Towards Tutorial Dialog to Support Self-Explanation: Adding Natural Language Understanding to a Cognitive Tutor*

Vincent Aleven, Octav Popescu, and Kenneth R. Koedinger
Human Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
aleven@cs.cmu.edu , octav@cmu.edu, koedinger@cs.cmu.edu

Abstract. Self-explanation is an effective metacognitive strategy, as a number of cognitive science studies have shown. In a previous study we showed that self-explanation can be supported effectively in a cognitive tutor for geometry problem solving. In that study, students explained their own problem-solving steps by selecting from a menu the name of a problem-solving principle that justifies the step. They learned with greater understanding, as compared to students who did not explain their reasoning. Currently, we are working toward testing the hypothesis that students will learn even better when they provide explanations in their own words rather than selecting them from a menu. We have implemented a prototype of a cognitive tutor that understands students' explanations and provides feedback. The tutor uses a knowledge-based approach to natural language understanding. We are entering a phase of pilot testing, both for the purpose of assessing the coverage of the natural language understanding component and for gaining insight into the kinds of dialog strategies that are needed.

1. Introduction

How can we take intelligent tutoring systems (ITSs) to the next level? ITSs are not yet as effective as human tutors, even in the domains where ITSs have been most successful. Quite possibly, the difference has something to do with the power of tutorial dialog. A growing number of researchers therefore are working on finding out whether adding dialog capabilities to ITSs will result in more effective systems [Rose and Freedman, 2000].

In designing tutorial dialog systems, it is important to focus on areas where dialog is likely to have the highest pay-off. The dialog must support learning activities that are most likely to improve students' understanding. A number of cognitive science studies have shown that self-explanation is an effective learning strategy. Students studying examples or textbook text learn with greater understanding when they explain the study materials to themselves [Chi, et al, 1989; Bielaczyc, et al, 1995]. We present a research project that focuses on developing a tutorial system that supports self-explanation by engaging students in dialog.

In previous research we showed that a cognitive tutor for geometry problem solving is more effective when it supports students' self-explanation, even without natural language understanding (NLU) or dialog [Aleven, et al., 1999]. In that study we compared two tutor versions. One group of students used a tutor version that required them to provide correct

* **Acknowledgements:** This research is sponsored by an NSF grant to the Center for Interdisciplinary Research on Constructive Learning (CIRCLE), a joint research center located at the University of Pittsburgh and Carnegie Mellon University.

In J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.), *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001* (pp. 246-255). Amsterdam: IOS Press.

Table 1: Examples of correct and incorrect explanations of the triangle sum theorem

Correct and Complete Explanations	**Incomplete Explanations (ctnd)**
The angles of a triangle sum to 180 degrees	180 degrees in a triangle
sum of all angles in a triangle is 180	the total sum of the angles need to be 180
angles must add up to 180 degrees in a triangle.	adding all the angles is 180
	a triangle's sum adds up to 180
Incomplete Explanations	because they all equal 180
A triangle is comprised of 180 degrees	they add to 180
triangle equals 180	it equals out to be 180 degrees

explanations for their steps, by referencing problem-solving principles. A second group of students used a tutor version that required them only to solve problems but not to explain their steps. We found that students who explained their steps learned with greater understanding. They were less inclined to rely on shallow guessing heuristics, were better able to explain their steps, and could better solve transfer problems.

The support for self-explanation in that study had a number of potential limitations: Students did not explain in their own words, selected explanations from menus, and provided very abbreviated explanations. We are preparing to test the hypothesis that a better way to support self-explanation is to have students explain in their own words, require that they provide complete explanations, and give them feedback and assistance in constructing explanations.

It may not be obvious that feedback is needed. In a number of cognitive science studies, self-explanation helped learning even when students did not receive feedback on their explanations. However, studies of students' self-explanations also found substantial individual differences in students' ability to self-explain and found that even when prompted, many students do not provide good self-explanations [Chi et al., 1989; Renkl, et al, 1998]. Feedback is likely to help students to generate better explanations, a point made also by [Conati and VanLehn, 2000]. Further, when students receive feedback on explanations, this makes it more likely that they will actually provide explanations when prompted to do so. Ensuring that students comply with prompts for explanations is important especially when dealing with instructional technology. A recent study of ours underscored both concerns [Aleven and Koedinger, 2000]. In this study, students worked with a tutor version that prompted them to explain their answers in their own words, but did not analyze the explanations or provide feedback. We found that without feedback, very few students were able to provide good explanations. Furthermore, students left the majority (64%) of explanation boxes blank or entered irrelevant comments.

