Representing a Student's Learning States and Transitions

Denise W. Gürer, Marie desJardins, and Mark Schlager
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025 USA
{gurer, marie}@erg.sri.com and mark_schlager@qm.sri.com

Abstract

We describe an ongoing project to develop an adaptive training system (ATS) that dynamically models a student's learning processes and can provide specialized tutoring adapted to a student's knowledge state and learning style. The student modeling component of the ATS, ML-Modeler, uses machine learning (ML) techniques to emulate the student's novice-to-expert transition. ML-Modeler infers which learning methods the student has used to reach the current knowledge state by comparing the student's solution trace to an expert solution and generating plausible hypotheses about what misconceptions and errors the student has made. A case-based approach is used to generate hypotheses through incorrectly applying analogy, overgeneralization, and overspecialization. The student and expert models use a network-based representation that includes abstract concepts and relationships as well as strategies for problem solving. Fuzzy methods are used to represent the uncertainty in the student model. This paper describes the design of the ATS and ML-Modeler, and gives a detailed example of how the system would model and tutor the student in a typical session. The domain we use for this example is high-school level chemistry.

Introduction

Employing computational models of how and what people learn can improve human-machine interactions. By modeling users' learning methods and knowledge state, systems can adapt to their particular needs and abilities. For example, collaboration between two experts in different domains can be facilitated through a model of their individual areas of expertise and interaction styles. Intelligent learning environments can provide appropriate feedback and assistance, based on a model of the student's learning process and current belief state. In addition, a machine-based "collaborative student" or co-learner can be created, who is at the same level as the student, and can learn with the student, allowing them to learn more effectively by interacting with a peer.

We are developing an adaptive training system (ATS) that incorporates a machine-learning based model, ML- (for Machine Learning) Modeler, that emulates the novice-to-expert transition by representing the knowledge state of the student and the learning mechanisms used to reach that state[1]. We use the name "Adaptive Training System" to emphasize the ability of the system to dynamically model students' learning processes and to adapt its tutoring methods to individual students. The knowledge representation used by ML-Modeler includes not only procedural strategies for problem solving (e.g. rules and procedures), but networks of abstract concepts and relationships.

An key feature of ML-Modeler is its ability to model a student's learning mechanisms such as specialization and generalization. When the student makes a mistake, ML-Modeler first determines the corresponding error or misconception, where an error is a mathematical or algebraic error and a misconception is a concept the student is having trouble understanding. If the mistake is a misconception, ML-Modeler attempts to obtain a deeper understanding of the misconception through a case-based analysis of the student's answer.

This unique approach allows the system to model a student's conceptual understanding, as well as their problem-solving skills. By consulting the student model, the ATS can identify weaknesses and errors in the student's understanding of the domain, and can take appropriate remedial action.

In the next section, we summarize some related work in the fields of intelligent tutoring systems and cognitive modeling. We then describe the representation used for the student and expert models and the techniques used by ML-Modeler, including the learning methods, model-tracing procedure, and representation of uncertainty. We give a detailed motivational example of a training session in the domain of chemistry equilibrium problems, and conclude with a discussion of research directions and contributions of this work.

Related Work

To date, cognitive learning theories have mainly focused on computational models of skill acquisition through the process of procedural compilation (Anderson, 1983; Chan, Chee & Lim, 1992; Möbus, Schröder & Thole, 1993) and procedural chunking (Newell, 1990). In addition to procedural skill acquisition, deeper cognitive structures are needed that can model abstract principles and concepts, and that can represent relations among these abstract concepts through the use of causal links. One of the goals of our work is to represent this type of knowledge as it is acquired by the student through analogical and inductive

1. The work described here is an ongoing project, and the ATS system, including ML-Modeler, is only partially implemented.

reasoning, in addition to procedural chunking and compilation.

It has been demonstrated that novices focus on the superficial or surface features of a problem, while experts focus on functionality or underlying domain principles (Chi, Glaser & Rees, 1982). In addition, when learning a new task, people often refer back to a previous problem to refresh their memories or to use as an analogy to solve their current problem (Ross, 1989). Pirolli and Recker (1994) showed that good students use self-explanation of example problems using abstract concepts.

Elio and Scharf (1990) developed a case-based model that could dynamically represent a novice-to-expert transition, based on a shift from focusing on surface features to abstract principles. Our work uses a similar case representation.; however, ML-Modeler goes beyond the case-based reasoning methods, emulating other learning methods in addition to case-based reasoning. Using a network representation of the student's knowledge with nodes representing problem statements, surface features, and abstract concepts, we model case-based reasoning, generalization, and constructive induction (the creation of new features) in a uniform framework.

