Towards reflective tutoring systems: self-representation and self-improvement.

Pierre DILLENBOURG and Peter GOODYEAR

Centre for Research on Computers and Learning.
University of Lancaster (U.K)

Abstract

Most current ITS implementations are consonant with results from research on human teaching which state that teaching is mainly the execution of pre-defined tutoring sequences rather than rational decision making based on explicit pedagogical theory. Nevertheless, human teaching research shows that the critical competence of the expert tutor is his/her ability to monitor the execution of pre-defined sequences, escape when they fail and reflect upon her/his experience. Analysis of the functions supporting such reflective activities leads us to propose an architecture for a reflective self-improving tutoring system.

1. The nature of pedagogical expertise

One may view the ideal tutor as an intelligent agent which uses some formalized declarative theory of tutoring in order to produce coherent and adequate teaching behaviour. Unfortunately, we know there are some serious problems with this approach:

a) We do not have a coherent and complete theory of tutoring expressible in computational terms. We have only fragments of such theory.

b) Experiments trying to prove the efficiency of some tutoring strategy across the curriculum have generally showed that tutoring knowledge is partially domain-specific.

c) There are advantages to having at least some tutoring activity generated by "compiled" teaching sequences (procedures). This can be computationally more efficient (reducing response times) and can offer a more consistent tutoring interface to the learner (moment-by-moment decision making may produce disjointed tutoring which disorients the learner).

d) Most important, there are many strategies (such as those employed by experienced teachers) for which we have prima facie evidence about effectiveness but for which theories are unable to provide an explanation with sufficient rigour to be expressible in computational terms (e.g. Leinhardt and Smith, 1985).

These problems cause us to look at a contrasting view of tutoring expertise, i.e. as a set of pre-defined tutoring sequences (or teaching plans), accompanied by the knowledge required for
selecting the appropriate sequence. In most current ITSs, tutoring knowledge is actually much closer to this sequence-based view than to the declarative theory-based approach, despite the rhetoric of ITS design.

There are parallels with research on human teaching: teachers' behaviour cannot adequately be described using the theory-based approach. Such a rationalized view does not readily fit with observations, as far as real teaching problems of any significant interest are concerned. For example, studies of elementary classroom teachers indicate that a teacher will typically engage in 200 to 300 distinguishable interactions per hour. In this time, they are unlikely to take more than six tutoring decisions, in the sense of considered choices between alternative courses of action (Calderhead, 1984).
2. The reflective teacher

Rather than characterising teachers’ classroom decision making as a steady, deliberate, purposive problem-solving activity, it is better to see it as the implementation of previously planned activities accompanied by a careful monitoring of classroom events (Peterson and Clark, 1978; Leinhardt and Greeno, 1986). Deliberate decision-making is activated when monitoring reveals too great a discrepancy between anticipated and actual classroom behaviour, and is essentially remedial (Calderhead, 1984).

Results of this kind support the hypothesis that experienced teachers have and use a repertoire of pre-defined tutoring sequences. These sequences may permit a certain amount of modification "on the fly" but they are essentially compiled procedures used to reduce cognitive load in a demanding task environment. Nevertheless, the crucial determinants of the teacher's success will be his/her ability to monitor the execution of the sequence, and know how to use feedback from the teaching situation to cue a switch to a new sequence.

There are interesting questions about the genesis of a particular tutoring sequence: is it copied as a fully-formed procedure - for instance through the practice of "model lessons" during teachers' training - or derived by reasoning from some declarative educational theory and routinized through practice (Anderson, 1983; Olson, 1984)? This has implications for the major weakness of reliance on proceduralized knowledge: its opaqueness creates a problem when it fails, for instance when the teacher is confronted by a new situation. The capacity to explicate tacit "knowledge in action", in the context of novel problems, represents an important constituent of professional expertise.

We all engage in many forms of action which have become routinized through practice over time (driving a car, calculating, using a word-processor...). If we want to understand some detail of how we perform these tasks, we try to observe ourselves in action. This is not always easy and may cause perturbations in task performance, but it is possible. Moreover, we can learn something merely by imagining or remembering ourselves performing the task: we can learn by manipulating a symbolic representation of our task performance. With respect to tasks such as teaching, reflecting on action in this way has been identified as a key ability distinguishing expert from novice (e.g. Schon, 1987).

Finally we need to consider when or why a teacher engages in this reflective process, i.e. what are the precipitating circumstances. We discriminate three sets of circumstances which vary according to the "urgency" of the process:

- dramatic real-time failure of tutoring (e.g. absence of student reaction);
- post-hoc reasoning about recent tutoring, activated by the existence of some unsatisfied tutoring goals;
- on-going reasoning about tutoring motivated by a top-level goal of self-improvement.
From the evidence on human tutors' reflection, we would propose that the cognitively demanding reasoning necessitated by self-monitoring and real-time repair (reflection in action) will require different mechanisms than those driving more "leisurely" reflection (reflection on action).

