# Building a 3rd Generation ITS for Symbolization: Adding a Tutorial Model with Multiple Tutorial Strategies

**Neil T. Heffernan (neil@cs.cmu.edu)**
**Kenneth R. Koedinger (koedinger@cmu.edu)**
School of Computer Science
Carnegie Mellon University
**Pittsburgh, PA 15213**

**Abstract**

Symbolization is the ability to translate a real world situation into the language of algebra. We believe that symbolization is the single most important skill students learn in high school algebra. We present research on what makes this skill difficult and report the discovery of a "hidden" skill in symbolization. Contrary to past research that has emphasized that symbolization is difficult due to both comprehension difficulties and the abstract nature of variables, we found that symbolization is difficult because it is the articulation in the "foreign" language of "algebra". We also present *Miss Lindquist*, an Intelligent Tutoring System (ITS) designed to carry on a tutorial dialog about symbolization. Miss Lindquist has a separate tutorial model encoding pedagogical content knowledge in the form of different tutorial strategies that was partially developed by observing an experienced human tutor. We discuss aspects of this human tutor's method that can be modeled well by Miss Lindquist and share examples of what she can do. Through testing, we plan to learn about the relative effectiveness of the different tutorial strategies Miss Lindquist uses.

## Introduction

The mission of the Center for Interdisciplinary Research on Constructive Learning Environments (CIRCLE) is 1) to study human tutoring and 2) to build and test a new generation of tutoring systems that encourage students to construct the target knowledge instead of telling it to them (VanLehn et. al., 1998). CAI (Computer Aided Instruction) systems were 1st generation tutors. They presented a page of text or graphics and depending upon the student's answer, put up a different page. Model tracing ITSs are 2nd generation tutoring systems that allow the tutor to follow the line of reasoning of the student. ITS have had notable success (Koedinger et. al., 1997) despite the fact that human tutoring can look very different (Moore, 1996). One way they are different is that there is a better sense of a dialog in human tutoring and maybe this is important. After analyzing over 100 hours of untrained tutors in naturalistic tutoring sessions Graesser et. al. (in press) believe "there is something about interactive discourse that is responsible for learning gains."

The members of CIRCLE are working on 3rd generation tutoring system that are meant to engage in a dialog with students, using multiple strategies to allow students to construct their own knowledge of the domain. We are building a new ITS, called Miss Lindquist, that will not only be able to model trace the student's actions, but can be more human-like in carrying on a running conversation, complete with probing questions, positive and negative feedback, follow-up questions in embedded sub-dialogs, and requests for explanation as to why something is correct. In order to build Miss Lindquist we have expanded the model tracing paradigm so that Miss Lindquist not only has a model of the student, but also has a model of tutorial reasoning (e.g. Clancey, 1982). Based on observation of an experienced tutor and cognitive research, this tutorial model has multiple tutorial strategies at its disposal.

The task domain we are working on is symbolization, which is the task of writing an algebraic expression given a real-world problem context, often presented in the form of a word problem. An example problem is shown in Figure 1. Symbolization is important because if students can't translate problems into algebra, they will not be able to apply algebra to solve real world problems. This domain makes it easy to avoid some difficult natural language issues because we can ask students to write algebraic expressions and those expressions are easy for the computer to "understand". We take advantage of this property of the domain to avoid any serious natural language processing. We do ask students to formulate explanations in words, but in such cases we provide them pull down menus to construct their sentences. Instead, we focus our energies on modeling tutorial reasoning which includes capturing the *pedagogical content knowledge (Shulman, 1986)* of an experienced human tutor. Pedagogical content knowledge is the knowledge that a good tutor has about how to teach a specific skill. A good tutor is not just one who knows the domain, nor is it simply one who knows generally tutoring rules. A good tutor is one who also has content specific strategies that can help a student overcome common difficulties. We have set out to observe and model some of these strategies for our specific domain of symbolization. Lets look at one example of a content-specific pedagogical strategy for symbolization. The following was collected and transcribed from a one hour long one-on-one tutoring session between an experienced human tutor and an eighth grade student working on the "bike-trip" problem, which we use as one of several running examples.

240. Student: [reads problem[1]] Cathy took a "m" mile bike ride. She rode at a speed of "s" miles per hour. She stopped for a "b" hour break. Write an expression for how long the trip took.
241. S: uhm [ writes "s/m+b" but should be "m/s+b"]
242. Tutor: How do you calculate the amount of time it takes you? If you're, if you're, if you're riding at, let's make it simple. If you are riding at 20 miles per hour, OK, and you go 100 miles, how many hours did that take you?
243. S: Um 5
244. T: 5. And how did you get that 5? How did you use the numbers 100 and …
245. S: 100 miles divided by miles per hour
246. T: So you took the miles and divided it by the [garbled, but possibly "speed"]
247. S: Miles divided by s plus b equals time [writes m/s+b ]
248. T: Right.

The tutor in the above dialog, appears to have done two things to scaffold this problem. First, the tutor focused on the problem of calculating the time actually on the bikes by decomposing what had been a problem with two operators into a problem that had only one operator. Presumably this is because the student indicated he understood that the goal quantity was found by adding for the amount of the break ("b") to the time actually on the bikes.

The second scaffolding move the tutor did was to change the problem question from a symbolization question to a presumably simpler *compute question* by asking the student to calculate the speed using 100 and 20 rather than "m" and "s". Then in line 244 the tutor asked for the *articulation step* of "How did you get that 5?" Finally, the student is prompted for the *generalization step* of writing the expression using variables.