Conati (1995), the other paper in this symposium that deals specifically with intelligent tutoring systems, outlines an extension to model-tracing diagnosis techniques that can recognize and reason about multiple problem solving styles. In some ways, this is similar to our method of matching the network representing a student's solution to an expert solution: in both cases, the order of the solution steps is less important than the concepts represented by the steps, but is still useful for understanding a student's overall problem-solving strategy.

Conati's work is built on top of the OLAE system (Martin and VanLehn, 1993), which represents a student's conceptual understanding of a domain with a Bayesian network of concepts. This probabilistic representation is analogous to our fuzzy representation of the uncertainty attached to each conceptual link in the student model. It is not clear where the conditional probabilities used in the belief network in OLAE come from (i.e., how does one compute the probability that a student actually applies a particular concept, given that they have familiarity with that concept?) We discuss some of the difficulties of using a Bayesian model and argue in favor of a fuzzy representation later in the paper.

Several papers in this symposium deal with introspective reasoning, or the problem of representing a system's internal belief and reasoning structure so that the system can analyze and correct misconceptions and missing knowledge in its own internal model. This problem is directly analogous to the problem we are addressing: representing and reasoning about a *student's* internal belief and reasoning structure to diagnose problems with their model.

Leake (1995) describes a framework for introspective reasoning under computational constraints in information retrieval. The knowledge represented in his framework is a superset of our student model: for example, his model includes knowledge about information needs and goals. Rosenblatt and Vera (1995) describe the GOMS (Goals, Methods, Operators, and Selection rules) framework for modeling mental states; the GOMS approach can be used to model either a system's internal knowledge for introspective reasoning, or a user's knowledge to do plan recognition and user modeling. Modeling the student's goals explicitly might add a useful dimension to our technique, by allowing us to understand the context in which the student is working. For example, a student who is learning new material will behave differently, and want different types of feedback, than a student who is reviewing previously learned material for a final exam.

Fox and Leake (1995) model case-based planning strategies as a set of expectations about system behavior. They use failures in expectations to trigger diagnosis and repair of failures, by identifying new indices for case retrieval. This failure-driven method is similar to our method for comparing the student and expert solutions to identify missing conceptual links (i.e., missing indices for case retrieval that we might wish to teach the student). Oehlmann and Edwards (1995) also use case-based techniques to do knowledge-level and introspective reasoning; their work focuses on re-use of regulation processes that plan and monitor cognitive activities.

Knowledge Representation and Reasoning

ML-Modeler uses a single representation for both the static expert knowledge and the dynamic student model. Knowledge is contained in a network of procedures, experiences, and concepts, in structures similar to memory organization packets (MOPs) (Kolodner, 1983; Schank, 1982). The learning mechanism consists of adding new experiences and concepts in the form of MOPs to the network, generalizing by collapsing the MOPs, and indexing and reorganizing the MOPs in the network.

Each MOP represents a problem, its solution, and the concepts used to solve it. The expert domain MOPs are used as a base against which to compare students' solutions, where deviations from the expert solution are considered to be either errors or misconceptions.

The expert MOPs are discriminated from one another by using both literal features and concepts as indices, causing some problems to be indexed in more than one place. Indexing by concepts represents the expert point of view, where functionality and basic concepts discriminate one problem from another. However, students often focus on the surface features of a problem. Thus, we also index the MOPs using the literal features of the problem, such as the values given in a problem statement. This provides a richer representation that ML-Modeler can draw upon to gain a deeper understanding of students' mistakes.

Problem 1: At 500K, 1.00 mol of ONCl(g) is introduced into a one-liter container. At equilibrium the ONCl(g) is 9.0% dissociated. Given the following equilibrium equation:

$$2\ ONCl\ (g) \leftrightarrow 2NO\ (g) + Cl_2\ (g)$$

Calculate the value of K_c for the equilibrium at 500K.

