ED 388 280	IR 017 429
AUTHOR	Oberem, Graham E.
TITLE	Transfer of a Natural Language System for Problem-Solving in Physics to Other Domains.
PUB DATE	94
NOTE	09p.; In: Educational Multimedia and Hypermedia, 1994. Proceedings of ED-MEDIA 94--World Conference on Educational Multimedia and Hypermedia (Vancouver, British Columbia, Canada, June 25-30, 1994); see IR 017 359.
PUB TYPE	Reports - Descriptive (141) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; Computer Simulation; Higher Education; *Intelligent Tutoring Systems; Law Related Education; Mathematics; *Natural Language Processing; Physics; *Problem Solving; *Programmed Tutoring; Secondary Education

ABSTRACT

The limited language capability of CAI systems has made it difficult to personalize problem-solving instruction. The intelligent tutoring system, ALBERT, is a problem-solving monitor and coach that has been used with high school and college level physics students for several years; it uses a natural language system to understand kinematics problems and can teach students to solve physics problems by engaging the student in a plain English dialogue. The four programs described in this paper demonstrate the transportability of the ALBERT natural language systems across diverse subject areas, programming languages and hardware platforms. FREEBODY allows students to draw free-body diagrams on the computer with a mouse and also assess whether or not the resulting diagram is reasonable in terms of the situation described in the problem statement. ILONA is a program that helps students translate logical statements into mathematical form; the natural language system used in ILONA was also derived from ALBERT. CICERO is a program that was developed as a part of a series of lessons in the field of Roman law; it allows students to participate in a simulated lawyer-client interview. ALBERT's language system provides an opportunity for dialogue between the student-lawyer and the computer-client. ECLIPS is a program designed to understand and solve chemistry problems of molarity typed in plain English; it uses both the natural language system and the generic problem-solver from ALBERT. Moving ALBERT's natural language system to a new subject area is a relatively simple programming task that takes the form of minor adjustments to the vocabulary elements and the syntactic pattern database. (Contains 16 references.) (AEF)

Transfer of a Natural Language System for Problem-Solving in Physics to Other Domains

GRAHAM E. OBEREM
Department of Physics FM-15, University of Washington, Seattle, WA 98195, U.S.A.
E-Mail: oberem@u.washington.edu

Abstract: A natural language system originally designed to understand kinematics problems has been transferred to different topics in Physics and to subject areas as diverse as Logic and Roman Law. These intelligent tutoring systems are able to understand plain English text and can engage students in a natural language dialogue. Although its capabilities are limited, the portability of this natural language system and the advantages it offers over traditional approaches to answer judging in computer assisted instruction make it an attractive alternative for many CAI applications. This paper describes the language processing system and several programs that use it.

INTRODUCTION

For many students, problem-solving in physics consists of a matching formula with information given in a problem statement (Chi, Feltovich, & Glaser, 1981). To some extent this can be done without a proper grasp of the physical concepts involved, but this can be very frustrating for students. Ideally, problem-solving should reinforce the concepts that have been learned in class. Coaching a student in problem-solving involves dialogue and such dialogues are often Socratic in style (Mestre, Gerace, Hardiman, & Lochhead, 1987).

It has been suggested that the computer could be used to personalize problem-solving instruction, even in large classes, but the limited language capability of CAI systems has made it difficult to achieve this goal (Trowbridge & Chioccariello, 1985). Patrick Suppes (1990) stresses the need for adequate language processing in CAI in general, and some intelligent tutoring systems have attempted to address this issue (Brown, Burton, & de Kleer, 1982). However, the majority of CAI programs, even intelligent tutoring systems, still avoid the problem by constraining the user to menu selections, keyword answers, or multiple-choice questions.

The intelligent tutoring system, ALBERT, is a problem-solving monitor and coach that has been used with high school and college level physics students for several years (Oberem, 1987). ALBERT uses a natural language system to understand kinematics problems stated in plain English. The program not only understands physics problems, but also knows how to solve them and can teach a student how to solve them by engaging the student in a plain English dialogue.

ALBERT was originally implemented on a mainframe computer to obtain the speed required for the language processing and problem-solving. However, advances in microcomputer hardware and software over the last few years have made it possible to move ALBERT to the PC and Macintosh platforms. Due to the modular construction of ALBERT, it has been possible to use the natural language system and the expert problem solver in several new programs for teaching physics and other subjects. In this paper we discuss four programs that incorporate the natural language system developed for ALBERT.

THE NATURAL LANGUAGE SYSTEM

The natural language system developed for ALBERT is intended to augment existing CAI tools. It is not a generic natural language processor. Its robustness in the programs described here arises out of the fact that it is called upon to operate only in a very limited domain of expertise for any given application. In the case of ALBERT, the language system is only used to understand textbook kinematics problems and student-computer dialogues about kinematics. ALBERT's language system uses a syntactic pattern matching technique to parse the input sentences and includes semantic routines to complete the process of understanding. It also has the ability to deal with syntactically incomplete sentence fragments, a feature required for dealing with the vagaries of the more colloquial student-computer dialogues. The departure from the keyword matching approach of traditional CAI enables ALBERT to engage students in a fairly natural problem-solving discussion.