Our experienced tutor often invited the student to use concrete numbers. We call this strategy the *concrete articulation strategy (Koedinger & Anderson, 1998[2]).* McArthur et. al. (1990) also observed that human tutors often used what he called *curriculum scripts* and *micro-plans*, which often involved a series of questions designed to remediate particular difficulties. We call these scripts *knowledge construction dialogs* to emphasis the fact that that we are trying to build a tutor that encourages students to build their own knowledge by less often *telling* them a hint and more often *asking* them a question.

The impediments to building a third generation tutor is not just technical. We think that if you want to build a good ITS for a domain you need to:

- Study what makes that domain difficult, including discovering any hidden skills, as well as determining what types of errors students make.
- Construct a theory of how students solve these problem. (We instantiated that theory in a cognitive model.)
- Observe experienced human tutors to find out what pedagogical content knowledge they have and then build a tutorial model that, with the help of the theory of domain skills, can capture and reproduce some of that knowledge.

We look at these each of these steps in turn.

# What Makes Symbolization Difficult?

Symbolization is a difficult task for students. For example, one month into an algebra class only 13% of students could answer the symbolization problem in the caption to Figure 1. To determine what makes symbolization difficult we conducted two *difficulty factors assessments* (Koedinger & MacLaren, 1997) which are paper and pencil tests that we gave to groups of 80+ students (Heffernan & Koedinger, 1997 and 1998). First, we identified three hypotheses about what makes symbolization difficult.

The first of these is the *comprehension hypothesis*. Much of the prior research (Cummins et. al., 1988; LeBlanc & Weber-Russell, 1996; Lewis & Mayer, 1987; Paige & Simon, 1979) on word problem solving has focused on students' comprehension abilities. Cummins et. al. "suggest that much of the difficulty children experience with word problems can be attributed to difficulty in comprehending abstract or ambiguous language." The general conclusion from the above research is that comprehension rules are key knowledge components students must acquire to become competent problem solvers.

A second hypothesis is the *generalization hypothesis*. According to this hypothesis, symbolization is difficult because students must learn how to use variables to generalize arithmetic procedures..

More recent research by Koedinger and Anderson (1998), and which we confirmed (Heffernan & Koedinger, 1997 and 1998), showed that students could comprehend many problems well enough to find a numerical answer, but they nevertheless failed to correctly symbolize. Although this refutes the comprehension hypothesis it does not refute the generalization hypothesis because the symbolization problems had variables in them. Therefore, we compared students' ability to symbolize a problem that contained a variable (with an answer like "800-40m") to their ability to symbolize a problem with just constants. In the "constants" case the students were asked to write an expression for their answer (i.e. "800-40*3") instead of finding a numerical solution (like "680"). Even if we counted as correct the very few students who did not follow the directions and evaluated the answer, we found that the presence of the variable in the problem did not make problems more difficult. Therefore, the generalization hypothesis was refuted.

So what can explain why symbolization is so difficult? We propose the *articulation hypothesis* which suggests that there is a "hidden" skill that is not obvious to most teachers and researchers. The hidden skill is the ability to produce symbolic sentences in the language of algebra. It appears that many students are able to figure out all the

[1] Throughout this paper, text in square brackets are comments, and S and T stand for "student" and "tutor" respectfully.
[2] Then called the *inductive support* strategy.

conceptual relations in a problem, but are not able to express those relationships in algebra. If we asked students to translate a story written in English into Greek we would not be surprised if many fail because they don't know Greek. But teachers and researchers often fail to realize that algebra too is a language. And a language that students have had relatively little practice in "speaking" By "speaking" we mean producing sentences of symbols, not verbalizing. This was demonstrated anecdotally by one of our students who when asked to symbolize a problem with the answer of "(72-m)/4" responded with "72-m=n/4=". Many commentators have noted that students will incorrectly use an equal sign in a way that makes sense if "=" means "results in." Sfard et. al. (1993) gives the following example "3*4=12-5=7." Another example is the student who when working on a problem with an answer of "550/(h-2)" answered with

h-2 h)$\overline{550}$

This student means to suggest that first she would subtract 2 from "h." The arrow seems to indicate that this new decremented value of h should be assigned back to the symbol "h". Then 550 should be divided (indicated with the grade school way of expressing division) by this new value of "h." Both of these examples indicate students who probably understand the quantitative structure and the sequence of operations that should happen, but nevertheless, failed to express that structure in normative algebra. What does such a student need to learn? A computer scientist or linguist might say that the student needs to learn the correct grammar for algebraic expressions. The novice student knows how to write one-operator expression like "5+7" using the following simple grammar:

<expression> = <literal> <operator> <literal>
<literal> = 1|2|3|4….
<operator> = "+" | "-" | "*" | "/"

But the competent student knows how to write multiple operator expression indicated by these grammar rules:

<expression> = <expression> <operator> <expression> | "(" <expression> ")" | <literal>

Phrased differently, what the student needs to be told is that "You can always wrap parentheses around an expression and substitute an expression anywhere you normally think a number can go. There are also rules for when you can leave out the parenthesis but you can always put them in to be sure that your expression won't be misinterpreted."

We found experimental evidence that supports the articulation hypothesis when we performed the following manipulation (Heffernan & Koedinger, 1997 and 1998). We started with a two-operator problem, like

> **Composed**: Ann is in a rowboat in a lake. She is 800 yards from the dock. She then rows for "m" minutes back towards the dock. Ann rows at a speed of 40 yards per minute. Write an expression for Ann's distance from the dock.

and decomposed the problem into two new separate questions like the following.

> **Decomposed**: A) Ann is in a rowboat in a lake. She is 800 yards from the dock. She then rows "y" yards back towards the dock. Write an expression for Ann's distance from the dock.
>
> B) Ann is in a rowboat in a lake. She then rows for "m" minutes back towards the dock. Ann rows at a speed of 40 yards per minute. Write an expression for the distance Ann has rowed.

