Learning Computational Logic with an Intelligent Tutoring System: SIAL

A. Simón, A. Martínez, M. López, J.A. Maestro, J.M. Marqués and C. Alonso
Dpto. Informática, Universidad de Valladolid, Valladolid, Spain
arancha@infor.uva.es

Abstract

SIAL is an Intelligent System for the learning of …rst order logic. It has been developed as a laboratory tool for Arti…cial Intelligence courses of Computer Science curricula. Special care has been given to issues as ‡exibility, accuracy of the error diagnosis messages, and ease of use. This paper presents how these issues have in‡uenced the design of the system, and the intended use in the laboratory.

1 Introduction

University is demanding new formulas to improve the quality of teaching, promoting motivation and a more individualized learning. The use of educational software is seen as a valuable tool to ful…ll these demands.

Intelligent Tutoring Systems are computer-based instructional systems with models of instructional content that specify what to teach and teaching strategies that specify how to teach [Mur99]. This ability to model expertise enables the system to conduct …ne-grained interactions with the learner. The types of interactions supported by AI techniques are specially relevant when the goal is to acquire complex problem solving skills [Dil94].

We have developed an ITS for the learning of logic, called SIAL, which we will apply in the laboratories next year. Our experience started three years ago, with the development of a previous system, SLI [SMM98], a demonstrator of propositional and predicate calculus theorems.

The rest of this paper is structured as follows: next section starts by introducing our previous work with SLI. Then, we describe SIAL: its objectives, the subjects of computational logic which are involved in this tool, modes of interaction with the student, the user interface, the architecture of the system and some speci…c features related to error diagnosis and teaching methodology. Finally, we present the conclusions.

2 Previous work

SLI is a …rst order logic resolution system. The student introduces the axioms and the negated theorem to be proved, and the system presents the resolution tree in case there's one and it can …nd it. This way, the student can observe the substitutions o¤ered for the resolution of the problem. SLI has additional options to show the clauses generated in the resolution and the intermediate steps taken to arrive at them.

Although SLI was not initially designed for educational purposes, it was actually used as a pedagogic tool for teaching logic in the laboratory sessions of Arti…cial Intelligence during the 97/98 and 98/99 courses. The main bene…ts of this experience were a higher student participation in the learning process, which increased their motivation and interest in the subject, and a better understanding of the concepts presented [SMM98].

The experience with SLI led us to start a new project: SIAL (Intelligent System for the Learning of logic). The design guidelines of SIAL were exposed in [MS99]. In this paper, we describe the characteristics of the tool focusing the description on the pedagogical point of view.

3 Description of SIAL

The main objectives we want to achieve with SIAL are the following:

- To improve the learning by means of a software application which facilitates the assimilation of abstract concepts and problem solving skills.
- Availability of an application at the students' laboratories that will enable the follow-up of the student learning.

In the following sections, we describe our tool: the subjects of logic which are implemented in it, the modes of interaction with the student, the user interface, the architecture of the system and the way SIAL carries out the error diagnosis.

3.1 Subjects of computational logic involved

The program implements twelve thematic levels, ranging from the simplest skills to the most complex ones. Levels 1-6 include clausulation of well-formed formulas, uni…cation of predicates, binary resolution, resolution refutation, and application of factoring rule. From level 7 simpli…cation techniques for resolution that allow to eliminate generated clauses are introduced, as elimination of pure literals, tautologies, and subsumed clauses. Set of support strategy has been included as a form of controlling the progressive increase of generated clauses. Level 12 implements hyperresolution rule.

The main di¤erence between the levels lies in the type of interaction that the system allows to the user and in the di¤erent forms of performing error diagnosis.

3.2 Modes of interaction with the student

The in‡uence of the interaction over learning has been deeply studied, (see [ACKP95]) and remains as an active …eld in educational software research [LL99].

SIAL implements three di¤erent interaction styles with the student:

(i) Strongly guided or tutorized mode (levels 1-6) for the low-level learners, where the proposed exercises have to be solved step by step.

(ii) Weakly guided or free mode (levels 7-12), where the student can present the …nal solution to the system with no intermediate steps.

(iii) Student controlled or automatic mode, in which the student proposes and solves his own problems and calls the automatic resolutor to contrast the results or …nd them.

This di¤erentiation of levels follows the idea of knowledge decomposition presented in [CA95]. The exercises have been classi…ed by its di¢culty level, and in order to be able to solve the exercises at one level it is needed to have acquired some expertise in the lower levels. The problems in the …rst levels are solved step by step with the assistance of the system, so that it can follow the solution path of the learner, and advise him in case of a deviation of the correct solution path. This is bene…cial for the beginners, but at some level of expertise, the user must be able to perform the tasks by himself. Apart from this pedagogical reason, the step-by-step solution can become a load, instead of a help, for the experts, which are expected to prefer a more ‡exible interface. For this reason, we have included the free interaction style in the higher levels.

