one object motion in respect of a reference point;
3. meeting and overtaking problems;
4. timetable problems (e.g., problem 3 in Figure 1);
5. average speed problems; and
6. problems on interpreting distance vs. time graphs representing piecewise uniform motion.

First Week

The following topics were realized in the given order in five days (one topic per day within 2 hours).

1. *Problem Solving Methodology.* Pólya's phases of problem solving. An example. Basic heuristics (redefinition, specialization, generalization and analogy) and examples of their use. Schoenfeld's control of problem solving. An example.
2. *Problems on One Object Motion.* Problems with speed, elapsed time (appears directly or indirectly via starting and finishing times) and covered distance (appears directly or indirectly via initial and final distances from reference point). Sketching and interpreting quantitative and qualitative distance vs. time graphs representing piecewise uniform motion.
3. *Problems on Two Object Motion.* Problems on meeting and overtaking. Sketching and interpreting quantitative and qualitative distance vs. time graphs representing piecewise uniform motion. Reducing two object motion on one object motion (specialization).
4. *Advanced Problems on Motion.* Timetable problems. Average speed problems. Examining one object motion as two object motion (generalization). Motion on the river and in the air (analogy).
5. *Problem Solving by Expert Systems.* What is an expert system? Application areas. An expert system regarding problems on percentage (experimentation). Expert system architecture. Knowledge representation by if-then rules. Analysis of the consulted knowledge base.

Second Week

The following topics were realized in the given order in four days (one topic per day within 2 hours).

1. *Problems on One Object Motion* (first part). Work with text editor and PROLOG system. Developing and testing first knowledge base regarding basic relations among speed, elapsed time (appears directly or indirectly via starting and finishing times) and covered distance.
2. *Problems on One Object Motion* (second part). Developing and testing knowledge bases regarding one object motion in respect of reference point. Developing and testing knowledge bases concerning the interpretation of distance vs. time graphs representing piecewise uniform motion (optional).
3. *Problems on Two Object Motion.* Developing and testing knowledge bases regarding problems on meeting and overtaking (first part).
4. *Problems on One and Two Object Motion.* Developing and testing knowledge bases concerning problems on meeting and overtaking (second part). Developing and testing knowledge bases regarding problems on average speed. Developing and testing knowledge bases in respect of timetable problems (optional).

Figure 2. Experimental program

Knowledge Base Development

The subjects developed knowledge bases for some related problems by using the text editor. These bases were composed of facts, rules and explanations regarding the activation and/or conceptualization of these rules:

- The facts expressed the number of moving objects and their properties that could be asked by the shell, such as:

```
objects(1).
askable(covered_distance(X)).
```

When the shell was utilized, the first fact required the user to enter a concrete moving object like "train," whereas the second fact, if used, enabled asserting an additional item of information like

```
known(covered_distance(train))  /  unknown(covered_distance(train)).
```

depending on the user "yes"/ "no" answer to the shell.

- The rules expressed strategies whereby some classes of problems on motion can be solved. (Such rules do not solve problems quantitatively.) They had the following form:

```
if object(X) and
     speed(X) and
     covered_distance(X)
then answer( elapsed_time (X) = covered_distance(X) / speed(X) ).
```

While X stands for any object (car, bus, train, etc.), speed(X) denotes that the speed of a specified object X is known.

- The explanations contained messages regarding the activation and/or conceptualization of the developed rules, which respectively captured items of procedural and/or conceptual knowledge. In other words, the explanations clarified why a particular rule is to be used in respect of its procedural and/or conceptual background.

The validity of the developed knowledge bases was determined with the aid of the shell. While the initial validity was determined by the subjects themselves, the final validity was determined by the teacher with the collaboration of the subjects. Armed with the text editor, the subjects themselves

improved the developed knowledge bases, taking into account the results of the initial and final validations.

RESULTS

The subjects (nine pairs of students) developed about 30 knowledge bases, which were saved in nearly 300 versions by using the above mentioned PROLOG program. All these knowledge bases were carefully examined and for each pair of students we determined the values of the following variables:

- the number of developed rules;
- the number of correctly developed rules;
- the number of incorrectly developed rules;
- the number of problems whose solutions are correctly expressed; and
- the number of rules providing for that.

The means and standard deviations of these variables are reported in Table 1. Note that the correctness of a rule was determined in respect of both a relevant problem on motion and the applied knowledge base formalism.

Table 1
Means and Standard Deviations of the Measured Variables Regarding Knowledge Base Development

	VARIABLE	M (SD)
1.	number of developed rules	16.8 (5.6)
2.	number of correctly developed rules	13.4 (5.5)
3.	number of incorrectly developed rules	3.4 (2.2)
4.	number of problems whose solutions are correctly expressed	8.1 (3.3)
5.	number of rules providing for that	11.8 (5.6)

For each pair of students we also determined the number of problems whose solutions are correctly expressed for the examined problem types. The means and standard deviations of these variables are displayed in Table 2.