In this paper, we present a prototype system that helps student improve explanations through dialog. We describe the system's knowledge-based NLU component, show examples of dialogs with the system, and discuss how we plan to develop the system further.

2. The Geometry Explanation Tutor

In developing a system that provides support for constructing self-explanations, our focus has been on getting the student to explain precisely—on helping students to improve explanations that seem to get at the right idea, but are not sufficiently precise. This goal followed from our analysis of several small corpora of student explanations. Those corpora indicated, besides the fact that it is difficult for students to state geometry rules in their own words, that there are many ways of stating geometry rules correctly, and even more ways of stating them incorrectly. Some examples are shown in Table 1. Students' attempts at stating geometry rules are often incomplete, for example because they omit part of the conditions

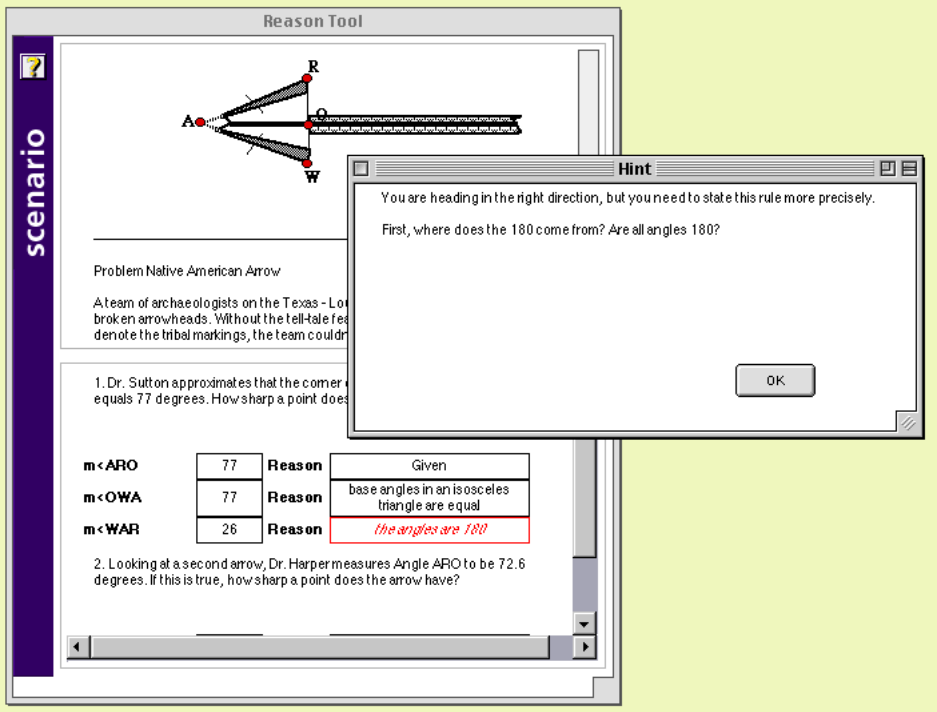

Figure 1: The Geometry Explanation Tutor

under which the rule applies. Further, students use language that is ill-formed both syntactically and semantically.

What was needed, therefore, is a robust NLU component that can perform a fine-grained analysis of students' explanations, so that the tutor can respond to the types of omissions we often see in students' explanations. We are adding such facilities to an existing cognitive tutor, the PACT Geometry Tutor [Aleven, et al., 1999]. This tutor has been developed by our research group, together with a new high-school geometry course of which the tutor is an integrated part. Following guidelines developed by the National Council of Teachers of Mathematics [NCTM, 1989], the curriculum emphasizes the application of geometry to "real world" problems. In courses based on the curriculum, students spend 40% of total classroom time working on the tutor, in the school's computer lab. During these sessions, the teacher is present to assist students who need help beyond what the tutor can offer. The curriculum and tutor are in regular use in about five schools, mostly in the Pittsburgh area.