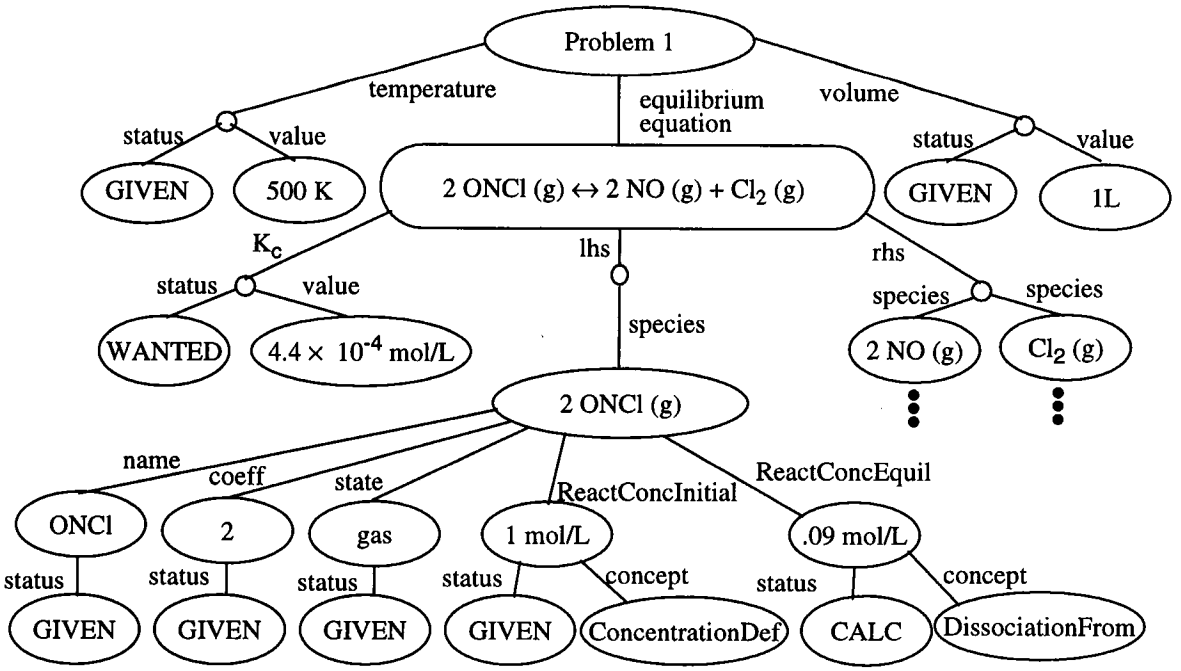

Figure 1: A chemical equilibrium problem and its corresponding conceptual network, where K_c is the equilibrium constant, lhs and rhs are left-hand-side and right-hand-side of the equation respectively, coeff is the coefficient before a species in the equation, ReactConcInitial and ReactConcEquil are the reactant's (ONCl's) concentration before and during equilibrium, GIVEN implies the value was given in the problem statement, and CALC implies the value was calculated during problem solving.

A student's knowledge state and learning mechanisms are represented using the same structure as the expert MOPs, where the indices include concepts, procedures, and learning mechanisms that ML-Modeler believes that the student is using with high probability (see the next section for a discussion of our representation of uncertainty). These indices change over time as the student progresses through a series of problems and instruction, reflecting the student's constantly changing knowledge and learning state.

Each student MOP represents a problem solving episode that consists of a conceptual network of literal features, facts, and conceptions and a set of equations and/or procedures. An example of a problem typical in the domain of chemical equilibrium is shown in Figure 1, along with its corresponding network. Figure 2 illustrates the series of equations that yield the correct solution.

The links in the conceptual network indicate state (e.g., temperature was given in the problem statement), factual knowledge (e.g., temperature is 500K), and concepts (e.g., an understanding of the dissociation of the reactant ONCl is needed to calculate its concentration at equilibrium).

The equations represent steps that are necessary to arrive at the solution. Each equation is broken into its constituent parts and associated with its corresponding concepts or procedures. For example, equation 1 corresponds to the procedure of taking the ratio of the concentrations of the reactants over the products, and the (100-9) portion of equation 2 corresponds to the concept of dissociation, in this case the dissociation of ONCl by 9%.

Initially, the student conceptual network consists of only the literal features contained in the expert conceptual network. As shown in the figure, this includes nodes that have the status of GIVEN. As the student solves a problem, appropriate links are added and updated along with annotations (e.g., possible explanations of misconceptions and pedagogical techniques used such as coaching or scaffolding), values calculated by the student, and a relative time stamp that allows tracking of each step of the student's solution.

ML-Modeler creates and generalizes new indices, enabling the system to represent the mental states and learning mechanisms of a novice-to-expert transition. In its initial state, ML-Modeler can only represent the student

1. $K_c = \frac{[NO]^2 [Cl_2]^1}{[ONCl]^2}$ (Coeff(NO), Coeff(Cl_2), Coeff(ONCl))

numerator = EquilEquProducts(rhs)

denominator = EquilEquReactants(lhs)

whole equ. = EquilConstantEqu

2. $[ONCl] = ((100-9)/100)\ (1 mol/L)$ (DissociationFrom(ONCl), PercentageCalc, Units(Conc))

whole equ. = ReactConcEquil(ONCl)

3. $[ONCl] = (0.91)\ (1 mol/L)$

whole equ. = Calc(Equ2)

4. $[ONCl] = 0.91 mol/L$

whole equ. = Calc(Equ3)

