USING CLIPS AS THE CORNERSTONE OF A GRADUATE EXPERT SYSTEMS COURSE

Kwok-bun Yue

University of Houston - Clear Lake
2700 Bay Area Boulevard, Houston, TX 77058

Abstract. This short article describes the effective use of CLIPS as the cornerstone in a graduate expert systems course. The course included about 8 to 9 hours of in-depth lecturing in CLIPS, as well as a broad coverage of major topics and techniques in expert systems. As part of the requirement of the course, students solved two small yet non-trivial problems in CLIPS before went on to develop a toy expert system in CLIPS in an incremental manner as the term project. Furthermore, students were required to evaluate CLIPS programs by their classmates. An anonymous questionnaire at the end of the semester revealed that the students responded very favorably about the course, especially their experience with CLIPS.

INTRODUCTION

This article describes the experience of teaching and using CLIPS as the expert system shell language in a graduate expert systems course in the Spring semester of 1990. The department of computer science at the University of Houston - Clear Lake offered two graduate courses in the artificial intelligence area. The first course, Artificial Intelligence, is a survey course of general searching techniques and knowledge representation issues, as well as various application areas such as learning, natural language processing, vision, etc. The second course, Expert Systems, is intended to include an in-depth coverage of expert systems theory and programming techniques. Currently, the department also offers a graduate course in Artificial Neural Systems.

There were several papers in the literature describing the teaching of expert systems courses (Bahill and Ferrell 1986, Brown 1987, Warman and Modesitt 1989, Wolf and Rozanski 1990). (Bahill and Ferrell 1986) discussed the advantages of using an expert system shell and emphasized the importance of a student term project. However, the course did not cover many expert system techniques such as those in the areas of uncertainty handling and knowledge acquisition. (Brown 1987) emphasized the evaluations of expert systems and were more suitable for students majoring in business rather than computer science. Both (Warman and Modesitt 1989) and (Wolf and Rozanski 1990) centered their courses on group projects and were quite practically oriented.

When we developed our course, we felt that it is more suitable to have a relatively complete coverage of major topics in expert systems, especially since the technology has matured significantly in recent years. On the other hand, we would also like to teach an

expert system shell language quickly and in-depth so that the students can develop their expert system term projects as soon as possible and with confidence.

CLIPS was used because it is relatively simple and can be mastered in relatively short time. CLIPS is also free and a copy of it (version 4.0) is enclosed in our textbook, (Giarratano and Riley 1989). Furthermore, most of our students are working in NASA related companies that use CLIPS extensively.

The first week of the course was devoted to the introduction and general theory of expert systems. To enable the student to develop skills in CLIPS as early as possible, we covered CLIPS in the second to fourth weeks of the course for a total of about 8 to 9 hours. This encompassed Chapters 7 to 12 in (Giarratano and Riley, 1989). Various assignments, as described in the next section, were assigned to the students to enhance their CLIPS programming skills.

Since CLIPS is a forward chaining expert system language, we spent time to cover languages in other paradigms to complement CLIPS. For example, an hour was used to lecture on VP-Expert and M1 to illustrate backward chaining shell languages. Framed-based systems and object-oriented based systems were also discussed. The exception is logic-based systems, which were assumed to be well covered in the prerequisite graduate Artificial Intelligence course.

From the fifth week on, various important topics in expert systems theory were elaborated. This included languages and tools, evaluation techniques, verification and validation, and explanation facilities. We spent especially generous amount of time in knowledge acquisition, where more than 75% of the materials in (McGraw and Harbison-Briggs 1989) were covered, and uncertainty handling, where techniques as advanced as Dempster-Shafer theory and fuzzy logic were elaborated. The last class was used for student presentations of selected term projects.

HOMEWORK ASSIGNMENTS

Believing that active participation is the best way of learning, a heavy dose of assignments was included in our courses. A brief description of the assignments is shown in Table 1 in the next page. Our assignments have the following characteristics.

(a) Nearly all assignments are related to CLIPS. It is hoped that this concentration on a single language will allow the students to develop true expertise in the language.
(b) Two somewhat traditional yet non-trivial problems were included to enhance student programming skills in CLIPS.
(c) Evaluation of other classmate programs were emphasized. This forced the students to read programs and learn by comparison and contrasting.
(d) The final term project was built in an incremental manner which better imitates how expert systems are constructed.

Each assignment is described in more details now. The purpose of the first assignment is to let the students to compare the flavor of a rule-based programming language (CLIPS) to a more traditional language like Lisp.

Homework #1: Implement the CLIPS program for the block world in the textbook (pp. 417-418) in LISP or PROLOG. Test your program with the test cases in the textbook. Briefly discuss the relative merits and difficulties of your LISP implementation.

Homework #2: Extend the CLIPS program for the block world problem in Homework #1 to handle multiple move-goals. Test your new CLIPS program with sufficient number of cases and hand in both the program and the output.

Homework #3: Design and write a CLIPS program that serves as an interpreter of nondeterministic finite state machines. Again, test it with sufficient number of cases. Describe briefly the difficulties that will encounter if you were using LISP or PASCAL.

Homework #4: Interview an expert in any problem of your interest. Hand in:

(1) a brief description of the problem,
(2) a brief listing of the reasons for selecting the problem, and
(3) about 10 non-trivial rules that the expert may use and that may form the core of a small expert system (be sure to start in a high enough level of abstraction.)

Homework #5: Implement the rules you obtained in Homework #4 in CLIPS. Be sure to include enough information so that other people can use your toy expert system. Hand in three copies.

Homework #6: Evaluate two toy expert systems by your classmates.