We have built a prototype tutorial dialog system, based on the PACT Geometry Tutor, which we call the Geometry Explanation Tutor. This system covers one of the six units that make up the curriculum of the original PACT Geometry Tutor, namely, the unit that deals with the geometric properties of angles. Like all cognitive tutors, the Geometry Explanation Tutor selects suitable problems on an individual basis and provides feedback, as students enter solutions or intermediate steps. The tutor provides context-sensitive hints on students' request. It decides, also on an individual basis, when to advance a student to the next curriculum section. Unlike other cognitive tutors, the Geometry Explanation Tutor requires that students explain their answers to geometry problems, by stating—in their own words—a geometry definition or theorem that justifies the step. The tutor checks that the explanation is an accurate and complete statement of an appropriate geometry rule. If the explanation is incomplete, or if the student is focusing on the wrong rule, the tutor provides feedback, as is illustrated in Figure 1.

Table 2: Two examples of constructing an explanation with the Geometry Explanation Tutor. Shown on the left are a student's successive attempts at explaining the given geometry rule (what was added since the previous attempt is underlined) and the tutor's responses. Shown on the right is the tutor's classification of the student's explanation.

Explanation of the isosceles triangle theorem (step 2 in Figure 1, m∠OWA)	
Student: angles are equal	CONGRUENT-ANGLES
Tutor: Are any two angles congruent? What tells you that there are two congruent angles here?	
Student: angles in a triangle are equal	CONGRUENT-ANGLES-IN-TRIANGLE
Tutor: That is a bit imprecise. Are ALL angles in ALL triangles congruent?	
Student: angles in an isosceles triangle are equal	CONGRUENT-ANGLES-IN-ISOSCELES-TRIANGLE
Tutor: Correct, except for one thing: are all three angles in an isosceles triangle congruent?	
Student: base angles in an isosceles triangle are equal	ISOSCELES-TRIANGLE-BASE-ANGLES
Tutor: CORRECT.	

Explanation of the triangle sum theorem (step 3 in Figure 1, m∠WAR)	
Student: they are 180	EQUAL-180
Tutor: Could you be more specific? Who are "they"? And what tells you that they are 180?	
Student: the angles are 180	ANGLES-180
Tutor: You are heading in the right direction, but you need to state this rule more precisely. First, where does the 180 come from? Are all angles 180?	
Student: the angles in a triangle are 180	ANGLES-OF-TRIANGLE-180
Tutor: You are awfully close. But is each angle measure equal to 180? Precisely what is 180?	
Student: the sum of angles in a triangle are 180	TRIANGLE-SUM
Tutor: CORRECT	

A sense of dialog emerges as the student incrementally improves an explanation, helped by the tutor's detailed feedback. This is illustrated in Table 2, which shows two dialogs with the system about two different geometry theorems. In each step, the hypothetical student adds to her explanation-under-construction and then resubmits it to the tutor. (On the right are listed the system's internal classification of the student's explanations, as is explained below.) The tutor's responses often take the form of a question, meant to make the student see that the rule she stated is overly general. After several iterations, the student has refined an explanation that initially was imprecise and overly general into an accurate statement of the geometry rule.

3. The Architecture of the Geometry Explanation Tutor

The Geometry Explanation Tutor is based on the standard cognitive tutor architecture [Anderson, et al., 1995], augmented with facilities for natural language understanding (see Figure 2). The task of the NLU component is to classify students' explanations with respect to a set of categories of correct and partially correct explanations, as explained below. The task of the cognitive tutor component is to check whether the student's explanations are correct, based on the classification determined by the NLU component and to present appropriate feedback to the student.

We opted for a hybrid approach to natural language understanding. The tutor relies primarily on a knowledge-based approach to recognize sentences as correct or partially correct explanations. It uses a statistical text classifier when the knowledge-based method fails, to determine whether the explanation is in the ballpark (i.e., focuses on the right geometry rule), not whether it is complete. The reasons why we prefer a knowledge-based