5. $[NO] = (9/100)\ (1/1)\ (mol/L)$ (Dissociation To(NO), Percentage Calc, CoeffRatio (NO, ONCl), Units (Conc))

6. $[NO] = 0.09 mol/L$

whole equ. = Calc(Equ5)

7. $[Cl_2] = (9/100)\ (1/2)\ (mol/L)$ (Dissociation To(Cl_2), Percentage Calc, CoeffRatio (Cl_2, ONCl), Units (Conc))

whole equ. = ProdConcEquil(Cl_2)

8. $[Cl_2] = 0.045 mol/L$

whole equ. = Calc(Equ7)

9. $K_c = \frac{(0.09 mol/L)^2 (0.045 mol/L)}{(0.91 mol/L)^2}$ (Substitute(Equ6), Substitute(Equ8), Substitute(Equ4))

whole equ. = Substitute(Equ1)

10. $K_c = 4.40 \times 10^{-4} mol/L$

Figure 2: The expert solution to the chemical equilibrium problem depicted in Figure 1. Each equation has its corresponding concepts attached to it.

as a single case or set of primitive procedures to choose from, using a generic metric of similarity on surface features. As experiences and concepts are added to the network, the system can use more abstract and functional features of a problem in order to index and refine the student MOPs of the net.

In addition to developing abstract concepts, the MOPs can be collapsed, corresponding to the compiling of procedures into general rules. These rules can be retrieved and directly applied to a problem, emulating skill acquisition.

Our approach provides a smooth integration of case-based learning and rule-based problem solving with the application of inductive reasoning for improving and refining the procedures and indices stored in the model.

Modeling the Student

To apply ML-Modeler to student modeling, the learning mechanisms, concepts, and procedures to be used must be guided by the student's problem-solving behavior over time. The output of this process is an evolving representation of the student's understanding of the domain.

The core student model consists of the most probable knowledge and learning mechanisms exhibited by the student. These are represented through the way the student MOPs are indexed, since a concept or learning mechanism must be highly probable before becoming an index. Both example problems and problems worked on by the student are represented in the core student model. In addition to the core model, a set of plausible inferences represent extensions of the core model that the student may have formed. These are represented in each MOP's internal conceptual network, where plausible inferences are attached to concept nodes.

ML-Modeler maintains a set of alternative hypotheses about the student's new mental state or what it believes the student may have learned during the most recent problem solving episode. Maintenance is accomplished through the use of a domain model, the current core model of the student, and an observable history of the student which includes traces of the student's problem solving behavior,

the student's answers to questions, questions the student has asked, and browsing the student has performed. Using this information, ML-Modeler updates the core student model by increasing or decreasing its confidence in appropriate plausible inferences, causing some to be added to the core model and others to be discarded.

The most probable concepts, learning mechanisms, and procedures are added to the core student model by adding them as indices to the student MOPs. Thus, when it is determined that it is probable the student understands a particular concept, that concept is added as an index to any MOP that contains that concept in its problem solving episode. In other words, the concept is updated globally throughout the MOP network.

Generalization can occur when two MOPs contain many similar indices. The similar components can be collapsed into a single MOP, with indices leading to any discriminating features below the "generalized" MOP. This may cause fragments of different problems to appear in a single MOP. This process can emulate a student's transition to more generalized problem solving approaches and concepts, while also modeling specialization or exceptions to the more generalized rules. In terms of the MOP structure, the more generalized concepts and rules surface to the top of the network, while the specialized cases sink towards the bottom.

The Modeling Process

Our ATS focuses on teaching the mathematical and highly conceptual domain of chemistry, specifically chemical equilibrium. Currently, we are focussing on quantitative, equation-rich problems. During the modeling process, each student's equation is matched to the expert solution equation set through a matching algorithm discussed in Gürer (1993), that represents how similar one equation is to another, where algebra, known values, and mathematics are taken into consideration.

The similarity between equations is represented by a likeness factor that ranges from 0 to 100, where 100 is a perfect match. The best match is taken, and ML-Modeler traces back from the matched equation to the originating equation, checking off the solution steps as it goes. This allows ML-Modeler to determine whether a mistake is likely to correspond to a concept or to a careless mathematical or an algebraic error. For example, suppose a student's equation matches expert equation 4 in Figure 2 with a likeness factor of 81 (e.g., the student writes [ONCl] = .89 mol/L). We can trace back to equation 3 and then again to equation 2. If none of these equations has been checked off yet, it is assumed that the mistake corresponds to a missing concept, in this case one or both of the concepts DissociationFrom and PercentageCalc. If equation 2 was checked off (i.e. the student used that equation previously), it is assumed the student has made a mathematical calculation error.

