Knowledge Co-construction and Initiative in Peer Learning Interactions

BY

CYNTHIA KERSEY
B.B.A., University of Wisconsin - Madison, 1985
M.S., Governors State University, 2001

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2009

Chicago, Illinois

To John P. Nurse,

who told me girls can do anything that boys can.

I wish you could see this.

ACKNOWLEDGMENTS

Many people have contributed to this research and it is a pleasure for me to acknowledge as many of them as I can.

First, and most importantly, my advisor Barbara Di Eugenio supported me in too many ways to list. Secondly, Pam Jordan, who was like a co-advisor, helped run experiments and build systems and provided me with encouragement when I needed it the most. I'd also like to thank my other committee members — Martha Evens, Leilah Lyons, Tom Moher, and Robert Sloan — along with Sandy Katz for their thoughtful suggestions.

I've been incredibly lucky to work with a great group of graduate students in the NLP lab — Davide Fossati and Joel Booth, who were always available and willing to discuss a research question with me, Lin Chen, Swati Tata and Alberto Tretti.

To earn a PhD at this stage in my life required an incredible amount of juggling of responsibilities and I couldn't have done it without my family and friends. I would like to thank my mother, brother and sister-in-law for their unfailing support and encouragement, Shirley Kersey for being my role model, Dave Kersey, and my dear friend, Mary Rittgers, who never failed to step in when I needed an extra set of hands. But most of all, I'd like to thank my children, Alyssa and Danny, for putting up with late dinners, missed events and a mom who seemed to be *always* working.

ACKNOWLEDGMENTS (Continued)

This work has been supported by the National Science Foundation (ALT-0536968) and in part by the Office of Naval Research (N000140010640).

CK

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF TABLES (Continued)

LIST OF FIGURES

LIST OF ABBREVIATIONS

CS	Computer Science
CSCL	Computer Supported Collaborative Learning
ITS	Intelligent Tutoring System
KC	Knowledge Component
KCC	Knowledge Co-Construction
NL	Natural Language
NLU	Natural Language Understanding
SMDP	Student Model/Domain Planner

SUMMARY

Peer learning has been shown to be an effective mode of learning for all participants. However the study of peer learning from a computational perspective is still in the early stages. In this research, I developed a computational model of peer learning and embedded this model in a peer learning agent.

To endow the agent with appropriate behaviors, I undertook an extensive corpus analysis in order to identify correlates of knowledge co-construction. This construct explains the effectiveness of peer learning by postulating that learning is enhanced when students work together to construct knowledge. I identified that linguistically based initiative shifts seem to capture the notion of collaborative construction. In the corpus analysis, I found a strong relationship between knowledge co-construction and initiative shifts and moderate correlations between initiative shifts and learning.

The results of this analysis were incorporated into KSC-PaL, an artificial agent that can collaborate with a human student via natural-language dialog and actions within a graphical workspace. Evaluations of KSC-PaL showed that the agent was able to encourage shifts in initiative in order to promote learning and that students learned using the agent.

CHAPTER 1

INTRODUCTION

In recent years there has been an increase in interest in the area of collaboration and collaborative learning. While collaboration can be unsuccessful when members refuse to contribute or dominate the interaction (Barron, 2000), groups working together cooperatively are able to arrive at solutions that none could come up with individually. Students working together have more frequent generation of new ideas and a higher level of reasoning (Johnson and Johnson, 1999).

Research also shows that collaboration promotes learning, potentially for all of the participants (Brown and Palincsar, 1989; Fisher, 1993; Tin, 2003). Similarly, studies in peer tutoring demonstrate that there are cognitive gains for both the tutor and the tutee (Birtz et al., 1989; Cohen et al., 1982; Rekrut, 1992).

In addition to studies of collaborative learning, collaboration in dialogue has long been researched in computational linguistics (Chu-Carroll and Carberry, 1998; Constantino-González and Suthers, 2000; Jordan and Di Eugenio, 1997; Lochbaum and Sidner, 1990; Soller, 2004a; Vizcaíno, 2005). There has also been research in collaborative dialogue in the area of Computer-Supported Collaborative Learning (CSCL), however much of the research focuses on conversational grounding (Baker et al., 1999; Dillenbourg and Traum, 1999) and not the relationship with learning.

While there has been a focus on using natural language for intelligent tutoring systems (Evens et al., 1997; Graesser et al., 2004; VanLehn et al., 2002), peer to peer interactions are notably different from those of expert-novice pairings, especially with respect to the richness of the problem-solving deliberations and negotiations. Using natural language in collaborative learning could have a profound impact on the way in which educational applications engage students in learning.

The study of peer learning from a computational perspective is still in the early stages. Although some researchers have attempted to develop simulated peers (Chan and Baskin, 1988; Vizcaíno, 2005), there is very little research on what constitutes effective peer interaction to guide the development of effective peer learning agents.

Previous research has suggested several mechanisms that explain why peer learning is effective for all participants. Among them are: self-directed explaining (Chi et al., 1994), other-directed explaining (Ploetzner et al., 1999; Roscoe and Chi, 2007) and knowledge co-construction (KCC) (Hausmann et al., 2004). KCC episodes are defined as portions of the dialogue in which students are jointly constructing a shared meaning of a concept required for problem solving. This last mechanism is the most interesting from a peer learning perspective because it is a truly collaborative construct and also because it is consistent with the widely accepted constructivist view of learning.

Since KCC is a high-level concept, it doesn't provide insights into what happens within any such episode. Although (Hausmann et al., 2004) extends the analysis of KCC by incorporating relations such as *elaborate* and *criticize* within KCC episodes, such relations are very difficult for an artificial agent to recognize. Thus, a simpler but principled model of KCC is required to implement a peer learning agent.

Potentially, the concepts of control and initiative from computational linguistics can play a part in recognizing KCC. It is generally accepted that one or more threads of control pass between participants in a dialogue. This implies that tracking the transfer of control can be useful in determining when KCC is occurring. In theory, frequent transfer of control between participants would indicate that they are working together to solve the problem and perhaps also to construct knowledge.

Our domain in this research is problem solving in basic data structures and algorithms, which is part of the foundations of Computer Science. While in recent years, interest in CS in the US has dropped dramatically, CS is of enormous strategic interest, and is projected to foster vast job growth in the next few years (AA. VV., 2006). I believe that by supporting CS education in its core we can have the largest impact on reversing the trend of students' lack of interest. My belief is grounded in the observation that the rate of attrition is highest at the earliest phases of undergraduate CS curricula. This is due in part to students' difficulty with mastering basic concepts (Katz et al., 2003), which require a deep understanding of static structures and the dynamic procedures used to manipulate them (AA. VV., 2001). These concepts also require the ability to move seamlessly among multiple representations, such as text, pictures, pseudo-code, and real code in a specific programming language.

Surprisingly, few educational software systems address CS topics, for example teaching a specific programming language like LISP (Corbett and Anderson, 1990) or database concepts (Mitrović et al., 2004). Additionally, basically they are all intelligent tutoring systems, where the relationship between the system and the student is one of “subordination”. Only two or three of these ITSs address foundations, including: Autotutor (Graesser et al., 2004) which addresses basic literacy, but not data structures or algorithms; ADIS (Warendorf and Tan, 1997) which tutors on basic data structures, but its emphasis is on visualization, and it appears to have been more of a proof of concept than a working system; ProPL (Lane and VanLehn, 2003) which helps novices design their programs, by stressing problem solving and design skills.

The goal of this dissertation is to derive a model of effective collaborative problem solving which is feasible to implement into an artificial agent. It consists of 3 subgoals.

1. Derive a model of knowledge co-construction that can be recognized by an artificial agent.
2. Analyze the role of initiative and balance of initiative in successful collaborative problem solving.
3. Embed this model in an artificial agent to investigate the effectiveness of using such a model with an artificial agent.

1.1 Contributions

This dissertation makes contributions in the areas of computer-supported collaborative learning, artificial intelligence in education and computer science education.

In performing a comprehensive analysis of collaborative problem solving, I

- validated the effectiveness of certain types of peer learning interactions
- derived a definition of task initiative for this domain
- annotated collaborative dialogues with knowledge co-construction, dialogue initiative, task initiative and rhetorical relations

I operationalized knowledge co-construction via shifts in task initiative. This included analyzing the relationship between initiative and learning, KCC and learning and initiative and KCC. Based on this analysis, I developed a model to recognize KCC episodes based on shifts of initiative and built classifiers to automatically identify task initiative.

I developed an artificial peer learning agent that encourages learning by managing shifts in task initiative. To create the agent I developed Bayesian network-based student models for peer-led interactions. Additionally, I derived methods to automatically identify and interpret student drawing and coding actions. I also developed a planner to track and manage shifts in task initiative which incorporated the results of an analysis of cues for initiative shifts to guide the agent's behavior.

My evaluation of the peer agent showed that students learned when working with the agent and those students whose interaction with the agent involved more shifts in initiative learned more. I also found that the agent was able to encourage initiative shifts through the use of cues identified in the analysis.

1.2 Outline

This dissertation is organized as follows:

Chapter 2 discusses related research in the areas of collaborative modeling, initiative, knowledge co-construction and peer learning environments.

Chapter 3 introduces the problem solving domain and describes the collection of collaborative problem solving episodes between human students using a computer-mediated interface. I used these interactions to derive my model of knowledge co-construction. This chapter also contains an analysis of the effectiveness of the problem solving episodes.

Chapter 4 details the extensive analysis of the collaborative problem solving episodes. I began with an initial analysis based on easily identifiable features of the dialogues. I then explored in more detail features of both dialogue and graphical actions in the shared workspace. This was followed by annotation of knowledge co-construction and initiative and in-depth analysis of the relationship between learning, knowledge co-construction and initiative.

Chapter 5 explains how I used machine learning to build a classifier to identify the initiative holder in an utterance. It also contains an analysis of methods for encouraging shifts in initiative.

Chapter 6 describes the design of the peer learning agent, KSC-PaL, which has as its core the TuTalk dialogue management system (Jordan et al., 2007). This includes the system architecture and the extensions to TuTalk that were required in order to create a peer learning agent. These extensions include an interface, a student model and a planner to manage shifts in initiative.

Chapter 7 describes the evaluation of two versions of KSC-PaL. One version of KSC-PaL attempts to maintain a high-level of shifts of initiative in order to encourage learning. In the second version, KSC-PaL does not encourage task initiative shifts and acts more as a tutor. My evaluation explored the differences in learning calculated using pre-test and post-test score. I also analyzed the ability of KSC-PaL to manage shifts in initiative. Lastly, I examined student satisfaction based on a user survey.

Chapter 8 contains conclusions and plans for future work.

CHAPTER 2

RELATED WORK

In this chapter, I will review a selection of research from several different areas that are related to my work. I begin with collaborative modeling, followed by initiative and collaboration. Next, I review related work in the area of knowledge co-construction. I conclude with a survey of collaborative learning environments.

2.1 Collaborative Modeling

Sidner proposes a model of collaboration based on a series of exchanges consisting of proposal/acceptance or proposal/rejection (Sidner, 1994). She suggests that collaboration requires negotiation to share and reach agreements on individual beliefs in order to achieve common goals. Chu-Carroll and Carberry build on this work with a recursive Propose-Evaluate-Modify framework (Chu-Carroll and Carberry, 1998) which was developed by analyzing three corpora of collaborative planning dialogues. Their analysis suggests that collaborative behavior follows this pattern: (1) after a proposal is presented, the partner evaluates it based on her private knowledge to determine whether to accept or reject the proposal; (2) if the partner doesn't have enough information to make a decision, she will initiate an information exchanging dialogue with the other party; (3) if she rejects the proposal, she will initiate a negotiation dialogue to modify the proposal instead of rejecting it outright.

While the above models capture the essence of collaborative exchange, they are difficult to extend to domains outside of the ones in which they were developed. The Balance-Propose-Dispose model of Di Eugenio et al. (Di Eugenio et al., 2000) is a model that is more widely applicable to collaboration in general. To develop the model, computer-mediated dialogs where collected where two people work together on a design task — to buy furniture for a living room and a dining room. Each participant was given a separate budget and inventory and the dyads collaborated to furnish the rooms given their joint inventory and budget. Using the collected dialogues, a model was developed based on an agreement process that consists of three stages:

1. Balance. When neither agent is able to deliberate to the point where they can make a commitment to a change in the problem state, information is balanced among the participants and they will partially deliberate. This phase continues until at least one of the agents is in a position to fully deliberate and make a commitment to a change in the problem state.
2. Propose Options. At this stage an option has been fully deliberated and one of the pair makes a proposal for a change in the problem state. This stage can only be reached when enough information has been exchanged for a proposal to be made.
3. Dispose of proposal. After a proposal is made, it is either implicitly or explicitly accepted or rejected by the partner.

The balance phase of the model seems similar to a KCC episode. My modeling of KCC is potentially useful in extending this model for use by an artificial agent.

2.1.1 Initiative and Collaboration

One commonly referenced work in the area of dialogue initiative is that of Walker and Whittaker (Walker and Whittaker, 1990). In dialogue each participant will at some point take the initiative or conversational lead in the dialogue. When initiative passes from one participant to another, so does control of the dialogue. It is through such mixed-initiative dialogue that information is exchanged. Using rules proposed by Whittaker and Stenton (Whittaker and Stenton, 1988) to parse the dialogue into control segments, Walker and Whittaker suggest that each control segment corresponds to different subgoals in the discourse plan. They validate this claim by showing the distribution of anaphora to the various control segments in both advisory dialogues and task-oriented dialogues.

Jordan and Di Eugenio propose that control and initiative are two separate features in collaborative problem solving dialogues (Jordan and Di Eugenio, 1997). While control and initiative might be synonymous for the dialogues analyzed by Walker and Whittaker where a master-slave assumption holds, it is not the case in collaborative dialogues where no such assumption exists. Jordan and Di Eugenio argue that the notion of control should apply to the dialogue level, while initiative should pertain to the problem-solving goals. The pair shows several dialogue instances where control and initiative diverge and suggest that this distinction is important to assist in recognizing when problem solving is stalled, when work begins on a new part of the problem and when an earlier commitment might be blocking the attainment of a solution.

In a similar vein, Chu-Carroll and Brown also argue for a distinction between control and initiative, which they term dialogue initiative and task initiative (Chu-Carroll and Brown, 1998). They suggest that in a collaborative problem-solving environment it is necessary to differentiate between the lead in the pursuit of the current problem-solving goal (task initiative) and the lead in the current discourse focus (discourse initiative). To aid in this distinction they propose a model that tracks initiative shifts between participants based on the combined effect of a set of cues. These cues can be recognized based on linguistic and domain knowledge alone and fall into three categories: explicit cues, discourse cues and analytical cues. Explicit cues are explicit requests by the speaker to give up or take over initiative. Discourse cues are cues that can be recognized using linguistic information from an utterance while analytical cues require an evaluation of the speaker's proposal using the hearer's private knowledge. Using their model, Chu-Carroll and Brown were able to correctly predict task initiative holders in 99.1% of turns and dialogue initiative holders in 87.8% of turns in the corpus on which the model was trained. Additionally, they were able to increase prediction accuracies by 2-4% for task initiative holders and by 8-13% for dialogue initiative holders in other domains as compared to a simple prediction method without use of cues.

Guinn (Guinn, 1998) agrees that there is a distinction between the two types of initiative. In his study, problem-solving agents collaborate to solve a problem. These agents take a top-down approach to planning. Each participant solves a goal by (1) seeing if the goal is trivially true or (2) decomposing the goal into subgoals. When a problem-solver cannot satisfy a goal trivially by looking it up in its knowledge base and it also cannot decompose the goal into subgoals, it has the option of requesting that the other participant satisfy the goal. However, the problem-solver will only exercise this option if it believes that the other participant is capable of satisfying the goal. Based on this restricted scenario, Guinn defines task initiative as belonging to the participant who dictates which decomposition of the goal will be used by both participants during problem-solving. Dialog initiative belongs to the participant who is expected, by both parties, to communicate next and it should always pass immediately to the participant who is best able to handle the current task. Despite the distinction between initiative types, he suggests that whoever has task initiative over the current goal also has the dialogue initiative. Initiative is attached to each goal and during problem-solving, initiative may shift back and forth between participants depending on which goals the two participants are working on, however the initiator of a goal retains initiative for that goal. Guinn does not differentiate between communicative goals and problem-solving goals and as such is able to refute the theory of multiple threads of control and equate dialog initiative with task initiative.

Strayer and Heeman (Strayer and Heeman, 2001) propose that initiative is held by the segment initiator, the participant that initiated the current topic under discussion. The non-initiator can make utterances that contribute to the purpose of the current discourse segment, but initiative remains with the segment initiator. In contrast to Chu-Carroll and Brown's model that suggests that dialogue initiative is subordinate to task initiative, Strayer and Heeman's definition makes initiative subordinate to intentional structure.

Core, Moore and Zinn undertook a systematic investigation of the role of initiative in the area of human-human tutorial dialogues (Core et al., 2003). Using a series of computer-mediated tutorial sessions collected in an earlier study, the dialogues were annotated for initiative using Whittaker and Stenton's initiative assignment rules. The study compared initiative in didactic tutoring versus Socratic tutoring. They found, contrary to their hypothesis, that students had initiative more often in didactic dialogues (21% of turns) than in Socratic dialogues (10% of turns). Additionally, they found no direct relationship between student initiative and learning. However, they did find that the Socratic dialogues were more interactive and this interactivity correlated with higher learning gains. Specifically, there were positive correlations between percentage of words produced by the student and learning gain, percentage of utterances produced by the student and learning gain and percentage of tutor utterances that were questions and learning gain. This led to a hypothesis that student language production is an indication of student knowledge construction. Additionally, they hypothesize that while students did not have dialogue initiative, they perhaps had task (learning) initiative as proposed by Chu-Carroll and Brown.

Jordan and Siler (Jordan and Siler, 2002) also explore student initiative in human tutorial dialogues. In their study they determine that a broader definition of initiative, "StudentInitiative", better captures the idea of topic control than do the models of dialogue initiative such as the commonly used definitions of Whittaker and Stenton and Walker and Whittaker which miss a large part (78%) of actual student initiative.

My work validates the work of those that distinguish between two types of initiative and I extend the definition of task initiative in a collaborative problem solving domain. Additionally, I expand on the concept of StudentInitiative, as well as Core, Moore and Zinn's work, but I approach it from the perspective of collaborative problem solving instead of human tutoring.

2.2 Knowledge Construction

Constructivism is a theory of learning that claims that knowledge is constructed by students rather than received and absorbed. Cognitive constructivism maintains that effective learning is dependent on individual self-monitoring, elaborations and representational constructions of the user (Resnick, 1989). It asserts that learning is particularly effective when it is situated in action and participation in activities (Ben-Ari, 1998).