3. Architecture of a reflective tutoring system.

This section aims to describe the architecture (i.e. some distribution of functions among various components) which could enable the tutor to perform the reflective activities previously described. As figure 1 shows, the tutor's functions are conceptually shared between two agents, the "monitor" and the "mentor", which both handle the set of tutoring sequences. With one exception, these agents respectively perform reflection in action and reflection on action.
The Reflective Tutor

3.1. The set of tutoring sequences.

In accordance with the previous sections, the tutor's knowledge should be represented as a set of pre-defined sequences. Each sequence is to be defined by four subsets of knowledge:

1. The "actions", i.e. a list of functions that the tutor must call successively when it applies a tutoring sequence. These functions are defined in the interface, the domain model or another component of the system.

2. The "conditions", i.e. the criteria for selecting a sequence, expressed in terms of student model state (or category) or in terms of difference between the student model state and the goal state.

3. The "experience" wherein, each time a sequence is applied, the system records the student model state and the effectiveness of the tutoring sequence (or the student model states before and after the sequence application).

4. The "description" which contains a more or less elaborate declarative description of the procedural definition of the sequence.

The use of subsets 3 and 4 will be described in section 3.3. Nevertheless, we can already see the symmetrical structure of this knowledge construct, one face being oriented towards reflection-in-action (1 & 2) and one face dedicated to reflection-on-action (3 & 4).

3.2. The monitor.

The first task of the monitor is to select a tutoring sequence by comparing its "conditions" part to the current student model state. If, for some sequence, these conditions are satisfied, then
the monitor's next task is to run this sequence. Since the sequences are composed of calls to other components of the system, the tutor generally loses the control of the sequence during its execution. This is not acceptable in a reflective tutor (reflection in action) : it must monitor the sequence execution and hence keep some control of the sequence computation.

This control must be defined by some monitoring knowledge, which would be included within the monitor when it concerns all the sequences and within the sequence "actions" subset when it concerns a particular sequence.
Here are some examples of monitoring knowledge:

- monitoring the interface:
  "If, given the current state of the interface, the strategy execution will lead to more than 5 windows being open, then close any windows which are obsolete."

- monitoring the domain computation:
  "If, given the current domain knowledge, the domain computation of some function in the strategy will last longer than 20 seconds, then print "Please, be patient ..."."

- monitoring the diagnosis process:
  "If the student has not given any response during the three last activities, then escape from the current sequence."

- monitoring the monitoring itself:
  "If no sequence may be selected for this student (because the sequences related to his/her state have already failed), then shift from the monitor to the mentor."

How may this kind of monitoring (including some self-monitoring) be performed by the monitor? The anticipatory character of the monitoring knowledge examples shows that this control may not be reduced to some computation of system variables but implies some reasoning about computational processes and performances. This is nevertheless a simple form of reasoning. Hence, the monitor implementation should draw its inspiration from the current work in artificial intelligence on procedural reflection (Maes, 1988). Procedural reflection is defined as "the ability for a process to describe and analyse itself while running" (Ferber, 1988).

This procedural reflection is generally implemented through the design of interpreters, which handle some "reified" form of the program to execute and some description of themselves. The permanent updating of this self-representation is achieved by writing the interpreter in the same language that is processed.

3.3. The mentor

The mentor has to achieve "reflection on action", more precisely to resolve the conflicts emerging between its knowledge and its experience. (Schon's concept of "surprise", 1987). The conflict resolution requires knowledge such as:

Over-generalization rule

| IF | I know that the diagnostic X is the "conditions" part of tutoring sequence Y | AND |
|    | I experienced that the diagnostic of the student Z was X                     | AND |
|    | I experienced that the sequence Y has been inefficient for a student Z       |     |
| THEN| a) I remove diagnostic X from the "conditions" part of sequence Y              | OR  |
|     | b) I change the diagnostic process so that it does not produce diagnostic X for the student Z | OR  |
|     | c) I do nothing (considering this experience as a statistical accident)       |     |

Under-generalization rule
IF I know that the diagnostic $X_1$ is the "conditions" part of tutoring sequence $Y_1$ AND I know that the diagnostic $X_2$ is the "conditions" part of tutoring sequence $Y_2$ AND I experienced that the diagnostic of the student $Z$ was $X_1$ AND I experienced that the sequence $Y_1$ has been inefficient for a student $Z$ AND I experienced that the sequence $Y_2$ has been efficient for a student $Z$

THEN a) I add diagnostic $X_1$ to the conditions part of sequence $Y_2$ OR b) I change the diagnostic process so that it does produce diagnostic $X_2$ for the student $Z$ OR c) I do nothing (considering this experience as a statistical accident).
The conclusions a) and b) of these rules actually have the same effect on the performance of the system: for the under-generalisation rule for instance, the result will be to select the sequence Y2 for any future student behaving as the student Z. The difference lies in the location of the knowledge modifications: the first conclusion updates tutoring knowledge while, the second changes the cognitive diagnosis process. In the context of the reflective tutor, we will opt for the first one: changing the tutor knowledge (more precisely, the conditions part of tutoring sequences).