Figure 1 illustrates the major components of ALBERT's language processing system. The vocabulary of one dimensional kinematics is very limited and largely free of ambiguity. Words in the lexicon are therefore grouped into approximately twenty five categories based on their semantic function in this domain of physics. Traditional linguistic groupings such as noun and verb are not used. In the first pass, the words in the input text are parsed into a numerical string where each number in the string corresponds to the lexical category of the associated word. Regular patterns occur in these numerical strings and these patterns correspond to syntactic entities containing important information.

A knowledge-base of approximately one hundred commonly occurring syntactic patterns is maintained in ALBERT. This list of patterns was obtained by analyzing the syntax of textbook physics problems and student-computer dialogues. In the second phase of the parse, the numerical string derived from lexical analysis is searched for these syntactic patterns. When a particular pattern is identified in the input string, a parameter-driven semantic routine is called to extract the information from that part of the sentence. Very few semantic routines are needed, only five in the case of ALBERT. When semantic processing is complete, all the information in the typed input will have been used to instantiate variables in computational models of the kinematics expert , the student, and the tutorial process.

In addition to the natural language processor, ALBERT includes a generic expert problem-solver, a tutorial model that is based on the results of research on the teaching of problem-solving (Reif, 1981), and a tutorial management system that is used to guide ALBERT's interaction with the student. Although all of these components are modular, the language processor has been of the greatest value in the development of new intelligent tutoring systems requiring natural language capability.

MOVING THE LANGUAGE SYSTEM TO A NEW DOMAIN

Moving ALBERT's natural language system to a new subject area is a relatively simple programming task, but it does require a significant amount of effort in analyzing the language of the new domain. Three components of the language system need to be modified for a new application: a) the lexicon, b) the syntactic pattern database, and c) the semantic routines. The first two take the form of alphanumeric data. Modifying them is a straightforward editing task. The third component comprises a small collection of sub-routines, some of which are generic and can be transferred to new programs with only minor modification. For example, the routines that extract and store the numerical values of physical quantities can be used with little or no change. Modifying the semantic routines or producing additional ones is a programming task the complexity of which is determined by the nature of the target domain.

Developing a new lexicon involves analyzing examples of the language that will be typical in the new area of interest. Each word must be classified according to its semantic function and added to the appropriate group in the vocabulary. A given word may be grouped differently for different applications and, in new programs, it has been found that new groups need to be added to the lexicon while some of the old ones can be discarded. The vocabulary can be refined and expanded once the program is in use.

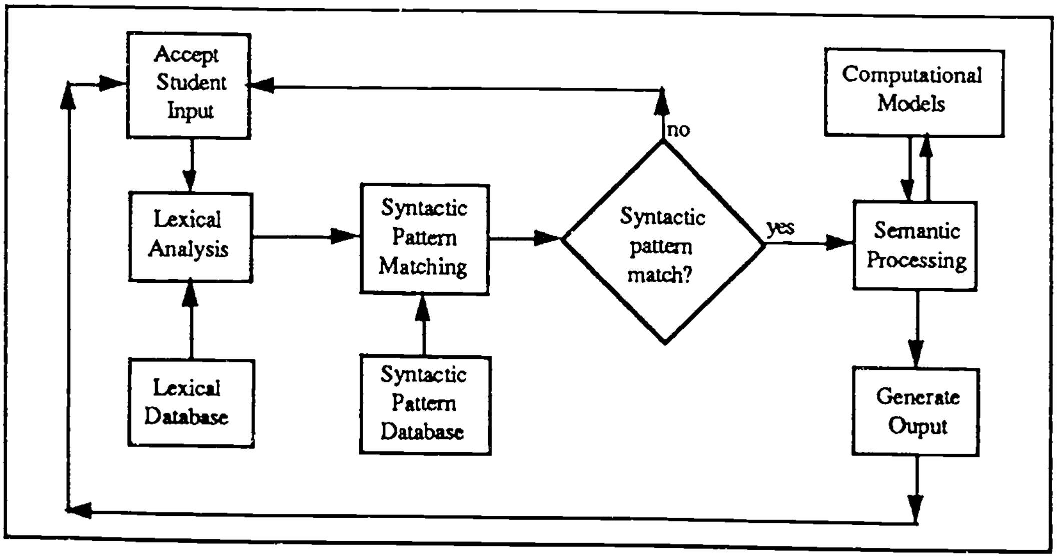


Figure 1. The structure of ALBERT's natural language system

The commonly occurring syntactic structures are also identified by analyzing examples of text from the new domain. These must be encoded as numerical patterns to match the word classifications in the lexicon. A small number of data elements are associated with each syntactic pattern to act as parameters for the appropriate semantic routines. Inevitably, additional syntactic patterns will be identified during the initial testing phase and can be included at that time.