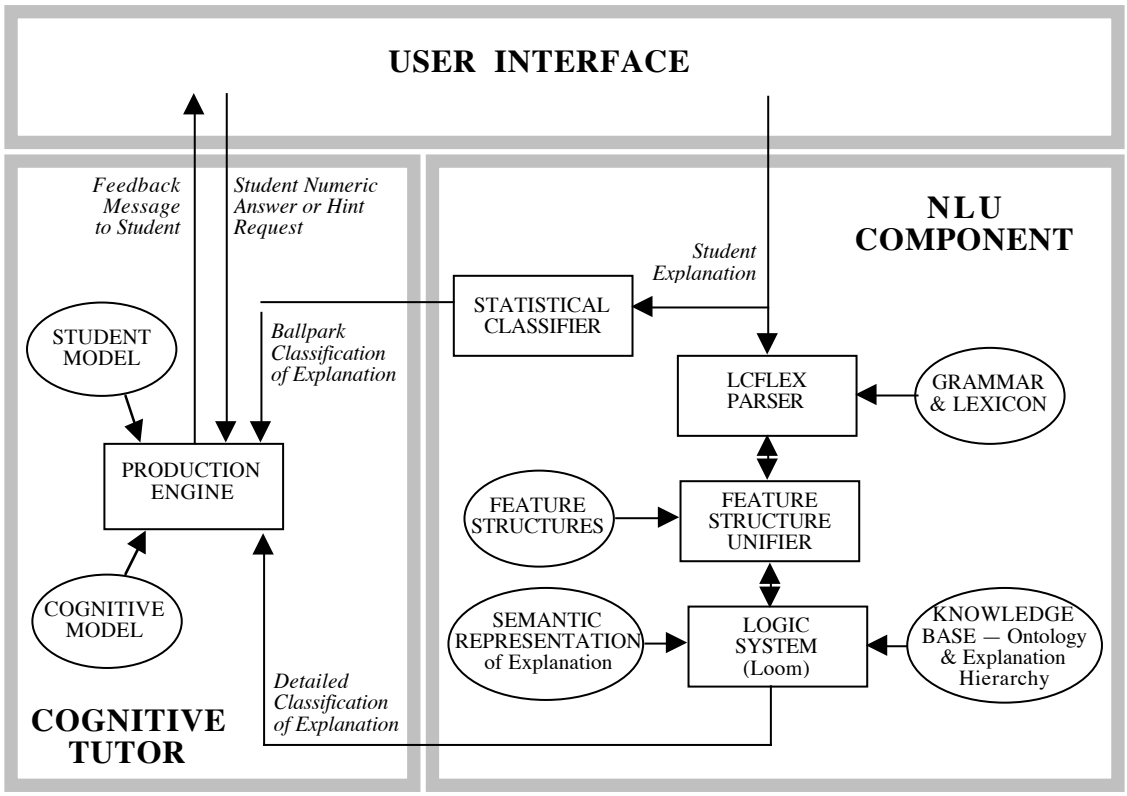

Figure 2: Architecture of the Geometry Explanation Tutor

approach over a purely statistical approach, as was used for example in AutoTutor [Wiemer-Hastings et al, 1998], are explained in [Popescu and Koedinger, 2000].

The knowledge-based NLU component parses the student explanation, creates a semantic representation, and classifies that representation. We use the term "NLU component" to refer to the knowledge-based part. The main knowledge source used by the NLU component is a geometry knowledge base, implemented in the Loom term description logic. The knowledge base covers the topics addressed in the Angles unit of the tutor curriculum. It contains a basic ontology of the domain, which includes geometry objects such as angles and lines, as well as relations such as congruency, adjacency, etc. We use a portion of the Generalized Upper Model [Bateman, et al, 1995] to anchor our geometry ontology. Further, the knowledge base contains a hierarchy of explanation categories, representing correct and partially-correct ways of stating the geometry rules. A small part of this hierarchy is shown in Figure 3. Each category in the hierarchy represents a class of explanations that have the same meaning but may have widely different surface forms. The hierarchical ordering of the categories reflects the degree of completeness of the explanations. More complete explanations are represented by more specific concepts in the hierarchy (i.e., concepts lower down in Figure 3). Conversely, incomplete explanations, or overly general statements of geometry rules, are represented by concepts higher up in the hierarchy. The hierarchy is based in part on our analysis of the corpora of student explanations mentioned earlier. Currently, the knowledge base contains definitions for about 250 concepts and 100 relations. Among those are definitions for about 70 explanation categories, corresponding to correct and incorrect ways of stating 30 geometry rules.