Then we compared the ability of a student to answer the composed problem with their ability to get both decomposed parts correct. We found that the composed problems were much harder. Why? We speculated that many students could not compose the two decomposed expressions together; just because you know that you need to first add two quantities together and then multiply them by a number, doesn't mean you know how to express this correctly in the language of algebra. The following is an example of a student who appeared to be missing just this skill of composing expressions together. This example occurred while the first author was tutoring a student on the following "two-jobs" problem:

> T: Debbie has two jobs over the summer. At one job she bags groceries at Giant Eagle and gets paid 5 dollars an hour. At the other job she delivers newspapers and gets paid 7 dollars an hour. She works a total of 30 hours a week. She works "g" hours bagging groceries. Write an expression for the total amount she earns a week. [the correct answer is "5g+7(30-g)"]
>
> S: A=5*g, B=30-g, C=7*B and D=A+C

This student clearly understands the 4 math operations that need to be performed, and the order in which to perform them. This student spontaneously introduced new variables (A, B, C, and D) to stand for the intermediate results. We were surprised to find that this student could not easily put this together and write "5g+7(30-g)". This student appears to be ready for a strategy that will help him on just one skill; combining expressions by substitution. (We also turn this idea into a tutoring strategy which is presented below in the section on *Tutorial Strategies*.)

To see if substitution really is a hidden component skill in symbolization, we designed the following transfer experiment. Thirty-nine students were given one hour of group instruction on algebraic substitution problems like the following:

> Let X= 72-m. Let B= X/4. Write a new expression for B that combines these two steps.

The student were guided in practicing this skill. The students got better at this skill, but that is not the interesting part. By comparing pre-tests and post-tests, we found statistically significant increases in the students ability to do symbolization problems, even though they did not get instruction involving word problems! The students transferred knowledge of the skill of substitution to the skill of symbolization revealing a shared skill of being able to "speak" complicated (more than one-operator) sentences in the foreign language of algebra. This is strong supporting evidence for the articulation hypothesis.

This research has put a new focus on the production side of the translation process. This work also has ramifications for sequencing in the algebra curriculum. If learning how to do algebraic substitution involves a sub-skill of symbolization, perhaps algebraic substitution should be taught much earlier. In many curriculums (e.g. Larson, 1995)

it is not taught until students get to *systems of equations* half-way through the year .

## Cognitive Student Model

Our student model is similar to traditional student models. We use the Turtle (Anderson & Pelletier, 1991) production system, which is a simplification of the ACT (Anderson, 1993) Theory of Cognition. A production system is a group of if-then rules operating on a set of what are called *working memory elements (wmes).* We use these rules to model the cognitive steps a student could use to solve a problem. Our student model has 68 production rules. Our production system can solve a problem by being given a set of wme that encodes the problem at a high level. Figure 1 shows an example involving 5 quantity wmes and 2 relation wmes.

We model the common errors that students make with a set of "buggy" productions. From our data, we compiled a list of student errors and analyzed what were the common errors. We found that the following list of errors was able to account of over 75% of the errors that students made. We illustrate the errors in the context of the "two-jobs" problem which has a correct answer of "5g+7(30-g)".

1) Wrong operator (e.g. "5g-7(30-g)")
2) Wrong order of arguments (e.g. "5g+7(g-30)")
3) Missing parentheses (e.g. "5g+7*30-g")
4) Confusing quantities (e.g. "7g+5(30-g)")
5) Missing a component (e.g. "5g+7g" or "g+7(30-g)" or "5g+30-g")
6) Omission: correct for a subgoal. (e.g. "7(30-g)" or "5g")
7) Combinations of errors (e.g. "5g+7*g-30" has the wrong order for "g-30" and is missing parenthesis)

These "buggy" productions are used to allow us to make sense of a student's input even if she has made several incorrect steps. We don't want a computer system that can't understand a student if she gives an answer that has parts that are completely correct and parts that are wrong. We want the system to be able to understand as much as possible of what a student says and be able to give positive feedback even when the overall answer to a question might be incorrect.

Traditional model tracing tutors have a bug message attached to each buggy production that generates a message through the use of a template. We do not do that. We feel such an architecture confuses student reasoning with tutorial reasoning. We instead have the student model report its full diagnosis (which is represented with a set of wmes) to the tutor model that will then decide what to do.

If the student makes several errors, traditional model tracing tutors are sometimes in a quandary as to what to do. Some ITSs do not deal with multiple bugs and instead rely on breaking down the problem into finer steps. A problem with this approach is that you can't break down a skill like symbolization easily without decreasing the overall difficulty. Another solution is to ask the student what the subgoals should be and then tutor them on the subgoals individually (Corbett & Anderson, 1995.) However, a problem remains about what the ITS should do if the student makes more than one distinct error in a given input. This is addressed below.

## The Tutorial Model

As mentioned already, we collected and transcribed one hour of experienced human tutoring. We wanted to observe what experienced tutoring in this domain looked like. The tutor worked as a full time math tutor for over a year before teaching middle school math for 5 years. She was given a list of symbolization problems and told her goal was get the student to learn how to solve such problems.

After transcribing the dialog we have been able to extract some regularities in term of the tutorial strategies. One caveat: our tutorial model is informed by this observation of human tutoring, but it doesn't model any one individual or make claims to being the most effective model. The model we have come up with has multiple strategies it can choose among. Through empirical study, we plan to learn which strategies are most effective.

Now we will look at the components of the tutorial model in Figure 2. It is composed of a tutorial agenda component, as well as tutorial questions that can be used alone or in combination to make a tutorial strategy. The model is implemented with 77 productions. First, we deal with how Miss Lindquist decides what to focus problem attention upon.

