TOWARD COGNITIVE TUTORING IN A COLLABORATIVE, WEB-BASED ENVIRONMENT

BRUCE M. MCLAREN, KENNETH R. KOEDINGER, MIKE SCHNEIDER
Human-Computer Interaction Institute, Carnegie Mellon University
Pittsburgh, PA USA
E-mail: bmclaren@cs.cmu.edu, koedinger@cmu.edu mike.schneider@cs.cmu.edu

ANDREAS HARRER, LARS BOLLEN
Collide Research Group, University Duisburg-Essen, Duisburg, Germany
E-mail: harrer@collide.info, bollen@collide.info

While intelligent tutoring has been applied to collaborative learning environments, it has met with little success so far because of the complexity involved in adding a tutoring component to a collaborative environment. We propose to tackle this problem by using Cognitive Tutors as the basis for our approach and by applying a technique we call Bootstrapping Novice Data (BND). The BND approach involves feeding student log files from a problem-solving tool into tutor development software to create the beginnings of a tutor for the tool. We describe an initial implementation of our approach in which Cool Modes, a collaborative software tool, is integrated with the Behavior Recorder, tutor-authoring software that supports development by demonstration. We show how our initial implementation provides a foundation for an intelligent tutor for collaboration but also discuss some of the challenges ahead.

1 Introduction

Intelligent Tutoring Systems (ITS) [12] have typically been developed for one-on-one (machine-to-student) instruction. We are interested, however, in how we could integrate tutors into collaborative, web-based learning environments. While efforts have been made toward providing tutoring in collaborative environments [3], [6] there are many and varied challenges still ahead. Consider, for instance, the complexities of building tutors for computer-mediated collaborative environments in which users in remote locations, communicating over the Internet, work together to solve problems [14]. The large space of possible actions and interactions between users, in addition to the problem representation itself, make it more challenging to analyze learner behavior and build tutors for collaborative tasks than for the single-student case.

As an alternative to the traditional approach to tutor development and as a step toward addressing the complexities of a collaborative environment [4] we propose a tutor-development approach that leverages actual problem-solving data not only to guide tutor design, as has been done before [8], but also to contribute directly to tutor implementation. In this approach, called *Bootstrapping Novice Data* (BND), we provide groups of collaborating novice users with a computer-based tool, let them attempt to solve problems with the tool, and record that problem-solving activity in a tutor-specific representation. This integrated record of user activity has two important benefits: (1) we learn a great deal about how users engage in collaborative problem-solving activities, and (2) it provides the initial representational structure for a tutor that could support collaboration. Our initial step in this direction has been to develop a prototype integration between a system for authoring collaborative modeling software, Cool Modes (Collaborative Open Learning and MODEling System) [11], and a tutor authoring environment, the Cognitive Tutor Authoring Tools (CTAT) [7]. The Cool Modes software generates computer log files of user activity that are, in turn, used by CTAT as an initial representation of the tutor. In this paper, we discuss how we have effected this integration and how we could use this model as a way of collecting user data to create an initial, skeletal tutoring system.

An important underpinning of this work is the notion of component-based development. Our approach relies on taking an existing software application (what we term a *tool*) and integrating it, with little or no modification, with a tutor or tutor agent. Using off-the-shelf or pre-existing software as the basis for building tutoring systems could result in substantial time savings, as compared to the traditional approach of building tutors *from scratch* [9], [13]. This is particularly important in developing tutors for collaboration, as the underlying interaction model is much more complex than in the single-student scenario.

A longer-term aim of our work, which will be greatly facilitated by both the component-based approach and our initial steps in developing BND, is to explore how we can fully integrate cognitive tutoring techniques in a computer-mediated collaborative environment. We want to use the integration of Cool Modes and CTAT not just as a means of leveraging user log files to help in building a skeletal, initial tutor, but also as a step toward developing a fully integrated, pedagogical model to provide real-time tutoring in Cool Modes and other collaborative software environments.

2 Creating the Initial Representation of a Tutor for Collaboration

The BND process we have developed to create an initial, skeletal model of a tutor through log data involves the integration of two software components, each with complementary features.