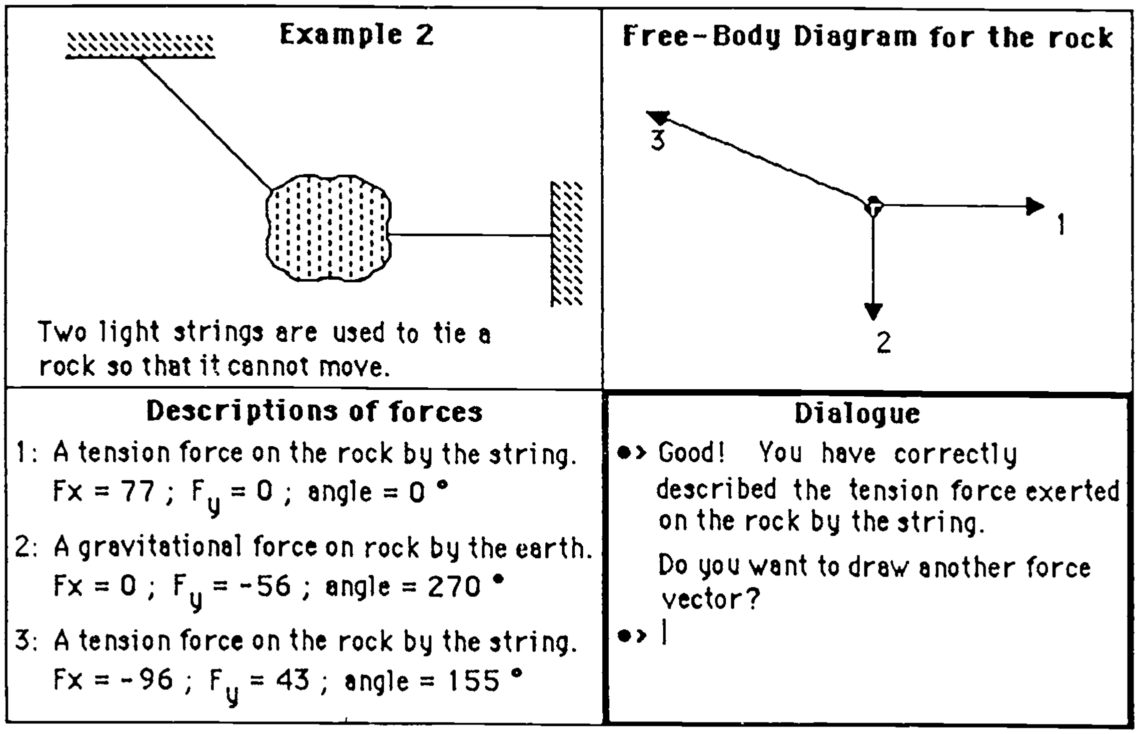
The examples that follow show how the process just described has been used to produce intelligent tutoring systems that incorporate a natural language interface.

FREEBODY

Research has shown that students have a great deal of trouble learning some of the most basic concepts in introductory physics (McDermott, 1991). One way of helping students to understand these concepts is to engage them in a qualitative dialogue about carefully chosen physical situations. This often requires students to draw and discuss diagrams. In addition to the need for language processing, one must address the problem of understanding student diagrams drawn on the computer. Although research is providing the techniques needed for interpreting text and graphics in complex physical situations (Novak & Bulko, 1993), the problem is often simplified by the unambiguous nature of the context, for example, when students are required to draw vectors.

Many students have trouble identifying the forces exerted on a given object (McDermott, 1984). A "free-body diagram" is a formal representation in which the forces on an object of interest are shown as vectors. The student is expected to identify the type of each force and the agent producing the force. Some students find it difficult to identify all of the forces. They propose non-existent forces and often cannot identify the agent producing a given force. FREEBODY allows students to draw free-body diagrams on the computer with a mouse. The program discusses each of the forces that the student draws and helps the student to correct any misconceptions that he or she might have. FREEBODY also assesses whether or not the resulting diagram is reasonable in terms of the situation described in the problem statement. For example, in cases where the net force on an object is zero, the sum of the force vectors drawn by the student should be zero.

Figure 2 shows a typical screen display from FREEBODY. The problem is displayed in the upper left quadrant of the screen. A summary of the forces is maintained in the lower left quadrant. Students can refer to that area of the screen for information about forces already drawn and described. Students draw the free-body diagram in the upper right part of the screen and the dialogue takes place immediately below that. The flexibility of the intelligent tutor permits the student to draw the force vectors in any order and to adjust the length and orientation of a vector during the discussion. The program interprets both plain the English dialogue



Example 2

Two light strings are used to tie a rock so that it cannot move.

Free-Body Diagram for the rock

Descriptions of forces

1: A tension force on the rock by the string.
Fx = 77 ; F_y = 0 ; angle = 0 °

2: A gravitational force on rock by the earth.
Fx = 0 ; F_y = -56 ; angle = 270 °

3: A tension force on the rock by the string.
Fx = -96 ; F_y = 43 ; angle = 155 °