### *Dealing with the diagnosis: The Focusing Heuristic*

Miss Lindquist uses a heuristic to decide what to focus the conversation on. In cases when the student model's diagnosis indicates that the student had some correct elements and some incorrect elements. For instance, we considered giving the following positive feedback on an answer like that in line 242 : "Your answer of 's/m+b' has some correct elements; it is true that you need to add the time of the break to the time on the bikes to find the total trip time." This feedback was meant to confirm the "+b" portion of the answer. After looking at what our human tutor did we decided not to give positive feedback unless the student has two operands correct and the correct operator. We give an example of this in the context of the "two-jobs."

T: [problem with answer of 5g+7*(30-g)]
S: 5g+7*g
T: No, but, 5*g does represent the amount Debbie earned bagging groceries. Let me ask you a simpler question. Can you tell me how much she made delivering newspapers?

If the student has made more than one error, the tutor decides to come up with a strategy to deal with each error. The errors are considered in the order they would be encountered in a post-order traversal of the parse tree of the correct answer (i.e visited "bottom-up.") Therefore, the tutor might add multiple questions to the tutorial agenda depending upon the tutorial strategy selected for each error.

If a student says something the student model doesn't understand (e.g. says "5/30-5*7/g" when the answer is "5g+7(30-g)") we will still want a robust ITS to be able to pick a reasonable strategy for a response. This is important because many times the tutor (humans or computers) will not be able to make sense of the student's input. Graesser et. al. (in press) reports in their study of human tutors that they "found that the human tutors and learners have a remarkably incomplete understanding of each other's knowledge base and that many of each other's contributions are not deeply understood… Most tutors have only an approximate assessment of the quality of student contributions." We want our ITS to be able to operate under these same difficult conditions and still be robust enough to say something reasonable.

### Tutorial Agenda

Miss Lindquist has a data structure we called the agenda, that stores the ideas she wants to talk about next. This agenda ordinarily operates like a push down stack, but we give an example of when the stack order is violated below in the section on the Concrete Articulation Strategy.

### Tutorial Questions

The tutorial model can ask the following kinds of tutorial questions illustrated with an example of how the question can be phrased:

1) Q_symb : Symbolize a given quantity ("Write an expression for the distance Anne has rowed?")
2) Q_compute: Find a numerical answer ("Compute the distance Anne has rowed?")
3) Q_explain: Write a symbolization for a given arithmetic quantity. This is the articulation step. ("How did you get the 120?")
4) Q_generalize: Uses the results of a Q_explain question ("Good, Now write your answer of 800-40*3 using the variables given in the problem (i.e. put in 'm')")
5) Q_represents_what: Translate from algebra to English("In English, what does 40m represent?" (e.g. "the distance rowed so far"))
6) Q_explain_verbal: Explain in English how a quantity could be computed from other quantities. (We have two forms: The reflective form is "Explain how you got 40*m" and the problem solving form is "Explain how you would find the distance rowed?")
7) Q_decomp: Symbolize a one operator answer, using a variable introduced to stand for a sub-quantity. ("Use A to represent the 40m for the distance rowed. Write an expression for the distance left towards the dock that uses A.")
8) Q_substitute: Perform an algebraic substitution ("Correct, that the distance left is given by 800-A. Now, substitute "40m" in place of A, to get a symbolization for the distance left.")
9) Q_order_of_ops: Ask a clarification question about the order of operations. ("Which operation do you do first/second/third/etc?")

You will notice that questions 1, 3, 4, and 8 all ask for a quantity to symbolize. Their main difference lies in when those questions are used, and how the tutor responds to the student's attempt. Questions 5 and 6 ask the student to answer in English rather than algebra. To avoid natural language the student is prompted to use pull down menus to complete this sentence "The distance rowed is equal to <noun phrase> <operator> <noun phrase>." The noun phrase menu contains a list of the quantity names for that problem. The operator menu contains "added to", "subtracted from", "multiplied by" and "divided by."

Below we will see how these questions can be combined into strategies.

### Tutorial Strategies

We will now look at a few examples of strategies that the tutorial model is able to reason about. First, we look at the concrete articulation strategy and how the cognitive research described earlier applies to it.

#### *Concrete Articulation*

The concrete articulation strategy usually uses three questions of type Q_compute, Q_explain and Q_generalize. Here is an example of the dialog generated by Miss Lindquist which we believe parallels the experienced human tutor closely.

T1: [Doing the "bike-trip" problem] Please write an expression for the total time the trip took. [Q_symb]
S1: s/m+b
T2: No. Let me ask you an easier question. Please compute the time actually on bikes if the speed was 20 miles per hour, and the distance ridden was 100 miles? [Q_compute]
S2: 5
T3: Correct. What math did you do to get that five? [Q_explain]
S3: 100/20
T4: Correct. Now write your answer of 100/20 using the variables given in the problem (i.e. put in 'm' and 's')? [Q_generalize]
S4: b+m/s
T5: Correct.

We will walk through what Miss Lindquist did here. First, the student model diagnosed S1 ("s/m+b") as having the quantity "s/m" with reversed arguments (bug type #2.) The diagnosis also said that the student understood that the total trip time was the time of the break ("b") plus the time actually on the bikes. This diagnosis was passed to the tutorial model that could have selected among competing strategies. In this case the tutorial model decided to conduct a *concrete articulation* knowledge construction dialog. Therefore, the tutor model generated questions for the *compute*, *explain/articulation* and *generalization* steps. The tutor put these three questions on the agenda.