The NLU component combines the LCFLEX active-chart parser [Rose and Lavie, 1999] and a feature structure unifier to construct a semantic representation of the student's sentence. We have developed a grammar that has about 200 rules. The semantic representation of

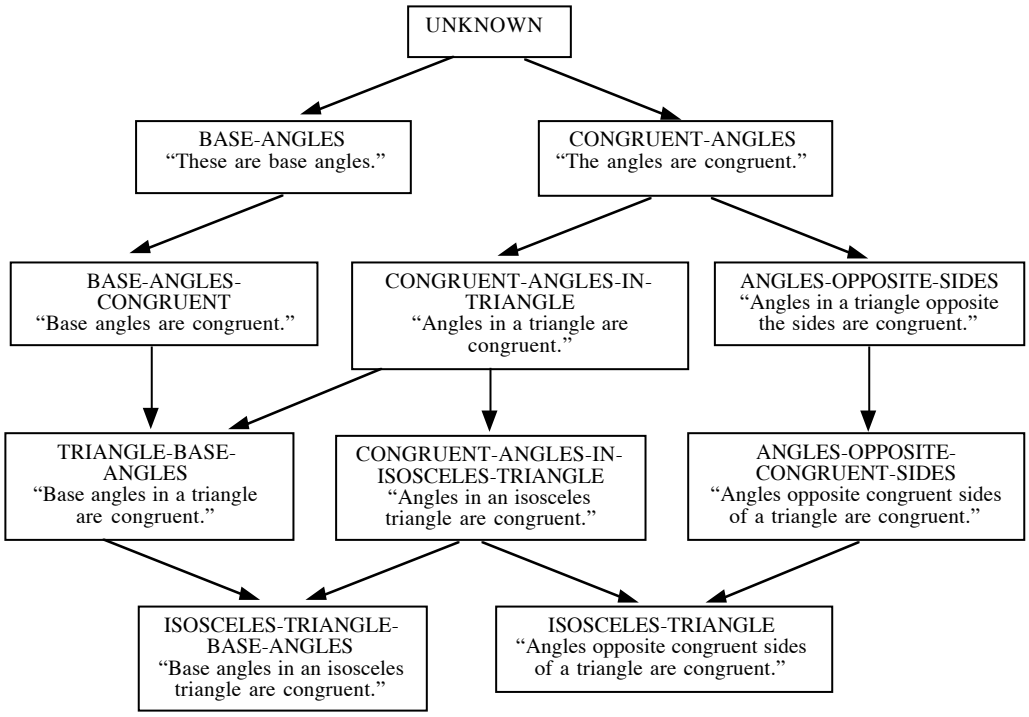

Figure 3: Excerpt from the Geometry Explanation Tutor's hierarchy of explanation categories. Each category represents a class of explanations which can be paraphrased as shown. The system's knowledge base contains a defintion for each category, expressed in Loom's term descriptoin language. The two categories at the bottom represent complete explanations, all other categories represent partial explanations.

sentences is logic-based and is represented in the Loom term description logic system [MacGregor, 1991]. Once this representation has been constructed, it is classified automatically, by Loom's classifier, with respect to the explanation categories in the knowledge base.

The cognitive tutor component determines what feedback to give to the student, based on the classification of the explanation. In order to be considered correct, the student's explanation must be a complete and correct statement of an appropriate geometry rule, that is, a theorem or definition that can justify the problem-solving step being explained. Therefore, the tutor first determines which geometry rule or rules can justify the current step. It does so using its cognitive model of geometry problem solving, which represents the skills of an ideal student, modeled as a set of production rules. After determining the geometry rule(s) to be explained, the tutor selects a feedback message, taking into account whether the student's explanation was (a) a complete explanation of one of these rules, (b) a partial explanation of one of these rules, (c) a reference to one of these rules, meaning that the student entered the name of a rule rather than a statement of the rule itself or (d) whether the student focused on the wrong rule.

If the student explanation is classified as a complete statement of an appropriate geometry rule, the tutor accepts the explanation. For example, in the last step of the first dialog shown in Table 2, the NLU component classifies the student's explanation under category ISOSCELES-TRIANGLE-BASE-ANGLES, shown on the bottom left in Figure 3. This category represents full explanations of the isosceles triangle theorem, which indeed justifies the step being explained. The tutor therefore accepts the explanation.