Hausmann, Chi and Roy propose that constructivism is the force behind three potential mechanisms for learning during collaboration (Hausmann et al., 2004):

- Other-directed explaining where one peer explains to or instructs a partner
- Co-construction, which occurs when a peer shares knowledge, which can then be criticized or elaborated on by his/her partner
- Self-directed explaining or self-explaining with an audience

In this study, students with a limited knowledge of physics, worked in pairs to solve problems in the domain of kinematics. Learning gains were analyzed based on improvement from pre-test to post-test. All three of these mechanisms were shown to impact learning but to different degrees. Other-directed explaining led to learning gains for the listener 45% of the time. However, learning gains were most consistently produced by self directed explaining, occurring in 71% of the self-directed explaining episodes. But this effect was only for the speaker. Learning gains for listener occurred less often, on order with other-directed explaining. Co-construction, however, was beneficial to both the speaker and the listener when it occurred. Co-construction is defined as the joint construction of knowledge. This gives support to the constructivist perspective, that participation is important to learning. Also, this is the mechanism most relevant to collaboration, as it can only occur in a collaborative environment.

2.3 Collaborative Learning Environments

One system that is especially relevant to this research is the Clarissa (Collaborative Learning As Realised In Simple Simulated Agents) system (Burton et al., 2000) which allows the exploration of collaboration with respect to dialogue roles. The authors used artificial agents to study variations in distribution of dialogue roles. Successful collaboration was measured by the variety in dialogue roles used along with the perceived benefit from collaboration measured by changes in the agent's knowledge base.

In *free* collaboration, one agent selects a dialogue role and the other agent selects a different, non-conflicting dialogue role. For example, if the first agent selects *question* the other agent cannot select the role *question* or any role that conflicts with *question* such as *argue*. In *free* collaboration there was a tendency for dialogue roles to be used unevenly which had an uneven impact on the agents' knowledge bases. This suggests that students that collaborate in such a manner are likely to benefit unequally from the experience.

This contrasts with *MultiSwap* collaboration where roles are kept until there was a focus change. This type of collaboration was much more successful according to their metric. Such results would seem to support my hypothesis that initiative plays a role in successful collaborative learning.

A selection of other collaborative learning environments follows. These environments can be dividend into two general areas: peer learning agents and computer supported collaborative learning.

2.3.1 Peer Learning Agents

Peer learning agents fall into three areas. In one case the agent acts as a companion to the student during learning activities. The learning companion learns from the teacher along with the human student. Chan and Baskin (Chan and Baskin, 1988) developed a learning companion to explore competition between the student and the peer agent as well as collaboration between the student and the agent.

Other peer agents act as tutees who are taught or coached by a student. One study examined the role of a learning companion as a student of the human student in the domain of binary Boolean algebra (Uresti, 2000). This research explored the hypothesis that a learning companion with less proficiency than the human student would be beneficial for the student learning. The system implemented two companions with different levels of expertise. Results from a empirical evaluation suggested that subjects interacting with a less capable companion have a trend of more improvement than subjects interacting with a more capable companion. This is similar to what I found in my analysis of KCC, that students with a lower pre-test score benefited more from KCC.

Lastly, some peer agents encourage reciprocal tutoring and can play either the role of tutor or tutee. However, while the agent can take on different roles in different interactions, the role is fixed for the duration of the interaction. In (Chou et al., 2003; Chou et al., 2002), the agent and the student take turns playing the role of a tutor and a tutee. The system includes computational support for the student when playing the role of the tutor as well as an artificial agent that can take on the role of a tutor or a tutee.

2.3.2 Computer-Supported Collaborative Learning

CSCL provides supports for student collaboration. An example of such a system is the C-CHENE system (Baker and Lund, 1997; Baker et al., 1999). This system was designed to support dyads of students collaborating in the construction of diagrams of energy chains, which are qualitative models for energy storage, transfer and transformation.

One recent project in the area of computer science collaborative problem solving is COLER, an internet based collaborative learning environment focusing on the domain of entity relationship modeling (Constantino-González and Suthers, 2000). COLER's focus is how domain specific reasoning can guide the coaching of collaborative interaction. It provides users with a chat area along with an individual graphical workspace as well as a shared graphical workspace. The system enhances the collaborative process by using the socio-cognitive conflict theory, which states that discussion of conflicts in beliefs can lead to learning. Software-based coaches at both group and individual levels are used to find structural differences between students' individual and group solutions. The coaches then encourage the students to share and discuss the difference in their solutions. A decision tree is used to generate coaching advice.

HabiPRo is another system that is designed to assist in collaborative problem solving (Vizcaíno, 2005). It assists students working together on Java programming and facilitates interaction by using a "Simulated Student" to monitor both student actions and collaborative conversations for three negative situations - off-topic discussion, unbalanced participation level and problem solving impasses. Off-topic conversations are identified by matching the content of utterances to words in a database classified as either domain words or off-topic words. Participation level is monitored by tracking the frequency of utterances and actions, while learning problems are identified by comparing a proposed solution to the actual problem solution. When a negative situation takes place, the Simulated Student acts, trying to prevent any decrease in collaboration or motivation. Reported results show that the use of the Simulated Student allowed participants to solve more exercises and reduce off-topic discussion. Additionally, the Simulated Student was able to successfully identify passive students.

Coler and HabiPro were designed to assist in collaborative learning by monitoring for predefined measures of successful collaboration, while EPSILON (Soller, 2004a) uses student data to explore collaborative behavior and develop a model of collaboration. EPSILON contains a graphical interface that allows users to collaborate on object-oriented modeling problems. It provides a chat interface with sentence starters and an area to build object-oriented models which are shared among the participants. Data collected from sessions conducted with this system were used to create a model of knowledge sharing episodes, which are segments of interaction (including student utterances and workspace actions) in which one student attempts to present new knowledge to his peers and his peers attempt to understand it. New knowledge is considered knowledge that the receiving peer(s) did not know before the collaborative learning session. Dialogues were manually segmented to find successful and unsuccessful episodes of knowledge sharing. Each type was then used to train a Hidden Markov model (HMM). The models were shown to detect successful and unsuccessful knowledge sharing episodes 74% of the time. Soller also attempted to identify why such episodes were effective or ineffective by using both quantitative and qualitative analysis. A combination of HMMs, Multidimensional Scaling and clustering were used to produce groups of effective and ineffective HMMs (Soller, 2004b). Each grouping was then compared to a separately prepared qualitative analysis of student activity within each group and these combinations of analyses were used to create descriptions of effective and ineffective knowledge sharing episodes (Soller and Lesgold, 2003).

Another project that uses collected data to identify problems in collaboration is that of Goodman et al. (Goodman et al., 2005). This study uses a collaborative agent to intervene when it detects problems such as a dominant or passive participant or problem solving confusion. Like the EPSILON project, Goodman and his colleagues studied participants collaborating on a computer science topic. In this study the domain was software engineering problems using the Object Modeling Technique (OMT). Their initial goal was to use dialogue acts and participant instructional roles as predictors of successful collaboration and to intervene when such features were absent. Following the approach of Soller and Lesgold (Soller and Lesgold, 2003), Hidden Markov Models were used. However, because HMMs were unable to accurately detect poor collaboration, their goal was revised. Instead of trying to identify good and bad collaboration per se, the group chose to identify timely responses to questions and suggestions, as this is a feature of good collaboration. Using a combination of dialogue acts and some surface features, two neural networks were constructed - one to detect utterances requiring a response and the other to detect the response. The results from this model along with indicators of learner understanding and deliberative discussion which were derived from analyzing dialog acts were used to build a student model that was incorporated in an intelligent peer agent.

Research by Harrer et al. also uses student data to build a model of successful collaboration which is then used to assist in building tutors for collaboration (Harrer et al., 2006). Data was collected using a collaborative software tool, Cool Modes, which contains a chat feature as well as a graphical modeling facility. Although the chat facility was included, the study focused on using only the actions of the collaborators. Two studies were undertaken, each involving pairs of students solving a graphical modeling problem. The first study was used to identify properties of collaboration that might be important to successful collaborative experiences and the features in a graphical problem-solving domain that could be used to identify these properties. The five properties identified were conceptual understanding, visual organization, task coordination, task coherence and task selection. Task coherence is the ability of the dyad to use objects in a correct way, while visual organization refers to visually arranging those objects appropriately. Task coherence, task coordination and task selection reflect collaboration strategies. The second study was used to verify that these properties were indeed relevant and also that they were generalizable across different graphical modeling problems. After the second study, the group explored how to use only three different types of actions to build an appropriate tutor. The actions selected were chat actions (although content was ignored), move actions (repositioning an object in the workspace) and creation/deletion actions. They determined that each individual action is not sufficient for representing all of the identified dimensions. However, they had more success aggregating actions by number and participant. They suggest that this analysis could potentially be expanded by using data mining techniques to connect patterns of actions and by clustering sequences of actions at both the group and individual level.

The work by Soller, Goodman et al. and Harrer et al. are most similar to my work in that they build models of collaboration based on actual student data. Soller's project focuses on knowledge construction and sharing, however her primary focus is speech acts. Goodman's group also focuses on dialogue related features. Harrer and colleagues, however, exclusively use actions to model collaboration and do not relate these actions to knowledge construction.

Additionally, my work builds on the research of Chou et al., in that my peer learning agent can take on both the role of the more experienced and less experienced peer. However, unlike their agent, the interaction for my agent is not fixed for the duration of the interaction.

CHAPTER 3

DATA COLLECTION

In order to derive a model of peer learning, we collected dialogues between pairs of human students working together to solve problems in the domain of computer science data structures. In this chapter I will describe the domain, a pilot study to derive the interface and problems used in data collection, the collection of problem solving dialogues and a collection of baseline interactions.

The data structures that we focused on are (1) linked-lists, a set of data nodes connected by pointers (see example in Figure 1 drawing), (2) stacks, array-based last-in first-out structures and (3) binary search trees, a data structure where each node connects to at most two other nodes and whose ordering property makes it very efficient for retrieving data. This domain was chosen because data structures and their related algorithms are one of the core components of computer science education and a deep understanding of these topics is essential to a strong computer science foundation.

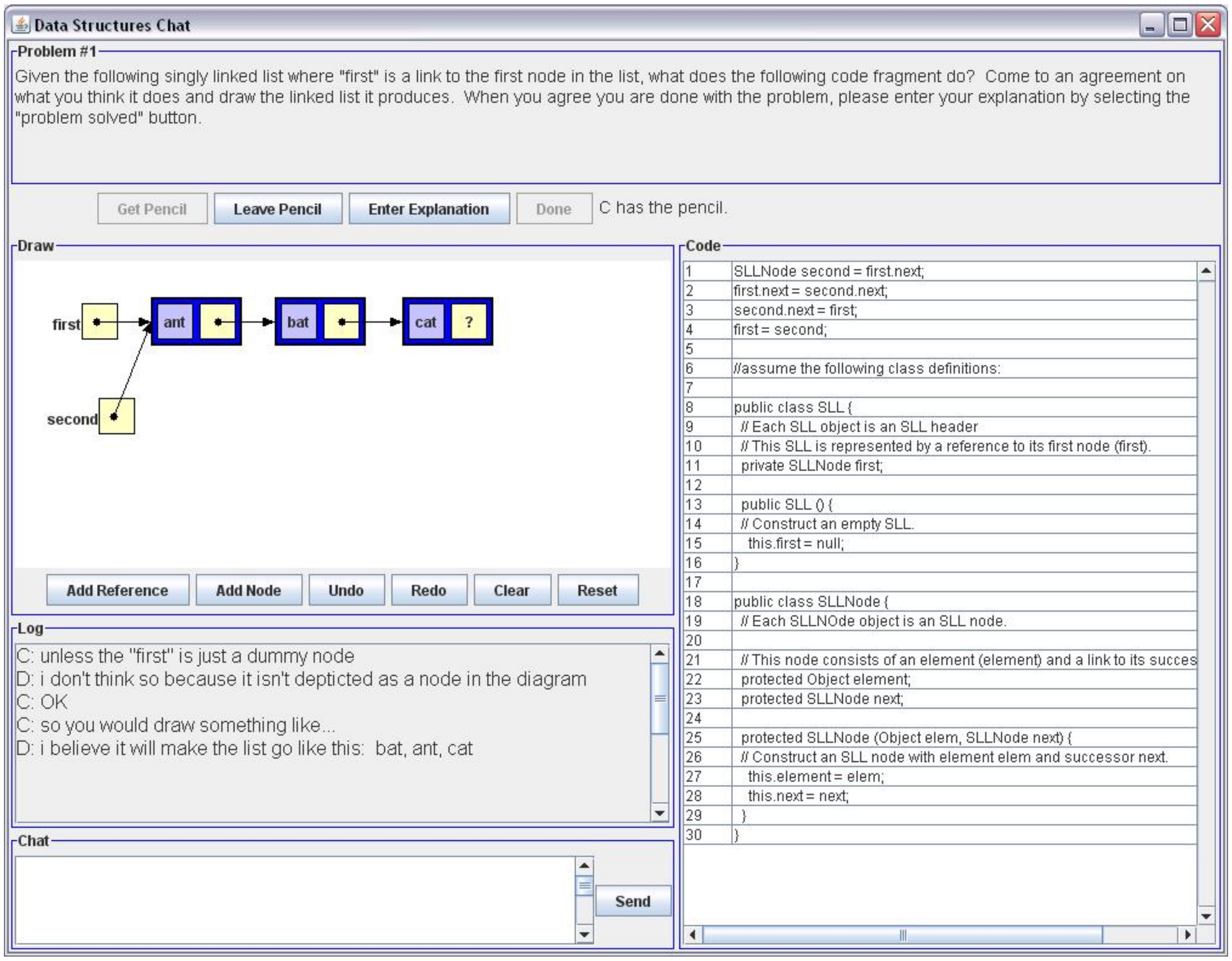
Before collecting the peer problem dialogues, we ran a pilot study in which we developed a set of data structure tasks and videotaped face-to-face collaborations of pairs of students solving these problems. These dialogues were then transcribed in order to:

1. verify that the task statements encourage collaboration
2. gauge whether students benefit from collaboration on these tasks

3. determine what other communicative channels students naturally use, in addition to natural language dialogue, that may be essential for sustaining collaborative behavior when moved to a computer-mediated environment

We verified that the tasks do encourage beneficial collaborations. Students discussed differences in their understanding of what the code presented to them did and what errors were within the code. Additionally, the post-test scores of most students were higher than their pre-test scores demonstrating that some learning did occur. We also observed that during their interactions, the peers frequently drew data structures and deictically referred to the code they were diagnosing or explaining. In addition they collaboratively marked up the code under discussion. Given these observations we developed a computer-mediated environment, originally based on a graphical interface developed by Davide Fossati for an intelligent tutoring system in the same domain. A computer-mediated environment was chosen to more closely mimic the situation a student faces when interacting with KSC-Pal, the artificial peer agent. This interface consists of four distinct areas (see Figure 1) :

1. Problem display: Displays the problem description that is retrieved from a database.
2. Code display: Displays the code from the problem statement. This section numbers each line of the code for easy reference by the students. Additionally, the students are able to use a menu-based facility to make changes to the code, such as crossing-out lines and inserting lines, as well as undoing these corrections.
3. Chat Area: Allows for user input and an interleaved dialogue history of both students participating in the problem solving. The history is logged for future analysis.



Given the following singly linked list where "first" is a link to the first node in the list, what does the following code fragment do? Come to an agreement on what you think it does and draw the linked list it produces. When you agree you are done with the problem, please enter your explanation by selecting the "problem solved" button.

C has the pencil.

```
C: unless the "first" is just a dummy node
D: i don't think so because it isn't depticted as a node in the diagram
C: OK
C: so you would draw something like...
D: i believe it will make the list go like this:  bat, ant, cat
```

```
SLLNode second = first.next;
first.next = second.next;
second.next = first;
first = second;

//assume the following class definitions:

public class SLL {
  // Each SLL object is an SLL header
  // This SLL is represented by a reference to its first node (first).
  private SLLNode first;

  public SLL () {
  // Construct an empty SLL.
    this.first = null;
}

public class SLLNode {
  // Each SLLNOde object is an SLL node.

  // This node consists of an element (element) and a link to its succes
  protected Object element;
  protected SLLNode next;

  protected SLLNode (Object elem, SLLNode next) {
  // Construct an SLL node with element elem and successor next.
    this.element = elem;
    this.next = next;
  }
}
```

Figure 1. The data collection interface

4. Drawing area: Here users can diagram data structures to aid in the explanation of parts of the problem being solved. The drawing area has objects representing nodes and links. These objects can then be placed in the drawing area to build lists, stacks or trees depending on the type of problem being solved.

The changes made in the shared workspace (drawing and code areas) are logged and propagated to the partner's window. In order to prevent users from making changes at the same time, I implemented a system that allows only one user to draw or make changes to code at any point in time. In order to make a change in the shared workspace, a user must request the "pencil" (Constantino-González and Suthers, 2000). If the pencil is not currently allocated to her partner, the user receives the pencil and can make changes in the workspace. Otherwise, the partner is informed, through both text and an audible alert, that his peer is requesting the pencil. The chat area, however, allows users to type at the same time, although they are notified by a red circle at the top of the screen when their partner is typing. While this potentially results in interleaved conversations, it simplifies communication between the peers.

The data collection sessions began with the presentation and signing of consent forms. Each student then individually took a pre-test (see Appendix A) to measure his/her knowledge of data structures prior to the collaborative interaction. This was followed by a short tutorial on using the interface. The students then engaged in the collaborative problem-solving activity using the computer-mediated interface. The session concluded with each student individually completing a post-test (identical to the pre-test) and a short questionnaire regarding the interaction.

When solving the problems, the participants were able to interact both verbally and through actions in the shared workspace as shown in the dialogue sample in Figure 2. In this excerpt the dyad is working on problem 1. After discussing a misconception of C's, in

```
14:01:56 C:  unless the "first" is just a dummy node
14:02:20 D:  i don’t think so because it isn’t depicted
             as a node in the diagram
14:02:28 C:  OK
14:03:13 C:  so you would draw something like...
14:03:24 D:  i believe it will make the list go like this:
             bat, ant, cat
14:03:40 C:  draw: add pointer second (n100)
14:03:44 C:  draw: move n100
14:03:46 C:  draw: link n100 to null
14:03:47 C:  draw: link n100 to n002
```

Figure 2. An excerpt from one of the pilot dialogues

14:03:40 C begins to illustrate the effect of the first line of code in the code sample given to the pair. First a pointer variable is added to the drawing and then in 14:03:47 it is made to point at a node in the drawing.

Thirty participants were recruited from introductory computer science courses at the University of Pittsburgh and the University of Illinois at Chicago. The majority of the participants were male students currently enrolled in a data structures course (see Table I). However, gender and course enrollment did not play a role in the formation of dyads. Participants were matched with a partner from the same university based on the scheduling restrictions of the participants.

Dialogues were collected for a total of 15 dyads where each dyad was presented with five problems, 3 linked list problems, 1 stack problem and 1 tree problem. The problems were either error diagnosis problems or code explanation problems (see Appendix C). The initial

	Data Structures Semester 1	Data Structures Semester 2	Other	Total
Pitt	**10**	**10**	**4**	**24**
Male	8	10	4	22
Female	2	0	0	2
UIC	**5**	**1**	**0**	**6**
Male	4	1	0	5
Female	1	0	0	1
Combined	**15**	**11**	**4**	**30**
Male	12	11	4	27
Female	3	0	0	3

TABLE I

DATA COLLECTION PARTICIPANTS

exercise was used to let the participants become acquainted with the interface. Students were allowed to ask questions regarding the interface and were limited to 30 minutes to solve this problem. The remaining exercises had no time limits, however the entire session (including administrative tasks and testing) could not exceed three hours. Therefore not all dyads completed all five problems.