At S2, the student answers the compute question correctly, and the tutor decides to continue with its plan. It has to be sensitive to what the student typed. If the student typed 100/20 instead of 5, then both the compute and explain questions would be removed from the agenda and the tutor would skip to the generalize step. An example of this sort of

flexibility is demonstrated in T5. T4 was a question to articulate the time actually on bikes. The tutor was hoping the student would type "m/s" but instead the student answered "b+m/s." Only a pedantic tutor wouldn't accept that answer as it indicates an understanding that subsumes the understanding required for the sub-goal. This is where the tutorial agenda behaves differently than a stack because Miss Lindquist pops both of the questions off the agenda.

To show you an example of how the previous situation could have come out differently, lets look at the following dialog

T1: Hello. [Doing the "bike-trip" problem] Please write an expression for the total time the trip took. [Q_symb]
S1: help!
T2: Let me ask you an easier question. Please compute the time actually on bikes if instead the speed was 20 miles per hour, the time of the break was 2 hours, and the distance ridden was 100 miles? [Q_compute]
S2: 7
T3: Correct. What math did you do to get that 7? [Q_explain 7]
S3: 2+5
T4: Okay, but how did you get the 5? [Q_explain 5]
S4: 100/20
T5: Correct Now lets try this again, what math did you do to get the 7? [Q_explain 7]
S5: 2+100/ 20
T6:Correct. Now write your answer of '2+100/20' using the variables given in the problem (i.e. put in 'm', 'b' and 's')? [Q_generalize]
S6: b+m/s
T7: Correct.

This example is interesting in the way Miss Lindquist responds at T4. The system has bug rules to apply to questions other than symbolization questions. In this case, the student model reported that the student's answer of "2+5" is a partially complete articulation but that the student didn't spell out where the "5" came from. So the tutorial model is able to "dig-down" into the student 's reasoning with an embedded sub-dialog. This involved adding the additional question of explaining how the student got the "5".

### *Substitution with decomposition*

A second example of a tutorial strategy is what we call the *substitution with decomposition* strategy, which was alluded to above when we looked at the student who answered the "two-jobs" problem with "A=5*g, B=30-g, C=7*B and D=A+C". Below we present an example of Miss Lindquist tutoring with this strategy. (As this article went to press we had just conducted our first test with a human student to help us debug the system. The following example is verbatim from this test.) This strategy uses both Q_decomp and Q_substitute questions. Question Q_decomp sets the stage for this strategy, and is used to break down the originally composed problem into separate decomposed one-operator problems (as shown below in T2 and T3), introducing new variables when appropriate (as in T3.) Question Q_substitute is used to scaffold the substitution step as shown in T4.

T1: Hello. Jane is "j" years old. Peg is "p" years old. Mary is Peg's age minus Tom's age. Tom is 3 years younger than Jane. Peg is "x" years older that Mark. Write an expression for Mary's age. [Q_symb]
S1:(p-3)-(3-j)+(p+x) [correct answer is "p-(j-3)"]
T2:Hmm. No. Let me ask you and easier question. Use j and 3 to calculate Tom's age. [Q_decomp]
S2:3-j
T3:Hmm. No. Try again please. What is Tom's age? Hint: pick one of these: j-3, 3-j, j+3, j*3, j/3, or 3/j. [This is a rephrasing as explained in the hint section below]
S3:j-3
T4:Correct. Calculate Mary's age. Use A instead of j-3 for Tom's age. [Q_decomp]
S4:p-a
T5:Correct. You are correct that p-A is Mary's age. Now, can you state this in terms of the numbers given in the problem. [Q_substitute]
S5:p-(j-3)
T6:Correct.

We note that this strategy is not one inspired by looking at human tutors but by our cognitive analysis. Suggestively, the human student seemed to profit from this strategy. We look forward to a controlled evaluation of this strategy to determine its relative effectiveness.

### *Reflective Explanation Strategy*

Mathematical communication is increasingly seen as a goal for students, and in particular translating between an algebraic representation, an English verbal representation, and a concrete representation of numerical instances (Schoenfeld et. al., 1992, Koedinger & Nathan) Other researchers are also interested in wanting an ITS to elicit explanations from students (e.g. Conati et. al., 1997.) We noticed in the following dialog that our experienced tutor made many requests for explanations that required the student to translate from the symbolic representation to a verbal representation (lines 204, 210 and 212.) Or sometimes they were asked to explain in English without having been told the algebra (line 192 and 196.)

190. S: [reads problem]Bob left at 3 P.M. and drove 550 miles from Boston to Pittsburgh to visit his grandmother. Normally this trip takes him "h" hours, but on Tuesday there was little traffic and he saved 2 hours. What was his average driving speed? [Q_symb]
191. S: ….Well … Ah … so he save two hours …ahum…
192. T: Do you know how to calculate average driving speed? [Q_explain_verbal]
193. S: I think, but I forget
194. T: Well, average speed… As your mom drove here did she drive the same speed the whole time. [tutor tries to make concrete]
195. S: No
196. T: But she did have an average speed. How do you think you calculate the average speed? [Q_explain_verbal]
197. S: It would be h hours divided by 550 miles an hour.
198. T: So which way is it? It's miles PER[verbal emphasis] hour. So which way do you divide?
199. S: It would be 550 divided by h. [writes "550/h=mph"]
200. T: OK, so now,.. that's how you calculate miles per hour. So now how about for this problem? Read the problem again. Because you got the right idea. You know how to calculate average speed. But what exactly do you have to do for this trip.
201. S: Um. Well he save two hours, but I don't know how that is important
202. T: Well, how do you calculate... Not for Bob but for your mom, how did you calculate what her average speed was driving to CMU this morning? [prompting for "average speed is equal to total distance divided by total time" like Q_explain_verbal]

203. S: Ahm, I guess you would I would have done it 550 divided by h
204. T: yeah [even though the 550 is not for his mom] That's how you calculate average speed but what exactly is it? 550 represent what? [like Q_represents_what]
205. S: Miles per hour
206. T: No.
207. S: Oh 550 miles
208. T: Right
209. S: Divided by h
210. T: Which represents? [like Q_represents_what]
211. S: Miles per hour
212. T: No what does h represent? [like Q_represents_what]
213. S: Hours [This problem continues for 15 more conversational turns.]