When the student's explanation is classified as being a partial explanation of an appropriate geometry rule, the tutor selects a feedback message that points out how to

improve the explanation. Most of the time, the tutor will simply select the feedback message that is attached to the category under which the explanation was classified—each category of partial explanations has its own feedback message. For example, the student's statement shown in Table 2 that "angles in a triangle are equal", is classified under category CONGRUENT-ANGLES-IN-TRIANGLE, shown in the middle of Figure 3. The feedback message shown is the one associated with this category.

If the student's explanation classifies as a reference (e.g., if the student says "triangle sum" instead of stating the triangle sum rule), the tutor selects a feedback message saying that the student should explain the rule more fully. Finally, if the student's explanation is classified under a category not associated with any of the geometry rules that could justify the current step, the tutor selects a message pointing out that the student is focusing on the wrong rule. Technically, the feedback messages are implemented by means of bug rules in the tutor's cognitive model.

Efficiency is an important concern in developing interactive NLU systems. We took the following steps to enhance the efficiency of our NLU system. First, syntactic parsing and the construction of a semantic representation are interleaved so that semantic constraints can be employed to prune the search, that is, to eliminate from consideration partial paths that are grammatically correct but for which no coherent semantic interpretation can be found. Further, the NLU component works concurrently with the student: It receives each word as soon as the student types it, and thus is able to construct a significant portion of the semantic representation while the student is still typing. The NLU component is described in greater detail in [Popescu and Koedinger, 2000].

As mentioned, the NLU component uses a statistical text classifier as a backup. This classifier is based on the Naïve Bayes classification method [Mitchell, 1997, Ch. 6]. When the NLU component fails to build a semantic representation for a sentence, or builds a semantic representation that does not classify under any of the explanation categories, the system falls back on the statistical classifier to determine whether the student's explanation is focusing on the right geometry rule. If so, the tutor will indicate in its response that the student is on the right track, although the current version does not consider the explanation to be correct. The tutor will say "You appear to be focusing on the right geometry rule. However, the tutor does not understand your explanation, could you please state it in a different way." Without the statistical classifier, the tutor feedback could include only the second of these two sentences. Thus, the statistical text classifier enables the tutor to provide more informative feedback in response to (some) unexpected input, sentences that express geometry rules in an unusual way. The Naïve Bayes text classifier performs a more coarse-grained classification task than the NLU component. It classifies students' explanations with respect to a subset of the categories that the NLU component uses. It has one category per geometry rule, representing explanations that focus on that rule, whether they are complete or incomplete explanations. The statistical classifier computes the probability of each category by multiplying the conditional probabilities of each category, given each word.

4. Discussion and Conclusion

We present a preliminary architecture for a tutorial dialog system that supports self-explanation. One might characterize the basic approach of the system as "classify-and-react" —in each dialog cycle, the system classifies the student's explanation and then responds based on that classification. This process is rather similar to the model-tracing strategy of cognitive tutors [Anderson, et al., 1995], except that the system's classification process is more sophisticated than that of the standard model-tracing tutors.

So far, we have focused on developing the NLU component of the Geometry Explanation Tutor. The dialog management component is currently not very sophisticated. It does not maintain a dialog history and does not do dialog planning. In spite of this, users can

have a sense of dialog when interacting with this system—as the examples hopefully illustrate, and as a few people who have used the system have remarked. However, one does not have to look far to find tutorial strategies and dialog phenomena that are not covered and that do not fit easily in the classify-and-react framework. Here, we consider three categories of limitations.

