ARIADNE’S THREAD: AN INTRODUCTION TO LOGIC PROGRAMMING

VITTORIO MIDORO¹, AUGUSTO CHIOCCARIELLO², DONATELLA PERSICO¹,
LUIGI SARTI¹ and MAURO TAVELLA¹

¹Istituto Tecnologie Didattiche, Consiglio Nazionale Ricerche, Genova, Italy
²Honeywell Information System, Milano, Italy

PERGAMON PRESS
OXFORD · NEW YORK · BEIJING · FRANKFURT
SÃO PAULO · SYDNEY · TOKYO · TORONTO
1988
ARIADNE’S THREAD: AN INTRODUCTION TO LOGIC PROGRAMMING

VITTORIO MIDORO¹, AUGUSTO CHIOCCARIELLO², DONATELLA PERSICO¹
LUIGI SARTI¹ and MAURO TAVELLA¹

¹Istituto Tecnologie Didattiche, Consiglio Nazionale Ricerche, Genova, Italy
²Honeywell Information System, Milano, Italy

Abstract—Ariadne’s Thread is a package composed of a self-instruction text addressed to high school students coupled with a commercially available prolog interpreter. The material is aimed at introducing computer novices to the concepts of logic programming.

The examples and tasks in the package are based on the metaphor of Theseus in Crete with the subject matter presented through a series of dialogues between Ariadne and Theseus, introducing Prolog, a “genius” who will help him to find a way out of the labyrinth.

Thirty secondary school teachers of different subjects with little or no computer experience attended a week-long summer school based on Ariadne’s Thread. During the first two days they were introduced to Prolog in lectures, in the remaining three days they developed small Prolog applications in groups in the lab. with one computer and a tutor assigned to each group. Data was collected through questionnaires.

The results showed that the material was quite easily understood and motivating, and the participants were able to devise applications in their subject area but only with substantial help from the tutors. Prolog alone did not prove to be a suitable vehicle for an introduction to logic programming concepts. While simple examples were easily learnt, more difficult problems required mastery of the language far beyond the level attained.

This study indicates that the learning environment needs to be developed to facilitate the use of problem solving techniques, and the course requires to be extended to two weeks duration.

INTRODUCTION

A society shaped by the computer revolution demands the mastery of intellectual tools, such as communication, information retrieval, and problem solving skills, up to now not explicitly taught in school.

To respond to the general demand for “computer literacy,” new courses and suitable strategies need to be developed. In defining the contents one of the main objectives should be that of enhancing higher-order thinking skills among learners. Training of a large and heterogeneous teachers’ population to use the new learning materials in the school system is also needed. Two key ideas underlay our approach:

— The learners’ higher-order level thinking skills could be improved through acquisition of suitable mental tools;
— these tools can be acquired through an appropriate programming environment [1, 2].

The following computer science areas may be the basis for defining those contents:

— structured programming [3];
— object oriented programming;
— concurrent programming [4];
— logic programming.

This paper is concerned with logic programming; within this area relevant educational topics are:

— formal representation of knowledge related to the subject domain;
— inference mechanisms capable of solving problems in this domain;
— the capability of learning from experience in such a way that the computer program modifies itself;
— the use of recursive algorithms as tools to model complex situations.

Prolog has been chosen as one of the languages that incorporates the above features [5–7].
THE LEARNING MATERIAL: ARIADNE’S THREAD

Ariadene’s Thread is a package composed of a self-instructional book [8] addressed to high school students, coupled with a commercially available Prolog interpreter. This material is aimed at introducing computer novices to the above mentioned four concepts.

The learning material requires the students to write Prolog programs to solve problems of increasing difficulty. This operational approach is aimed at developing mental skills rather than transferring programming language notions.

The examples and tasks in the package are based on the metaphor of Theseus in Crete with the subject matter presented through a series of dialogues between Ariadne and Theseus. Ariadne introduces Theseus to Prolog, a genius who will help him to find the way out of the labyrinth.