3.3 User interface

One important objective of SIAL was to o¤er a functional and ‡exible interface because this tool was planned for university students.

Other aim was to reduce the syntactic errors as much as possible for focusing the student on concept errors. In this respect, we have designed the interface in such a way that it acts as a supervisor of the user input, limiting the number of possible errors.

In the strongly guided mode, the interface restricts the symbols that the user can introduce, the actions that can be made over the expressions, and the order of these actions. Actually, the mouse is the only communication device with the computer, and consequently the keyboard is disabled. This is a form of reducing the complexity of the diagnosis process.

Two ways of processing are allowed to the user: to modify the whole expression or to select a subexpression to e¤ect an isolated treatment. To facilitate the selection of the expressions and subexpressions, we have developed an intelligent interface, which selects the whole expression a¤ected by a symbol. This feature is not only good for the ease of use, but it also achieves a pedagogical

Figure 1: View of a window in SIAL

goal: it helps the user to think always on the subexpression level, never on single characters [ASF99].

The window that processes the expression or subexpression is divided into two parts, one of them contains the problem proposed to the user, and the other is where the user can carry out the modi…cations in a simple way. For composing the expressions, the user only has to drag the desired part from the upper part to the inferior one, and/or select the adequate connectors in a button bar speci…cally designed for SIAL, introducing them in the correct place. This speci…c bar has a button for each conector (^; : ; _; $;...) and additional buttons for the renaming of variables, skolemization, elimination of expressions, and introduction/elimination of carriage return.

We have added a code of colours that facilitate the use of SIAL. For example, the set blue-yellow allows to determine the subexpression that is to be dragged if the user clicks the left-button of the mouse in the actual position, or that which is to be shown in a new window if the user does a double-click.

SIAL o¤ers a graphic output whenever this is possible, although in those cases the system can also provide an output in text mode. This happens in the resolution refutations that the user obtains from level 5, included the hyper-resolutions, since they are represented as a resolution tree with the necessary bindings for carrying out the resolutions.

The application counts with a context sensitive help developed in HTML. When an error is detected, the help shows theoretic explanations referring to the cause of this error. This feature is closely related to the error diagnosis process which is explained later in this section. Additional information, as the user manual is also included in the help system.

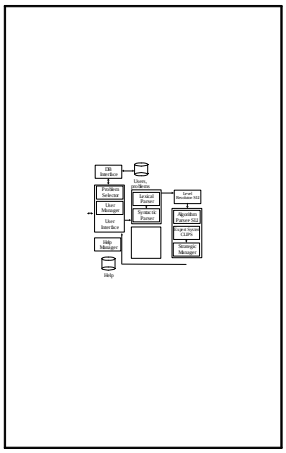

Figure 2: Architecture of the system

3.4 Architecture of the system

The architecture consists of the modules described below (…g 2).

- The **interface module** takes charge of the interaction with the user. It contains two additional components: the User Manager and the Problem Selector which manage the access to the remote data of the user accounts and the problems to be solved respectively. This module is closely related to the Help Manager and the Data Base Interface.
- The **lexical-syntactic parser module** which controls the user input.
- The Level Resolutor which is based on SLI as the expert model; the Algorithm Parser which uses SLI to compare the expression introduced by the user with that obtained from the expert in the clausulation and resolution refutation processes; a constraint base implemented in CLIPS, which helps to determine the cause of some type of errors; and the Strategic Manager -not implemented yet- which would help in the error diagnosis when the expert system fails.
- The **automatic resolutor module** that includes OTTER [WOLB92], a …rst order logic resolution system.

Both, SIAL and SLI have been developed in C++.

3.5 Error diagnosis

Error diagnosis is the task of inferring the learner's knowledge by analysing his or her behaviour [DS92]. It is a necessary task to support individually-adapted instruction.

As we indicated above, the error diagnosis is in‡uenced by the level of interaction between the system and the student. Within the strongly guided mode, the student has to include all the steps that lead to the solution. The input is entered by direct manipulation using a graphic interface. This interface acts as a supervisor of the student input, restricting the number of possible errors. When an error is detected, the system interprets it by direct inference, matching the error with the student's bug.

Within the weakly guided style, the student can write the …nal solution with no intermediate steps, which makes a direct inference diagnosis method almost impossible and very unreliable in the part corresponding to the clausulation. However, the errors produced in the resolution refutation, control strategies and simpli…cation techniques are diagnosed by direct inference as in strongly guided mode.

We have designed the diagnosis process as follows.

Any expression introduced by the student is …ltered through a lexical-syntactic parser, in order to detect the presence of non-valid characters or symbols that do not match the grammar speci…cations of the logic language used. In case the parsers detect an error, the diagnosis module rejects the expression signaling the error committed.