Cool Modes is a collaborative software tool designed to support ”conversations” and shared graphical modelling facilities between collaborative learners [11]. It is a domain independent tool that supports a variety of modelling and learning tasks and provides users with various plug-in objects, such as Petri nets, a turtle programming environment, text widgets, and a ”chat” area, each of which has its own semantics and underlying representation. All of the Cool Modes objects are available on a palette from which students may drag and drop objects into workspaces to build models. Cool Modes is extensible; new objects adhering to a well-defined API may be added to the palette as required. Translation between different object types is achieved through reference frames, a set of entities and rules that facilitate semantic object mapping. Cool Modes also provides operational facilities, such as execution of simulations and automated calculations. Each Cool Modes user runs a client program that contains a private workspace in which objects can be privately created and updated. In addition, all users have access to a shared workspace, which is rendered in all of the collaborating clients and may be updated by any of the participants.

The Cool Modes client program communicates with a server, called MatchMaker [5] that maintains the shared workspace and handles all communication between the collaborating clients.

Cool Modes does not assess or critique students' solutions apart from helping the students create syntactically correct models and allowing them to execute and observe the models. Thus, the tutoring component of the integration is provided by **CTAT** [7], an authoring tool for intelligent tutors. A Cognitive Tutor is composed of a problem representation and a set of production rules that model both desired and buggy behavior. Cognitive Tutors are difficult to develop, typically requiring AI programming expertise. On the other hand, a specialized type of Cognitive Tutor also supported within CTAT is a Pseudo Tutor, a tutor developed by demonstration. A Pseudo Tutor behaves much like a regular Cognitive Tutor, except that it only provides instruction for a single problem instance and is much easier to develop. A software component of CTAT called the Behavior Recorder (BR) is used to record all of the actions taken when demonstrating a problem solution and stores the actions in a structure called a behavior graph. Each edge of a behavior graph represents an action taken by the student on a particular widget of the GUI. More specifically, each action is represented as a Selection (the GUI widget selected, such as TextArea1), Action (the type of action taken, such as `Update Text`), and Input (the value provided as input, such as `20`). Each node of the graph represents a state of the interface after traversing a path of edges from the root to that node.

The basic idea of Bootstrapping Novice Data, depicted in Figure 1, is to use a tool, in this case Cool Modes, to generate a log of user actions and have those actions recorded in the BR, providing the beginnings of a real tutor. Translated user log files (similar in format to [4]) , which contain recorded user actions, are provided as input to the BR. Because the BR is a component with a well-defined message interface (i.e., it accepts XML messages we call Dormin Messages), this integration was relatively easy to develop. All that was required was a translator, using XSL-T, to take the Cool Modes XML log files and convert them to XML Dormin messages.

The BND approach involves having groups of collaborating users, depicted on the left side of Figure 1, generate (possibly different) correct and faulty solutions to the same problem. The logs of the different collaborating groups are then translated into a single behavior graph in the Behavior Recorder. After the behavior graph is generated, a tutor author manually updates it by adding hints and bug messages, annotating buggy paths, and adding skills to the edges. Not only does the BND approach provide the tutor author with examples of actual correct and buggy paths taken by users, it also presents the author with traversal frequencies of those paths. The edge traversal counts are good indicators of which of the correct solution paths might be considered primary, which secondary, as well as which types of error paths occur frequently enough to merit writing specific bug messages.

Instead of an individual author crafting Pseudo Tutors from scratch, tapping only their own experience or incorporating student data by hand, as in traditional ITS development, the BND approach leverages the empirical problem-solving data of a wide range of users engaged in actual, collaborative problem-solving. This

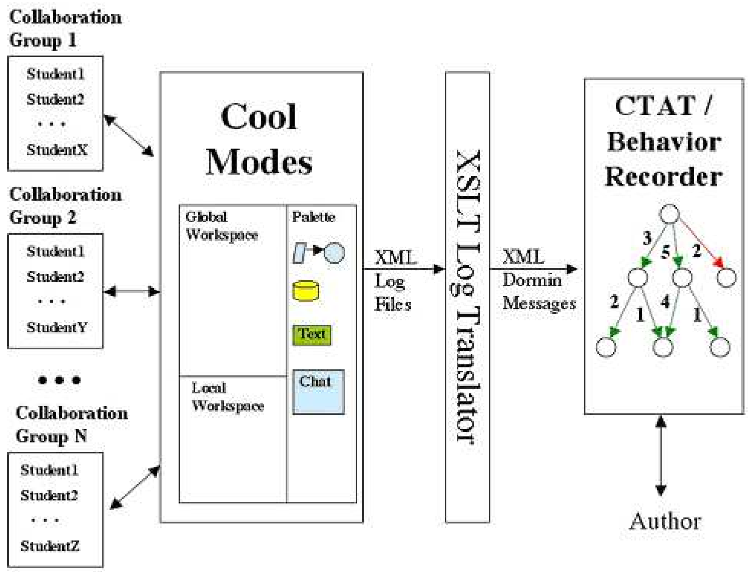

Figure 1. Bootstrapping Novice Data through the integration of Cool Modes and CTAT