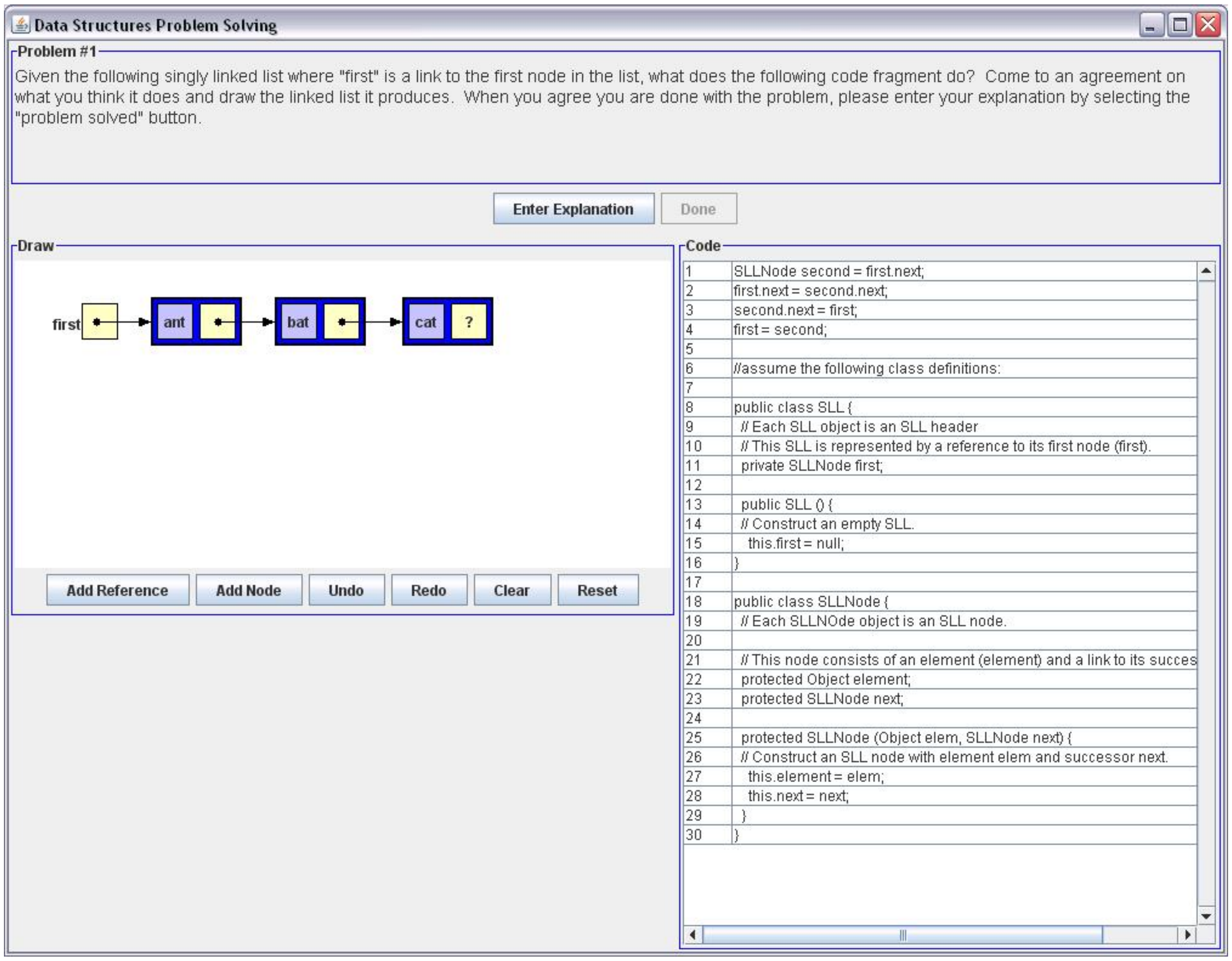
3.1 Learning Gains

After the completion of data collection, I established that the interface and task were conducive to learning by conducting a paired t-test on the pre-test and post-test scores. This analysis showed that the post-test score was moderately higher than the pre-test score ($t(30) = 2.83$; $p = 0.007$; effect size (d) = 0.30).

3.2 Baseline Data Collection and Analysis

In order to demonstrate the benefit of collaborative learning, baseline data was collected with individual students using the interface in Figure 3. This interface is identical to the interface used in data collection with the exception of the dialogue area (chat box and log), which is excluded. The student had access to all other features including the graphical work area. Each student took the pre-test prior to using the interface. The students worked individually on the same problems that were presented to the dyads (see Appendix C). The session concluded with the student taking the post-test.

Surprisingly, the students did learn in these interactions ($t(20) = 2.28$, $p = 0.034$, effect size (d) $= 0.25$). However, the results of this analysis are biased by the make-up of the subject pool. Of the 20 students that participated in the baseline experiments, 15 had taken their last data structures course one or more semester prior to the experiment. It appears that the learning gains are not reflective of actual learning, but more likely the interaction with the interface refreshed concepts that the students had previously learned.



Given the following singly linked list where "first" is a link to the first node in the list, what does the following code fragment do? Come to an agreement on what you think it does and draw the linked list it produces. When you agree you are done with the problem, please enter your explanation by selecting the "problem solved" button.

```
SLLNode second = first.next;
first.next = second.next;
second.next = first;
first = second;

//assume the following class definitions:

public class SLL {
 // Each SLL object is an SLL header
 // This SLL is represented by a reference to its first node (first).
 private SLLNode first;

 public SLL () {
 // Construct an empty SLL.
  this.first = null;
}

public class SLLNode {
 // Each SLLNOde object is an SLL node.

 // This node consists of an element (element) and a link to its succes
 protected Object element;
 protected SLLNode next;

 protected SLLNode (Object elem, SLLNode next) {
 // Construct an SLL node with element elem and successor next.
  this.element = elem;
  this.next = next;
 }
}
```

Figure 3. The baseline data collection interface

CHAPTER 4

CORPUS ANALYSIS

The corpus consists of the peer learning interactions described in Chapter 3. The full interaction between a dyad was subdivided into as many dialogues as problems that they solved. Thus, the corpus consists of a total of 69 dialogues.

I began with an initial analysis of easily extractable features of the dialogues to show there is some correlation of student activity to learning. This was followed by a detailed analysis of dialogue features and graphical actions. I next analyzed the relationship of KCC to learning. Given my hypothesis that initiative shifts can identify KCC, I explored the impact of initiative shifts and learning and then analyzed the relationship between initiative shifts and KCC. The analysis concluded with an examination of predictors of problem solving score.

4.1 Initial Analysis

In order to direct further corpus analysis, I first analyzed the dialogues from exercises 3, 4 and 5, which are error diagnosis problems (1 problem for each type of data structure), to identify features that positively impacted learning and problem solving. Since both chat and shared workspace actions were logged for each user, I was able to automatically extract the following features for each exercise (the labels used in reporting results in tables are listed in parentheses):

- Total number of words (words)
- Words per turn (words per turn)
- Time spent on graphical actions (time on graphical actions)
- Total number of turns (total turns)
- Drawing turns excluding those that only rearranged the drawing (actual drawing turns)
- Code turns (code turns)

Additionally, all pairs received a score on the solutions that they submitted (problem score). For each problem completed, a pair could earn a maximum of five points based on the correctness and completeness of their solution. Since each pair was presented with 5 problems, they could earn a total of 25 points.

Linear regression analysis revealed significant correlations and trends toward correlations between some of these features and learning (see Table II). In problem 3 there is a trend toward correlation of drawing with post-test score. This suggests that use of the graphical workspace was beneficial to the students. The remaining correlations and trends to correlation also suggest that participation in general is an important factor in collaborative learning.

4.2 Detailed Analysis - Dialogue Features

A more detailed analysis of dialogue features was performed using multiple linear regression to predict post-test score. Separate regressions were run for each of the problem

Predictor	Problem 3 (Lists)	Problem 4 (Stacks)	Problem 5 (Trees)
Words	0.19 ($p = 0.02$)		
Words per Turn	0.14 ($p = 0.05$)		
Time on graphical actions	0.15 ($p = 0.04$)		
Problem Score	0.31 ($p = 0.00$)		
Total Turns	0.11 ($p = 0.09$)		
Actual Drawing Turns	0.11 ($p = 0.09$)		
Code Turns			0.14 ($p = 0.08$)

TABLE II

BASIC FEATURES AS PREDICTORS OF POST-TEST SCORE (R^2)

types: list (problems 2 and 3), stack (problem 4) and trees (problem 5). Problem 1 was excluded from the analysis since its purpose was to let the participants become familiar with the interface. Pre-test was used as a covariate because there was a significant positive correlation between pre-test score and post-test score (see Table III).

The list problems (problems 2 and 3) showed trends toward correlation with words typed and total turns (see Table IV). This suggests that student's actions have a positive effect on post-test score and learning. However, these are only trends toward correlation and it is not likely that the actions themselves that correlate but rather what occurs during these actions.

There were no significant correlations of dialogue features in the stack problem. In the tree problem there is a trend toward a negative correlation of words typed and post-test

	β	R^2	p
All problems	0.88	0.77	0.00
Linked list problems	0.60	0.36	0.00
Stack problems	0.81	0.66	0.00
Tree problems	0.81	0.66	0.00

TABLE III

PRE-TEST SCORE AS A PREDICTOR OF POST-TEST SCORE

score ($\beta = -0.27$, $R^2 = 0.04$, $p = 0.11$). This suggest that the more discussion there was, the less students learned. This could reflect the fact that there was a wide variation in the students' experience solving tree problems and the students that typed the most were solving the problems individually with little input from their partners.

	β	R^2	p
Words	0.22	0.07	0.15
Total turns	0.20	0.06	0.18

TABLE IV

DIALOGUE FEATURES AS PREDICTORS OF POST-TEST SCORE (LIST PROBLEMS)

4.3 Detailed Analysis - Graphical Features

Multiple linear regression was also used to identify features from the graphical workspace that were predictors of post-test score. Again, pre-test score was used as a covariate because of its significant positive correlations with post-test score. For the list problems, drawing actions in the graphical workspace had a significant correlation with learning (see Table V). Interpreted drawing actions count a sequence of drawing turns that together draw an identifiable step in the problem solving process as a single drawing action. There was a significant correlation ($R^2 = 0.20$, $p = 0.00$) of interpreted drawing actions and post-test score. However, there were no significant correlations of actions in the graphical workspace for the stack and tree problems and post-test score. This result implies that diagramming linked lists helps students learn while drawing stacks and trees is less helpful. For drawing actions, having task initiative (see section 4.5) did not seem to be a factor in learning. When considering only those drawing actions where the drawer held task initiative, there is no significant correlation of drawing with learning.

4.4 Knowledge Co-construction

My analysis of the basic, easily extractable features showed that there is some correlation between participation and learning. However, it seemed likely that participation alone is not the key, but rather what occurs during the collaborative episodes. This justified doing a deeper analysis of the collaborations.

Since knowledge co-construction has been shown to lead to learning (Hausmann, 2005), I began by analyzing the relationship between KCC and learning. Then, in an effort to

	β	R^2	p
Time on graphical actions	0.36	0.12	0.01
Actual drawing turns	0.34	0.11	0.02
Interpreted drawing actions	0.46	0.20	0.00

TABLE V

GRAPHICAL FEATURES AS PREDICTORS OF POST-TEST SCORE (LIST PROBLEMS)

identify correlates of KCC, I analyzed dialogues for various ways in which co-constructing knowledge can occur, such as a student elaborating on what a partner said or criticizing a partner's contribution.

4.4.1 Annotation

In order to examine the impact of knowledge co-construction on learning, the dialogues were annotated with KCC episodes. A KCC episode is defined as a series of utterances and graphical actions where students are jointly constructing an understanding or shared meaning of a concept required for problem solving (Hausmann, 2005). An example of a KCC episode is shown in Figure 4 beginning at timestamp 15:57:04 and ending at timestamp 15:58:34. In this excerpt, R and C are co-constructing knowledge related to a segment of code. Both students are making contributions to the understanding and correction of the code. This episode is in contrast with the excerpt shown in Figure 5. Here, there is no real collaboration between the students. S is directing the problem solving and C adds nothing of substance.

```
15:56:49 C: which is what list they wanted
15:56:55 R: right
15:57:04 C: and printing seems to be fine
15:57:34 R: um, does it increment correctly?
15:57:46 C: yeah
15:58:01 R: we dont change where head is.
15:58:13 C: we just move p
15:58:23 C: oh wait
15:58:24 C: i see
15:58:30 C: right p = p.next
15:58:34 R: correct
15:58:49 C: so that it?
15:58:56 R: I think so
```

Figure 4. A knowledge co-construction episode

Using this definition, an outside annotator and I coded 30 dialogues (approximately 46% of the corpus) for knowledge co-construction episodes. In annotating the beginning and end of a KCC episode, we did not partition episodes based on topic under discussion. Therefore, sequential episodes are counted as a single episode. Nested episodes would also be counted as a single episode, however we did not encounter any in our annotation efforts. The resulting intercoder reliability, measured using the Kappa statistic (Carletta, 1996), is considered excellent ($\kappa = 0.80$).

```
18:46:09 S: The bottom par is after lines 3-5, produced by doing the following.
18:46:33 S: Draws ins->elem
18:46:54 S: Pred == null in this case, we were not given a pred, and there is
            no pred in the list.
18:47:38 S: Do you understand the creation of ins.
18:50:01 C: yup
18:48:03 S: In this case, pred == null results in true.
18:48:22 S: We're assuming it, if we hit lines 4 and 5.
18:50:45 C: ok
```

Figure 5. No knowledge co-construction

Annotation of co-construction types was more difficult. We started by annotating the same dialogues as above for four different types of knowledge co-construction:

- criticize: A student critically evaluates her peer's input.
- elaborate: A student adds additional information to the topic under discussion.
- justify: A student adds support to a statement made by a peer.
- summarize: A student recaps the suggestion made by his partner.

In this annotation effort, we were unable to achieve an acceptable level of agreement. Since both coders had difficulty differentiating justification from elaboration, the justify annotations were re-coded as elaborate. Additionally, there were very few instances of summarize, so we removed those annotations. By making those changes, the intercoder reliability statistic increased substantially to an acceptable level ($\kappa = 0.64$).

4.4.2 Knowledge Co-construction and Learning

To study the relationship between knowledge co-construction and learning, the dialogues were analyzed using multiple linear regression using pre-test score plus a measure of knowledge co-construction to predict post-test score. Four different measures of knowledge co-construction were used.

1. KCC episodes: The number of KCC episodes.

2. KCC utterances and graphical actions: The total number of utterances plus all graphical actions within KCC episodes.

3. KCC actions: The number of utterances and *interpreted* graphical actions that occurred during KCC episodes. Statistics relating to KCC actions within a KCC episode are shown in Table VI.

4. KCC initiative shifts: The number of times control of the problem solving shifts between participants during KCC episodes. This measure is meant to reflect the density of the KCC episodes, where many initiative shifts reflect more active knowledge co-construction.

Neither of the first two measures had a correlation with learning in any of the problems. The number of episodes likely understates the amount of co-construction that actually occurs since short episodes are weighed the same as long episodes. The total number of utterances plus all graphical actions likely overstates the amount of co-construction since it often takes many individual graphical actions to create a single object, such as a linked list.

	M	Mdn	SD
List Problems	13.63	11.67	6.50
Stack Problem	9.34	9.25	2.92
Tree Problem	12.17	11.75	6.01

TABLE VI

STATISTICS FOR KCC ACTIONS IN KCC EPISODES

For the remaining two measures of KCC, I found no correlations with learning in the stack or tree problems (see section 4.9). However, correlations in the list problems showed that being active in the learning process has a positive impact on learning for both the individual and the dyad.

	β	R^2	p
KCC actions as predictor of **mean** post-test score	0.43	0.14	0.02
KCC actions as predictor of **individual** post-test score	0.33	0.08	0.03
KCC actions as predictor of individual post-test score (low pre-test subjects, n=14)	0.61	0.37	0.03
KCC actions as predictor of individual post-test score (high pre-test subjects, n=16)	0.33	0.09	ns

TABLE VII

KCC ACTIONS AS PREDICTORS OF POST-TEST SCORE (LIST PROBLEMS)

In Table VII, the first row shows the benefit for the pair overall by correlating the mean post-test score with the mean pre-test score and the dyad's KCC actions. The second row shows the benefit for individuals by correlating individual post-test scores with individual pre-test scores and the dyad's KCC actions. The difference in the strength of these correlations suggests that members of the dyads are not benefitting equally from knowledge co-construction. If the subjects are divided into two groups, those with a pre-test score below the mean score and those with a pre-test score above the mean score, it can be seen that those with a low pre-test score benefit more from the knowledge co-construction episodes than do those with a high pre-test score (rows 3 and 4 in Table VII) . Table VIII which uses KCC initiative shifts as the measure of co-construction shows similar results. It is interesting to note that when the outlier pairs are removed (see section 4.5.2), the correlation for the low pre-test score subjects becomes much stronger ($R^2 = 0.45$, $p = 0.02$) when KCC initiative shifts are used as the measure of co-construction.

4.4.3 Types of Knowledge Co-construction and Learning

Each of the two types of knowledge co-construction, elaborative and critical, was analyzed for its relationship with learning using multiple linear regression. In this corpus, I found no statistically significant correlations between post-test score and criticisms (after removing the impact of pre-test score) or between post-test score and elaborations (after removing the impact of pre-test score). There was a trend toward correlation for elaborative co-construction when using the partner's elaborations as a predictor of the student's score ($R^2 = 0.11$, $p = 0.10$). This suggests that to receive an explanation of a concept from a

	β	R^2	p
KCC initiative shifts as predictor of **mean** post-test score	0.46	0.15	0.01
KCC initiative shifts as predictor of **individual** post-test score	0.35	0.09	0.02
KCC initiative shifts as predictor of individual post-test score (low pre-test subjects, n=14)	0.41	0.17	0.16
KCC initiative shifts as predictor of individual post-test score (low pre-test subjects, outliers removed n=12)	0.67	0.45	0.02
KCC initiative shifts as predictor of individual post-test score (high pre-test subjects, n=16)	0.10	0.01	ns

TABLE VIII

KCC INITIATIVE SHIFTS AS PREDICTORS OF POST-TEST SCORE (LIST PROBLEMS)

peer is beneficial for the student receiving the explanation but not for the student making the explanation.