Table 2

Means and Standard Deviations of the Number of Problems Whose Solutions are Correctly Expressed for the Examined Problem Types

PROBLEM TYPE	1	2	3	4	5	6
M (SD)	2.0(1.1)	1.0(1.1)	2.4 (1.1)	.8 (.8)	1.5 (1.4)	.4 (1.3)

The subjects met a number of difficulties in the course of the knowledge base development. The main difficulties and their frequency (the number of pairs who encountered them) are summarized in Table 3.

Table 3

Main Difficulties in Knowledge Base Development and Their Frequency

DIFFICULTY	N
1. wrong rules utilization	7
2. distance relations regarding one object motion in respect of reference point	4
3. meeting and overtaking time when objects' starting times are different	4
4. taking time interval as time point	3

DISCUSSION

Two important findings emerged from this study: First, armed with a text editor and an expert system shell, the subjects successfully developed, within a few hours, small-scale knowledge bases. Second, the subjects primarily encountered difficulties in respect of wrong rules utilization, which were of the following types: rules with surplus variables, wrongly linked rules, viciously circled rules and wrongly ordered rules.

The subjects had no prior experience with the externalization, formalization and representation of their own knowledge. Furthermore, they were provided with very little help from the teacher. Despite that, the development of the knowledge bases was successfully realized, especially if we have in mind the ratio of the number of correctly developed rules to the number of incorrectly developed rules. It is true that the elicitation of expert knowledge is a highly complex enterprise (Parsaye & Chignell, 1988), but the study showed that the LISD students could successfully develop small-scale knowledge bases[3] These bases comprised around ten (hierarchically organized) rules expressing how several related problems can be solved. The study

evidenced that the LISD development of such a knowledge base can be realized within a few hours. It therefore supports the above reported Lippert's (1990) speculation yet regarding ninth grade students of above average intelligence and mathematical ability.

The knowledge base development dealt with six types of problems. In order to solve them, the subjects created a number of interesting rules such as:

```
8 rule
if object(X) and
 usual_speed(X) and
 speed_for_delay(X) and
 delay(X)
then answer( traveling_time(X) = speed_for_delay(X) * delay(X) /
 (speed_for_delay(X) - usual_speed(X)) ).

11 rule
if object(X) and
 left_first(X) and
 straight_second(X) and
 right_third(X) and
  greater_slope_first(X)
then answer( $It moves quickly backward, stops and then
 moves slowly forward.$ ).
```

Note that whilst rule 8 is concerned with a timetable problem, rule 11 deals with the interpretation of a qualitative distance vs. time graph representing one object piecewise uniform motion. Although the subjects succeeded in solving problems of type 1, 3 and 5, two difficulties were observed. First, three pairs took a time interval as a time point, like in the following example

```
meeting_time(X, Y) is distance(X, Y) / (speed(X) + speed(Y)).
```

(In this and other examples "is" is not a PROLOG operator. It stands for "is equal to" or "is calculated via".) Second, only two pairs developed a correct knowledge base regarding meeting and overtaking problems when objects have different starting times. This resulted from the fact that other knowledge engineers (four pairs) did not take into account that different starting times do affect the initial distance between the objects.

The subjects failed to solve problems of type 2, 4 and 6. This is understandable if we have in mind the duration of the treatment as well as the complexity of the problems of type 4 and 6. It is surprising, however, that the subject had difficulties with the externalization of distance relations concerning one object motion in respect of a reference point. Since we observed the same difficulty in our pilot study, further research may examine it in fuller detail.

In the last two paragraphs we have underlined some difficulties regarding the confusion of time conceptions and the confusion of distance conceptions. These confusions, which have already been realized in physics education, may be summarized as follows.

> It is impossible to deal clearly and correctly with instantaneous quantities without discriminating between instants (or "clock readings") and time intervals. It is impossible to deal with back-and-forth motion without discriminating between positions, changes in position, and distances travelled by the body (three different concepts to which the term "distance" is frequently indiscriminately applied.) These are indeed sophisticated ideas; that is why it took the human mind so long to penetrate them. It is unrealistic to expect students to make the penetration in the short time and through the shortcuts that are so frequently imposed. (Arons, 1990; p. 21)

Most difficulties were observed in respect of wrong rules utilization. We summarize them under the following labels: (a) rules with surplus variables, (b) wrongly linked rules, (c) viciously circled rules, and (d) wrongly ordered rules. These difficulties, however, did not significantly affect knowledge base development.