Dialogs with the Geometry Explanation Tutor are smooth as long as each student attempt at explaining progresses toward a more complete explanation, or to put this more technically, as long as each explanation attempt is classified under a more specific explanation category than the previous attempt. (This assumption is met in the two examples given in this paper.) The tutor however cannot detect when a student "regresses" or "stagnates", that is, when a subsequent attempt at explaining a step is worse or no better than a previous attempt on the same step. In such cases, the tutor should point out that the student's previous explanation attempt was better and perhaps offer to undo the regressive step. Instead, it currently blithely gives a locally-appropriate feedback message without any knowledge of the global lack of progress. Similarly, the tutor currently cannot detect "lateral movement" with respect to the classification hierarchy, that is, situations where a subsequent explanation attempt is better in one respect than the previous attempt, but worse in another respect. The tutor is not in a position to point out that the student deleted a part of her explanation that was essential. In order to address these problems, the tutor would need to keep a history of how students' explanation attempts were classified. Further, it would need to be able to display appropriate feedback messages when it detects regression, stagnation, or lateral movement. It is an open question to what extent these messages need to be context-sensitive and how such context-sensitivity can best be achieved. Addressing these problems makes sense if students indeed regress or stagnate with some frequency. We intend to analyze tutor data to see if they do.

A second way in which the system's dialog capabilities are limited is that the system's on-request help messages are not sensitive to state of the student's explanation-under-construction. When the student asks for help, the help message suggests that the student look up the relevant geometry rule in the tutor's on-line Glossary of geometry knowledge. (The Glossary contains a short description in English of each geometry rule, illustrated with a short example. Students can browse this Glossary freely as they work with the tutor.) However, the dialog would be more smooth and the system more helpful if the help message would focus specifically on what the student still needs to do in order to complete the explanation that she is working on. For example, if only a single word were missing, the tutor might simply give it. If the missing word was a technical term, the tutor might use the opportunity to give an explanation of the term. A related limitation is that the system's feedback messages and on-request help messages are not well-coordinated. The hint messages do not follow up on questions asked by the feedback messages. For example, if the student asks for help after receiving a feedback message such as "how do you know that the angles are congruent?" the help message should address the question asked in the feedback message. ("You have an isosceles triangle here. What angles in an isosceles triangle are congruent?") Addressing this problem makes sense if tutor use data indicate that there is room for improvement, for example if students underuse the tutor's help facilities or are often not able to improve their explanations after receiving feedback.

A third limitation of classify-and-react is that it is based on the assumption that the student's goal in the dialog is fixed, namely, to provide a correct and complete statement of a general geometry theorem, and therefore that each utterance by the student is an attempt at achieving the same goal. This assumption does not hold in many tutorial dialogs. The problem-solving goal that the student works on (and that the dialog focuses on) may change for example due to scaffolding by the tutor [Heffernan and Koedinger, 2000]. Students or the tutor may ask questions. They may elaborate on or retract previous statements. However, the current Geometry Explanation Tutor does not have a representation of the dialog goal and

therefore it cannot deal with changing goals. This restricts its repertoire of dialog strategies. For example, the tutor cannot branch into a subdialog on a more specific topic, such as a particular condition of the theorem to be explained. As another example, when the student states on overly general rule, it would be great if the tutor could show a counterexample, a technique that is a mainstay of mathematical reasoning. Within the current architecture it would be straightforward to have feedback messages that present counterexamples. But without the ability to branch into a subdialog about why the counterexample is a counterexample—as would be necessary if the student did not understand why—it does not seem wise to have such feedback messages. Before we decide whether to add these strategies, we plan to gather evidence related to their pedagogical effectiveness through the time-honored method of Wizard of Oz studies. In these studies, some of the smarts of a computer tutor are provided by a human tutor, who is communicating over the network, usually unbeknownst to the student working with the computer tutor. This setup provides a relatively straightforward way to pilot test new tutor features before incurring the cost of actually implementing them.

It may well turn out to be necessary to augment the tutor architecture so that it maintains a dialog history. It is probably a good idea to add facilities for dialog planning. We are considering whether to adapt an existing approach [Core, et al, 2000; Freedman, 2000; Heffernan and Koedinger, 2000] to our domain. In considering these extensions, our main goal is to find out what really improves student learning. For the short term, we have a tutor that is far enough along to start pilot-testing and to start exploring the questions raised above, through a careful combination of pilot testing, Wizard of Oz studies, and studying human tutors. Ultimately, our goal is to conduct a controlled experiment to test the hypothesis that natural language dialog makes a difference in supporting self-explanation.

References

Aleven, V., and K. R. Koedinger, 2000. The Need for Tutorial Dialog to Support Self-Explanation. In *Building Dialogue Systems for Tutorial Applications, Papers of the 2000 AAAI Fall Symposium,* edited by C. P. Rose and R. Freedman, 65-73. Technical Report FS-00-01. Menlo Park, CA: AAAI Press.