Semantic networks, trees, binary trees and their Prolog representations are used by Ariadne to describe the possible labyrinths in which the Minotaur lives. The depth first search method is explained by Ariadne to make Theseus understand how Prolog gives answers to his questions; the assert and retract built-in predicates and their “unusual” arguments are the pretext to introduce the basic ideas which make a program modifiable. Recursion, backtracking and list manipulation are the elements which allow Theseus to find the way out of the maze.

According to Malone’s proposal [9], three components must be taken into account in order to develop intrinsically motivating learning material: fantasy, challenge and curiosity. In Ariadne’s Thread the metaphor and the mythological environment address the fantasy component while the tasks of increasing difficulty proposed by Ariadne to Theseus are related to the challenge component. The conflict between the naive Theseus’ way of thinking and the formal logic underlaying Prolog is aimed at creating the curiosity component.

To use this material in a self instructional way, an IBM/PC and a suitable Prolog interpreter are required. The user must enact Theseus’ role during the dialogue with Ariadne.

After the first validation tests, this material has been used as the basis for the teacher training course described below [10].

A TEACHER TRAINING COURSE

About thirty secondary school teachers of different subjects (math, humanities, philosophy and geography) with little or no computer experience attended a summer school based on Ariadne’s Thread.

The key idea in a teacher training course is for the teachers to receive the same instructional material to be used in the classroom. This approach takes into account both the methodological and the content aspects of the curriculum innovation. Of course, teacher training is an on-going process and it will continue with further studies and experiences.

The main aims of the course were:

— to test the instructional and motivational effectiveness of the material developed and collect teachers’ comments on the material;
— to evaluate its impact on teachers, and to verify the suitability of this material for teacher training;
— to explore the possible uses of AI techniques in the context of different high school subject matters.

Ariadne’s Thread and some case studies constituted the week-long course. During the first two days the participants were introduced to Prolog through Ariadne’s Thread, in the remaining three days they chose suitable topics and developed small Prolog applications. The first part was conducted in lecture mode with the lecturer following the book structure and programming examples using a personal computer connected to a projector.

In the second part the participants were divided into six small groups according to their topics and a tutor was assigned to each group. The activities were carried out in a computer lab with one computer per group.

Data were collected through:

— one questionnaire submitted to the participants after the lecture part of the course, aimed at
gathering information on understanding of topics and examples, quality of exposition, level of learning, appreciation and interest (Fig. 1);
— a second questionnaire submitted after the lab activities, aimed at collecting data on time spent on different tasks, practical problems in using the computer and difficulties encountered in using Prolog (Fig. 2);
— subjective evaluation by the tutors, who interacted very tightly with each group and provided feedback on the actual level of mastery and difficulties encountered by the students.

The qualitative nature of all the collected data was aimed at gathering feedback on the impact of the learning material and the course structure on the teachers and possibly redirecting our approach.

The results showed that, although the material was quite easily understood and fairly motivating, the programming skills acquired by the participants in the lecture phase were not sufficient to carry out the lab activities without substantial help from the tutors.

Nevertheless, the teachers could devise meaningful applications in their subject area using the above mentioned techniques.

Furthermore, Prolog alone did not prove to be a suitable vehicle for an introduction to logic programming concepts. In general while simple introductory examples are straightforward and easily learnt, more difficult problems require mastery of the language far beyond the introductory level.

This suggests that more “controlled practice” on these techniques and a more gradual transition from a learning environment to real applications are needed. Moreover, the use of Prolog to deal

![First Questionnaire](image)

**Fig. 1.** Most significant items of the first questionnaire. Each participant was asked to rate the levels of items on the left side. For each item the distribution of answers is shown on the right side.
with actual school topics is beyond the average programming skills which teachers can acquire with the current introductory course. Two things are needed to implement this change:

- a problem solving learning environment to introduce children to logic programming techniques;
- special purpose shells developed for their specific subjects, to stress the conceptual aspects sheltering the implementation details.

Finally, this provided indications of the necessary revisions for the teacher training course:

- to expand the required course time to two weeks;
- to develop a software learning environment to facilitate the use of problem solving techniques.

THE LEARNING ENVIRONMENT

The learning environment which has been designed and now is being implemented at ITD will be described below.