Four pairs developed *rules with surplus variables*, such as:

```
1 rule
if object(X) and
 starting_time(X) is ST and
 finishing_time(X) is FT
then answer( elapsed_time(X) = FT - ST ).
```

Since rules are generally concerned not with solving particular problems but with describing how some problems can be solved, this use of variables is not needed. The students quickly realized this fact and did not repeat the mistake. But, two pairs gave up using variables and developed knowledge bases comprising isolated rules.

As the subjects were not familiar with PROLOG unification, they initially linked developed rules in the following way:

```
1 rule
if object(X) and
 speed(X) and
 elapsed_time(X) is T
then answer( covered_distance(X) = speed(X) * T ).

4 rule
if object(X) and
 starting_time(X) and
 finishing_time(X)
then T is finishing_time(X) - starting_time(X).
```


This mistake—termed *wrongly linked rules*—was made by five pairs. Having noticed the mistake, we explained to the students how PROLOG unifies terms and how rules are to be linked. After our explanation, four pairs continued to represent their knowledge by hierarchically organized rules. Two pairs developed rules yielding identities of the type "X is X", such as:

```
6 rule
if object(X) and
 starting_time(X) and
 finishing_time(X)
then elapsed_time(X) is finishing_time(X) - starting_time(X).

9 rule
if object(X) and
 starting_time(X) and
 elapsed_time(X) is T
then answer( finishing_time(X) is starting_time(X) + T ).
```

These rules—called *viciously circled rules*—can be circumvented by: (a) adding a suitable condition to the rule and/or by changing the order of the rules, or (b) developing another set of rules. In small-scale knowledge bases viciously circled rules can easily be located and bypassed. Their appearance should therefore probably have only a minor negative influence on the outcomes of the knowledge base development. It may be that the appearance of these rules will slow down the process of knowledge base development, but is quite certain that bypassing them will help students to reflect on and make use of their knowledge in a better way.

Three pairs of students developed *wrongly ordered rules*. These students were told how PROLOG fires developed rules and how rules should therefore be ordered. We applied the following rule of thumb:

- in a set of related rules, more specific rules precede others; and
- in a whole knowledge base, main rules precede subordinate rules.

Note that our shell displayed the way in which the rules were fired if its tracing facility was set on.

In general, learning through knowledge engineering gives students an opportunity to develop and test their own epistemologies. As knowledge bases are developed through the process of knowledge (re)construction shaped by various classroom interactions, this type of learning does support the socio-constructivist nature of knowledge and learning (e.g, Ernest,

1991; Grabinger, 1996). A number of recent analyses underline that promoting this nature should be an important goal of research in computer science (e.g, Leith, 1990; Self, 1990). This issue is particularly relevant to educational software development, which should involve students and their teachers yielding products that are not based upon foreseen but upon real learning experiences.

To summarize: the findings reported in this study undoubtedly suggest the relevance of LISD to teaching/learning mathematics. Further research may primarily be directed toward: (a) developing multimedia knowledge bases[4] regarding various mathematical topics; and (b) examining the outcomes of this kind of learning relating to computational and mathematical issues.

References

Arons, A.B. (1990). *A guide to introductory physics teaching*. New York: John Wiley & Sons.

Bratko, I. (1990). *Prolog Programming for Artificial Intelligence* (second edition). Wokingham: Addison-Wesley.

Ernest, P. (1991). *The Philosophy of Mathematics Education*. Basingstoke: Falmer Press.

Galle, P., & Kovács, L.B. (1992). The logic of worms: A study in architectural knowledge representation. *Environment and Planning B: Planning and Design, 19*, 5-31.

Grabinger, R.S. (1996). Rich environments for active learning. In D.H. Jonassen (Ed.), *Handbook of research for educational communications and technology* (pp. 665-92). New York: Macmillan.

Harel, G., & Hoz, R. (1990). The Structure of Speed Problems and its Relation to Problem Complexity and Isomorphism. *Journal of Structural Learning, 10*(3), 177-96.

Harel, I., & Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments, 1*, 1-32.

Hiebert, J., & Carpenter, T.P. (1992). Learning and Teaching with Understanding. In D.A. Grouws (Ed.), Handbook of Research on Mathematics Teaching and Learning (pp. 65-97). New York: Macmillan.

Hofstadter, D.R. (1980). *Gödel, Echer, Bach: An Eternal Golden Braid*. London: Penguin.

Jonassen, D.H. (1996). *Computers in the classroom: Mindtools for critical thinking*. Englewood Cliffs, NJ: Prentice-Hall.

Kadijevich, D. (1993). Learning, Problem Solving and Mathematics Education, report 93/3. University of Copenhagen: Department of Computer Science.

Kadijevich, D. (1998). A Didactic Approach to Learning Mathematics through Knowledge Engineering (submitted for publication). Internet [http://www.mi.sanu.ac.yu/~djkadij/no9.htm].