approach contrasts sharply with the usual method of developing single-student ITSs in which a domain expert author first creates *expert* problem solutions. The critical advantage of BND is how it directly captures and encodes incorrect and inefficient novice solutions - information that will prove invaluable in building a full ITS for a collaborative system.

3 An Example: The NASA Exercise for Group Dynamics

We tested our BND approach and integration of Cool Modes and BR using a scenario called the NASA exercise. In this exercise, a group of students is presented with a fictional space travel problem. After their shuttle crashes on the light side of the moon, the students must decide which of a set of 15 items on the shuttle (e.g., matches, a bottle of oxygen, a bottle of water) are most important for survival while journeying to the mother ship 200 miles away on the moon's surface. The task facing the students is to assign a strict priority ordering to all items (e.g., bottle of oxygen 1 (highest), matches 15 (lowest)). Using scientific knowledge about light, the need for oxygen, the atmosphere of the moon, etc., NASA has proposed an optimal solution to the problem. The game is typically first played by each individual and then as a collaborative group. Interestingly, the collaborative solution is typically better than the average individual solution and even better than the best individual's.

Both the individual and group phases of the exercise can be easily conducted in Cool Modes using a visual discussion language which employs text cards for the items, numerical indicators for priorities, and links to connect items to priorities. We obtained several potential solutions, both correct (shown in figure 2) and incorrect, with a simplified scenario of 5 items (15 is the usual number), by logging Cool Modes sessions.

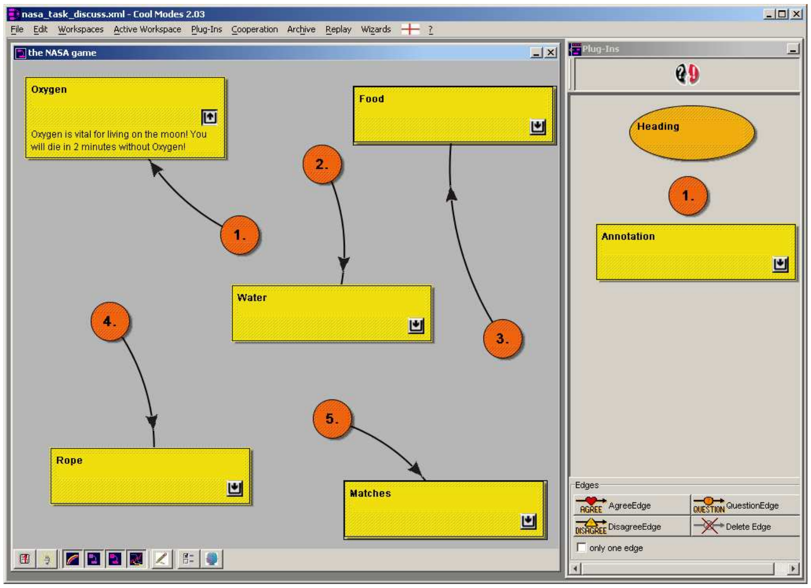

Figure 2. Correct solution of the simplified NASA exercise within Cool Modes

As per Figure 1, the Cool Modes XML log files were transformed by XSLT into XML Dormin messages. These were then fed into CTAT's Behavior Recorder, resulting in a behavior graph with weighted edges, showing the frequency of steps taken by the users across all solution attempts. This graph is shown in Figure 3.

The left branch at the top represents the fact that 4 of the users correctly chose a bottle of oxygen as the highest priority item (by connecting priority 1 to the text card labelled `oxygen` in Cool Modes), while the right branch at the top indicates that 2 users first (and also correctly) assigned matches as the lowest priority item. The branch emanating from state11 to state12 is a buggy path, in which oxygen was assigned a low priority (4). Since some of the users naturally chose different, yet still correct sequences of priority assignments, the graph has some confluent paths. These are equally valid paths in solving the problem.