When a mathematical or algebraic error is detected, ML-Modeler simply notes the mistake and coaches the student on his or her error. On the other hand, when a concept is involved, ML-Modeler attempts a deeper understanding of the student's mistake through modeling case-based reasoning.

First the core model is searched to ascertain whether the student has already exhibited some case-based learning strategies using the concept in question. ML-Modeler then produces a set of plausible inferences by perturbing a copy of the core student model. If one of these matches the student's activity, it is attached to the concept in the conceptual network contained in the current problem's MOP. The ATS can use this, along with other information contained in the core student model, to make an instructional decision.

Our research direction is leading to perturbations of the core student model being done through the use of analogy and other learning mechanisms. If a case is found with the same surface features, a direct analogy is drawn and the problem is re-solved to that point using the incorrect analogy. If the solution matches the student's mistake, it is highly likely that the student is focusing on the literal features of the problem rather than the conceptual features.

Two other types of perturbation we may employ are overgeneralization and incorrect use of analogy. The student may have remembered a previous problem (either an example that was presented to them or a problem that they solved earlier) with the same conceptual structure. However, the student may have applied the analogy incorrectly or inappropriately applied the analogy to the current problem which is a specialized case. These can be emulated through applying the conceptual aspects of the analogous case to the new problem (where it is not appropriate to do so).

Overspecialization can also occur when a student does not realize he or she can apply a general rule or case to the current problem. This can be determined by inspecting the placement and representation of the current problem's expert MOP through its concept indices (as opposed to its literal indices).

These perturbations allow our ATS to model not only the errors and conceptual areas the student is having difficulty with, but also possible incorrect uses of generalization, specialization, and analogy.

Representing Uncertainty

Fuzzy methods (Zadeh, 1965) are used to evaluate the plausible alternatives in ML-Modeler; the most plausible heuristics and concepts (i.e., those that best predict the student's behavior) are attributed to the student. We chose to use fuzzy techniques to represent the uncertainty in the student model, rather than Bayesian or evidential techniques, for a number of reasons:

1. Students and (human) teachers interacting with the system are more comfortable assigning and interpreting

fuzzy membership categories than point probabilities.

2. The prior probabilities and conditional probabilities required by Bayesian methods would be difficult to estimate, and fuzzy methods provide a straightforward and low-complexity alternative.
3. The results given by fuzzy techniques will yield a good first approximation to Bayesian probabilities.
4. The fuzzy representation and inference techniques could be replaced with Bayesian or evidential methods without requiring significant modifications to the rest of the model, so we can explore alternative methods in the future.

The basic type of uncertainty that we need to represent is the probability that the student knows a particular concept, given that they perform a particular step in their solution. We can write this uncertainty as a conditional probability,

$$P(C \mid S)$$

where C represents the state in which the student knows concept C and S represents the fact that the student has performed solution step S. Using Bayes' rule, we can rewrite this probability as

$$P(C|S) = \frac{P(C) \cdot P(S|C)}{P(C)}$$

$P(C)$, the prior probability that the student knows the concept, can be approximated based on a profile of the student and the student's past performance: a new student with no knowledge of chemistry is very unlikely to know the given concept; a student who has solved, or been shown, similar problems in the tutoring system is likely to know it.

Knowing a concept is different from being able to apply it, however. The second term in the equation above, $P(S|C)$, represents the probability that a student performs a particular solution step, given that they know the concept. This can be quite difficult to estimate, particularly for dynamically generated problems where there is not a history of statistical data to draw on. The factors that determine this probability include whether the concept is used in, or relevant to, the solution step (if it is not, then $P(S|C)$ is just the same as $P(S)$ since knowing C will be independent of applying S); how good the student has been historically at applying their knowledge; and whether the concept is likely to be retrieved at the appropriate time.

Computing $P(S)$ is also problematic; again, this is confounded by the dynamic nature of the tutoring situation, where statistics cannot realistically be kept for each concept and each solution step. This probability represents the probability that a random student would use this solution. One way to estimate this probability is to rewrite it, again using several applications of Bayes' rule, as

$$P(S) = P(S|C) \cdot P(C) + P(S| \neg C) \cdot P(\neg C)$$

Notice that we have computed all of these probabilities above, except for $P(S| \neg C)$, which will be computed similarly to $P(S|C)$.

Because of the complexity involved, although it might be possible to get better results if a good model for computing the Bayesian probabilities were available, we have decided to simplify this aspect of the student modeling problem by using fuzzy set memberships to represent uncertainty.

The fuzzy methods we propose to use were adapted from the techniques used in Sherlock II (Katz *et al.*, 1993). In Sherlock II, fuzzy probability distributions (fpds) are associated with knowledge variables that correspond to concepts in our network. We also assign fpds to links in the network, and can use the relationships between concepts in the network to help compute the fpds.