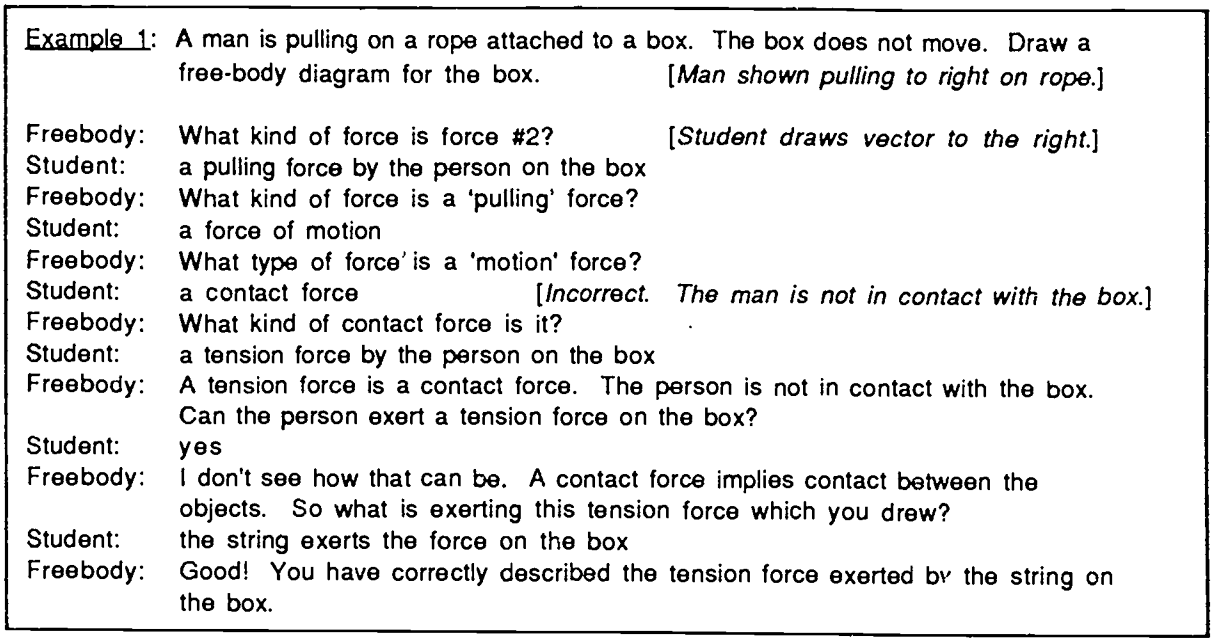
Dialogue

•> Good! You have correctly described the tension force exerted on the rock by the string.

Do you want to draw another force vector?

•> |

Figure 2. The screen layout for FREEBODY.



Example 1: A man is pulling on a rope attached to a box. The box does not move. Draw a free-body diagram for the box. [*Man shown pulling to right on rope.*]

Freebody: What kind of force is force #2? [*Student draws vector to the right.*]
Student: a pulling force by the person on the box
Freebody: What kind of force is a 'pulling' force?
Student: a force of motion
Freebody: What type of force'is a 'motion' force?
Student: a contact force [*Incorrect. The man is not in contact with the box.*]
Freebody: What kind of contact force is it?
Student: a tension force by the person on the box
Freebody: A tension force is a contact force. The person is not in contact with the box. Can the person exert a tension force on the box?
Student: yes
Freebody: I don't see how that can be. A contact force implies contact between the objects. So what is exerting this tension force which you drew?
Student: the string exerts the force on the box
Freebody: Good! You have correctly described the tension force exerted bv the string on the box.

Figure 3. An extract from a student dialogue with FREEBODY.

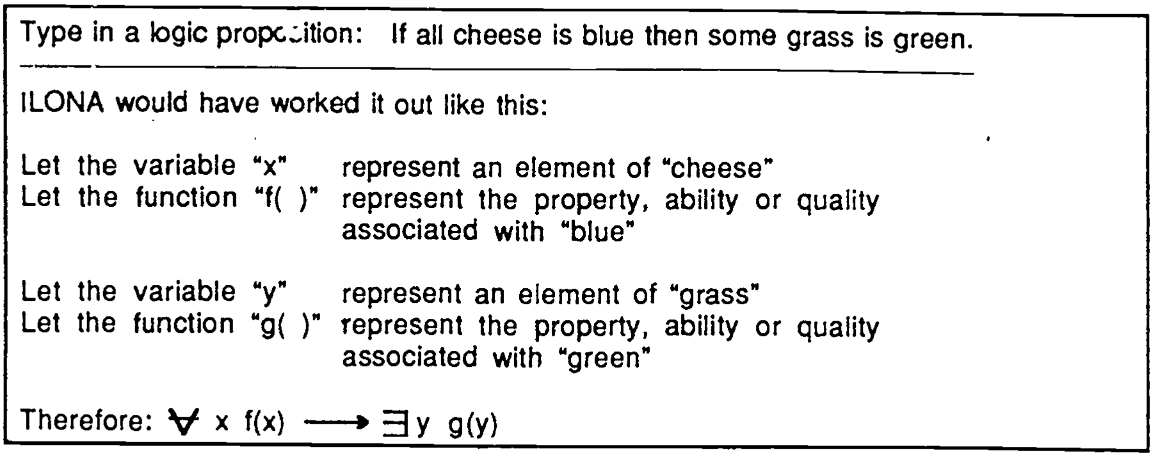
and the graphics input. FREEBODY has been used extensively in the introductory calculus-based physics courses at the University of Washington for the last two years. All the dialogues are recorded in full and this allows us to use the program as a research tool to probe student understanding of force (McDermott, 1990). Figure 3 is an extract from a discussion between FREEBODY and a physics student that shows this student's difficulty with contact forces. We are currently investigating the effectiveness of this program in dealing with such misconceptions (Oberem, Shaffer, & McDermott, 1993).