The author has not yet manually updated the behavior graph shown in Figure 3. For instance, the author would ultimately mark the path on the right below state11 as buggy, which would change the selection/action/input of the rightmost edge emanating from state11 to red (the color we use to indicate a buggy action). This, in turn, would result in the deletion of all states on the path below state12, since the Behavior Recorder assumes that all buggy states are dead ends.

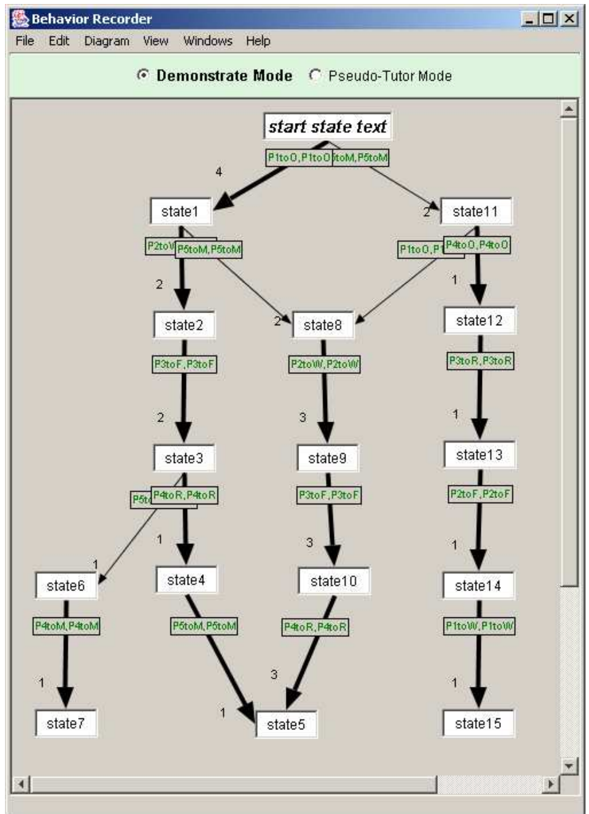

Figure 3. The Behavior Recorder after multiple user solutions of the NASA exercise

4 Discussion and Conclusion

The NASA exercise illustrates the value of the BND approach and the Cool Modes / Behavior Recorder integration. By having students attempt to solve this problem, we not only record the variety of paths taken, but we actually produce the backbone representation of an actual tutor. Given this initial representation, an author then annotates the behavior graph with hints, bug concepts, and bug messages. The traversal counts guide the author by indicating which paths are anomalous and which are common solutions/mistakes. An annotated behavior graph can be run as a fully functioning Pseudo Tutor.

But there is still much work to be done. The NASA problem, while intended to be tackled collaboratively, was, for our purposes, attempted only by individual users in order to perform a simple test of the Cool Modes / BR integration. To make the BND process work for actual collaboration, there is still research and development work required. For instance, the behavior graphs in typical mid-range to difficult problems are likely to be quite bushy, interconnected, and thus difficult to read and manipulate manually; for more complex problems it is even hard to grade solutions as correct or buggy. One of our goals, therefore, is to write software to automatically read and evaluate the graphs and, in turn, provide advice to authors (for instance, that an infrequently traversed edge may be buggy). One very basic change to the BR required to support collaboration is a way to record the user who performed each action, information Cool Modes has readily available. Since the BR was originally designed to support single-student tutoring, this data is currently not captured in the behavior graph, so we will extend the selection-action-input representation to include and reason about an `Actor` field. Since collaboration environments typically involve dynamic instantiation of objects (e.g., new Petri net nodes), the Behavior Recorder must also be capable of handling dynamic object definition and recognition. In other words, the BR must be able to identify similar selection-action-inputs across collaborative sessions with selections referring to different instances of objects. Since Cool Modes uses a consistent, internal naming scheme, we can leverage this to identify common objects across sessions. Ultimately, for collaborative systems in general, we need mapping tables to help the BR identify similar nodes and paths across sessions.

We may also need a more powerful representation than behavior graphs for modelling behavior and delivering tutoring for more complex collaborative activities. Existing collaborative systems typically employ more complex models, such as action-based collaboration analysis [10], which derives higher-level activity descriptions from user actions using plan recognition, or Hidden Markov Models [15], which have been used to model effective and ineffective student interactions. The production system approach of full Cognitive Tutors appears to be a promising way to model more complex collaborative behavior. In particular, we could treat collaboration as a special type of meta-cognitive behavior, by specifying a plausible model of collaboration similar to what we've done in the area of help seeking [1], [2]. Given this direction, the collaborative tutor building approach described in this paper could be viewed as an initial cognitive modelling / cognitive task analysis, initial test case development, and programming-by-demonstration input for building full Cognitive Tutors.