ML-Modeler uses seven values (definitely, probably, possibly, maybe, possibly not, probably not, definitely not) in the fpds that are attached to each concept and link in the student model network. An fpd can thus be represented as a 7-tuple $(p_1, p_2, p_3, p_4, p_5, p_6, p_7)$ where p_1 is the probability mass associated with the value *definitely not* and p_7 is the probability mass associated with the value *definitely*.

Initial values for the fpds will depend on the source of the knowledge, i.e., on why they were added to the student model. Concepts and links that were added during the translation of the student's solution steps into the network representation will be biased toward the upper end (e.g., the fpd might be (60, 30, 10, 0, 0, 0, 0)). Concepts that are added as the result of a plausible hypothesis formed by ML-Modeler will initially be "unknown" ($\frac{1}{7}$, $\frac{1}{7}$, $\frac{1}{7}$, $\frac{1}{7}$, $\frac{1}{7}$, $\frac{1}{7}$, $\frac{1}{7}$). These concepts will then be upgraded or downgraded using updating rules based on those used in Sherlock II. These updating rules "shift" the mass of the probability toward the more probable (for upgrading) or less probable (for downgrading) ends of the distribution. The upgrading rule is

$$p_1 = p_1 - p_1 c$$

$$p_i = p_i - p_i c + p_{i-1} c$$

$$p_7 = p_7 + p_6 c$$

where c is a constant that controls the rate at which upgrading occurs. [2] Similarly, the downgrading rule is

$$p_1 = p_1 + p_2 c$$

$$p_i = p_i - p_i c + p_{i+1} c$$

$$p_7 = p_7 - p_7 c$$

For example, the hypothesis that the student has used case-based reasoning to apply the solution of a previous problem to this problem would cause the system to add a

2. Sherlock II uses a *range vector* to further control the speed at which downgrading and upgrading occurs; for simplicity, we omit this here.

number of concepts and links that represent this case-based method. These concepts and links would be initialized with an fpd indicating that their probability is unknown. They would then be analyzed for their consistency with the existing student model and with the evidence provided by the student's behavior. Concepts and links in the new hypothesis that are inconsistent (e.g., a link indicating familiarity with a concept that the student has not yet learned) will be downgraded. Those that are supported (e.g., an additional concept that is used in the student's next solution step) will be upgraded.

An Example

The following example illustrates how ML-Modeler works. In order to develop a realistic scenario, high school chemistry students were given a series of chemical equilibrium problems to solve. This process was videotaped and the protocols analyzed. The students were given the example problem shown in Figure 3, and were provided with an explanation of the constant of equilibrium equation and some basic concepts pertaining to equilibrium. The students were then given Problem 1, shown in Figure 1, to solve. A combination of two typical mistakes exhibited by the students is shown in the resulting solution of Figure 4.

For the reaction

$$N_2O_4(g) \rightleftharpoons 2\,NO_2(g)$$

the concentrations of the substances present in an equilibrium mixture at 25° C are

$[N_2O_4] = 4.27 \times 10^{-2}$ mol/L

$[NO_2] = 1.41 \times 10^{-2}$ mol/L

What is the value of K_c for this temperature?

SOLUTION:

$$K_c = \frac{[NO_2]^2}{[N_2O_4]} = \frac{(1.41 \times 10^{-2}\ \text{mol/L})^2}{(4.27 \times 10^{-2}\ \text{mol/L})}$$

$$= 4.66 \times 10^{-3}\ \text{mol/L}$$

Figure 3: The example problem and solution given to the student before solving other problems.

In analyzing this solution, ML-Modeler sequentially analyzes the student's equations in the order presented. Student equation 1 is a perfect match to expert equation 6. This is traced back to expert equation 5 which has the DissociationTo, PercentageCalc, CoeffRatio, and Units concepts. Expert equations 5 and 6 are checked off and all four concepts, which we suppose were initially "unknown," are upgraded using $c = 0.25$ to (.1, .14, .14, .14, .14, .14, .18)

Student equation 2 is also a perfect match to expert equation 7 with the numbers multiplied out. This causes expert equation 7 to be checked and concepts DissociationTo, PercentageCalc, CoeffRatio, and Units to be further upgraded to (.08, .13, .14, .14, .14, .14, .22).

1. $[NO] = 0.09 mol/L$

2. $[Cl_2] = \frac{0.09 mol/L}{2}$

3. $K_c = \frac{(0.045 mol/L)\ (0.09\ mol/L)^2}{(1\ mol/L)^2}$

Figure 4: A student's solution to the chemistry problem given in Figure 1.