Nevertheless, we need some rule of a higher level to choose between conclusions a) and c) (or to chose between rules if we translate each of these rules into production rules with a single conclusion). For the under-generalization rule, this meta-rule would be something like:

\[
\begin{align*}
\text{IF} & \quad \text{There is theoretical evidence that "X1 is a condition of Y2" OR} \\
& \quad \text{There is statistical evidence that "X1 is a condition of Y2"} \\
\text{THEN} & \quad \text{Chose conclusion a) (change knowledge)} \\
\text{ELSE} & \quad \text{Chose conclusion c) (do nothing)}
\end{align*}
\]

The theoretical evidence refers to the ability of the system to prove the statement "X1 is a condition of Y2" by using the description part of the strategy Y2, some explicit representation of the student model X1 and a partial theory allowing it to build the link between these descriptions. Knowing that some theorem is provable within some theory relates to another side of AI work on reflection, which is called declarative (Maes, 88) or conceptual (Ferber, 88) reflection. While procedural reflection corresponds to some form of self-control, declarative reflection manipulates some explicit description of the process reflected (self-representation), which is supposed to give some "self-understanding" to the system.

Most authors touch on the recursive character of reflection: if you reflect on your computation, since reflection is a form of computation, you must be able to reflect on your reflection! In this example, the mentor could reason on the reflection performed by using this kind of meta-meta-rule:

\[
\begin{align*}
\text{IF} & \quad \text{Rule Z1 led to adding conditions X1,X2,X3, X4..., Xi to sequence Y} \\
& \quad \text{Rule Z2 led you later to remove conditions X1,X2,X3, X4..., Xj from sequence Y} \\
\text{THEN} & \quad \text{Remove Rule Z1}
\end{align*}
\]

These reflection activities should be performed when the system is not in use with a particular student, for instance after being used by a sample of 30 students. But, besides this kind of post-action reflection, the mentor must also guarantee some real-time repair, i.e. helping the monitor to take a decision when it is not any more able to do it. This activity requires knowledge like:

\[
\begin{align*}
\text{IF} & \quad \text{I know that the diagnostic X is the "conditions" part of tutoring sequence Y} \\
& \quad \text{I experienced that the diagnostic of the student Z was X} \\
& \quad \text{I experienced that the sequence Y has been inefficient for student Z} \\
& \quad \text{There is no other sequence whose conditions equal diagnostic X} \\
\text{AND} & \quad \text{AND} \\
\text{AND} & \quad \text{AND}
\end{align*}
\]

The concept of a reflective tutor is a generalization of the concept of a self-improving system, since improvement is the result of a particular kind of reflection-on-action: the tutor may discover new conditions to select more efficiently its tutoring sequences. The architecture described results partially from our attempt to overcome the restrictions we met in our previous work on self-improving systems, especially the tutor's inability to discriminate between valid knowledge and results due to chance.
The meta-rule presented in the previous section indicates the kind of machine learning mechanisms involved in this improvement-oriented reflection: "theoretical evidence" clearly refers to explanation-based learning methods (Mitchell et al, 1986; Dejong et al, 1986) while the "statistical evidence" refers to similarity-based methods, i.e. methods based on induction.

For now, we believe that the originality of the proposed architecture lies in the idea that improving the interaction between the learner and the system implies not only that the system has a good representation of the learner but also an appropriate self-representation.

The architecture proposed in this paper has not been implemented. It results from the convergence of reflections drawn from our respective experiences in tutoring knowledge (Goodyear, 1986, 1988, 1989) and self-improving systems (Dillenbourg, 1988, 1989). This architecture implies a multiple representation of tutoring sequences, at least in procedural and declarative forms. It is interesting to notice that the same trend has also appeared in work on domain expertise and student modelling. A second notable convergence is in work on reflection itself within the informing disciplines of ITS development: in educational research (teacher as reflective practitioner), in cognitive psychology (learning through reflection on experience (Boud et al, 1985)) and in computer science (computational reflection). Our current work is exploring further facets of reflection, both as a powerful tool for enhancing learning and as an aid to thinking about self-evaluation by intelligent agents.

6 Acknowledgments
We are grateful to fellow participants in the workshop on tutoring knowledge at the second European Seminar on ITS (Le Mans, November '88) and to John Self for comments on an earlier drafts of this paper. Responsibility for its final content is ours alone.

7. References
GOODYEAR, P. Ed. (in press) Teaching Knowledge and Intelligent Tutoring, Norwood New Jersey, Ablex.
OHLSSON, S. (1986) Some principles of intelligent tutoring. Instructional science (14) 293-326