A clear, but we believe appropriate, limitation of the direction outlined in this paper is emphasis on student problem-solving actions, rather than on collaborative social activities such as dialogue between collaborators [3]. While we acknowledge that dialogue is important to many, if not most, collaborative activities, we believe that a significant class of collaborative problems can be dealt with by computer support at the level of observed actions. Our focus is also a matter of tackling one piece of the complex collaboration puzzle at a time.

In general, to reach the ultimate goal of deploying cognitive tutoring in a collaborative work environment, we need to better understand the intersection of collaborative workspaces, such as Cool Modes, and individualized cognitive tutoring, such as that provided by CTAT. By applying BND techniques and taking a component-based approach, we are well positioned to experiment with cognitive tutoring in a collaborative environment.

References

1. Aleven, V., McLaren, B. M., Roll, I., and Koedinger, K. (2004). Toward Tutoring Help Seeking: Applying Cognitive Modeling to Meta-Cognitive Skills. To appear in: Proc. of Intelligent Tutoring Systems (ITS), Maceio, Brazil.
2. Aleven, V., McLaren, B. M., and Koedinger, K. (forthcoming). Towards Computer-Based Tutoring of Help-Seeking Skills. In: Help Seeking in Academic Settings: Goals, Groups and Contexts, Karabenick, S. ,Newman, R. (eds)
3. Goodman, B., Hitzeman, J., Linton, F., and Ross, H. (2003). Towards Intelligent Agents for Collaborative Learning: Recognizing the Role of Dialogue Participants. In Proc. of Artificial Intelligence in Education (AIED), IOS Press, Amsterdam.
4. ISO/IEC JTC1 Subcommittee36 Workgroup2 (2003). Collaborative Technology - Collaborative Workplace - http://collab-tech.jtc1sc36.org/
5. Jansen, M. (2003) Matchmaker - a framework to support collaborative java applications. In Proc. of Artificial Intelligence in Education (AIED), IOS Press, Amsterdam.
6. Lesgold, A., Katz, S., Greenberg, L., Hughes, E., Eggan, G. (1992). Extensions of Intelligent Tutoring Paradigms to Support Collaborative Learning. In S. Dijkstra, H. Krammer, J. van Merrienboer (Eds.), Instructional Models in Computer-Based Learning Environments. Berlin: Springer-Verlag, 291-311.
7. Koedinger, K. R., Aleven, V., Heffernan, N., McLaren, B. M., and Hockenberry, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. To appear in: Proc. of ITS, Maceio, Brazil.
8. Koedinger, K. R. and Terao, A. (2002). A cognitive task analysis of using pictures to support pre-algebraic reasoning. In C. D. Schunn, W. Gray (Eds.), Proc. of Cognitive Science Society, 542-547.
9. McArthur, D., Lewis, M. W., and Bishay, M. (1996). ESSCOTS for learning: Transforming commercial software into powerful educational tools. Journal of Artificial Intelligence in Education, 6 (1), 3-33.
10. Muehlenbrock, M. (2001). Action-based Collaboration Analysis for Group Learning. IOS Press, Amsterdam.
11. Pinkwart, N. (2003). A Plug-In Architecture for Graph Based Collaborative Modeling Systems. In Proc. of Artificial Intelligence in Education (AIED), IOS Press, Amsterdam.
12. Polson, M. C. and Richardson, J. J. (1988). Foundations of Intelligent Tutoring Systems. Lawrence Erlbaum Associates Publishers.
13. Ritter, S. and Koedinger, K. R. (1996). An Architecture For Plug-In Tutor Agents. In: Journal of Artificial Intelligence in Education, 7 (3 / 4), 315-347.
14. Suthers, D. D. (2003). Representational Guidance for Collaborative Learning. In Proc. of Artificial Intelligence in Education (AIED), IOS Press, Amsterdam.
15. Soller, A. and Busetta, P. (2003). An Intelligent Agent Architecture for Facilitating Knowledge Sharing Communication. Proc. of Workshop on Humans and Multi-Agent Systems at AAMAS-03, Melbourne, Australia, 94-100.