Student equation 3 is a non-perfect match to expert equation 9. At this point, ML-Modeler can only recognize that the student has substituted the wrong number (1 mol/L instead of .91 mol/L) and this is traced back to equation 2, causing expert equations 2, 3, 4, and 9 to be checked. The discrepancy corresponds to the concepts DissociationFrom, PercentageCalc, and Units. As a result, DissociationFrom is downgraded one step from "unknown" to (.18, .14, .14, .14, .14, .14, .1) and PercentageCalc is downgraded back to (.1, .14, .14, .14, .14, .14, .18). Units, which was applied correctly, is upgraded again, yielding the distribution (.06, .11, .13, .14, .14, .14, .25).

Notice that the weight on Units is gradually shifting to the more probable values: that is, the Units concept is becoming more likely. If we used a different learning constant c, this shift would happen at a different rate. For example, if we used $c = 0.95$, the probability distribution for Units after three upgrades would be (0, 0, .03, .14, .14, .14, .53).

Before further analyzing this misconception, ML-Modeler analyzes what was done correctly. In this case the student used the correct numbers in the numerator. These can be traced back to expert equations 6 and 8, which is further traced back to equation 7. Equations 6 and 7 were already checked off, so the concepts corresponding to these equations are *not* upgraded, and equation 8 is checked off. In addition, expert equation 9 traces back to expert equation 1, causing equation 1 to be checked and the concepts and procedures EquilEquProducts, EquilEquReactants, and EquilConstantEqu to be upgraded once and Coeff upgraded three times.

At this point there is evidence that the student's mistake corresponds to DissociationFrom since there is prior evidence that he or she understands PercentageCalc and Units. However, ML-Modeler now attempts to apply case-based reasoning techniques to come to a deeper understanding of the student's mistake.

First the student's MOP network is analyzed for previ-

ously observed learning mechanisms, but not much is useful at this point since the MOP network only contains the example case and the current problem. Since the example problem does not have the same conceptual structure as the current problem, overgeneralization and overspecialization are not attempted. An analogy based on surface features is attempted.

ML-Modeler searches the student and expert MOP networks for cases or problems that have the same surface features as the current problem. The example problem is found to have many of the same literal features (e.g. the equilibrium constant equation is used where product concentrations appear in the numerator and reactant concentrations appear in the denominator, and concentrations given in the problem statement are placed in the equilibrium equation).

ML-Modeler performs a 1-1 mapping from the example case to the current solution, keeping in mind the location of the student's error. Since the student placed the product and reactant concentrations in the correct spots, this commonality is not pursued. However, the student did place the given value for a concentration directly into the equilibrium equation, which is what the example problem did. ML-Modeler performs this step, re-solves the expert solution up to the match (equation 9), and compares the results to student equation 3. There is an exact match and the inference that the student was focusing on surface features is created, assigned a highly probable value, and attached to the corresponding component of the student's conceptual network. A further analysis compares the differences between the value the student used and the correct value. This difference is found to be directly related to what the concentrations are before and after equilibrium and the dissociation of ONCl.

The ATS may want to intervene at this point or further analyze the rest of student's solution. There is enough information for the ATS to prioritize focusing the student on the dissociation concept and explaining the difference between the before and during states of equilibrium.

If ML-Modeler continues to analyze the student's solution, it will find a non-exact match of student equation 4 to expert equation 10. This is traced back to expert equation 9 which has already been checked off. Therefore, this mistake is attributed to a calculation error. Note that since the expert solution was re-solved with the student's previous mistake, the error is recognized as not matching $[(.045)(.09)^2]/(1)^2 = 3.645\text{x}10^{-4}$ mol/L not the original expert solution of $4.40\text{x}10^{-4}$ mol/L (see Gürer, 1993 for more details of re-solving).

Discussion

Currently, student models can be categorized into two varieties: overlay, where a student's knowledge is represented as a subset of an expert model, and model tracing where a student's concepts are represented as rules and mistakes are attributed to buggy procedural rules. Like the overlay model, ML-Modeler is able to identify concepts and procedures a student understands and those that are misconceptions. In addition, ML-Modeler attempts to perturb the student and/or expert model in order to determine a deeper understanding of the student's misconceptions.

Previous perturbation models have focused on perturbing procedural rules and attributing these to a student's mistakes (Langley, Wogulis & Ohlsson, 1990; Sleeman et al., 1990). A problem with perturbing procedural rules, is that there is no deeper cognitive representation of a student's mistakes. We believe that ML-Modeler can go beyond procedural rule perturbation to modeling a student's mistakes through analogy, overgeneralization, and overspecialization. In addition to recognizing incorrectly performed procedures or rules, ML-Modeler can identify general problem solving strategies and deeper misconceptions.