Kaput, J. (1992). Technology and Mathematics Education. In D.A. Grouws (Ed.), *Handbook of Research on Mathematics Teaching and Learning* (pp. 515-56). New York: Macmillan.

Keitel, C., & Ruthven, K. (Eds.) (1993). *Learning from Computers: Mathematics Education and Technology*. Berlin: Springer.

Kroening, M. (1995). Using Rules to Code the Hart and Soul of VB Application. *VB Teach Journal.*

Leith, P. (1990). *Formalism in AI and Computer Science*. New York: Ellis Horwood.

Lippert, R. (1990). Teaching Problem Solving in Mathematics and Science with Expert Systems. *Journal of Artificial Intelligence in Education, 1*(3), 27-40.

Merritt, D. (1989). *Building Expert Systems in Prolog*. New York: Springer-Verlag.

Nathan, M.,J., & Young, E. (1990). Thinking situationally: Results With An Unintelligent Tutor For Word Algebra Problems. In A. McDougall & C. Dowling (Eds.), *Computers in Education* (pp. 425-30). Amsterdam: Elsevier Science Publishers.

Parsaye, K., & Chignell, M. (1988). *Expert Systems for Experts*. New York: John Willey.

Pólya, G. (1990). *How To Solve It* (2nd edition). London: Penguin.

Prilepko, A.I. (Ed.) (1985). *Problem Book in High-School Mathematics*. Moscow: Mir Publishers.

Ruthven, K. (Ed.) (1989). Informational Technology and Mathematics Education (special issue). *Educational Studies in Mathematics, 20*, 3.

Schoenfeld, A.H. (1985). *Mathematical Problem Solving*. Orlando: Academic Press.

Self, J. (1990). *Theoretical Foundations for Intelligent Tutoring Systems. Journal of Artificial Intelligence in Education, 1*(4), 3-14.

Sterling, L., & Shapiro, E. (1986). *The Art of Prolog*. Cambridge: MIT Press.

Tall, D. (1994). Computer environments for the learning of mathematics. In R. Biehler, R.W. Scholz, R. Straesser, & B. Winkelmann (Eds.), *Didactics of mathematics as a scientific discipline* (pp. 189-99). Dordrecht: Kluwer.

Tripp, S.D., & Bichelmeyer B. (1990). Rapid Prototyping: An Alternative Instructional Design Strategy. *Educational Technology Research and Development, 38*, 31-44.

Wilson, B., & Cole, P. (1991). A Review of Cognitive Teaching Models. *Educational Technology Research and Development, 39*(4), 47-64.

Acknowledgement

The study used Cogent Prolog which had been generously provided by Amzi! Inc. (http://www.amzi.com).

Notes

1. The mean and standard deviation of the subjects' nonverbal IQ were 115.5 and 7.8, respectively. This ability was assessed by Daniels' Figure Reasoning Test. This test is structurally similar to Ravens Progressive Matrices and highly correlates with it. The assessment was realized by a group of psychologists.
2. We did not take into account problems (rules) dealing only with speed, covered distance and elapsed time.
3. We tried to make the LISD learning requirements as simple as possible. Thus, neither dealt we explicitly with programming in logic and PROLOG, nor did we mention the LISD phases (Kadijevich, 1998) and utilize LISD according to them.
4. Having in mind recently proposed requirements for computer environments in mathematics education (Tall, 1994), multimedia knowledge bases may be implemented in Amzi PROLOG since it can easily be embedded in Microsoft Visual Basic (Kroening, 1995).

Jl. of Interactive Learning Research (1998) **9**(3/4), 249-269

Network Similarity (NETSIM) as a Method of Assessing Structural Knowledge for Large Groups

K. DAVID PINKERTON
University of Denver
College of Education
Denver, CO, USA
and
Cherry Creek Public Schools
Smoky Hill High School
Aurora, CO, USA
dpinkert@shhs1.smoky.org

Structural assessments such as concept maps and semantic networks can provide unique learning and evaluation opportunities. Large-scale implementation however is inhibited by psychometric and pragmatic issues. This study reports on the features of a computer program (KNOT) which utilizes a network similarity index (NETSIM) to compare novice and expert concept maps. Treatment in three high school physics classes consisted of three levels of "language-rich" teaching: low, medium, and high. Results suggest that alpha reliability is at least .85 and stability reliability is .73. The validity coefficient is .85. Content, construct, concurrent, and predictive validities are high. Time to collect and analyze data is minimal and enhances the possibility of using NETSIM in large-scale structural assessments in conjunction with objective and subjective evaluative measures.

Information is not understanding. Science students who understand develop a rich set of relations among concepts in a domain (Ruiz-Primo & Shavelson, 1996). Interrelated concepts form cognitive structures (Diekhoff, 1983).