Since this seems common in our tutorial dialog we have implemented similar questions in the form of Q_represents_what and Q_explain_verbal as shown below.

T1: Hello. [Doing the "bike-trip" problem] Please write an expression for the total time the trip took. [Q_symb]
S1: m/s
T2: What does "m/s" represent (e.g. the speed on the bike?)[Q_represents_what]
S2: the total time of the trip
T3: No, "m/s" represents the actual time on the bikes. Can you explain how you would find the total time of the trip using the time on the bikes? [Q_explain_verbal]
S3: The total time of the trip is equal to the amount of time for the break added to the actual time on the bikes. [Composed using pull down menus.]
T4: Good, now say what the total trip time is in symbols [Q_symb]
S4: b+m/s
T5: Correct

There are many different contexts where you might want to use explanations. We are currently using them in situations like the one that occurs at line S1 where the student might have made an error of omission, rather than of commission. By asking the student what they mean by "m/s" we might prompt them to think more deeply about what that quantity represents. Another place where an explanation seems appropriate is in a reflective context as mentioned in the Hints section below.

### *Missing Parentheses Errors Remediation*

A fourth example of a tutorial strategy is applicable only when the student fails to use parentheses. We model the pedagogical decision of what to do in such a case as having to decide between treating the error as a slip (the student knows how to use parentheses but forgot) or a serious conceptual problem that demands a more through tutorial intervention. Our plan is to use the knowledge tracer component, that keeps track of the probability the student knows a particular skill, to decide if the error was a slip or not. If it is was a slip then we simply point out the error. On the other hand, if its not a slip then we can engage in the following dialog (assume the student said 7*30-g instead of 7(30-g))

T: What operation should you do first in 7*30-g? The multiplication or the subtraction? [Q_order_of_ops]
S: minus (tutor realizes that the student knows what should be done first but lacked the conceptual understanding for the need for parentheses)
T: [Start a dialog about the rules for order of operations, such as "If evaluating 7-3*4/(5+2), what operations would you do first and why?"]

### *Hints*

Giving the students a hint is the simplest tutorial strategy we have and is a common way that a traditional model tracing tutors gives help. Each question has a *hint chain* which is simply a list of rephrasings of the question. The rephrasings get progressively more explicit, eventually, possibly, telling the student exactly what to do[3]. We speculate that students will learn more if asked a question to prompt their own knowledge construction, than if they were told a hint. If the tutor has to tell the student the most explicit hint message then there is a good chance the student might not understand why that answer is correct. We observed that our experienced human challenged students' answers when it looked like the students guessed. Therefore, if the student receives the most explicit hint, Miss Lindquist has a strategy to ask the student a follow-up reflective question about what they received a hint on.

## Other Tutorial Operations

In addition to the questions the tutorial model can pose, it can also decide to do other things. A few examples will suffice. The tutor can decide whether to provide positive feedback about a portion of the student's answer that is correct, for example

T: No, but the 5g is correct for the distance rowed.

The tutor can also decide to give simple negative feedback like the following.

T: You used a "z" in your answer but there is no "z" in this problem. Did you mean to make that an "x"?

Generally, we avoid simply reporting the results of the student diagnosis, as this message does, but this is an example of a error that is probably best remediated with a simple message rather than a knowledge construction dialogs, since the error is probably not a conceptual error.

A final element, but a very important one is making sure the tutor provides the English cues for conversational dialog coherence (Moore, 1993). For instance the tutor reasons whether a question should be prefaced with

T: No. Let me ask you an easier question…

or

T: No. What is …[no prefacing]

or

T: Correct, Now, lets go back to this earlier question …

or

T: Okay, but where did the 5 come from…

We recognize that this is a simple approach, but it has been able to take us a reasonable distance.

[3] The question of how explicit to make the "bottom-out", or last hint (i.e. "Do you tell the student the answer?") is not a question that we, or the architecture, makes a commitment about as it appears to depend greatly on context. Some of our hint chains end with multiple choice questions while others tell the student exactly what to type.

## How is Miss Lindquist different from traditional model tracing tutors?

Our experienced human tutor was very active, taking over 200 turns in an hour which averaged out to a comment every 20 seconds. Graesser et. al. and Lepper et. al. (1997) both reported that tutors asked at least 80% of the questions (our smaller sample was even higher.) Our experienced tutor did not wait for the student to ask for a hint, as many model tracing tutors do. Miss Lindquist is more like a human in that the dialog is largely through tutor initiative. But this difference with model tracing tutors should not be made too much of, as model tracing tutors could easily have their hints triggered automatically when an error is made.

McArthur et. al. criticized Anderson's et. al. (1985) model tracing ITS and model tracing in general "because each incorrect rule is paired with a particular tutorial action (typically a stored message), every student who takes a given step gets the same message, regardless of how many times the same error has been made or how many other error have been made. … Anderson's tutor is tactical, driven by local student errors (p. 200)" and goes on to argue for the need for a more strategic tutor. Miss Lindquist meets that criticism. Miss Lindquist's model of tutorial reasoning is both strategic (i.e. has multi-step plans) and tactical (i.e. reasons to produce output at the single question level.) She also intelligently handles multiple errors and reasons about the order in which to deal with them and then constructs a plan to deal with each of them. Miss Lindquist is a modest step on the path to making a more dynamic tutor and promises to teach us about the effectiveness of different tutoring strategies.