In developing FREEBODY it was found that fewer words were required in the lexicon than was the case for ALBERT. We also found that the syntax used to describe the forces could be matched with fewer syntactic patterns than were needed for ALBERT. The semantic routines were restructured to deal with the more qualitative data, since there is no numerical problem-solving in FREEBODY. The ease with which ALBERT's language system could be adapted made the development of this program relatively straightforward.

ILONA

ILONA is a program that helps students translate logical statements into mathematical form. The student types in a logical proposition and then "writes" it in mathematical form by clicking successively on the appropriate symbols that appear on the screen. This program is different from ALBERT and FREEBODY in that there is no dialogue as such. The natural language system is used to understand logical sentences such as: *If all cheese is blue then some grass is green.* ILONA can translate such input sentences into mathematical form and explain its reasoning by stepping through the operations in sequence. Figure 4 shows how ILONA would solve and explain the example above. ILONA is also able to check the student's answer and give specific feedback about errors which the student might have made. However, the degree of flexibility given to the student in the choice of variable names and function names makes a literal comparison of ILONA's answer with that of the student difficult. To overcome this problem, the student's answer is parsed into the canonical internal representation used for ILONA's answer. Answer judging is then relatively straightforward, as is the ability to provide specific feedback on errors. Figure 5 shows how ILONA can assist the student who has made an error.

The natural language system used in ILONA was also derived from ALBERT. In this case, the sentence structures were found to be very limited, thus requiring very few syntactic patterns. However, the vocabulary is extensive due to the unlimited range of topics. In fact it is impossible to include every conceivable noun and adjective in the lexicon of a CAI program like ILONA. Accordingly, ILONA's version of the language system was modified to cope with unrecognized words. Consider the sentence: *If it is the case that all big whatzits are flubsy, then some small thatzits are wubsy.* Clearly words like *whatzits* and *thatzits* will not be found in any dictionary. However, we can infer from the sentence structure that these two unknown words are probably



Type in a logic propc.ition: If all cheese is blue then some grass is green.

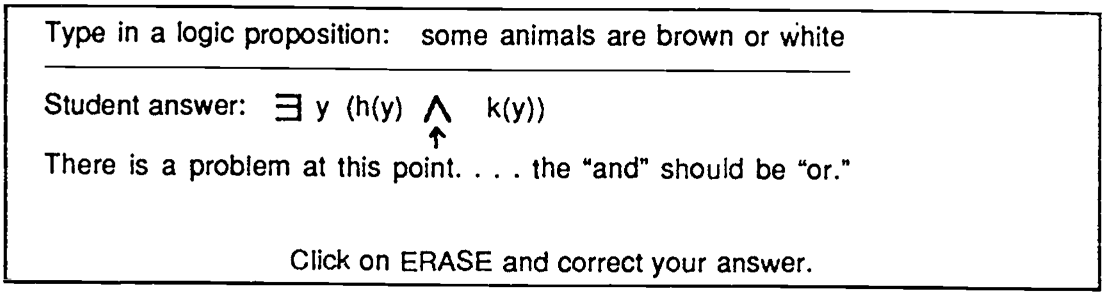
ILONA would have worked it out like this:

Let the variable "x" represent an element of "cheese"
Let the function "f( )" represent the property, ability or quality associated with "blue"

Let the variable "y" represent an element of "grass"
Let the function "g( )" represent the property, ability or quality associated with "green"

Therefore: $\forall x\ f(x) \longrightarrow \exists y\ g(y)$

Figure 4. An example of how ILONA solves a logic problem.



Type in a logic proposition: some animals are brown or white

Student answer: $\exists y\ (h(y) \underset{\uparrow}{\wedge} k(y))$

There is a problem at this point. . . . the "and" should be "or."

Click on ERASE and correct your answer.

Figure 5. An example of the feedback ILONA gives when a student makes an error.

nouns, and that the words *flubsy* and *wubsy* describe some property of these nouns. The natural language system in ILONA deals with unrecognized words by inferring their most probable function in the sentence from the sentence structure. This "fuzzy" parsing appears to work well for ILONA.