4.4.4 Balanced Co-Construction

I also explored whether equal participation from students might be a requirement for learning. In order to test this hypothesis, I computed balance using the formula found in Equation 4.1 where A is a measure of the student's contribution and A_p is the contribution of his/her partner. If actions are completely balanced, this measure returns a score of one and if all the actions are performed by a single partner it will return a value of zero.

$$1 - \frac{|A - A_p|}{A + A_p} \tag{4.1}$$

Table IX shows the result of multiple linear regression on various measures of student action plus pre-test score to predict post-test score. The measure of student activity used to compute A and A_p are one of the following:

1. KCC actions: The number of utterances and interpreted graphical actions contributed by the student during KCC episodes.
2. KCC-criticize: The number of criticisms recorded during KCC episodes.
3. KCC-elaborate: The number of elaborations recorded during KCC episodes.
4. Dialogue initiative: The number of utterances in KCC episodes where the student had control over the dialogue (see section 4.5).
5. Task initiative: The number of utterances and interpreted graphical actions in KCC episodes where the student had control over problem solving (see section 4.5).

Measure of KCC	β	R^2	p
KCC actions	0.21	0.01	0.18
KCC-criticize	-0.34	0.10	0.03
KCC-elaborate	0.14	0.08	ns
Dialogue initiative	0.27	0.10	0.09
Task initiative	0.10	0.01	ns

TABLE IX

BALANCED CONTRIBUTION OF KCC AS PREDICTOR OF POST-TEST SCORE (LIST PROBLEMS)

The results of the analysis do not confirm the intuition that balanced activity is better. KCC-criticize shows a negative correlation with learning. This implies that the pairs in which one student criticized more than the other learned better than the pairs where the criticism was balanced. The other trend toward correlation is for dialogue initiative, which shows that more balanced dialogue initiative results in learning.

Since balanced co-construction did not strongly correlate with learning, I hypothesized that perhaps the person who was more involved in co-construction would have larger learning gains. Student involvement was calculated using the ratio of a student's action to the actions of the pairs. The measures of student activity are the same as those described above. As shown in Table X there were no significant correlations with learning and involvement in co-construction.

Measure of KCC	β	R^2	p
KCC actions	-0.15	0.02	ns
KCC-criticize	-0.06	0.00	ns
KCC-elaborate	-0.11	0.02	ns
Dialogue initiative	0.15	0.03	ns
Task initiative	-0.04	0.00	ns

TABLE X

KCC ACTIVITY RATIO AS A PREDICTOR OF POST-TEST SCORE (LIST PROBLEMS)

While it appears that for learning to occur both students must be active participants in the problem solving process, the relative amount of participation is not a key factor.

4.5 Initiative

Since my attempts to annotate for KCC relations, such as criticize and elaborate, were only moderately successful and there were no correlations between these relations and learning, I looked for simpler but principled correlates of KCC. I chose to annotate for the linguistically motivated notion of initiative shifts in dialogue. Initiative shifts refer to control over the conversation being transferred from one person to another. Intuitively, initiative would shift between peers when they are working together to to solve the problem, and hence, that they are co-constructing the solution. An example of initiative shifts and KCC is found in Figure 4 where initiative shifts between C and R. C has initiative at timestamp 15:57:04, but control passes to R in the next line when R suggests that the code segment has an error. R maintains initiative until timestamp 15:58:13 when C elaborates on the cause of the error and C retains control until the end of the KCC episode at timestamp 15:58:34.

Before embarking on an exhaustive manual annotation of initiative, I first examined whether initiative may indeed affect learning in this context by automatically tagging for initiative using an approximation of Walker and Whittaker's utterance based allocation of control rules (Walker and Whittaker, 1990). In this scheme, each turn in the dialogue is tagged as either: (1) an assertion, (2) a command, (3) a question or (4) a prompt (turns not expressing propositional content). This was done automatically, by marking turns that end in a question mark as questions, those that start with a verb as commands, prompts from a list of commonly used prompts (e.g., ok, yeah) and the remaining turns as assertions. Control is then allocated by using the following rules based on the turn type:

1. Assertion: Control is allocated to the speaker unless it is a response to a question.
2. Command: Control is allocated to the speaker.
3. Question: Control is allocated to the speaker, unless it is a response to a question or a command.
4. Prompt: Control is allocated to the hearer.

Since the dialogues also have a graphics component, all drawing and code change moves had control assigned to the peer drawing or making the code change.

In this initial analysis, I found that in problem 3, learning gain positively correlates with the number of turns where a student has initiative ($R^2 = 0.16, p = 0.04$).

4.5.1 Annotation

Since this preliminary analysis showed a correlation of initiative with learning gain, I chose to begin a thorough data analysis by manually annotating the dialogues with initiative shifts. Walker and Whittaker claim that initiative encompasses both dialogue control and task control (Walker and Whittaker, 1990), however, several others disagree. Jordan and Di Eugenio propose that control and initiative are two separate features in collaborative problem solving dialogues (Jordan and Di Eugenio, 1997). While control and initiative might be synonymous for the dialogues analyzed by Walker and Whittaker where a master-slave assumption holds, it is not the case in collaborative dialogues where no such assumption exists. Jordan and Di Eugenio argue that the notion of control should apply to the dialogue level, while initiative should pertain to the problem-solving goals. In a similar vein, Chu-Carroll and Brown also argue for a distinction between control and initiative, which they term dialogue initiative and task initiative (Chu-Carroll and Brown, 1998). Since there is no universally agreed upon definition for initiative, I decided to annotate for both dialogue initiative and task initiative. For task initiative, I derived an annotation scheme based on other research in the area. According to Jordan and Di Eugenio, in problem solving (task) initiative the agent takes it upon himself to address domain goals by either (1) proposing a solution or (2) reformulating goals. Guinn (Guinn, 1998) defines task initiative as belonging to the participant who dictates which decomposition of the goal will be used by both participants during problem-solving. A third definition is from Chu-Carroll and Brown. They suggest that task initiative tracks the lead in development of the agent's plan. Since the primary goal of the dialogues studied by Chu-Carroll and Brown is to develop a plan, this could be re-worded to state that task initiative tracks the lead in development of the agent's goal. Combining these definitions, task initiative can be defined as *any action by a participant to either achieve a goal directly, decompose a goal or reformulate a goal.* Actions in our domain that show task initiative include:

- Suggesting a section of code to verify.
- Explaining what a section of code does.
- Identifying a section of code as correct or incorrect.

- Suggesting a correction to a section of code
- Making a correction to a section of code prior to discussion with the other participant.

The distinction between dialogue initiative and task initiative can be seen in Figure 4. While C takes control of the dialogue at timestamp 15:57:04, R maintains control of the task, which here is identifying a section of code as incorrect. It's not until timestamp 15:58:30 that C takes task initiative by suggesting a correction to a line of code.

Two coders, the author and an outside annotator, coded 24 dialogues (1449 utterances) for both dialogue and task initiative. This is approximately 45% of the corpus. For dialogue initiative, the rules derived by Walker and Whittaker were not used. Instead, the coders marked each utterance with the speaker who had control of the conversation. Task initiative was annotated using the definition given above. The resulting intercoder reliability, measured with the Kappa statistic, is 0.77 for dialogue initiative annotation and 0.68 for task initiative, both of which are high enough to support tentative conclusions.

4.5.2 Analysis

Using multiple linear regression analysis on these annotated dialogues, I found no correlation of either type of initiative or initiative shifts with learning, in the stack and tree problems (see section 4.9). However, in the list problems, there was a signicant correlation between post-test score (after removing the effects of pre-test scores) and the number of shifts in dialogue initiative and the number of shifts in task initiative (see Table XI). Since the purpose of this analysis is to model collaboration with an agent, two pairs whose problem solving collaboration had gone awry are excluded from this analysis. For one of the pairs, large portions of dialogue involved problem solving activities on a single line of code that was erroneously identified as an error. Since the peer agent will discourage this type of activity, it seems appropriate to remove this dialogue from the analysis. In the other excluded dialogue, there were frequent initiative shifts however the students often did not appear to be addressing each other. They were working in parallel as opposed to collaborating. Their utterances were generally self-directed and not in response to their partners. An example of this is shown in Figure 6. R's utterance at timestamp 20:12:09 is not in response to the previous utterance made by T, since T is referring to a return value and R does not mention one in his code segment. The following utterance by R seems to be a reference to his previous utterance since again it does not address T's question. At 20:12:49, T makes a suggestion for a correction but it seems self-directed since it is does not address R's utterance at 20:12:32 regarding the method of referencing head. This is followed by T responding to his original question about return values. Such a breakdown in collaboration would not occur with the agent.

This correlation of post-test score with initiative shifts seems to validate the hypothesis that collaboration results in learning when both students are active participants in collaborative problem solving.

4.6 Knowledge Co-construction and Initiative

Since separately KCC and initiative shifts correlate with learning, I continued the corpus analysis by investigating the relationship between KCC and initiative.

Predictor of Post-test	β	R^2	p
Dialogue initiative shifts	0.36	0.14	0.01
Dialogue initiative shifts (excluding outliers)	0.45	0.20	0.00
Task initiative shifts	0.23	0.08	0.14
Task initiative shifts (excluding outliers)	0.42	0.20	0.01
Utterances with dialogue initiative	0.28	0.06	0.06
Utterances with dialogue initiative (excluding outliers)	0.38	0.12	0.02
Utterances with task initiative	0.07	0.02	*ns*
Utterances with task initiative (excluding outliers)	0.14	0.04	*ns*

TABLE XI

INITIATIVE PREDICTORS OF POST-TEST (LIST PROBLEMS)

It seems likely that knowledge co-construction episodes involve frequent shifts in initiative, as both participants are actively participating in problem solving. To test this hypothesis, I calculated the average initiative shifts per line during KCC episodes and the average initiative shifts per line during problem solving outside of KCC episodes for each pair. A paired t-test was then used to verify that there is a difference between the two groups. The t-test showed no significant difference in average dialogue initiative shifts in KCC episodes compared with non-KCC problem solving. However, there is a significant difference between average task initiative shifts in KCC episodes compared with the rest of the dialogue (see Table XII). The difference between the two groups shows that there is a meaningful increase in the number of task initiative shifts in KCC episodes compared with

```
20:11:54  T:     should it return anything?
20:12:09  R:     removeBeginning(link h){ if(h == head)h = h.next;
                 return h; else return null;}
20:12:24  R:     that wont work
20:12:32  R:     head isnt a class variable
20:12:49  T:     if head != null
20:12:59  T:     or h !=  null
20:13:21  T:     actually we can always return h i think
20:13:33  T:     either it is the head or it is null
20:14:00  R:     acutally the list is
20:14:07  R:     1->2->3->4->5->
```

Figure 6. Students not collaborating

problem solving activity outside of the KCC episodes. Analyzing only the list problems shows an even larger difference.

	Average Shifts in KCC Episodes	Average Shifts outside KCC Episodes	Difference (d)	T-Test p-value
All Problems	0.2217	0.13040	0.49	0.0016
List Problems	0.2117	0.10233	0.65	0.0480

TABLE XII

FREQUENCY OF TASK INITIATIVE SHIFTS

4.7 Knowledge Scores, Initiative and Knowledge Co-construction

Since the agent uses a student model to assist in making decisions regarding its behavior, I analyzed the relationship of knowledge score, a measure of student knowledge based on probabilities derived from the student model (calculated as described in Chapter 6), with initiative and knowledge co-construction.

Table XIII shows that initiative shifts and knowledge co-construction are significant predictors of knowledge score. This strengthens the hypothesis that initiative shifts and knowledge co-construction result in learning by evaluating a student's knowledge of all concepts involved in problem solving, not just those presented to the students in the pre-/post-test.

	β	R^2	p
Task Initiative Shifts	0.56	0.31	0.00
Dialogue Initiative Shifts	0.56	0.32	0.00
KCC actions	0.37	0.15	0.05

TABLE XIII

INITIATIVE SHIFTS AND KCC AS PREDICTORS OF KNOWLEDGE SCORE (LIST PROBLEMS)

4.8 Correlations with Successful Problem Solving

Another measure of success for peer learning is successful problem solving, measured by the scores received by the pair on the correctness of problem solutions. For each problem completed the pair could earn a maximum of five points.

Table XIV shows the correlations and trends toward correlation of basic features of the dialogue. The only correlations are with problem solving time and total turns which would be expected to be correlated with problem solving success, especially in debugging problems such as problems 3 and 4.

Drawing and coding actions (see Table XV) were predictors of problem score only in problems 4 and 5. In this table code actions refer to all actions related to the code in the graphical workspace including actions that merely selected a line of code, while code changes involve making a change to the code. Coding actions are most likely to correlate with problem score because when more errors are detected and corrected there would be more code changes. In problem 4 there is a correlation with interpreted drawing actions, however since there were only 7 pairs who drew anything, this correlation is suspect.

Task initiative shifts were predictors of problem solving success for two debugging problems, problem 3 - lists and problem 4 - stacks (see Table XVI). This suggests that active collaboration leads to problem solving success. However, the other problems do not validate this conclusion. This could be because problem 2 is an explanation task instead of a debugging task and problem 5 deals with trees where there is a wide variation in student experience level (see section 4.9).

Problem	Feature	β	R^2	p
3	Total turns	0.48	0.23	0.08
	Problem solving time	0.43	0.19	0.12
4	Total turns	0.40	0.16	0.18
5	Problem solving time	0.44	0.20	0.15

TABLE XIV

BASIC DIALOGUE FEATURES AS PREDICTORS OF PROBLEM SCORE

There were no significant correlations of knowledge co-construction or balance of knowledge co-construction with problem score for any of the problems. This suggests that knowledge co-construction, while related to learning, may not be related to successful problem solving.

4.9 Discussion

Several factors potentially explain the fact that only the linked list problems showed a correlation of initiative and knowledge co-construction with learning. First, the lack of correlations in the tree problem is possibly caused by the wide variation in experience levels of the students. Of the pairs that solved the tree problem, only 33% had both members receiving an acceptable score (more than 60% of the possible points) on the tree related problems in the pre-test. This contrasts with 58% for the list problems and 68% for the stack problems. And secondly, since the students had a better understanding of stacks prior to problem solving, there was less discussion in solving the stack problems. Additionally,

Problem	Feature	β	R^2	p
4	Code actions	0.43	0.18	0.15
	Time on graphical actions	0.38	0.15	0.20
	Interpreted drawing actions	0.57	0.33	0.04
5	Code actions	0.67	0.45	0.02
	Code changes	0.56	0.32	0.06
	Time on graphical actions	0.74	0.55	0.01

TABLE XV

GRAPHICAL FEATURES AS PREDICTORS OF PREDICTORS OF PROBLEM SCORE

my experience in teaching data structures in the classroom is that students struggle more with the concepts related to linked lists than with those involved in understanding stacks. So, a better overall understanding of stacks is a possible cause of the lack of correlation of dialogue initiative with post-test score in the stack problems.

4.10 Conclusions

There are four key findings from this corpus analysis that suggest that KCC can be operationalized by shifts in task initiative:

- KCC correlates with learning but KCC is a high level concept that cannot be easily recognized by an artificial agent.
- Shifts in both dialogue initiative and task initiative correlate with learning.
- Task initiative shifts within KCC episodes correlate with learning.

Problem	Feature	β	R^2	p
3	Utterances with task initiative	0.43	0.18	0.13
	Task initiative shifts	0.53	0.28	0.05
	Dialogue initiative shifts	0.38	0.14	0.18
4	Utterances with task initiative	0.46	0.21	0.11
	Task initiative shifts	0.56	0.31	0.05

TABLE XVI

INITIATIVE PREDICTORS OF PROBLEM SCORE

- KCC episodes and non-KCC-dialogue portions significantly differ in the frequency task initiative shifts, validating my hypothesis that shifts in task initiative can be used to identify when KCC is occurring.

CHAPTER 5

MANAGEMENT OF INITIATIVE

Since my corpus study showed that the level of task initiative can be used to identify when KCC and potentially learning is occurring, the peer learning agent (KSC-PaL) is endowed with behaviors to manipulate shifts in task initiative in order to encourage KCC and learning. This required that the agent be able to recognize the initiative holder in each utterance or action and that the agent has the ability to appropriately encourage the shift of initiative from agent to student.

In this chapter I describe building a classifier to recognize shifts in task initiative. This is followed by a corpus analysis to identify cues that can be used by the agent to shift initative.

5.1 Recognizing Initiative Shifts

The first step in manipulating task initiative is to provide KSC-PaL with the ability to track shifts in initiative. In order to have the agent recognize shifts, it must identify the initiative holder in each utterance or action. Using the Weka Toolkit (Witten and Frank, 2005), I explored various machine learning algorithms and feature sets that could reliably identify the holder of task initiative. The following ten features were used as potential identifiers of task initative:

- Prior speaker: speaker in the previous turn

- Speech act: statement, command, question or prompt
- Prior turn speech act: speech act in the previous turn
- Partner's last speech act: speech act of partner's last utterance
- Knowledge score: a measure of student knowledge, calculated as described in section 6.3.2.1
- Length of the utterance
- Pause: time difference between utterances
- I think: Utterance contains the phrase "I think", a rough approximation of hedging
- Error: an incorrect statement
- Dialogue initiative: holder of dialogue initiative in the turn (automatically computed using the method described in Chapter 6)

Using all of the features, I trained various classifiers using the annotated corpus. Based on the results from this initial screening. I narrowed down further exploration to the following classifiers:

1. K *: An instance-based classifier. It classifies an instance by finding the training instances that are most similar to the test instance and assigning the test instance to same class as the training instances.
2. Logistic Model Trees: A classifier that combines tree induction with logistic regression.
3. Naive Bayes: A simple probabilistic classifier based on applying Bayes' theorem and applying strong independence assumptions.

4. Ridor: Weka's implementation of a RIpple-DOwn Rule learner. It generates the default rule first and then the exceptions for the default rule with the smallest error rate.

5. JRip: Weka's implementation of RIPPER, a propositional rule learner.

6. PART: A rule based classifier that creates good rule sets by learning one rule at a time.

7. SMO : Weka's implementation of a support vector machine.

Each machine learning algorithm was run over the power set of features to determine which feature set performed the best for each classifier. Based on the fact that KSC-PaL requires strict turn-taking between the agent and the student, I eliminated the first feature (prior speaker), since for the interactions with the agent it will always be the partner. In this analysis I found that the relevant features of an action in the graphical workspace were substantially different from those of a natural language utterance.

Based on the results of this analysis shown in Table XVII, I selected the following two classifiers: (1) K* (Cleary and Trigg, 1995), a clustering algorithm, for classifying natural language utterances which correctly classified 71.77% of utterances and (2) JRip (Cohen, 1995), a rule-based algorithm, for classifying drawing and coding actions which correctly classified 86.97% of the instances. The features used in the K* classifier are prior turn, knowledge score, utterance length and dialogue initiative. For the classifier of graphical actions, the only feature used is knowledge score. This suggests that a student takes the initiative in performing graphical actions when he/she has a high knowledge level.

Classifier	Type	Percent Correct Utterances	Percent Correct Graphical Actions
K*	Clustering	71.77	na
Logistic Model Trees	Tree	70.71	na
Naive Bayes	Bayesian	69.56	na
Ridor	Rule-based	69.65	86.87
JRip	Rule-based	69.56	86.97
PART	Rule-based	69.12	73.48
SMO	Function	68.94	na

TABLE XVII

TASK INITIATIVE CLASSIFIERS

5.2 Methods for Encouraging Initiative Shifts

I explored two different methods for encouraging initiative shifts. One is that student uncertainty may lead to a shift in initiative. The other is that certain cues for initiative shifts identified in related literature (Chu-Carroll and Brown, 1998; Walker and Whittaker, 1990) lead to initiative shifts.

