Effective Translation of Didactic Principles into Logic Tutoring Software

The purpose of this presentation is to demonstrate some of the innovative aspects of a logic tutoring programme that is being developed at the Philosophy Department of the University of Gent and to explain its overall design.
We'll divide the session into three parts:

- Short explanation of the didactic principles as applied to logic tutoring software
- Overall view of the purpose of the programme and implementation of the didactic principles
- Demonstration and explanation of an innovative algorithm for selecting exercises in order to steer the learning process as efficiently as possible

At the Centre for Logic and Philosophy of Science (University of Gent, Belgium) logic tutoring software has been fully integrated for more than ten years in the teaching of undergraduates. Prof. Dr. Batens has developed a tutoring programme, which excels in the interactive way it works. The result of this approach satisfies both teachers and students: in an inquiry at the end of the year 1998, 72 % of the students spontaneously declared that the programme had been their best help in studying the course. Since the current programme runs only under MS_DOS, it has been decided to create a further extension of the programme that will run under Windows. At the same time, this project gives us the opportunity, using the obtained experience, to make the new programme as user-friendly as possible and to adapt it to the latest didactic insights.

Didactic principles as applied to logic tutoring software

Recently, research in didactics seems to develop in two ways: first, there is a growing amount of literature about what is called "learning styles".[1] It seems that the way students learn can be divided into several categories e.g. print-oriented, visual oriented, and so on. Taking the results of this research into account could put useful constraints on user interfaces, e.g. use of colour, highlights, etc.
Second, there is a shift in the way the role of the teacher is viewed. She is no longer the provider of knowledge[2]. Instead, she is seen as a supervisor of learning processes. Pupils should become "engaged learners". In this field of work, it is emphasised that students should be aware of metacognitive abilities, learning goals, evaluation demands, and so on. In other words, pupils should be more involved with the overall learning process.
Following Kuittinen (1998), we use four natural points of view to a computer-aided instruction (CAI) application: (i) Domain-dependent demands e.g. logic in our case, (ii) instructional demands like motivation, structure of application, interaction and so on, (iii) pragmatic demands (hardware requirements) and (iv) user-interface demands. Thus, learning style theories could be implemented into the user-interface, while the "engaged learning" theories provide guidelines for the instructional criteria.
In general, educational software should take into account that learning is a dynamic process. This means that it should not only include appropriate questioning but also provide branching according to the student's answers.
Concerning the instructional demands, logic tutoring software should provide an overall framework of goals, results, progress of the student and so on, in which the exercises are embedded. This way, the students don't get lost and won't lose motivation.

---

[1] For a lot of references, take a look at: http://web.indstate.edu/ctl/styles/ls1.html
[2] De Block & Heene (1996)

A good user interface design can help achieving the learning goals: a clear, consistent screen layout helps students to get a quick insight into the structure of the programme, the use of well-chosen colours helps not to lose concentration and can provide necessary emphasis, and so on.
During the presentation, guidelines to efficient educational software design will be presented, based on an overview of recent literature.

Purpose of the programme and implementation of the didactic principles

The overall purpose of the logic course is twofold:

- To give insight in the meaning of sentences in natural language.
- To give an introduction to formal logic: classical propositional logic (PC), a simple yet very natural system for relevant logic (PCR) and predicate logic (PL)

To achieve these goals the programme contains the following exercises:

- Well-formed formulas and inference rule training for PC, PL and PCR
- interactive proofs with help-modules for PC, PL and PCR
- interactive tableaux with help-modules for PC, PL and PCR
- classes, relations and sets
- translation of sentences in natural languages into PC and PL
- recognition of different types of inferences

In order to meet high standards, we use Van Ditmarsch's categorisation of interfaces as an evaluation tool (Van Ditmarch '98) and Delphi as a programme language. Although the programme is meant to be a tool to make logic exercises, it should also be the first step towards an independent logic course on CD-ROM. Moreover, the programme is meant to be used by other logic teachers as well, and should therefore leave room for individual preferences concerning inference rules, used symbols, and so on. Therefore, the use of databases in Delphi is of the utmost importance, for they should be easily accessible and modifiable by teachers, but not by students.