User's expressions are represented in a standard notation, and are compared with the output of two tools: OTTER, a theorem prover, and SLI, a …rst order logic resolution system. OTTER is used to detect whether there is a refutation or not, and SLI is used to …ner-grained tests about the clausulation and resolution refutation processes (resolvents, hyperresolvents, factorizations, subsumptions, and elimination of pure literals and tautologies).

The comparison between the user's and the system's expressions is performed using uni…cation. If the …nal expressions match (i.e. unify), the response is validated, otherwise, it tries to deduce the cause of this error. When the cause cannot be inferred at this level, another type of knowledge, coded in a constraint base is needed [MO99]. This happens in the clausulation process because it is the most complex.

The constraint base performs the analysis at a logic level in the strongly guided mode. It applies the solution to a set of constraints, which can detect logical errors, and the cause of them. This base has been written in CLIPS. CLIPS has been embedded in the program through a DLL.

The expert system is built as a set of constraints of two classes: some of them test typical errors found in logic as to confound the main operator in a logic equivalence, for example, $a \rightarrow b \leftrightarrow \neg a \wedge b$, instead of $\neg a \vee b$. The rest of the rules perform an search testing all the possible errors produced by a single fault in the replacement of the main logical conector.

4 Teaching Methodology with SIAL

SIAL is a tool designed for use in the laboratory as a reinforcement of the theory classes. It is not intended to replace classroom instruction, but to complement

it providing a problem solving environment. The system assumes that students have been exposed to the basic concepts of computational logic in lectures.

In the …rst session, the system proposes a test to the student when he starts using the tool, which places him in the adequate level depending on his knowledge of the subjects.

The student will solve a set of exercises proposed by the system at each level. Previously, the teacher will have thought out those exercises carefully in order to assure that the student learns progressively. The student will have to give the correct response to each exercise so that the system may show the next one, and he will have to solve all the exercises of the same level correctly in order to advance to a higher level.

5 Conclusions

We have developed an ITS for learning of computational logic, in which the error diagnosis and interface design play a very important role in the use of the application for teaching. The system is still under test, so that we have not been able to validate the tool at the laboratory with the feedback of its users, but we will do it next course. Relying on our experience with SLI, we think that SIAL can achieve the proposed objectives and increase the student's motivation in this subject.

Acknowledgements

This work has been partially supported by Junta de Castilla y León within the project Sistema Tutor Inteligente para la Enseñanza-Aprendizaje de la Lógica (Ref.: VA04/99).

References

[ACKP95] J.R. Anderson, A.T. Corbett, K.R. Koedinger, and R. Pelletier. Cognitive tutors: lessons learned. Journal of Learning Sciences, 4:167–207, 1995.

[ASF99] S.R. Alpert, M.K. Singley, and Peter G. Fairweather. Deploying intelligent tutors on the web: an architecture and an example. International Journal of Arti…cial Intelligence in Education, 10:183–197, 1999.

[CA95] A.T. Corbett and J.R. Anderson. Knowledge decomposition and subgoal rei…cation in the ACT programming tutor. In Proceedings of the AIED 95: 7th World Conference on Arti…cial Intelligence in Education, 1995.

[Dil94] Pierre Dillenbourg. The role of arti…cial intelligence techniques in training software. In Proceedings of LEARNTEC 1994, Karlsruhe, Germany, november 1994.

[DS92] P. Dillenbourg and J. Self. A framework for learner modelling. Interactive Learning Environments, 2(2):111–137, 1992.

[LL99] Karen Littleton and Paul Light. Learning with Computers: Analysing productive interactions. Routeledge, London, 1999.

[MO99] A. Mitrovic and S Ohlsson. Evaluation of a constraint-based tutor for a database language. International Journal of Arti…cial Intelligence in Education, 10:238–256, 1999.

[MS99] A. Martínez and A. Simón. SIAL: An intelligent system for the learning of logic. In Proceedings of the Young Researchers Track, AIED 99: 9th World Conference on Arti…cial Intelligence in Education, 1999.

[Mur99] Tom Murray. Authoring Intelligent Tutoring Systems: An analysis of the state of the art. International Journal of Arti…cial Intelligence in Education, 10:1050–1059, 1999.

[SMM98] A. Simón, J.M. Marqués, and J.A. Maestro. Desarrollo de aplicaciones software que faciliten la asimilación de los contenidos teóricos de las asignaturas. In Proceedings of the IV Congreso de Innovación Educativa en las Enseñanzas Técnicas, Comunicaciones y Ponencias, Tomo I, pages 305–314, 1998.

[WOLB92] L. Wos, R. Overbeek, E. Lusk, and J. Boyle. Automated reasoning: introduction and applications. Mc. Graw Hill, 1992.