Intuitively, uncertainty by a peer might lead to his partner taking the initiative. One possible identifier of student uncertainty is hedging. So, along with an outside annotator, I annotated utterances in our peer dialogs with hedging categories as identified in (Bhatt et al., 2004). Using these categories we were unable to reliably annotate for hedging. But, after collapsing the categories into a single binary value of hedging/not hedging we arrived at an acceptable agreement ($\kappa = 0.71$).

Another identifier of uncertainty is a student's request for feedback from his partner. When uncertain of his contribution, a student may request an evaluation from his peer. Therefore, we annotated utterances with "request for feedback" and achieved excellent inter-annotator agreement($\kappa = 0.82$).

Chu-Carroll and Brown (Chu-Carroll and Brown, 1998) identify cues that may contribute to the shift of task and dialogue initiative. Since task initiative shifts appear to identify knowledge co-construction episodes, I chose to explore the following cues that potentially result in the shift of task initiative:

- Give up task. I tagged utterances in the dialogs where the student explicitly gave up the task using phrases like "Any other ideas?".
- Silence. A pause may suggest that the speaker has nothing more to say in the current turn and intends to give up his initiative. A silence of 60 seconds or more after an utterance or action was marked a pause.
- Prompts. A prompt is an utterance that has no propositional content. Prompts were annotated when the dialogs were annotated for dialog initiative.
- Repetitions. Redundant utterances suggest that the hearer should take over initiative. In analyzing the peer dialogs for repetitions, I found no repetitions. This is likely due to the fact that the dialog is typed instead of spoken and the students have a history of the entire interaction displayed on their screens.
- Analytical cues. These are cues that cannot be recognized without the hearer performing an evaluation of the speakers utterance. These utterance could be invalid

statements, suboptimal proposals or ambiguous statements. For this analysis, analytical cues were identified as utterances preceding criticisms, which were previously annotated as a type of knowledge co-construction.

Using these cues, I was able to identify 283 shifts in task initiative or approximately 67% of all task initiative shifts. The remaining shifts were likely an explicit takeovers of initiative without preceding cues.

Cue/Identifier	Led to initiative shift	Percent of instances that led to initiative shift
Invalid statement	51	38.64%
Prompt	164	29.29%
Pause	46	25.27%
Hedge	34	23.94%
Request feedback	51	21.88%
Give-up task	4	20.00%

TABLE XVIII

CUES FOR SHIFTS IN INITIATIVE

Since there are several ways to predict and encourage initiative shifts, I explored which of these cues more often resulted in an initiative shift and which cues whose resulting initiative shift more often led to an increase in knowledge score. Table XVIII shows the number and percentage of instances of each cue that resulted in an initiative shift. The most likely cue to lead to an initiative shift was an invalid statement.

Along with the likelihood of a cue leading to an initiative shift, I also explored the impact of an initiative shift on knowledge score, which is a measurement of student knowledge based on probabilities derived from the student model (see Chapter 6). This is an important characteristic since the goal of KSC-PaL is to encourage initiative shifts in an effort to increase learning. First, I examined initiative shifts to see if they resulted in an increase in knowledge score. Table XIX shows the result of that analysis. The first row in this table shows how often the knowledge score increased, decreased and remained unchanged immediately after the initiative shifted from one participant to the other. The second row shows the change in knowledge score from immediately before initiative shifted until it shifted again. A Chi-Squared analysis computed on this table shows that this is not a random distribution ($\chi^2 = 33.9258$, $p = 0.00$). Also, it is interesting to note that while knowledge score did not increase as often as it remained unchanged, the negative impact on knowledge score was infrequent.

	Increased	Decreased	Unchanged
On utterance where initiative shifted	81	25	318
Prior to next shift in initiative	157	22	245

TABLE XIX

CHANGE IN KNOWLEDGE SCORE AFTER SHIFT IN TASK INITIATIVE

Since initiative shifts, when taken as a whole, can lead to an increase in knowledge, I explored the impact of specific cues on knowledge. The results are shown in Table XX.

Cue/Identifier	Increased on utterance where initiative shifted	Increased prior to next shift in initiative
Invalid statement	6 (11.76%)	12 (23.53%)
Prompt	28 (17.07%)	54 (32.93%)
Pause	4 (8.70%)	7 (14.22%)
Hedge	4 (11.76%)	8 (23.52%)
Request feedback	6 (11.76%)	9 (17.65%)
Give-up task	0 (0.00%)	0 (0.00%)

TABLE XX

INCREASE IN KNOWLEDGE SCORE AFTER SHIFT IN TASK INITIATIVE

Using the results from these analyses, I selected the cues that were most likely to lead to initiative shift and to increase knowledge score. Pause was excluded since a pause by the agent might be misinterpreted by the student as a system error. Therefore, I chose to have the agent encourage shifts in task initiative by:

- using prompts
- making mistakes which will ideally lead to a student criticism
- hedging
- requesting feedback

CHAPTER 6

KSC-PAL

This chapter presents the development of the peer learning agent, KSC-PaL. It begins with an overview of the agent's architecture. This followed by discussion of the student interface, the student model and the planner that is used to manage shifts in task initative.

6.1 Agent Architecture

The core of KSC-PaL is the TuTalk system (Jordan et al., 2007). TuTalk is a dialogue management system that supports natural language dialogues for educational applications and allows for both tutorial and conversational dialogues. Since peer dialogues are conversational in nature, the ability to manage these types of dialogues was an important factor. Additionally, TuTalk was selected as the dialogue management system because of its ability to handle mixed-initiative dialogues. In developing the agent I extended TuTalk by adding a graphical user interface, replacing TuTalk's student model and augmenting TuTalk's planner to implement the model discussed above. Figure 7 shows the agent's components and the data flows between the components.

The interface manages communication between TuTalk and the student. Students communicate with the agent using natural language and graphical actions within a graphical user interface. The student input is processed by the interface and its related modules into an appropriate format and passed to TuTalk. Since TuTalk's interpretation module is not

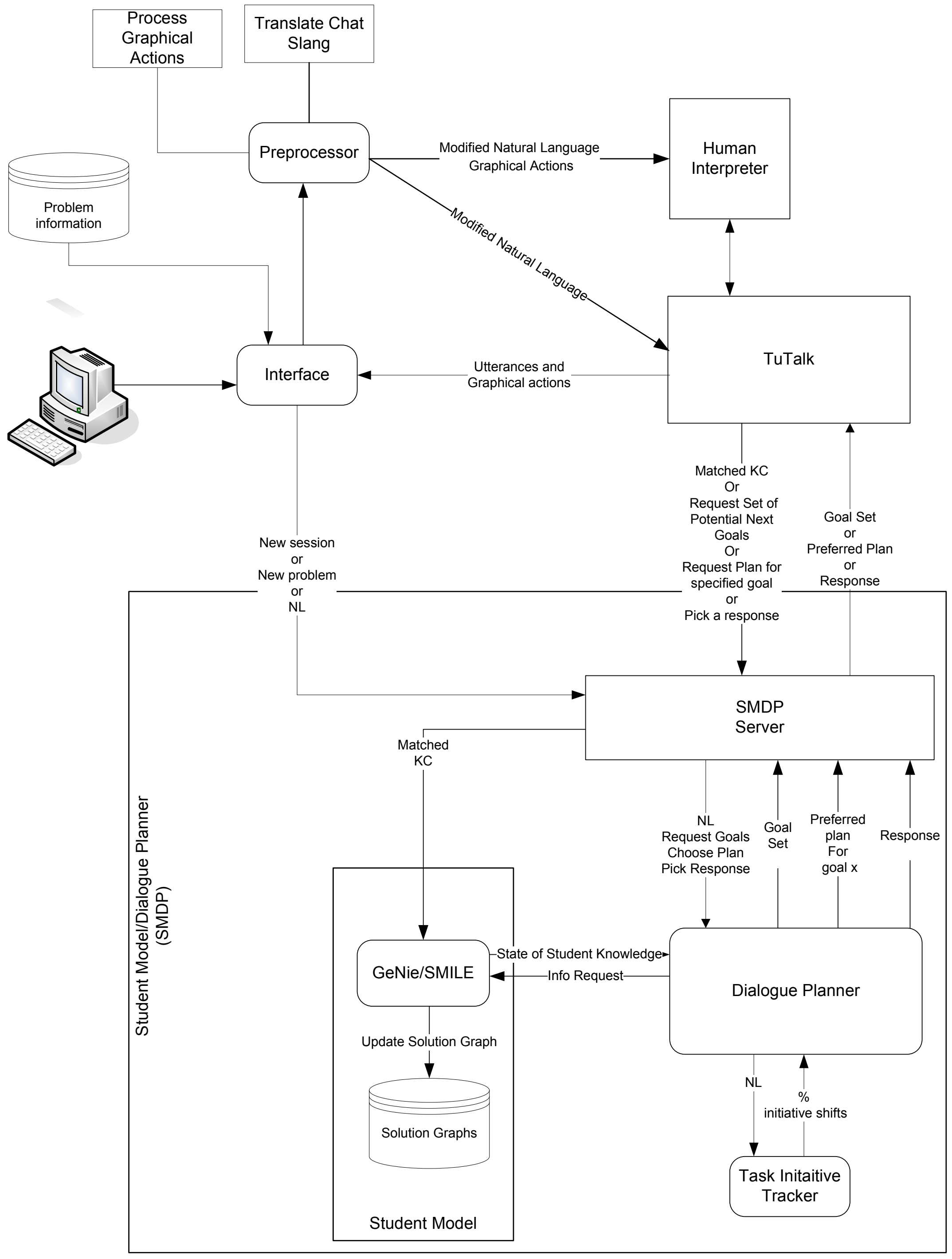

Process Graphical Actions
Translate Chat Slang
Preprocessor
Modified Natural Language Graphical Actions
Human Interpreter
Problem information
Modified Natural Language
Interface
Utterances and Graphical actions
TuTalk
Matched KC Or Request Set of Potential Next Goals Or Request Plan for specified goal or Pick a response
Goal Set or Preferred Plan or Response
New session or New problem or NL
SMDP Server
Matched KC
NL Request Goals Choose Plan Pick Response
Goal Set
Preferred plan For goal x
Response
Student Model/Dialogue Planner (SMDP)
GeNie/SMILE
State of Student Knowledge
Info Request
Dialogue Planner
Update Solution Graph
Solution Graphs
NL
% initiative shifts
Task Initaitive Tracker
Student Model

Figure 7. Agent implementation

able to appropriately handle all student utterances, a human interpreter assists in this process. Additionally TuTalk requests assistance from the Student Model/Dialogue Planner (SMDP) to manage the dialogue in order to appropriately shift initiative and encourage learning. These modules are described below in more detail.

6.2 Interface

The user interface is structured similarly to the one used in data collection (see Figure 1). However, additional features were added to allow a student to effectively communicate with KSC-PaL. One enhancement was a preprocessor module which takes as input a student's utterances and actions and modifies them so that they can be recognized by TuTalk. This preprocessor consists of a spell corrector and a graphical actions interpreter.

Since the interface used in data collection has a component that mimics an instant messaging application, it was not surprising to discover that the corpus dialogues contained abbreviations and slang that are common to text messaging. These abbreviations are not recognized by the English language spell corrector in the TuTalk system, so a chat slang interpretation module was added to the agent interface. The slang words and their traditional English equivalents were collected from various sites on the internet (AA. VV., 2007a; AA. VV., 2007b).

The second part of the preprocessor interprets the student's drawing and coding actions and passes them to TuTalk as natural language utterances, such as *draw: line 1* when the student has correctly drawn the first line of code in problem 1. Graphical actions are matched to a set of known actions and when a student signals that he/she has finished drawing or coding either by ceding control of the graphical workspace or by starting to communicate through typed text, the interface will attempt to match what the student has drawn or coded with its database of known graphical actions. Reporting only at these times minimizes reporting cases where the student is in the process of creating a drawing but has not completed it, for example when a student has added several nodes of the list but has not yet linked them together. The set of known graphical actions includes not only correct ones but also anticipated misconceptions that were collected from the data collection interactions.

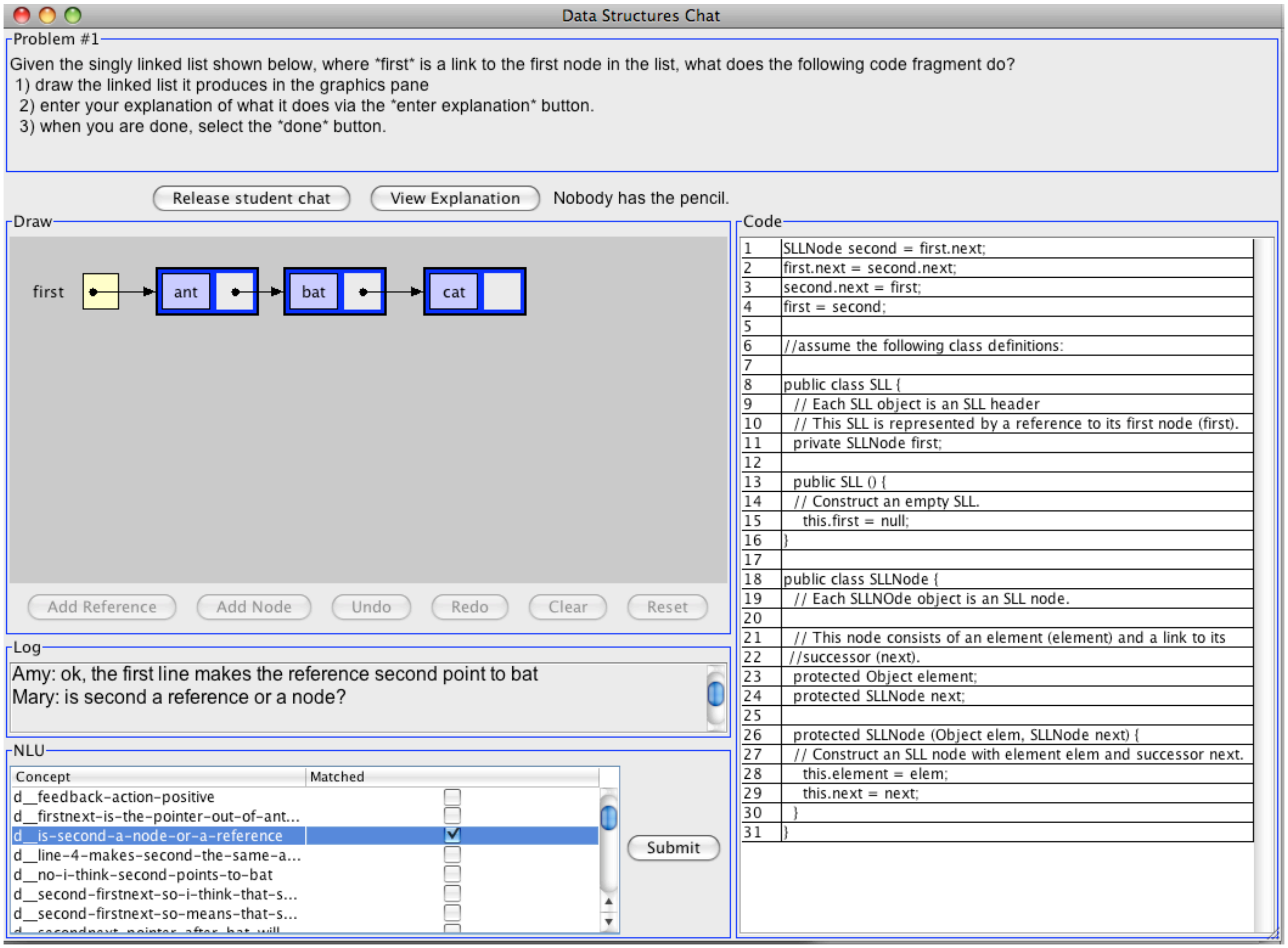
Since there are various ways to construct the same drawing, this interpretation is done using a constraint based approach. The linked list drawings can be very complex as the problems involve multiple pointers, nodes and connected components. Each node in a linked list can be pointed to by multiple nodes and/or pointers. For example, an expected drawing for exercise 2 involves a linked list structure: $first \rightarrow a \rightarrow b$ and $pred \rightarrow b$ which results in both a node and a reference pointing at the node containing b. To accurately reflect situations like this, the linked list drawings are represented as restricted graphs. The restriction is that each node is restricted to an outdegree of one, but its indegree can be unlimited. Both the student drawings and the constraints are represented as these restricted graphs and checking constraints involves matching the graph representing the student's drawing to the graphs representing the constraints.

Additionally, the preprocessor interprets modifications within the code area. As with drawing, a student may make the same change to code in various ways. For example one student may modify a line to make a correction while another might delete the incorrect line and insert a correction. So, instead of tracking changes, the current state of the student code is matched against a certain set of code states that the agent recognizes. The known states include proper correction of errors, as well as misconceptions.

6.2.1 Human Interpreter

Given the current technology available for natural language understanding, a human interpreter was incorporated to assist in the disambiguation of student utterances. The interpreter receives a student utterance along with a list of possible matching concepts from TuTalk. The interpreter then selects the most likely matching concepts from TuTalk thus assisting in natural language interpretation. If the student utterance doesn't match any of these concepts, a second list of concepts, containing student initiative utterances, are presented to the interpreter. If none of these match then all known concepts are presented to the interpreter for matching. Note that the interpreter has a limited, predetermined set of choices, corresponding to the concepts that TuTalk is aware of. In this way, his/her intervention is circumscribed. In Figure 8 the student has typed "is second a reference or a node?". The human interpreter is presented with a list of possible matches, including the matching concept *is-second-a-node-or-reference*. The wizard would validate this interpretation by pressing submit and the concept match would be sent to TuTalk.

The interpreter also plays a role in the interpretation of graphical actions. The natural language interpretation of the drawing or coding action is first sent to the interpreter. He/she then validates the interpretation and sends it on to TuTalk for processing. Ad-



Problem #1

Given the singly linked list shown below, where *first* is a link to the first node in the list, what does the following code fragment do?
1) draw the linked list it produces in the graphics pane
2) enter your explanation of what it does via the *enter explanation* button.
3) when you are done, select the *done* button.

Nobody has the pencil.

Code

```
SLLNode second = first.next;
first.next = second.next;
second.next = first;
first = second;

//assume the following class definitions:

public class SLL {
  // Each SLL object is an SLL header
  // This SLL is represented by a reference to its first node (first).
  private SLLNode first;

  public SLL () {
  // Construct an empty SLL.
    this.first = null;
}

public class SLLNode {
  // Each SLLNOde object is an SLL node.

  // This node consists of an element (element) and a link to its
 //successor (next).
  protected Object element;
  protected SLLNode next;

  protected SLLNode (Object elem, SLLNode next) {
  // Construct an SLL node with element elem and successor next.
    this.element = elem;
    this.next = next;
  }
}
```

Log

Amy: ok, the first line makes the reference second point to bat
Mary: is second a reference or a node?