As a consequence of these didactic developments, the following features will be built into the programme and receive a lot of attention:

- Division into more or less independent modules
- A practice mode and an 'examination' mode
- A clear explanation of the learning goals and what is expected from the students
- An easy-to-follow overview of the obtained results
- Suggestions concerning problems to focus on
- Hyperlinks from the relevant jargon to an explanatory word-list
- Exercises which rise in degree of difficulty
- Feedback as 'personalised' as possible
- Exercises are offered as varied as possible
- Students are offered automatically more exercises from the kind they do worst

During the presentation, some of the above-mentioned features of the programme will be demonstrated and – if necessary – the underlying algorithms and 'construction tricks' will be shown. Emphasis will be laid on context dependent feedback systems for proofs in natural deduction, since this is one of the highlights of the programme.

An innovative algorithm for the selection of exercises to guarantee effective learning

Already in the early sixties, experiments showed that offering a mixture of different exercises gives the best results in terms of 'long term gains' (De Block & Heene 1995).
Secondly, taking into account the individual differences between students' abilities, they should each be given the exercises they need, i.e. by means of branching, they should be offered relatively more exercises of the types they're least good at.
Most of the existing programmes we analysed, however, do not take these principles into account.

In order to improve the didactic quality of the programme we developed a relatively simple algorithm to solve the above-mentioned problems.
In short, it exists in giving an equal a priori probability to each type of exercise (e.g. N types) of 1/N. The computer randomly selects one of these types. If the chosen problem is correctly solved, the probability of this type of exercise of being randomly chosen again is reduced with a certain factor v/N, where $0 < v < 1$. The probability of the other types of exercises to be chosen is enlarged at the same time with the factor v/(N *(N-1)). Finally, if the probability of a certain type of exercise to be chosen drops below a critical value v/N, it cannot be chosen.
Therefore, the lower v, the more the selection of types resembles a completely random choice. The higher v, the more controlled the selection process is. The disadvantage of a completely random choice is the fact that it is hard to control the types of exercises the students will have to solve. This makes it possible that students get too many exercises of just a few types. On the other hand, the disadvantages of a large v is the fact that the type of exercise coming up next will be clear for the students, so that solving the problem becomes a reproductive process with a less effective learning process as a result.

It can be shown, however, that for certain values of v, the selection process has the following properties.

- Occurrence of types of exercises is not predictable
- Prevalence of types of exercises resembles a normal distribution already after N exercises – this within strictly controlled limits

From this, we can conclude that the use of the algorithm offers a solution for the first problem mentioned above.

More important, however, is the ease with which the algorithm can solve the second problem. It suffices to set the probability of a certain type to be chosen definitively at zero – after this type has been solved correctly for a number of times – and to redistribute the probabilities among the remaining types. Even more flexible is the following option: one can reset the probabilities when the student didn't solve the exercise correctly. That way, one can be sure that the pattern of prevalence of the correctly solved exercises of the different types resembles a normal distribution. Thus, the learning path of the students will naturally follow a way towards those types of exercises they do worst, which will increase the efficiency of the overall learning process.
Notice also that the algorithm has other advantages as well. Since it is, of course, independent from specific exercises – or exercises at all, for that matter – it can be used whenever one wants to 'fake' a normal distribution on a 'short notice'. Besides, it is relatively easy to 'attach' feedback systems to the types of exercises, thereby making a first start towards personalised help or even 'personalised software'.
During the presentation the algorithm and its applications will be explained and demonstrated.

References

Brickell, G: *Navigation and learning style.* Australian Journal of Educational Technology

De Block & Heene: *Inleiding tot de Algemene Didactiek.* Standaard Educatieve Uitgeverij, Antwerpen, 1995 (in Dutch)

Kuittinen, Marja: *Criteria for evaluating CAI applications.* Computers & Education 31 (1998) 1-16.

Van Ditmarsch, Hans: *User interfaces in natural deduction programs.* In: R.C. Backhouse (ed.), Proceedings of UITP 98, TUE CS report 98-08, 87-95. Eindhoven, University of Technology, 1998.

Equipment needs

- power point presentation with a laptop
- overhead projector