Homework #7:

(1) Describe and evaluate an expert system shell.
(2) Hand in a research paper in expert systems, together with your review.

Project: Develop and refine your toy expert system into a more powerful one in CLIPS with at least 40 non-trivial rules. Write a report about your expert system. You may need to give a presentation of its implementation. Hand in two copies.

Final Examination (Take home):

(1) Evaluate one of your classmates' expert systems.
(2) Answer several questions.

Table 1. Homework assignments and project.

The second and third assignments aimed to strengthen students CLIPS programming skills. Whereas there are ample of exercises in (Giarratano and Riley 1989), they are relatively easy and are much simpler than the complexity of an actual expert systems. Of course, students can develop CLIPS programming skills while developing their term projects. However, there are many tasks, such as problem definition, knowledge acquisition and

representation, in developing an expert system to distract the students from concentrating on practicing CLIPS. It is thus desirable to find some problems that are clearly defined, have relatively short solutions and are non-trivial.

The second and third assignments fitted these requirements well. In the second assignments, students were asked to modify the elegant solution (only 4 rules) in (Giarratano and Riley 1989) for planning in the block world of Winograd to handle the satisfactions of multiple goals (such as to move block A on top of block B and B on top of C). We were amazed that more than 5 different approaches were collected and though the solution has only about 10 rules, no student produced a correct solution which is also optimal (minimal number of movements of blocks). Same thing can be said of the third assignment, which implements an interpreter of nondeterministic finite state machines. Although the solution has only six rules, the rules are quite intricate and no student got everything right (most students overlook a potential infinite loop problem).

The remaining assignments were essentially the incremental development of a toy expert system and the evaluations of other classmate works. One major difference between our term project and the projects described in other literatures is that the students were requested to turn in prototypes (Homework #5) of their toy expert systems. This is used to emphasize the incremental nature of expert systems development. Another difference is the requirement to evaluate other classmates' expert systems. This reinforced the importance of the ability to evaluate expert systems as well as forcing students to read CLIPS programs. In general, we think that reading programs is an invaluable experience for gaining expertise in computer languages and it is not emphasized enough in many computer science curricula.

As in other expert systems courses, a wide variety of student expert systems were collected. The students reported no difficulty in implementing their expert systems in CLIPS and their CLIPS code are in general competent. On the other hand, some students had problems in identifying suitable expert system applications and finding experts. Two projects turned out not to be suitable for expert systems technology and it was obvious that at least two projects were not based on a real expert. The identification of suitable applications and experts will be of utmost importance in the future.

DISCUSSION AND CONCLUSION

Student evaluations at the end of the semester revealed that the course was very well received by the students, indicating that it is feasible to use CLIPS effectively as the cornerstone of an expert systems course.

An independent in-depth anonymous questionnaire were also filled by the students at the end of the semester. The student responses on the usefulness of lectures on various topics may range from 7 (most useful) to 1 (least useful). The average student response on the usefulness on the overall lectures is 6.1 whereas the average response to the lectures on more theoretical topics (such as verification and validation and uncertainty handling) ranged from 5.3 to 5.9. As a contrast, the average response of the usefulness of lectures on CLIPS is 6.7. This is significantly better than that of other topics.

There may be many reasons for the relatively low enthusiasm for more theoretical topics. For example, the theory on uncertainty handling requires solid mathematics

background. Topics like verification and validation of expert systems are still immature and are thus less organized and structured. Time limitation dictated that theory of knowledge acquisition can only be discussed in lectures but not practiced in the classroom, making it much less interesting and useful.

On the other hand, the responses positively indicated that the students were very interested in learning an expert system shell, CLIPS in our case, thoroughly so that they can practice the theory and develop their expert systems immediately. We feel that CLIPS is especially a good choice since it is concise, efficient, practical, free and can be mastered in relatively short time.

The student fondness of CLIPS was also shown in their responses on the usefulness of the assignments. Of all assignments, the response for the term project was the best with an average of 6.8. The two CLIPS assignments (homework assignments #2 and #3) draw an average response of 6.4 each. In contrast, the average responses of other assignments ranged from 4.9 (evaluation of an expert system shell) to 6 (development of the prototypes of the toy expert systems).

In conclusion, we believe that CLIPS can be used effectively as the cornerstone of a relatively complete graduate expert systems course, especially now that CLIPS version 5.0 provides object-oriented features to complement its rule-based programming paradigm. Well designed CLIPS problems are very helpful to the students for developing CLIPS programming expertise. A collection of such small yet illustrative problems should be very beneficial in teaching CLIPS in the future.

REFERENCES

Bahill, A. and Ferrell W. (1986). Teaching an Introductory Course in Expert Systems, ***IEEE Expert***, Vol. 1, No. 4, pp.59-63.

Brown, D. (1987). A Graduate-Level Expert Systems Course, ***AI Magazine***, Vol. 6, No. 3, pp.33-39.

Giarratano, J. and Riley, G. (1989). ***Expert System - Principles and Programming***, PWS-Kent, Boston, Massachusetts.

McGraw, K. and Harbison-Briggs, K. (1989). ***Knowledge Acquisition - Principle and Guidelines***, Prentice-Hall, Englewood Cliffs, New Jersey.

Wolf, W. and Rozanski E. (1990). Expert Systems: An Applied Course, ***SIGCSE Bulletin***, Vol. 23, No. 4, pp.23-24.

Warman, D. and Modesitt, K. (1989). Learning in an Introductory Expert Systems Course, ***IEEE Expert***, Vol. 4, No.1, pp.45-49.