It consists of two modules: Dedalus and Labyrinth. Dedalus is meant to be used by the teacher to create microworlds. Each of these microworlds is centered on a particular learning task. Some of them have already been set up as part of the course material and are enough to get started.

Labyrinth is meant to be used by the students to solve problems in these microworlds (Fig. 3).

DEDALUS: THE PROBLEM GENERATOR

The typical steps of problem generation are:

(a) the teacher decides on the problem he wants the students to face, stating on paper both the structure of the microworld and the Prolog primitives that should be available to the
Introducing logic programming

students. The microworld is usually structured as a maze, each room may contain objects, a robot-like character (named Theseus) may be put in one of the rooms.

(b) interacting with Dedalus, which includes a graphical editor, the teacher actually draws the map of the maze and associates to it the set of primitives (Prolog predicates and interfaces);

(c) the teacher implements the primitives, according to their previous specifications.

From what has been stated above, generating a problem is a complex task and requires time, teaching experience and a reasonable knowledge of the Prolog programming language. For this reason the learning material includes pre-defined microworlds and gives suggestions on how to use, modify and enhance them.

Figure 4a shows an example of microworld. The structure is a two roomed maze, containing objects (playing cards) and Theseus. Figure 4b shows the set of primitives associated to it.

LABYRINTH: THE STUDENT INTERFACE

Once the student is assigned a problem, he enters Labyrinth and identifies the microworld to be used. Using a Macintosh-like windows system, the drawing of the corresponding maze is displayed together with a set of icons allowing the student to activate the following functionalities.

(a) SHOW PRIMITIVES. Shows the set of primitives available in that particular microworld.

(b) SHOW SYMBOLIC LABYRINTH. Shows the maze represented as a list of Prolog clauses (Fig. 4c).

(c) EDIT. Enters an editor for the student to define his own clauses. The microworld primitives can be used as an enhancement of the Prolog language.

(d) TRACE. Activates a tracing facility of selectable students clauses.

(e) HELP. Activates an on line help facility.

The dialogue between the student and Labyrinth takes place in the dialogue window. The student queries the system, trying either the primitives or its own clauses. Whenever the execution affects the state of the objects in the maze, the effect is shown in the drawing.

To clarify how the environment works, let us consider the classical wolf, goat and cabbage problem: Theseus must move the King, the Queen and the Jack from room A to room B, carrying one card at a time. Should they be left alone, the King would quarrel with the Queen, and the Queen would tempt the Jack. King and Jack can be left alone.

Since this problem is not a trivial one, the student is guided to solve it through a sequence of
Fig. 4. (a) The initial state of the maze for the King, Queen and Jack problem. (b) Primitives available to the student. (c) Prolog description of the maze problem. (d) An example of interaction.

steps of increasing difficulty and generality. At the beginning, the student tries to find a solution "manually" (Fig. 4d) using the primitives listed in Fig. 4b. Then the student is guided to accomplish subtasks of the final task and finally he/she is required to find a general solution for the given problem. Summarising, the student puts into practice the learnt concepts in a microworld which provides:

— a graphical feedback giving an immediate representation of the consequences of the student's trials;
— built-in predicates for the student to solve each problem;
— editing, trace and help facilities.

CONCLUSIONS

The work done thus far suggests that Prolog can be effectively used to introduce certain AI techniques and logic programming concepts. Both the direct observation of students during the
course and the analysis of the questionnaire indicate that Prolog itself seems an easy language to learn, but as task difficulty increases, the mechanisms of the language constitute a barrier to the novice; this is in agreement with others [11]. A suitable microworld provides an arena in which to guide the learner to acquire the necessary skills operationally.

Further developments of the project concern the use of these skills in the context of traditional school subjects. Experience with teachers has shown that there are domains in which these skills can be used to facilitate the learning process. In general, however, such applications require programming competency far beyond the average skill of teachers and students. This tends to shift the focus of the learning experience from problem solving activities to technical details. A possible solution is to develop special purpose shells which, while concealing the Prolog implementation, support the conceptual aspects of the application. Dedalus & Labyrinth is an example of such shells.

REFERENCES