NLU

Concept	Matched
d__feedback-action-positive	☐
d__firstnext-is-the-pointer-out-of-ant...	☐
d__is-second-a-node-or-a-reference	✓
d__line-4-makes-second-the-same-a...	☐
d__no-i-think-second-points-to-bat	☐
d__second-firstnext-so-i-think-that-s...	☐
d__second-firstnext-so-means-that-s...	☐

Figure 8. Human interpreter interface

```
12:58:37:028 agent: i think we're done drawing
12:59:08:594 student: now we have to enter the explanation right?
12:59:21:869 Sent:  unanticipated-response
12:59:36:139 Sent:  off-topic__req-write-expl-ready
12:59:37:914 agent: ok, why don't you try?
13:00:28:847 student: solution second node and the first node is swapped.
13:00:35:187 Sent:  [[solution-prepared correct]]
13:00:38:032 Sent:  solution-correct
13:00:41:086 agent: that sounds right
13:00:41:089 agent: basically, it switches the first and second elements
13:00:56:072 student: good
13:01:00:467 Sent:  response-agree-proposal
```

Figure 9. An excerpt from a KSC-PaL problem solving episode

ditionally, if the interpretation was triggered by a student utterance both the drawing interpretation and the student utterance that triggered the interpretation are sent to the interpreter. The interpreter chooses whether to send the student utterance or the graphical action to TuTalk since TuTalk can only process one student utterance at a time. The other is discarded. Additionally, since interpreting student input of a solution would require extensive natural language processing, the interpreter also matches the solutions entered by the student to a limited range of possible solutions, such as *correct*, *incomplete* or *incorrect*.

An example of the work of the human interpreter is shown in Figure 9, which is an excerpt from a problem solving session with KSC-PaL. It begins with KSC-PaL telling the student that the drawing is complete. The student response to this utterances does not match to any of the expected responses, so *unanticipated-response* is returned to TuTalk in the first pass of interpretation. On the second pass, the utterance is matched to one of the expected student initiative utterances. The student then enters a solution that is sent to the interpreter. The interpreter identifies the solution as correct and sends the matching concept to TuTalk at 13:00:35:187 and then validates TuTalk's interpretation in the following line. The student response at 13:00:56:072 is an expected response to KSC-PaL's utterance in the previous line and is matched in the first pass.

6.2.2 Database

The interface retrieves information related to the problem being solved from a SQLite database. This database contains the problem description, initial drawing and initial code, as well as the constraints used in drawing and coding interpretation. Additionally, the database contains data that allows the agent to draw, make code changes and write problem solutions by sending a specially formatted natural language request to the interface.

6.3 Student Model/Dialogue Planner (SMDP)

KSC-PaL's planner selects scripts and responses to student initiative to manage task initiative shifts. TuTalk uses *scenarios* for guiding the dialogues. These scenarios contain both the *recipe* (script) and *concepts*, which are linguistic concepts used to realize the dialogue. Scripts are hierarchical in nature and consist of a sequence of goals for addressing a topic. Goals usually involve multiple steps where each step consists of an initiation followed by one or more responses (see Figure 10). Generally, the initiation is an agent utterance and responses are possible ways in which a student can respond. However, when using mixed-initiative, the initiation could represent a student utterance while the responses are potential agent replies to the student's utterance. Additionally, TuTalk allows for alternative recipes to achieve a goal. For example there are multiple ways for the agent to realize the goal of explaining a code segment in exercise two (see Appendix D).

Figure 10 is an excerpt from the script for problem 1. The *g* identifies *write-soln_solution-correct* as a goal. Each goal must contain at least one initiation. Initiations are identified by the word *say*. Possible responses to the initiation are listed preceded by the word *if*. The phrases following *say* and *if* are concepts (utterances) defined earlier in the scenario. Each concept can be followed by one or more types of labels. In this case, all concepts are followed with a *sem* label. Additionally, the goal has a *kc* label. This identifies a knowledge component (see below) that is covered through the execution of this goal. In this excerpt, the initiation is writing a solution to the problem, which is the the final step in the problem solving process. There are three likely responses to this initiation: agree with the proposal, disagree with the proposal or respond with a prompt. If the student was the initiating party, the agent would respond with one of the listed responses. The *sem* labels, the words preceded by *sem*, are used to aid the planner in selecting the response. Additionally, *sem* labels can be attached to goals. The planner uses these *sem* labels to chose one of the alternative recipes to achieve the goal. This is described in more detail below.

In drafting the scripts for KSC-PaL (see Appendix D), we authored goals that would encourage shifts in initiative as well as goals that would not encourage initiative shifts. Similarly in drafting responses to student initiative, we drafted both initiative shifting responses as well as responses that would not likely shift initiative. The agent encourages initiative

```
g sol-correct
  say write-soln__solution-correct kc +g8 sem solution
  if response-agree-proposal sem agree
  if feedback-action-prompt sem prompt
  if response-disagree-proposal sem disagree
```

Figure 10. An initiation with multiple responses

shifts by using prompts, hedging, requesting feedback from the student and encouraging student criticism by intentionally making errors in problem solving. TuTalk's planner does not manage these options to the level required by the agent, so a planning module was added to make choices on goal implementation and agent response with the objective of managing shifts in task initiative.

This planner was combined with the student model to create the Student Model/Domain Planner (SMDP). The SMDP consists of a server that manages communication with TuTalk, a student model, a task initiative module that tracks task initiative shifts and a planner that makes decisions based on the current state of task initiative and student knowledge.

6.3.1 SMDP Server

The server opens a port and waits for updates and requests from TuTalk. It anticipates four possible messages from TuTalk:

1. Matched knowledge component (KC): A matched KC message informs the SMDP that a student utterance matched a knowledge concept in the student model. This message is passed to the student model for processing.

2. Request for next possible goal to pursue: A goal request is generated when TuTalk is unable to match a student utterance to a concept. Initially, I believed that the student model could inform TuTalk's interpretation module of likely next goals, however the sparseness of the student model's solution graph made this impractical. Instead a list of concepts with which a student could take initiative was generated from the scripts. This list of goals is returned to TuTalk and is then passed to the human interpreter as possible matching concepts to the student utterance.

3. Request for goal implementation: A request for goal implementation is generated when TuTalk's dialogue manager is presented with alternate goal versions. This request is managed by the planner.

4. Request for agent response: A request for agent response is generated when the agent has multiple ways to respond to student initiative. The responses to these requests are also handled by the planner.

6.3.2 Student Model

The agent required a student model to track the current state of problem solving as well as estimate the student's knowledge of concepts involved in solving the problem in order to guide its behavior. Since TuTalk's student model does not provide these capabilities, a student model which incorporates problem solution graphs (Conati et al., 2002) was added to the agent. Solution graphs are Bayesian networks where each node represents either an action required to solve the problem or a concept required as part of problem solving. A user's utterances and actions are then matched to these nodes. This provides the agent with information related to the student's knowledge as well as the current topic under discussion.

After observing our pilot dialogues, we realized that the solutions to the problems in our domain are different from standard problem-solving tasks. The problems involve either code explanation or program debugging which fall in the area of program comprehension. Based on the models developed (von Mayrhauser and Vans, 1995), the strategies used to achieve the goal of program understanding are varied, but are most commonly classified as either top-down or bottom-up. We found instances of both strategies in the dialogues collected both during the pilot phase and the data collection phase. Therefore we cannot make any assumptions about the order in which a student will analyze code statements. Additionally, some of the problems required the identification of multiple unrelated "bugs". The order in which these bugs are identified is not relevant and the knowledge required to diagnose a certain error may not overlap with the knowledge required to locate a different error. Since the dialogues are peer led, the order and completeness in which these errors were diagnosed varied among the dyads. Therefore a graph comprised of connected subgraphs that represent sections of the code more closely matches our observations. KSC-PaL's student model is a modified version of solution graphs (see Figure 11) that has clusters of nodes representing facts that are relevant to the problem.[1] Each cluster contains facts that are dependent on one another. For example, one cluster represents facts related to the case where the

[1]This model was created using the GeNIe modeling environment developed by the Decision Systems Laboratory of the University of Pittsburgh (http://dsl.sis.pitt.edu).

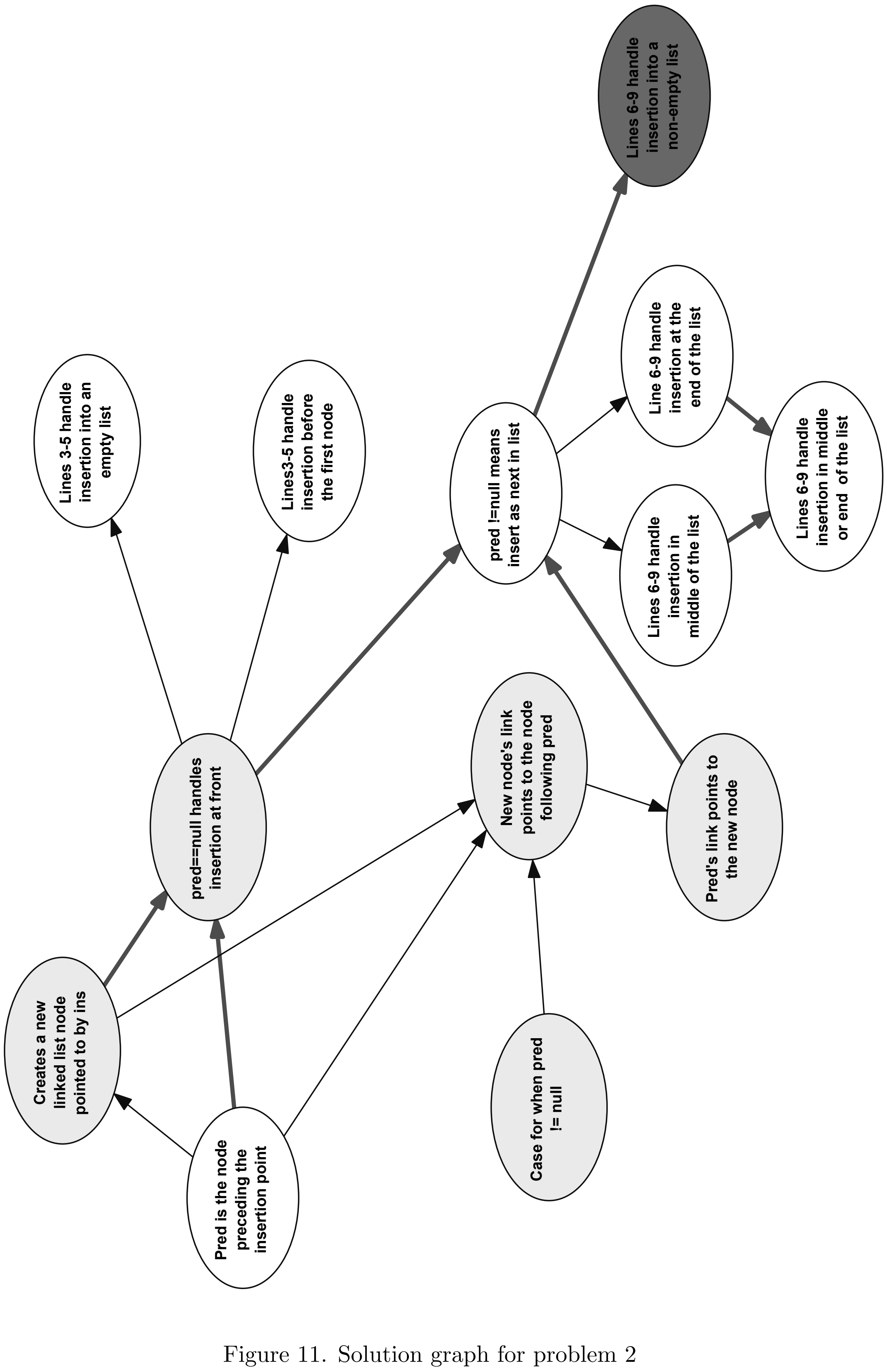

Lines 6-9 handle insertion into a non-empty list
Lines 3-5 handle insertion into an empty list
Lines3-5 handle insertion before the first node
pred !=null means insert as next in list
Line 6-9 handle insertion at the end of the list
Lines 6-9 handle insertion in middle of the list
Lines 6-9 handle insertion in middle or end of the list
pred==null handles insertion at front
New node's link points to the node following pred
Pred's link points to the new node
Creates a new linked list node pointed to by ins
Pred is the node preceding the insertion point
Case for when pred != null

Figure 11. Solution graph for problem 2

input node *pred* is equal to null. Knowledge of this fact is related to various other general knowledge concepts, but those concepts have no relationship to other concepts within the graph.

Using the solution graphs drafted after the pilot data collection, a manual annotation of utterances in dialogues collected during the peer interactions showed that a majority of nodes were either not matched or matched only one utterance showing that the concepts were too specific. In redrafting the solution graphs, nodes were merged and eliminated to more closely match what the students actually discussed.

In these revised solution graphs, there are various types of concepts which are shown in Figure 11 using different shades. The light gray nodes are knowledge concepts related to the main function (for example, the node containing "pred==null handles insertion at the front") while white nodes (e.g., "pred is the node preceding the insertion point") are general knowledge concepts. Those are concepts such as basic programming knowledge. The other major concept type is misconceptions which are colored dark grey (e.g., "lines 6-9 handle insertion into a non-empty list"). Misconception are those knowledge concepts that are incorrect, but the student believes to be correct.

To complete the solution graphs, probabilities of the concepts being known or unknown were assigned to all of the nodes in the graph. Misconception nodes were assigned probabilities calculated from the data collection dialogues. Since the remainder of the concepts were discussed by only a few dyads, I could not use the corpus to assign probabilities to the nodes. Instead, probabilities were assigned to the nodes using the following rules, based on rules defined in (Conati et al., 2002):

1. Those concepts which do not require knowledge concepts contained in other nodes (those nodes with no predecessors) were assigned a 50% probability of being known.
2. Concepts which require knowledge contained in a previous node are assigned a 70% chance of being known, if the parent concept is known. This represents a 30% probability that even if the parent knowledge concept is mastered, a student may fail to apply it, a phenomenon referred to as a slip.
3. Concepts which require knowledge contained in a previous node are assigned a 20% probability of being known if the parent concept is unknown. This represents a 20% probability that the student was able to correctly guess the knowledge concept.
4. When the parent node is a misconception node, the probabilities in items 2 and 3 are reversed.

For those nodes with multiple parents, some were treated as "or" where only one of the parent nodes was required to be known in order to update the probability of the child (thin arcs in Figure 11) while others required all of the parent nodes to be known for the child node's probability to be updated (thick arcs in Figure 11).

A KC sent by TuTalk is matched by the student model to the appropriate node in the solution graph and the probability of that node is updated. Since a matched KC could identify that a student either knows or does not know a problem-solving concept, the node is marked as known or unknown based on the match. Either way, this update will propagate probability updates throughout the Bayesian network.

6.3.2.1 Knowledge Score

To validate that the solution graphs accurately reflect student knowledge related to problem solving, I manually annotated the corpus dialogues with knowledge concepts. A student utterance or interpreted graphical action that reflected knowledge of a concept from the solution graph was marked as known followed by a concept identifier. An utterance or interpreted graphical action that was in opposition to a knowledge concept was marked as unknown followed by a concept identifier.

These annotated dialogues were then used to compute a knowledge score for each exercise for each dyad in the corpus. This was accomplished by updating the solution graphs based on the known or unknown concepts annotated in the dialogues. Using each dyad's solution graph at the completion of an exercise, a knowledge score was calculated by summing the probabilities of all nodes in the graph, except the misconception nodes, and then subtracting the probabilities of the misconception nodes. This score was then normalized by dividing by the maximum possible score for the solution graph. Linear regression, using knowledge score as a predictor of problem solving score, shows that the computed knowledge score is related to the knowledge required for problem solving (see Table XXI). I did not attempt to correlate knowledge score with post-test score since the goal of this analysis was to validate that the solution graph accurately reflected the specific knowledge required for solving a particular problem as opposed to the overall knowledge of data structures as reflected in the pre-/post-test.

Problem	β	R^2	p
2 (list-explanation)	0.58	0.34	0.02
3 (list-debugging)	0.80	0.63	0.00
4 (stack-debugging)	0.74	0.55	0.00
5 (tree-debugging)	0.80	0.64	0.00

TABLE XXI

CORRELATION OF KNOWLEDGE SCORE WITH PROBLEM SOLVING SCORE

6.3.3 Task Initiative Tracker

Based on the results of the analysis in section 5.1, the task initiative tracker contains a classifier for natural language utterances and a separate classifier for drawing and coding actions. On receiving a student or agent utterance or action from the SMDP server, the initiative tracker codes the turn with either student initiative or agent initiative. Natural language utterances are parsed using the Stanford Maximum Entropy Tagger (Toutanova et al., 2003) to provide the appropriate features for use by the task initiative classifier. When classifying a drawing or coding action, the initiative tracker retrieves the student knowledge score for use by the classifier. Once the turn is classified, it is determined whether a shift in initiative has occurred by comparing the current classification with the classification of the previous turn.

When requested by the planner, the task initiative tracker returns the average level of initiative shifts. This is computed by dividing the number of initiative shifts by the total number of turns.

6.3.4 Planner Module

Requests for goal implementation and requests for agent response are managed by the planner module. Two factors determine whether a goal implementation or response that encourages an initiative shift will be selected: (1) the current level of initiative shifts and (2) the change in the student's knowledge score. Task initiative shifts are tracked using the initiative tracker module described above and knowledge levels are maintained in the student model. Goals or responses are selected to encourage initiative shifts when the average level of initiative shifts is less than 0.2117 (mean initiative shifts in KCC episodes in list problems as calculated from corpus data) and the student's knowledge level has not increased since the last time a request for goal implementation or response was requested.

If the planner has determined that an initiative shift should be encouraged, it selects among alternatives based on the holder of task initiative in the previous utterance/action and the *sem* labels associated with each of the potential goal implementions or responses. The combinations of initiative holder and *sem* labels that encourage initiative shifts are shown in Table XXII. For example, to encourage an initiative shift when the the task initiative holder for the previous utterance/action was the student and the agent has the

Sem label	Holder of Task initiative on previous utterance/action
correct	student
partial-correct	agent
incorrect	agent
propose-correct	student
propose-incorrect	agent
propose-partial-correct	agent
disagree	agent
prompt	agent
hedge	agent
request-feedback	agent

TABLE XXII

COMBINATIONS THAT ENCOURAGE INITIATIVE SHIFTS

choice of goal implementations labeled *correct*, *partial-correct* and *incorrect*, the agent will select the goal implementation labeled *correct* because it is likely to result in a shift of initiative

6.4 Summary

The peer learning agent, KSC-PaL, is built on the TuTalk dialogue management system. Creating the agent required the addition of a graphical user interface, an enhanced student model to track student knowledge and a planner to recognize and manage shifts in task initiative.