Aleven, V., K. R. Koedinger, and K. Cross, 1999. Tutoring Answer Explanation Fosters Learning with Understanding. In *Proceedings of AIED-99,* edited by S. P. Lajoie and M. Vivet, 199-206. Amsterdam: IOS Press.

Anderson, J. R., A. T. Corbett, K. R. Koedinger, and R. Pelletier, 1995. Cognitive tutors: Lessons learned. The Journal of the Learning Sciences, 4,167-207.

Bateman, J. A, Henschel, R., and Rinaldi F. 1995. The Generalized Upper Model 2.0, GMD/IPSI, Project KOMET, Darmstadt, Germany.

Bielaczyc, K., P. L. Pirolli, and A. L. Brown, 1995. Training in Self-Explanation and Self-Regulation Strategies: Investigating the Effects of Knowledge Acquisition Activities on Problem Solving. *Cognition and Instruction,* 13 (2), 221-252.

Chi, M. T. H., M. Bassok, M. W. Lewis, P. Reimann, and R. Glaser, 1989. Self-Explanations: How Students Study and Use Examples in Learning to Solve Problems. *Cognitive Science,* 13, 145-182.

Conati, C. and K. VanLehn, 2000. Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. *International Journal of Artificial Intelligence in Education,* 11.

Core, M. G., J. D. Moore, and C. Zinn, 2000. Supporting Constructive Learning with a Feedback Planner. In *Building Dialogue Systems for Tutorial Applications, Papers of the 2000 AAAI Fall Symposium,* edited by C. P. Rose and R. Freedman, 1-9. Technical Report FS-00-01. Menlo Park, CA: AAAI Press.

Freedman, R., 2000. Using a Reactive Planner as the Basis for a Dialogue Agent. *Proceedings of the Thirteenth Florida Artificial Intelligence Research Symposium (FLAIRS 2000).* Orlando.

Heffernan, N. T., and K. R. Koedinger, 2000. Intelligent Tutoring Systems are Missing the Tutor: Building a More Strategic Dialog-Based Tutor. In *Building Dialogue Systems for Tutorial Applications, Papers of*

the 2000 AAAI Fall Symposium, edited by C. P. Rose and R. Freedman, 14-19. Technical Report FS-00-01. Menlo Park, CA: AAAI Press.

MacGregor, R, 1991. The Evolving Technology of Classification-Based Knowledge Representation Systems. In *Principles of Semantic Networks: Explorations in the Representation of Knowledge,* edited by J. Sowa. San Mateo, CA: Morgan Kaufmann.

Mitchell, T, 1997. *Machine Learning*. McGraw-Hill.

NCTM, 1989. Curriculum and Evaluation Standards for School Mathematics. National Council of Teachers of Mathematics. Reston, VA: The Council. See also http://standards-e.nctm.org/index.htm.

Popescu and Koedinger, 2000. Towards Understanding Geometry Explanations. In *Building Dialogue Systems for Tutorial Applications, Papers of the 2000 AAAI Fall Symposium,* edited by C. P. Rose and R. Freedman, 80-86. Technical Report FS-00-01. Menlo Park, CA: AAAI Press.

Renkl, A., R. Stark, H. Gruber, and H. Mandl, 1998. Learning from worked-out examples: the effects of example variability and elicited self-explanation. *Contemporary Educational Psychology*. 23. 90-108.

Rose, C. P. and R. Freedman, 2000 (editors). *Building Dialogue Systems for Tutorial Applications*. *Papers from the 2000 AAAI Fall Symposium*. Technical Report FS-00-01 Menlo Park, CA: AAAI Press.

Rose, C. P., and A. Lavie, 1999. LCFlex: An Efficient Robust Left-Corner Parser. User's Guide, Carnegie Mellon University.

Wiemer-Hastings, P., A. C. Graesser, D. Harter, and the Tutoring Research Group, 1998. The Foundations and Architecture of AutoTutor. In *Proceedings,ITS '98,* edited by B. P. Goettl, H. M. Halff, C. L. Redfield, and V. J. Shute, 334-343. Berlin: Springer Verlag.