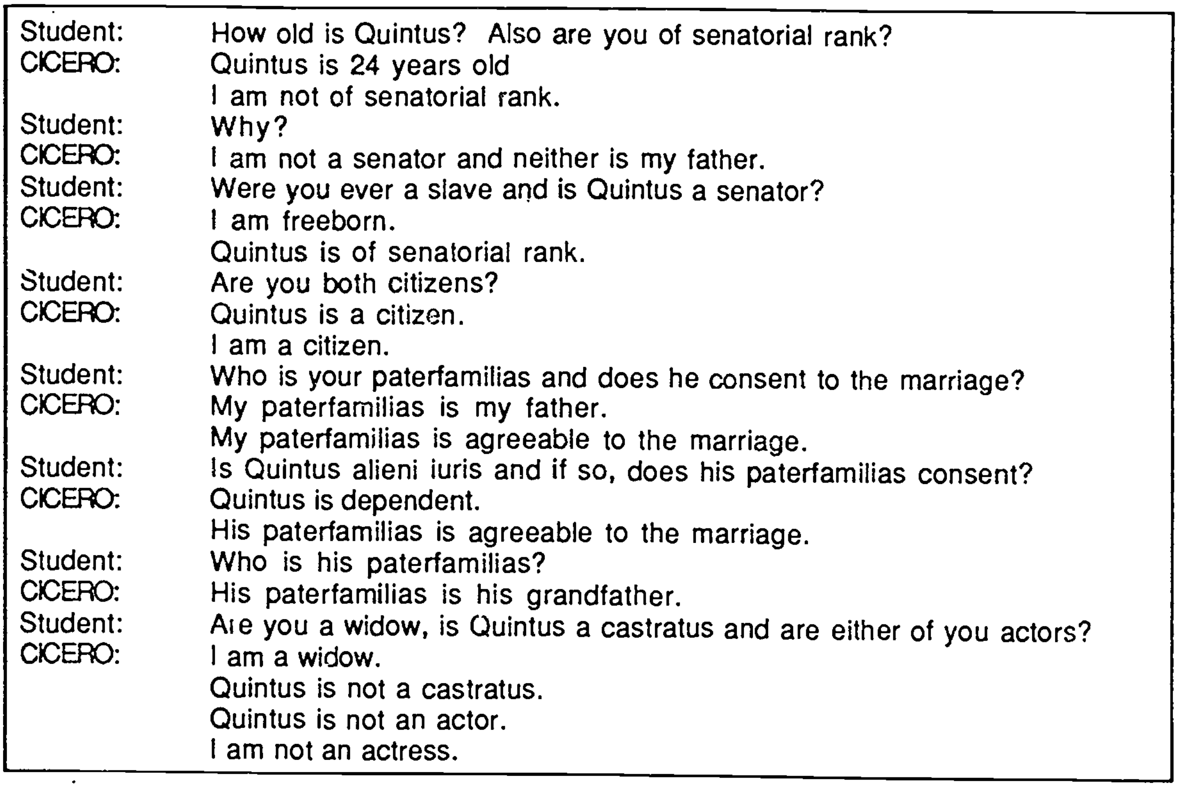
ILONA is part of a broader CAI tutorial that teaches the fundamentals of logic. ILONA provides students with a flexible work environment within which to practice what they have learned. The program is discussed more fully in Oberem, Mayer, & Makedon (1992).

CICERO

CICERO is a program that was developed as part of a series of lessons in the field of Roman Law (Lambiris, 1986). It allows students to participate in a simulated lawyer-client interview. The student, acting as a lawyer, asks the client questions to ascertain the facts of the case. The student then applies his or her legal knowledge to give advice to the client on the probable outcome of the case. A natural language processor is required to achieve a realistic and educationally rewarding flow of dialogue between the student-lawyer and the computer-client. It is important to allow a random sequence of questions, composite questions, pronouns and other linguistic devices common in the legal profession. The student must be allowed to ask questions in ordinary language, using alternative sentence structures and terms where appropriate.

The nature of the Roman Law of marriages is such that there is a limited range of relevant facts that are pertinent to the cases posed in the exercises. The relevant facts include such things as the age of the parties, whether they are related by blood and to what degree, their willingness to marry each other, and so on. It would not be relevant to ascertain what either party's tax liability is, or what their preferred sporting activity might be. Another factor which renders language understanding more tractable in this situation is the relative predictability of the form of the expected input. Input will be in the form of questions designed to elicit specific factual responses. The number of possible questions is thus finite as is the scope of the vocabulary that a lawyer would use. ALBERT's language system is well suited to this type of application.

The vocabulary needed for this program had been studied in an earlier attempt to design a lawyer-client simulation using conventional CAI answer-judging techniques. It was thus relatively easy to form the required lexicon by grouping the words according to their semantic function in the lawyers' questions. It was also



```
Student:   How old is Quintus?  Also are you of senatorial rank?
CICERO:    Quintus is 24 years old
           I am not of senatorial rank.
Student:   Why?
CICERO:    I am not a senator and neither is my father.
Student:   Were you ever a slave and is Quintus a senator?
CICERO:    I am freeborn.
           Quintus is of senatorial rank.
Student:   Are you both citizens?
CICERO:    Quintus is a citizen.
           I am a citizen.
Student:   Who is your paterfamilias and does he consent to the marriage?
CICERO:    My paterfamilias is my father.
           My paterfamilias is agreeable to the marriage.
Student:   Is Quintus alieni iuris and if so, does his paterfamilias consent?
CICERO:    Quintus is dependent.
           His paterfamilias is agreeable to the marriage.
Student:   Who is his paterfamilias?
CICERO:    His paterfamilias is his grandfather.
Student:   Aie you a widow, is Quintus a castratus and are either of you actors?
CICERO:    I am a widow.
           Quintus is not a castratus.
           Quintus is not an actor.
           I am not an actress.
```

Figure 6. An extract from a dialogue with CICERO.