CHAPTER 7

EVALUATION

I developed two versions of KSC-PaL to test the effectiveness of the model of KCC described above. In the *experimental* version of the agent (PaL), goal versions and responses to student utterances are selected by the planner to maintain a high level of shifts in initiative. In the *control* version (PaL-C), the planner is not consulted for goal versions or responses. Additionally, the script was modified to remove those utterances that were identified as likely to shift initiative: incorrect statements, hedges, prompts and requests for feedback (see Chapter 5). The script excerpts in Figure 12 and Figure 13 provide an example of how the scripts differ. These excerpts contain a goal that would be initiated by the student and the responses that would be made by KSC-PaL. In the experimental condition (Figure 12) the last response is a prompt which could potentially shift initiative. That response was removed from the control condition script (Figure 13).

```
g 2pal-which-is-pred
  say 2pal-d__which-node-is-pred
  if 2pal-i-think-that-pred-is-node-a sem correct1
  if 2pal-i-think-that-pred-is-node-b sem correct2
  if 2pal-i-think-that-pred-is-any-node-in-the-lis sem correct3
  if 2pal-response-dont-know sem prompt
```

Figure 12. An excerpt from the experimental condition script

```
g 2palc-which-is-pred
  say 2palc-d__which-node-is-pred
  if 2palc-i-think-that-pred-is-node-a sem correct1
  if 2palc-i-think-that-pred-is-node-b sem correct2
  if 2palc-i-think-that-pred-is-any-node-in-the-lis sem correct3
```

Figure 13. An excerpt from the control condition script

This chapter describes a user study was conducted in order to answer these questions:

- Does encouraging KCC by manipulating shifts of task initiative lead to improved learning?
- Do the identified initiative shifters have an impact on the level of initiative shifts?
- What impact does encouraging task initiative shifts have on student satisfaction with the agent?

I begin with a description of the method. Next I discuss the agent's impact on student learning. This is followed by an analysis of the effectiveness of the agent in shifting initiative and concludes with an examination of student satisfaction with the system.

7.1 User Study

In the user study, students were assigned to interact with one of the agents, PaL or PaL-C. At the beginning of the session, each student was given a five question pre-test (see Appendix B) to evaluate his or her knowledge prior to interacting with the agent. The first three questions of this test are identical to those on the test that was given prior to the data collection and baseline interactions (see Appendix A). Since, in the user study, the students solved only linked list problems, the remainder of the problems from the original test were removed and two additional problems were added on linked lists .

Prior to problem solving, the students were given a short tutorial on using the interface. They then solved two linked list problems with the agent. These problems were the same as the first two problems used in data collection (see Appendix C). At the conclusion of problem solving, students were given a post-test, identical to the pre-test. Additionally, they were asked to fill out a questionnaire to assess their satisfaction with the agent.

We collected interactions of 25 students, where 13 interacted with PaL and 12 interacted with PaL-C. The participants were recruited from introductory computer science courses at the University of Pittsburgh, the University of Illinois at Chicago, DePaul University and Carnegie Mellon University. The majority of the participants were male students who were currently taking a data structures course or had take a a data structures course within the past 6 months (see Table XXIII).

7.2 Effect on Learning

In order to investigate whether students learned using KSC-PaL, I performed a paired t-test of pre-test and post-test scores. This analysis showed that overall students learn using KSC-PaL (see Table XXIV). T-test analysis also shows that there is a significant difference between pre-test and post-test in the experimental condition and a trend toward

	Currently taking data structures	Took data structures previous semester	Other	Total
Pitt	**4**	**2**	**2**	**8**
Male	3	2	2	7
Female	1	0	0	1
UIC	**5**	**9**	**0**	**14**
Male	4	8	0	12
Female	1	1	0	2
DePaul	**1**	**0**	**0**	**1**
Male	1	0	0	1
CMU	**2**	**0**	**0**	**2**
Male	2	0	0	2
Combined	**12**	**11**	**2**	**25**
Male	10	10	2	22
Female	2	1	0	3

TABLE XXIII

USER STUDY PARTICIPANTS

Condition	N	Pre-test	Post-test	gain	t	p
KSC-PaL (all students)	25	0.61	0.68	0.07	2.90	0.01
PaL	13	0.60	0.66	0.06	2.55	0.02
PaL-C	12	0.62	0.69	0.07	2.03	0.06
PaL plus upper quartile PaL-C subjects	18	0.61	0.68	0.07	3.29	0.00
PaL-C less upper quartile PaL-C subjects	7	0.66	0.66	0.00	−0.96	ns

TABLE XXIV

LEARNING USING KSC-PAL

a significant difference in the control condition. However, there is no significant difference between the gains in the two groups.

7.2.1 Learning Comparison with Other Experiments

Given that there are three questions common to the tests given in the data collection interactions, the baseline interactions and the KSC-PaL user studies, I performed a statistical analysis using t-tests and ANOVA to see if there was any difference in learning across conditions. A summary of the mean test scores and mean learning gains for the three common questions is shown in Table XXV.

Using only the scores from the three common questions, paired t-tests showed that there was no significant difference between pre-test score and post-test score in any of the conditions. Additionally, an ANOVA analysis showed that there was no significant difference

Condition	N	Pre-test M	Post-test M	Gain M
KSC-PaL (all students)	25	0.65	0.68	0.03
PaL	13	0.65	0.63	0.03
PaL-C	12	0.69	0.73	0.04
Data Collection	30	0.69	0.72	0.03
Baseline	20	0.57	0.59	0.01

TABLE XXV

LEARNING ACROSS EXPERIMENTS (QUESTIONS 1-3)

across the conditions for learning gains which is defined as the difference between post-test score and pre-test score.

7.3 Initiative Shifts and Learning

In both conditions, the agent tracks the initiative holder in each utterance using the classifiers described in Chapter 6. A manual annotation of these utterance showed that the level of accuracy of the classifiers was as expected. 747 of 937 (80.15%) of utterances and drawing actions were correctly classified. However, in order to evaluate the effectiveness of initiative shifts, the following analysis uses the manually annotated data.

To examine the impact of initiative shifts on learning, I used two measures of shifts: (1) the number of shifts and (2) normalized initiative shifts, calculated by dividing the number of initiative shifts by the total number of utterances and drawing actions for the session.

Given that the control condition does not encourage initiative shifts but neither does it prevent them, I combined the subjects in the control condition with levels of initiative shifts in the upper quartile of the combined group with the experimental subjects. As shown in Table XXIV, when compared with the remaining control condition subjects there is a difference in learning between the two groups. A t-test performed on the gains between the two groups showed the difference is significant (t = 2.35, p= 0.03) . The effect size (d) is 0.18 which is considered a moderate difference.

Additionally, using multiple linear regression, the measures described above were used as predictors of post-test score after regressing out the impact of pre-test score. Table XXVI shows that while the correlations are significant or trending toward significance, the impact is relatively small. If this same analysis is applied to those subjects with a pre-test score below the mean, there is a larger impact of initiative shifts on post-test score. Analysis of high pre-test subjects showed no significant correlation of post-test score with initiative shifts or normalized initiative shifts.

7.4 Agent's Ability to Shift Initiative

In the experimental condition, KSC-PaL attempts to shift initiative in order to maintain a certain level of initiative shifts. This threshold appears to be set too low, since KSC-PaL rarely selected responses or goal implementations that would encourage a shift in initiative. Only 34 of the 200 requests for response or goal implementation (17%) resulted in a selection to shift initiative.

Predictor of Post-test	β	R^2	p
Initiative shifts	0.24	0.03	0.06
Normalized initiative shifts	0.28	0.01	0.02
Low pre-test subjects (n=14)			
Initiative shifts	0.45	0.07	0.09
Normalized initiative shifts	0.49	0.16	0.04

TABLE XXVI

IMPACT OF INITIATIVE SHIFTS ON LEARNING

Therefore, in order to examine the effectiveness of encouraging initiative shifts, I used an alternate method. As mentioned above, the script for the experimental condition included agent utterances that encourage initiative shifts while these types of utterances were generally excluded from the script for the control condition. Thus, the students in the experimental condition were more likely to encounter those utterances that encourage initiative shifts. To examine the impact of this difference, the dialogues in the user study were semi-automatically annotated with the following encouragers of initiative shifts:

- hedge
- request for feedback
- incorrect statements
- prompts

This was accomplished by collecting all of the agent responses and identifying those responses that fall into one of the categories listed above. Since the agent has a limited set of responses, the transcripts were queried for matching utterances and automatically coded with the appropriate labels.

First I investigated whether the number of shift encouraging utterances had an impact on learning by using multiple regression to predict post-test score using pre-test score + initiative shift utterances. This was not statistically significant.

I then ran a t-test to see if the number of utterances tagged as shift encouragers differed between control sessions and experimental sessions. An unpaired t-test showed that they were significantly different ($t = 3.28$, $p = 0.0036$). I then used linear regression to see if there was a correlation between the number of these shift inducing utterances and number of initiative shifts that occurred. This was also significant ($\beta = 0.40$, $R^2 = 0.16$, $p = 0.04$). This result suggests that these shift encouragers do have an impact on the number of initiative shifts that occur during a problem solving session.

7.5 Student Satisfaction

At the conclusion of problem solving, students were asked to complete a short survey relating to their satisfaction using KSC-PaL. The survey consisted of statements to which the students were to classify as agreeing or disagreeing. Responses were on a 5 point Likert scale, with 1 representing strong disagree and 5 representing strongly agree. The statements on the survey are shown in Table XXVII.

Statement	Control Condition M	Experimental Condition M
The agent helped me learn about linked lists	3.54	3.08
Working with the agent is like working with a classmate	3.23	3.38
I would use the agent on a regular basis, for other topics (like trees)	3.92	3.31
The agent understands what I am saying	3.77	3.23
The agent responds appropriately to what I am saying	3.54	3.46
I found what the agent said repetitive	3.00	3.08
I felt like I had control over solving the problems, and the agent wasn't trying to take charge too often.	3.31	3.69

TABLE XXVII

STUDENT SURVEY - AVERAGE RESPONSES

There were no significant differences between the responses to these questions for those in the control condition versus those in the experimental condition suggesting that attempting to shift initiative does not have a negative impact on student satisfaction with the agent.

7.6 Summary

My evaluation of KSC-PaL found that students learned using the agent. Although there was no significant difference between the conditions, I found those students who had higher normalized initiative shifts, regardless of condition, learned more. I also found that this effect was more pronounced for students who began with a lower level of knowledge regarding linked lists.

Additionally, in the experimental condition, KSC-PaL was successful in encouraging shifts in initiative using the identified shift encouraging cues. These attempts to shift initiative did not have a negative impact on student satisfaction with the agent, demonstrated by the fact that students in the experimental condition reported satisfaction levels equal to the satisfaction level of students in the control condition.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

This dissertation presented a model of knowledge co-construction that was incorporated into an artificial agent. It was divided in three major areas: (1) an analysis of peer learning interactions to identify computational correlates of knowledge co-construction; (2) an analysis of initiative and initiative shifts in order to embed the correlates of KCC into a peer learning agent; (3) the development of a peer agent and an evaluation of its effectiveness.

There are four major conclusions from my extensive analysis of problem solving interactions between pairs of human students.

- KCC does correlate with learning in the domain of computer science data structures. However, the KCC correlates previously identified in literature are not suitable for use in a peer learning agent, since there is no reliable way to automatically identify these correlates.
- Shifts in both dialogue initiative and task initiative correlate with learning. This finding is in agreement with the widely-accepted constructivist theory of learning.
- On average, KCC episodes contain more task initiative shifts per utterance than do problem solving activities outside of KCC episodes. This suggests that shifts in task initiative shifts can be used to identify episodes of KCC.

- Task initiative shifts within KCC episodes correlate with learning, especially for students who have a lower level of domain knowledge prior to the peer learning interaction.

These conclusions directed the second phase of my research. In this phase, I derived methods for a peer learning agent to manage shifts in task initiative in order to encourage KCC and learning. This required that the agent have the ability to recognize task initiative and to encourage shifts in task initiative when appropriate. In the first task, recognizing task initiative, I used machine learning to build classifiers of task initiative and found that two classifiers, one for natural language utterances and one for graphical actions, performed better than a single classifier. For the second task, encouraging task initiative shifts, I performed a second corpus analysis to identify cues that often lead to shifts in task initiative and whose resulting shift leads to an increase in knowledge. I discovered that the following cues were best at both tasks: invalid statements, prompts, hedges and requests for feedback.

Lastly, I built an agent, KSC-PaL, that incorporates the above analysis to promote learning. KSC-PaL is an innovative peer learning agent in that it attempts to shift task initiative between itself and the student. Therefore, unlike other peer learning agents, it shifts roles from more experienced peer to less-experience peer within a single problem-solving episode. A user study conducted with the KSC-PaL showed the students learned using KSC-PaL and liked working with the agent. Additionally, KSC-PaL was successful in encouraging shifts in initiative.

In addition to the research findings mentioned above, this research also contributed:

- a corpus of peer learning interactions
- one of the first systems that aids students in learning computer science data structures

8.2 Improvements on Current Work

Since in the current implementation of KSC-PaL, the agent choses to shift initiative based on a fixed level of average initiative shifts, it would be interesting to explore varying the threshold for initiative shifts. There may be some ideal level of initiative shifts that encourages learning without decreasing student satisfaction with the agent.

A second improvement to the current implementation of KSC-PaL would be to enhance the scripts used by the agent. The corpus generated from the user study with KSC-PaL could be used to identify additional topics initiated by students. These topics could then be scripted to enhance KSC-PaL's interaction with students.

KSC-PaL would also benefit from longer problem solving sessions that contained more problems. This would provide more opportunities to encourage shifts in initiative and potentially more opportunities for learning. Adding problems for additional data structures, such as stacks and trees could also enhance a student's experience with KSC-PaL.

Perhaps the most important improvement on current work would be to incorporate more sophisticated natural language understanding technology. With improved NLU, the human interpreter could be removed from the system. This would allow the system to be deployed in classrooms or potentially on the internet.

8.3 Long-term Research Directions

One important long-term research direction would be to extend this analysis to other domains. It's possible that these findings are specific to this type of problem solving. Further research would be necessary to show that these findings hold across other domains.

Extending the analysis across other domains or with more problems involve creating scripts for the new problems. Currently, creating scripts for KSC-PaL is a very manual and time-instensive activity. Further corpus analysis could derive interaction patterns or templates that could be used to generate scripts for problems in data structures or other domains.

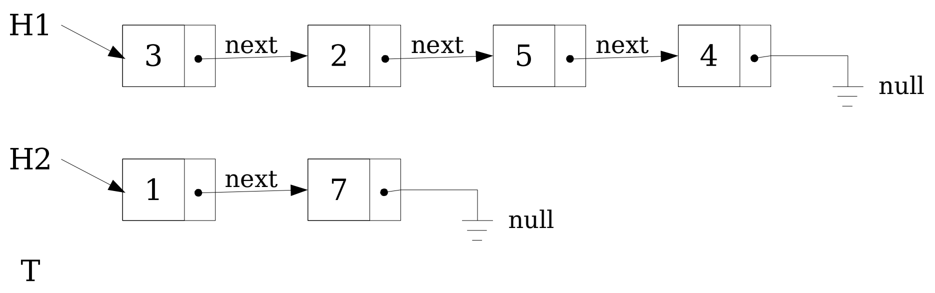
Finally, the graphical workspace is domain specific and it would be difficult to extend it to problems in other domains. Incorporation of sketch-based educational software, such as CogSketch (Forbus et al., 2008), would allow the graphical workspace to be used in any domain.

APPENDICES

Appendix A

DATA COLLECTION PRE-TEST AND POST-TEST

You have the following two *linked lists*, starting from the head pointers H1 and H2. You also have a temporary pointer T.



1. Look at the following procedure. The procedure is written in pseudo C/C++/Java, but don't worry about programming details such as declarations etc. What is the status of the data structures after its execution? Draw a picture representing them.

```
T = H2;
while (T.next ≠ null) {
    T = T.next;
}
T.next = H1;
```

2. Consider the following "variation" of the same procedure. Why doesn't it work?

```
T = H2;
while (T ≠ null) {
    T = T.next;
}
T.next = H1;
```

3. Look at the following code fragment for deleting an input node *del* from a singly linked list headed by first. It is written in Java but don't worry about programming details. What are the two possible singly linked list configurations for which it will not work?

```
public void delete (SLLNode del) {
  // Delete node del in this SLL.
   SLLNode next = del.next;
   SLLNode pred = first;
   while (pred.next != del)
      pred = pred.next;
   pred.next = next;
 }
```

You may assume the following class definitions:

```
public class SLL {
  // Each SLL object is an SLL header.
  // This SLL is represented by a reference to its first node (first).
  private SLLNode first;

  public SLL () {
   //Construct an empty SLL.
   this.first = null;
  }

  public class SLLNode {
   // Each SLLNode object is an SLL node.
   // This node consists of an element (element) and a link to its
   // successor (next).
   protected Object element;
   protected SLLNode next;

  protected SLLNode (Object elem, SLLNode next){
   // Construct an SLL node with element elem and successor next.
   this.element = elem; this.next = next;
  }
}
```

4. The following code fragment uses a singly-linked list to implement an unbounded stack. It is written in Java but don't worry about programming details. What one possible stack configuration would it not handle properly?

```
public Object removeLast () {
  // Remove and return the element at the top of this stack
  Object topElem = top.element;
  top = top.next;
  return topElem;
}
```

You may assume the following class definition in addition to the class definitions in question 3:

```
public class LinkedStack implements Stack {
  // Each Linked Stack object is an unbounded stack whose elements are
  // objects.
  // This stack is represented as follows: top is a link to the first
  // node of an SLL containing the stack's elements, in top-to-bottom
  // order.
  private SLLNode top;


////////// Constructor ///////////

public LinkedStack () {
  // Construct a stack, initially empty.
  top = null;
 }
```

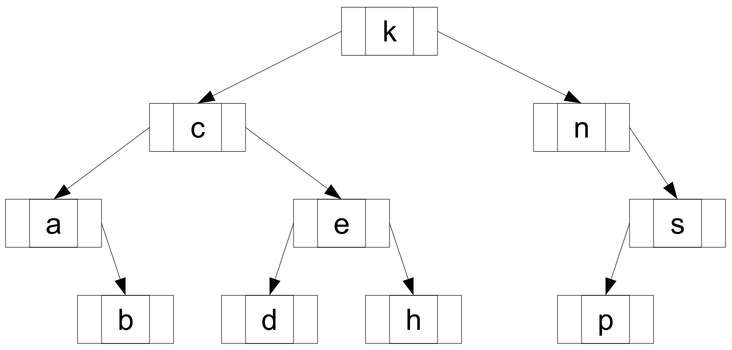
You have a *stack* data structure. The operations defined on it are *push*, *pop*, and *top*, with the usual semantics for stacks.

5. Your stack is initially empty. Write down the state of the stack and the content of the variable x after executing each of the following operations.

Operation	*Stack*	*x*
`x = "A";`		
`push("B");`		
`push("C");`		
`push(x);`		
`pop();`		
`x = top();`		
`push("A");`		
`push(top());`		

6. Mention an application (either a computer application or a real world situation) in which you think a stack structure may be appropriate.

The following picture represents a *binary search tree* (BST).