## Planned Experiment and Future Work

We believe that the good way to test theories of tutoring is to build concrete instantiations of that theory, as represented by computer programs, and perform experiments to determine which are most effective. We have taken a very modest step in that direction with our tutorial model. Though it has some general tutorial knowledge, most of its interesting aspects come from its content-specific pedagogical knowledge. We look forward to measuring the effectiveness of this system compared to effective benchmarked 2nd generation ITS (Koedinger et. al., 1997; Anderson et. al., 1995). We also look forward to comparing the effectiveness of different strategies. Finally we forward the possibility that through machine learning, Miss Lindquist could become a more effective tutor while at the same time instructing us on what makes a tutor more effective.


### Acknowledgements

This research was supported by NSF grant number 9720359 to CIRCLE and the Spencer Foundation. We would also like to thank for their helpful comments both Steve Ritter, John Pane, Douglas Rhode and Reva Freedman. Freedman's approach (Freedman, 2000) is similar.


## References


Anderson, J. R. (1993). *Rules of the Mind*. Hillsdale, NJ: Erlbaum.

Anderson, J. R., Boyle, D. F., & Reiser, B. J. (1985). Intelligent tutoring systems. *Science*, 228, 456-462.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995) Cognitive tutors: lessons learned. *The Journal of the Learning Sciences*, 4 (2), 167-207.

Anderson, J. R. & Pelletier, R. (1991) A developmental system for model-tracing tutors. In Lawrence Birnbaum (Eds.) *The International Conference on the Learning Sciences*. Association for the Advancement of Computing in Education. Charlottesville, Virginia (pp. 1-8).

Clancey, W. J., (1982) Tutoring rules for guiding a case method dialog. In D. Sleeman & J. S. Brown (Eds.) *Intelligent Tutoring Systems* London: Academic Press. (pp. 201-226.)

Corbett, A. T., and Anderson, J. R., (1995) Knowledge decomposition and subgoal reification in the ACT programming tutor. in *Proceedings of Artificial Intelligence in Education* (pp. 469-476)

Conati, C., Larkin, J. and VanLehn, K. (1997) A computer framework to support self-explanation. In : du Bolay, B. and Mizoguchi, R.(Eds.) Proceedings of AI-ED 97 World Conference on Artificial Intelligence in Education. Vol.39, pp. 279-276, Amsterdam: IO Press.

Cummins, D. D., Kintsch, W., Reusser, K. & Weimer, R. (1988). The role of understanding in solving word problems. *Cognitive Psychology*, 20, (pp. 405-438.)

Freedman, R. (2000 to appear) Using a reactive planner as the basis for a dialogue agent. In *Proceedings of the Thirteenth Florida Artificial Intelligence Research Symposium* (FLAIRS '00), Orlando.

Graesser, A.C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N., & the TRG (in press). Using latent semantic analysis to evaluate the contributions of students in AutoTutor. Interactive Learning Environments.

Heffernan, N. T., & Koedinger, K. R.(1997) The composition effect in symbolizing: the role of symbol production versus text comprehension. *Proceeding of the Nineteenth Annual Conference of the Cognitive Science Society* 307-312. Hillsdale, NJ: Lawrence Erlbaum Associates.

Heffernan, N. T., & Koedinger, K. R. (1998) The composition effect in symbolizing: the role of symbol production vs. text comprehension. *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, 307-312. Hillsdale, NJ: Lawrence Erlbaum Associates.

Koedinger, K. R., Anderson, J.R., Hadley, W.H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, *8,* 30-43.

Koedinger, K. R., & Anderson, J. R. (1998). Illustrating principled design: The early evolution of a cognitive tutor for algebra symbolization. In *Interactive Learning Environments*, 5, 161-180.

Koedinger, K. R., & MacLaren, B. (1997). Implicit strategies and errors in an improved model of early algebra problem solving. In Proceedings of the Nineteenth Annual Meeting of the Cognitive Science Society (pp. 382-7). Mahwah, NJ: Erlbaum.

Koedinger, K. R. & Nathan, M. J. (submitted to). The real story behind story problems: Effects of representations on quantitative reasoning. Submitted to *Cognitive Psychology*.

Larson, R., Kanold, T., & Stiff, L. (1995) *Algebra 1: An Integrated Approach.* D.C. Heath. Lexington, MA.

LeBlanc, M. D., & Weber-Russell, S.(1996). Text integration and mathematical connections: a computer model of arithmetic word problem solving. *Cognitive Science* 20,357-407.

Lepper, M. R., Drake, M. F., & O'Donnell-Johnson, T. (1997) "Scaffolding techniques of expert human tutors. In K. Hogan & M. Pressley (Eds.) *Scaffolding Student Learning: Instructional approaches and Issues.* Cambridge MA: Brookline Books (pp. 108-144.)

Lewis, A. B. & Mayer, R. E. (1987). *Journal of Educational Psychology*, *79(4),* 363-317.

McArthur, D., Stasz, C., & Zmuidzinas, M. (1990) Tutoring techniques in algebra. *Cognition and Instruction*. 7 (pp. 197-244.)

Moore, J. D. (1993) What makes Human Explanations effective? In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum. (pp. 131-136).

Moore, J. D. (1996) Discourse generation for instructional applications: Making computer-based tutors more like humans. *Journal of Artificial Intelligence in Education*, 7(2), 118-124

Moore, J. D. & Mittal, V. O. (1996) Dynamically generated follow-up questions. I*EEE Computer*, 75-86.

Ohlsson, S. (1986) Some principles for intelligent tutoring. *Instructional Science*, 14 293-326.

Paige, J. M. & Simon, H.(1979). Cognitive process in solving algebra word problems. in H. A. Simon *Models of Thought.* New Haven, Yale University Press.