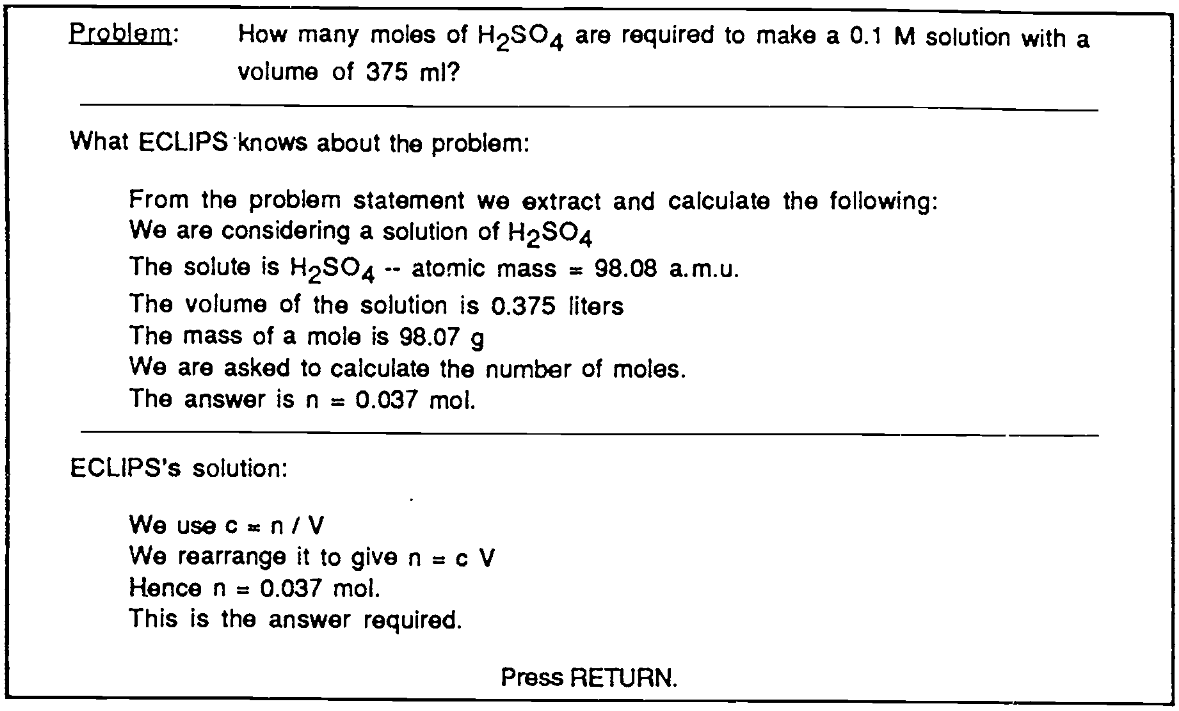
recognized that a relatively small set of syntactic structures are common to many of the questions asked. A single semantic sub-routine suffices to transform the input questions into one or more arrays of pointers that index into a knowledge-base of information about the two parties. This approach had been developed previously in a program designed to engage English second language (ESL) students in a computer-based dialogue (Oberem & Mostert, 1990). The answers given by CICERO are constructed using a very simple language generator that forms English sentences by placing pronouns, verbs, articles and other words around information contained in the knowledge-base. Each piece of knowledge in the database can also have an explanation associated with it. This allows students to ask "why" questions as illustrated in the dialogue shown in Figure 6.

CICERO keeps track of a student's progress during the course of the discussion. This information is used to judge whether or not a student has uncovered enough facts to give reliable advice and also enables CICERO to offer assistance to those students who have difficulty asking appropriate questions. Lambiris and Oberem have studied the use of this program and its earlier more traditional counterpart (Lambiris & Oberem, 1993). ALBERT's language system provides a more realistic opportunity for dialogue than the earlier versions of the simulation did. However, it was found that students who had used traditional CAI programs did not readily exploit the power of the natural language system without training in its use.

ECLIPS

Molarity is an area of chemistry where students are known to have difficulty (Gabel, & Sherwood, 1983). ECLIPS is a program designed to understand and solve chemistry problems of molarity typed in plain English. It uses both the natural language system and the generic problem-solver from ALBERT. The result is a program that closely resembles the original ALBERT. ECLIPS can understand and solve problems such as the one shown in Figure 7. Figure 7 also shows what ECLIPS "knows" about the problem and how ECLIPS solved it. At this time, ECLIPS does not include a tutorial component.

The vocabulary for ECLIPS was assembled by analyzing molarity problems found in a collection of introductory chemistry textbooks. The sentences used in these problems were examined and it was found that most of the syntactic patterns were identical to those required for ALBERT. The semantic routines from ALBERT needed some modification but, due to the inherent similarities in the way that information is presented in kinematics and in these problems, the changes were minor. One additional semantic routine was needed to identify chemical formulae and to extract the information from them. ALBERT's problem-solver is



Problem: How many moles of H_2SO_4 are required to make a 0.1 M solution with a volume of 375 ml?

What ECLIPS knows about the problem:

From the problem statement we extract and calculate the following:
We are considering a solution of H_2SO_4
The solute is H_2SO_4 -- atomic mass = 98.08 a.m.u.
The volume of the solution is 0.375 liters
The mass of a mole is 98.07 g
We are asked to calculate the number of moles.
The answer is n = 0.037 mol.

ECLIPS's solution:

We use c = n / V
We rearrange it to give n = c V
Hence n = 0.037 mol.
This is the answer required.

Press RETURN.