These capabilities can give an ATS the ability to respond more accurately to a student's misconceptions or knowledge gaps. Explanations are made easier through the conceptual nature of the case-based representation and processes. The ATS can dynamically generate new problems for the student to solve based on the case-based structure. Testing of the student's ability to transfer concepts can be done by creating problems that use the same abstract concepts as previous problems, but contain different surface features.

A record of the ATS's pedagogical interactions with the student is kept by attaching instructional decisions and the student's responses to the concept in the conceptual network of a MOP. In general, knowledge of the student's problem solving approach (e.g., focusing on surface features and not basic concepts) can further guide an ATS in its interactions with a student.

For the reasons stated above, we believe that ML-Modeler has much promise. We are currently implementing the ideas described in this paper. One area that we believe will prove difficult is the potential for computational complexity during cased-based perturbation. When the model contains a considerable case-library and student history, ML-Modeler will need some search rules to guide it in making likely perturbations and skipping the unlikely ones.

References

Anderson, J.R. (1983). *The architecture of cognition.* Harvard University Press.

Chan, T., Chee, Y.S. & Lim, E.L. (1992). COGNITIO: An extended computational theory of cognition. In *Proceedings of the 2nd International Conference on Intelligent Tutoring Systems* (pp. 244-251). New York, NY: Springer-Verlag.

Chi, M.T.H., Glaser, R. & Rees, E. (1982). Expertise in problem solving. In Sternberg, R.J. (Ed.), *Advances in the psychology of human intelligence,* Vol. 1. Hillsdale,

NJ: Lawrence Erlbaum Associates.

Elio, R. & Scharf, P.B. (1990). Modeling novice-to-expert shifts in problem-solving strategy and knowledge representation. *Cognitive Science*, 14, 579-639.

Fox, S. & Leake, D.B. (1995). Modeling case-based planning for repairing reasoning failures. In *Working Notes of the AAAI-95 Spring Symposium on Representing Mental States and Mechanisms*. AAAI Press, March.

Gürer, D.W. (1993). *A Bi-Level Physics Student Diagnostic Utilizing Cognitive Models for an Intelligent Tutoring System.* Doctoral Dissertation. Bethlehem: Lehigh University, Electrical Engineering and Computer Science Department.

Katz, S., Lesgold, A., Eggan, G. & Gordin, M. (1993). Modelling the student in Sherlock II. *Journal of Artificial Intelligence in Education*, 3(4), 495-518.

Kolodner, J.L. (1983). Maintaining organization in a dynamic long-term memory. *Cognitive Science*, 7, 243-280.

Langley, P., Wogulis, J. & Ohlsson, S. (1990). Rules and principles in cognitive diagnosis. In Frederiksen, N., Glaser, R., Lesgold, A. & Shafto, M. (Eds.), *Diagnostic monitoring of skill and knowledge acquisition* (pp. 217-250). Hillsdale, NJ: Lawrence Erlbaum Associates.

Leake, D.B. (1995). Representing self-knowledge for introspection about memory search. In *Working Notes of the AAAI-95 Spring Symposium on Representing Mental States and Mechanisms*. AAAI Press, March.

Martin, J. & VanLehn, K. (1993). OLAE: progress toward a multi-activity, Bayesian student modeler. In *Proceedings of the 1993 World Conference on AI and Education* (pp. 426-432). Association for the Advancement of Computing in Education.

Möbus, C., Schröder, O. & Thole, H.J. (1993). A model of the acquisition and improvement of domain knowledge for functional programming. *Journal of Artificial Intelligence in Education*, 3(4), 449-476.

Newell, A. (1990). *Unified theory of cognition.* Harvard University Press.

Pirolli, P. & Recker, M. (1994). Learning strategies and transfer in the domain of programming. *Cognition and Instruction*, 12(3), 235-275

Rosenblatt, J.K. & Vera, A.H. (1995). A GOMS representation of tasks for intelligent agents. In *Working Notes of the AAAI-95 Spring Symposium on Representing Mental States and Mechanisms*. AAAI Press, March.

Ross, B.H. (1989). Some psychological results on case-based reasoning. In Hammond, K. (Ed.), *Proceedings of the workshop on case-based reasoning (DARPA).* Pensacola Beach, Florida: Morgan Kaufmann.

Schank, R.C. (1982). Dynamic memory. Cambridge: Cambridge University Press.

Sleeman, D., Hirsch, H., Ellery, I. & Kim, I.Y. (1990). Extending domain theories: two case studies in student modeling. *Machine Learning*, 5, 11-37.

Zadeh, L. A. Fuzzy sets. (1965). *Information and Control*, 8, 338–353.