Sfard, A., & Linchevski, L. (1993). The gain and the pitfalls of reification- the case of algebra. *Educational Studies in Mathematics,* 00: 1-38.

Schoenfeld, A., Gamoran, M., Kessel, C., Leonard, M., Or-Bach, R., & Arcavi, A. (1992) Toward a comprehensive model of human tutoring in complex subject matter domains. *Journal of Mathematical Behavior*, 11, 293-319

Shulman, L. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher, 15*, 4-14.

VanLehn, K, Anderson, J., Ashley, K., Chi. M., Corbett, A., . Koedinger, K., Lesgold, A., Levin, L., Moore, M., and Pollack, M., NSF Grant 9720359. *CIRCLE: Center for Interdisciplinary Research on Constructive Learning Environments.* NSF Learning and Intelligent Systems Center. January, 1998 to January, 2003.

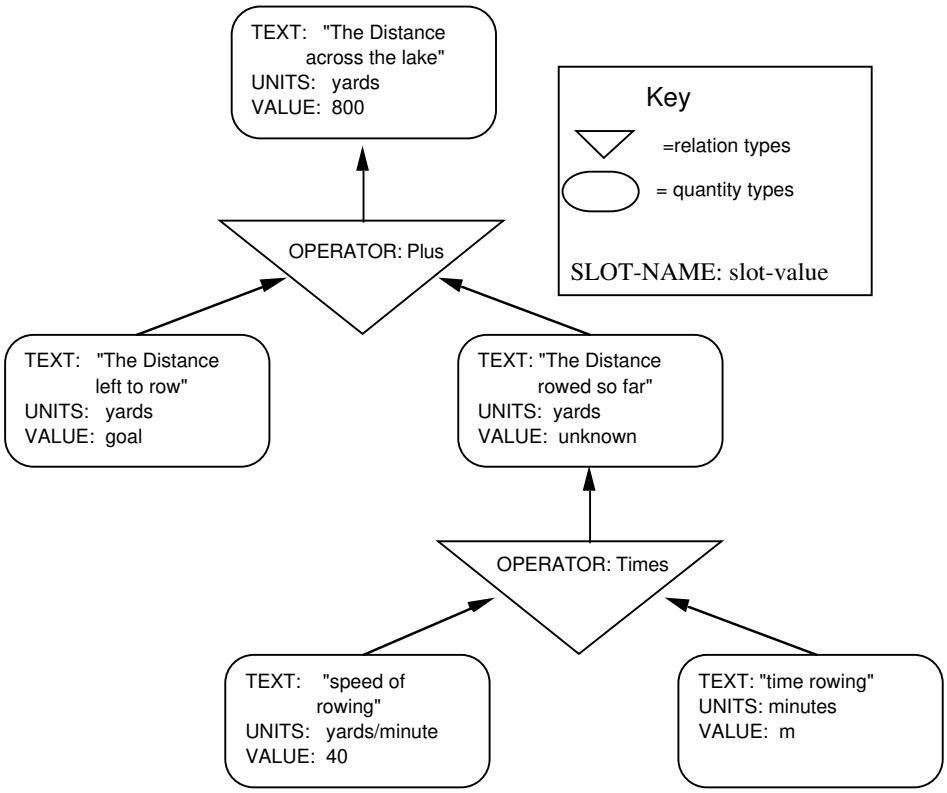


**Figure 1. A Picture of the working memory representation of a student after comprehending the problem "Anne is in a rowboat in a lake. She is 800 yards from the dock. She rows back towards the dock at a speed of 40 yards per minute for 'm' minutes. How far is Ann from the dock?"**

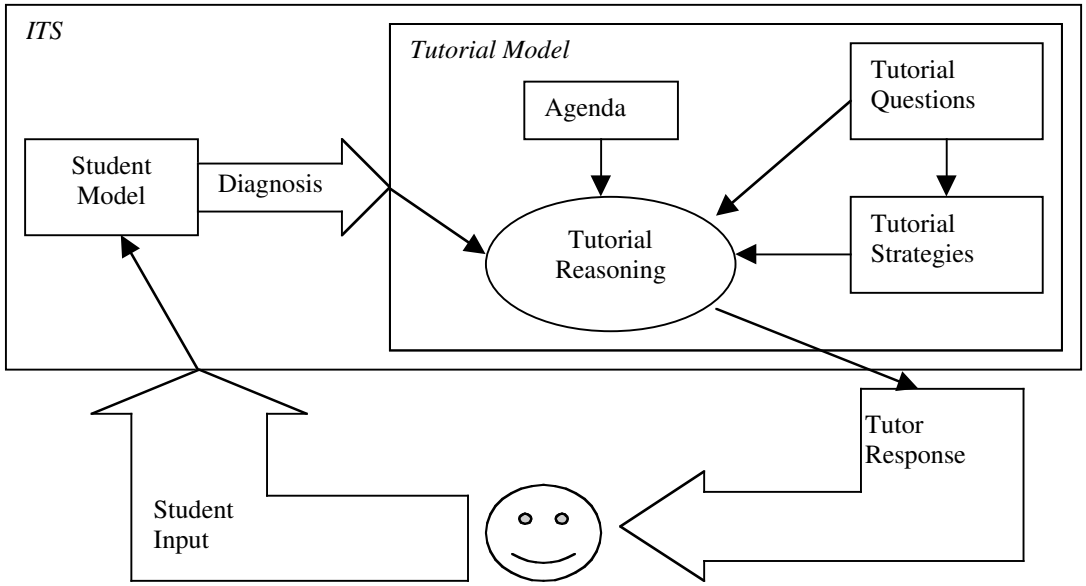


**Figure 2: A model of *Miss Lindquist*'s architecture.**