Figure 7. An example of a problem solved by ECLIPS.

generic and, in order to make it solve these problems, it was only necessary to substitute the molarity equations for the kinematical equations and change some of the variable names. Chronologically, ECLIPS was the first program derived from ALBERT. A tutorial component was not developed for ECLIPS, as the primary goal of this work was to demonstrate the transportability of the language system and the problem-solver.

CONCLUSION

In this paper we have described four programs that use a natural language system originally developed to teach problem-solving in physics. The success of the natural language processor in each case is due to the fact that the domain of operation is limited. We do not make the claim that this language system will work as well in every situation as it does these programs. However, the scope of many CAI tutorials and intelligent tutoring systems is sufficiently restricted to make ALBERT's language system both useful and robust. The programs described here demonstrate the transportability of the natural language system across diverse subject areas, programming languages and hardware platforms.

The original development of ALBERT required considerable design and programming effort and the work spanned several years. By comparison, the transfer of ALBERT's language system to other domains has required very little programming expertise and has been achieved quite quickly. Some tuning of the language system is required when it is first used in a new subject area. This takes the form of minor adjustments to the vocabulary elements and the syntactic pattern database. However, it has been our experience that this process converges quickly and that the language system soon functions very reliably.

New programs that use ALBERT's language system are under development. The emphasis in these programs is on engaging students in a natural and convincing dialogue to develop concepts that are difficult to learn in the traditional lecture environment. It is our hope that the work reported here will encourage others to enhance their CAI programs with more flexible language interfaces.

References

Brown, J.S., Burton, R.R. & de Kleer, J. (1982). Pedagogical, natural language and knowledge engineering techniques in Sophie I, II and III. In Sleeman, D. and Brown, J.S., editors, *Intelligent Tutoring Systems*. London: Academic Press, 227-282.

Chi, M.T.H., Feltovich, J.P. & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, 5 (2), 121-152.

Gabel, D.L. & Sherwood, R.D. (1983). Facilitating Problem Solving in High School Chemistry. *Journal of Research in Science Teaching*, 20 (2), 163-177.

Lambiris, M.A.K. & Oberem, G.E. (1993). Natural Language Techniques in Computer-Assisted Legal Instruction: A Comparison of Alternative Approaches. *Journal of Legal Education*, 43(1), 60-78.

McDermott, L.C. (1984). Research on conceptual understanding in physics. *Physics Today*, 37 (7), 24-32.

McDermott, L.C. (1990). Research and computer based instruction: opportunity for interaction. *American Journal of Physics*, 58 (5), 452-462.

McDermott, L.C. (1991). What we teach and what is learned - Closing the gap. *American Journal of Physics*, 59 (4), 301-315.

Mestre, J.P., Gerace, W.J., Hardiman, P.T. & Lochhead, J. (1987). Computer-based Methods for Promoting Thinking in Physics and Algebra: Incorporating Cognitive Research Findings into Software Design. A paper presented at the *Third International Conference on Thinking*, Honolulu, Hawaii.

Novak, G.S. & Bulko, W.C. (1993). Diagrams and Text as Computer Input. *Journal of Visual Languages and Computing*, 4 (2), 161-175.

Oberem, G.E. (1987). ALBERT: A Physics problem-solving monitor and coach. Proceedings of the *First International Conference on Computer Assisted Learning, ICCAL '87*, Calgary Alberta, Canada, 179-184.

Oberem, G.E. & Mostert, O. (1990). Adapting a physics-based natural language processing system for use with English second language students in a diagnostic dialogue setting. A paper presented at the *Tenth Annual Conference of the South African Applied Linguistics Association*, Rhodes University, Grahamstown.

Oberem, G.E., Mayer, O. & Makedon, F. (1992). ILONA: an advanced CAI tutorial system for the fundamentals of logic. *Education and Computing*, 8 (3), 245-254.

Oberem, G.E., Shaffer, P.S. & McDermott, L.C. (1993). Using a Computer to Investigate and Address Student Difficulties in Drawing Free-Body Diagrams. A paper presented at the *Summer Meeting of the American Association of Physics Teachers*, Boise, ID.

Reif, F. (1981). Teaching problem solving: a scientific approach. *The Physics Teacher*, 19 (5), 310-316.

Suppes, P. (1990) Uses of artificial intelligence in computer-based instruction. In Marik, V., Stepankova, O. & Zdrahal, Z, editors, *Artificial Intelligence in Higher Education*. New York: Springer-Verlag, 206-224.

Trowbridge, D. & Chioccariello, A. (1985). Computer dialogues for developing concepts of physics. In Duncan, K. and Harris, D., editors, *Computers in Education*. North-Holland Publishing, 381-385.

Acknowledgments

This work was sponsored in part by Rhodes University and the Institute for Research and Development in South Africa, the Computer Learning Research Center of the University of Texas at Dallas, and the National Science Foundation grant # MDR-8950322 to the Physics Education Group at the University of Washington. Mainframe ALBERT was developed using the TUTOR language of the Control Data PLATO system. PC-ALBERT, FREEBODY and ECLIPS were developed using cT, a product of Carnegie Mellon University. The TenCORE authoring language by the Computer Teaching Corporation was used for CICERO.