7. Show the keys with which “d” will be compared if you search for “d” in the given tree.

8. Insert a node containing “m” into the binary search tree. Draw the resulting tree.

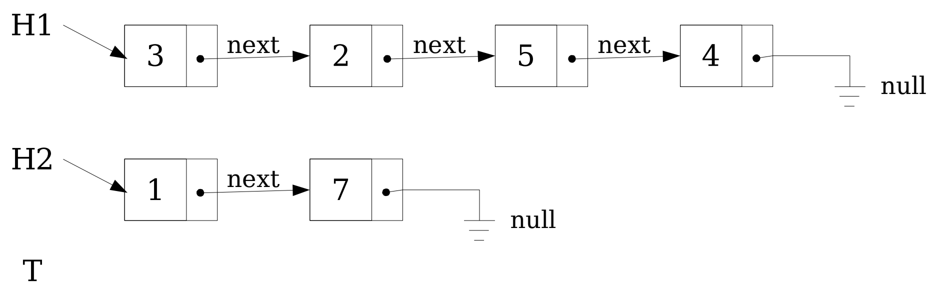
9. Delete the node that contains “c” from the original binary search tree. Draw two possible final trees.

10. In general, is it more difficult to insert a new node in a BST or to delete an existing node from a BST? Explain why.

Appendix B

EVALUATION PRE-TEST AND POST-TEST

You have the following two *linked lists*, starting from the head pointers H1 and H2. You also have a temporary pointer T.



1. Look at the following procedure. The procedure is written in pseudo C/C++/Java, but don't worry about programming details such as declarations etc. What is the status of the data structures after its execution? Draw a picture representing them.

```
T = H2;
while (T.next ≠ null) {
    T = T.next;
}
T.next = H1;
```

2. Consider the following "variation" of the same procedure. Why doesn't it work?

```
T = H2;
while (T ≠ null) {
    T = T.next;
}
T.next = H1;
```

3. Look at the following code fragment for deleting an input node *del* from a singly linked list headed by first. It is written in Java but don't worry about programming details. What are the two possible singly linked list configurations for which it will not work?

```
public void delete (SLLNode del) {
  // Delete node del in this SLL.
   SLLNode next = del.next;
   SLLNode pred = first;
   while (pred.next != del)
      pred = pred.next;
   pred.next = next;
 }
```

You may assume the following class definitions:

```
public class SLL {
  // Each SLL object is an SLL header.
  // This SLL is represented by a reference to its first node (first).
  private SLLNode first;

  public SLL () {
   //Construct an empty SLL.
   this.first = null;
  }

  public class SLLNode {
   // Each SLLNode object is an SLL node.
   // This node consists of an element (element) and a link to its
   // successor (next).
   protected Object element;
   protected SLLNode next;

  protected SLLNode (Object elem, SLLNode next){
   // Construct an SLL node with element elem and successor next.
   this.element = elem; this.next = next;
  }
}
```

4. Draw the list created by the following code. Assume the definition of SLLNode from problem 3.

```
SLLNode list = new SLLNode(10, null);
SLLNode list1 = new SLLNode(14, null);
list.next = list1;
list1 = new SLLNode(18, null);
list1.next = list.next;
list.next=list1;
```

5. A doubly-linked list contains nodes that have links to both the succeeding and preceding nodes. The following code is used with the doubly-linked list class DLL. Assume it is similar to the SLL class defined above except that each node has two references, next and previous. Explain what each line does and the end result of the execution of the method.

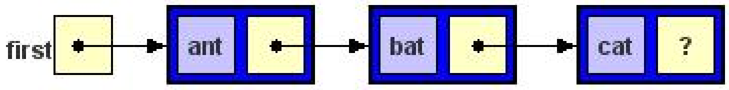
```
public void mystery(DLLNode node1, DLLNode node2){
  node1.next = node2.next;
  node2.next.previous = node1;
  node1.previous = node2;
  node2.next = node1;

}
```

Appendix C

COLLABORATIVE DATA STRUCTURE EXERCISES

1. Given the singly linked list shown below, where "first" is a link to the first node in the list, what does the following code fragment do?

1. Chat with each other to come to an agreement on what you think it does
2. Draw the linked list it produces in the graphics pane
3. Enter your explanation of what it does via the "enter explanation" button.
4. When you are done, select the "done" button

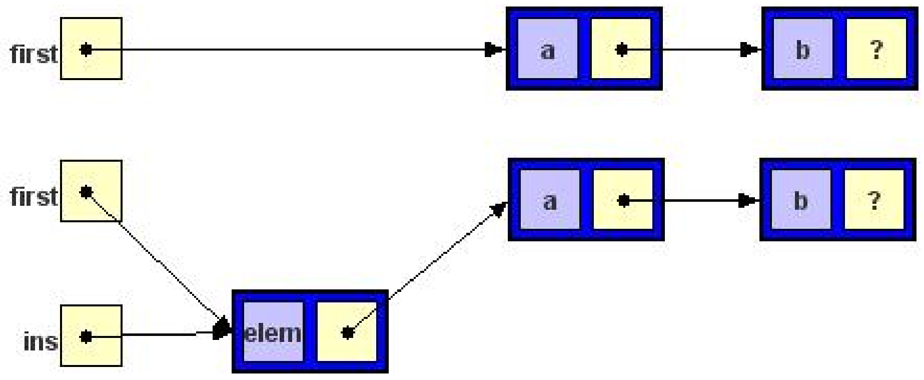


Initial Code:

```
SLLNode second = first.next;
first.next = second.next;
second.next = first;
first = second;

//assume the following class definitions:

public class SLL {
  // Each SLL object is an SLL header
  // This SLL is represented by a reference to its first node (first).
  private SLLNode first;

  public SLL () {
  // Construct an empty SLL.
    this.first = null;
}

public class SLLNode {
  // Each SLLNOde object is an SLL node.

  // This node consists of an element (element) and a link to its successor (next).
  protected Object element;
  protected SLLNode next;

  protected SLLNode (Object elem, SLLNode next) {
  // Construct an SLL node with element elem and successor next.
    this.element = elem;
    this.next = next;
  }
}
```

2. The method at line 1 inserts a new node with data "elem" before the node indicated by "pred" in a singly-linked list headed by the reference pointer "first". Line 2 creates the new node and what follows depends on where the new node is to be inserted. Lines 3-5 insert the new node before the first node. The new node's successor becomes "first", and "first" itself becomes a link to the new node. Lines 3-5 handle insertion in an empty linked list if "first" happens to be null as well as insertion before the first node of a nonempty linked list. This is illustrated below. Explain lines 6-9 in a similar manner:

1. Chat with each other to come to an agreement on what 2 insertion cases these lines handle
2. Draw an illustration of lines 6-9
3. Enter your joint explanation of lines 6-9 by selecting the "enter explanation" button
4. When you are done, select the "done" button



Initial Code:

```
1.  public void insert (Object elem, SLLNode pred){
2.      SLLNode ins = new SLLNode(elem, null);
3.      if (pred == null) {
4.          ins.next = first;
5.          first = ins;
```

```
     } else {
        ins.next = pred.next;
        pred.next = ins;
     }
}

//assume the following classes

public class SLL {
   // Each SLL object is an SLL header.

   // This SLL is represented by a reference to its first node (first).
   private SLLNode first;

   public SLL () {
   // Construct an empty SLL.
    this.first = null;
   }
}
public class SLLNode {
   // Each SLLNode object is an SLL node.

   // This node consists of an element (element) and a link to its
   // successor (next).
   protected Object element;
   protected SLLNode next;
   protected SLLNode (Object elem, SLLNode next) {
   // Construct an SLL node with element elem and sucessor next.
      this.element = elem;
      this.next = next;
   }
}
```

3. The following code is syntactically correct and is intended to create a linked list with the following structure 88->2->3->5 and print it out. Does the code do this? If not, come to an agreement on what the logic errors are and how to fix them. If there are errors:

1. Mark each logic error you find in the code with a comment as to what the error is
2. Mark up the code to fix any errors you find but do not be concerned about correct java syntax. An informal pseudocode or English description of how you'd change it is fine.
3. Enter a summary of any logic errors you find or indicate that there are none by selecting the "enter explanation" button
4. When you are done select the "done" button

Initial Code:

```
public class link {
        public int value;              // value of element
        public link next;              // reference to next

        // constructor
        public link(int n, link ln)
        {
               value = n;
               next = ln;
        }

        public static link insertBeginning (int v, link h)
        {
               link newhead = null;
               newhead = new link(v, h);
               return newhead;
        }

        public static link removeBeginning (link h)
        {
               link linktoremove = h;
               return linktoremove.next;
        }

        public static void removeAfterValue (int keyvalue, link headlink)
        {  //searches for the first occurrence of keyvalue and
           //removes the node after it
               link p = headlink;
               while (p != null) {
                     if (p.value == keyvalue){
                         p.next = p.next.next;
                         return;}
                     else
                         p=p.next;
               }
              return;
        }

        public static void main(String args[])
        {
               link head = null;  // initialize list head

               // add some entries to list
               for (int i = 5; i >= 1; i--)
                    head = new link(i, head);

               removeBeginning (head);
               insertBeginning (88, head);
               removeAfterValue (3, head);

               link p = head;
               while (p != null) {
                      System.out.println(p.value);
                      p = head.next;
               }
        }
}
```

4. The following code is syntactically correct and is intended to read in a string of characters, create a stack using the first 10 characters and then print the stack from top to bottom. For example, an input of `mississippi` results in an output of `ppississim`. Does the code work? If not, come to an agreement on what the logic errors are and how to fix them.

1. Mark each logic error you find in the code with a comment as to what the error is
2. Mark up the code to fix any errors you find but do not be concerned about correct java syntax. An informal pseudocode or English description of how you'd change it is fine.
3. Enter a summary of any logic errors you find or indicate that there are none by selecting the "enter explanation" button
4. When you are done select the "done" button

Initial Code:

```
// Stack.java: stack implementation
public class Stack {
   private int maxStack;
   private int emptyStack;
   private int top;
   private char[] items;

   public Stack(int size) {
      maxStack= size;
      emptyStack = -1;
      top = emptyStack;
      items = new char[maxStack];
   }

   public void push(char c) {
      items[++top] = c;
   }

   public char pop() {
      return items[top--];
   }

   public boolean full()  {
      return top + 1 == maxStack;
   }

   public boolean empty()  {
      return top == emptyStack;
   }
}


// Stackmain.java: use the stack
import java.io.*;
public class Stackmain {

  public static void main(String[] args)
       throws IOException {
    Stack s = new Stack(10); // 10 chars
    char ch;
    while ((ch = (char)System.in.read())
          != '\n')
       s.push(ch);
    System.out.print(s.pop());
    System.out.println();
  }
}
```

5. The following code searches for an input element x in a binary search tree. The code is syntactically correct but does it work? If not, come to an agreement on what the logic errors are and how to fix them.

1. Mark each logic error you find in the code with a comment as to what the error is
2. Mark up the code to fix any errors you find but do not be concerned about correct java syntax. An informal pseudocode or English description of how you'd change it is fine.
3. Enter a summary of any logic errors you find or indicate that there are none by selecting the "enter explanation" button
4. When you are done select the "done" button

Initial Code:

```
/**
 * Internal method to find an item in a subtree.
 * @param x is item to search for.
 * @param t the node that roots the tree.
 * @return node containing the matched item.
 */
private BinaryNode find( Comparable x, BinaryNode t )
{
    if( t == null )
        return null;
    if( x.compareTo( t.element ) > 0 )
        return find( x, t.left );
    else  //( x.compareTo( t.element ) < 0 )
        return find( x, t.right );

}
```

Appendix D

KSC-PAL SCRIPTS

D.1 Problem 2 - Experimental Condition

```
g prob__2 diff agent-only
  do 2pal-greeting
  do 2pal-explain-6-9 opt sem explain6-9
  do 2pal-top-draw opt sem did-draw
  do 2pal-finish-draw opt sem done-draw
  do 2pal-find-cases opt sem id-cases
  do 2pal-req-write-solution opt sem solution
  do 2pal-finish-problem
  do end__

g 2pal-greeting sem ver1 diff agent-only
  say 2pal-off-topic__ok-some-sort-of-insert sem correct
  if 2pal-off-topic__response-prompt sem prompt
  if 2pal-explain6-9-init do "2pal-explain6-9-init-response abort"
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  else do abort

g 2pal-greeting sem ver2 diff agent-only
  say 2pal-off-topic__all-we-have-to-do-is-explain-li sem correct
  if 2pal-yeah-3-5-is-explained-in-the-bottom-draw sem agree say 2pal-right
  if 2pal-off-topic__response-prompt sem prompt
  if 2pal-explain6-9-init do "2pal-explain6-9-init-response abort"
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  else do abort

g 2pal-greeting sem ver3 diff agent-only
  say 2pal-off-topic__ready-when-you-are
  if 2pal-off-topic__response-prompt sem prompt
  if 2pal-explain6-9-init do "2pal-explain6-9-init-response abort"
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  else do abort

g 2pal-greeting sem ver4 diff agent-only
  say 2pal-off-topic__this-problem-looks-harder
  if 2pal-off-topic__response-prompt sem prompt
  if 2pal-explain6-9-init do "2pal-explain6-9-init-response abort"
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  else do abort

g 2pal-explain-6-9 sem correct
  say 2pal-explain-6-9__insert-as-next kc +g5 sem explain6-9
  if 2pal-response-prompt
  if 2pal-explain6-9-init
  do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
    do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
    do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
  do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think sem prompt

g 2pal-wait diff agent-only
  say 2pal-wait__do-nothing
  if 2pal-explain6-9-init do "2pal-explain6-9-init-response abort"
  if 2pal-what-is-pred do "2pal-explain-pred abort"
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort"
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
  do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-unknown abort" sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort

g 2pal-respond-to-clarification-correct sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-clarification-correct sem agree diff agent-only
  say 2pal-response-agree
  do 2pal-wait

g 2pal-respond-to-clarification-correct sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-clarification-correct sem incorrect diff agent-only
  say 2pal-response-disagree
  do 2pal-wait

g 2pal-respond-to-clarification-incorrect sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-clarification-incorrect sem diagree diff agent-only
  say 2pal-response-disagree
  do 2pal-wait
```

```
g 2pal-respond-to-clarification-incorrect sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-clarification-incorrect sem incorrect diff agent-only
  say 2pal-response-agree
  do 2pal-wait

g 2pal-respond-to-question-general sem hedge diff agent-only
  say 2pal-respond-to-question__i-dont-know
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-correct sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-correct sem disagree diff agent-only
  say 2pal-response-disagree
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-correct sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-correct sem incorrect diff agent-only
  say 2pal-respond-to-counterpropose__i-agree-that-is-better
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-incorrect sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-incorrect sem agree diff agent-only
  say 2pal-respond-to-counterpropose__i-agree-that-is-better
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-incorrect sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-incorrect sem incorrect diff agent-only
```

```
  say 2pal-response-disagree
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-unknown sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-unknown sem agree diff agent-only
  say 2pal-respond-to-counterpropose__i-agree-that-is-better
  do 2pal-wait

g 2pal-respond-to-counterpropose-to-unknown sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-propose-correct sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-propose-correct sem agree diff agent-only
  say 2pal-response-agree
  do 2pal-wait

g 2pal-respond-to-propose-correct sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait

g 2pal-respond-to-propose-correct sem incorrect diff agent-only
  say 2pal-response-disagree
  do 2pal-wait

g 2pal-respond-to-propose-incorrect sem prompt diff agent-only
  say 2pal-response-prompt
  do 2pal-wait

g 2pal-respond-to-propose-incorrect sem agree diff agent-only
  say 2pal-response-agree
  do 2pal-wait

g 2pal-respond-to-propose-incorrect sem hedge diff agent-only
  say 2pal-response-hedge
  do 2pal-wait
```

```
g 2pal-respond-to-propose-incorrect sem correct diff agent-only
  say 2pal-response-disagree
  do 2pal-wait

g 2pal-explain-6-9 sem correct
  say 2pal-explain-6-9__insert-after-pred kc +m4 sem explain6-9
  if 2pal-response-prompt do 2pal-explain-6-9-cont sem prompt
  if 2pal-explain-6-9__and-preds-link-points-to-the-new-node
  kc +m5 sem correct
  if 2pal-explain6-9-init
  do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
  do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
  do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
     do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-explain-6-9-cont
  say 2pal-explain-6-9__and-preds-link-points-to-the-new-node
   kc +m5 sem correct
  if 2pal-ok sem hedge
  if 2pal-i-see-so-it-inserts-after-pred kc +g5 do abort sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
  do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
```

```
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
  do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-explain-6-9 sem correct
  say 2pal-explain-6-9__pred-not-null kc +m3 sem explain6-9
  if 2pal-response-prompt do 2pal-case-pred-not-null sem prompt
  if 2pal-when-will-pred-be-null kc -g8
   do 2pal-case-pred-not-null sem request-info
  if 2pal-why-would-you-make-pred-null do 2pal-why-pred-null
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort

g 2pal-explainlines6-9 sem partial-correct
  say 2pal-explain-6-9__insert-not-beginning kc +g5 sem explain6-9
  if 2pal-ok sem hedge
  if 2pal-right-lines-3-5-handle-insertion-at-the- kc +m2 sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-explain-6-9 sem partial-correct
  say 2pal-explain-6-9__insert-end kc +g7 sem explain6-9
  if 2pal-response-prompt sem prompt
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
     do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do  "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
```

```
   do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-explain-6-9 sem incorrect
  say 2pal-explain-6-9__6-9-help-add-a-node-to-the-beginning-of-
    kc "-m2 +mc2" sem explain6-9
  if 2pal-response-prompt do "2pal-remedy-beginning abort" sem prompt
  if 2pal-response-disagree
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-explain-6-9 sem incorrect
  say 2pal-explain-6-9__the-else-case-is-when-pred-is-not-at-the
   kc -g7 sem explain6-9
  if 2pal-response-prompt do "2pal-remedy-not-end abort" sem prompt
  if 2pal-response-disagree kc +g7 sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
```

```
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-remedy-beginning
  say 2pal-explain-6-9__no-i-was-wrong-3-5-handle-insertion-at-t
   kc +m2 sem correct
  say 2pal-not-sure-what-6-9-do sem agree
  if 2pal-me-neither do 2pal-remedy-beginning-2 sem prompt
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
     do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort

g 2pal-remedy-beginning-2
  say 2pal-explain-6-9__i-think-6-9-just-insert-the-node-as-next
   kc +g5 sem correct
  if 2pal-response-prompt do 2pal-detail-explain sem prompt
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-remedy-not-end
  say 2pal-explain-6-9__oh-itll-still-work-if-pred-is-at-the-end kc +g7
   sem correct
  if 2pal-response-prompt do "2pal-remedy-not-end1 abort" sem prompt
  if 2pal-explain-6-9__explain-end sem agree
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
  do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-remedy-not-end1
  say 2pal-explain-6-9__explain-end kc +g7 sem correct
  if 2pal-response-prompt sem prompt
  if 2pal-response-agree sem agree
  if 2pal-response-disagree sem disagree
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort
  do 2pal-what-you-think

g 2pal-detail-explain sem partial-correct
  say 2pal-explain-6-9__so-the-new-node-points-to-the-node-follo
   kc +m4 sem correct
  if 2pal-response-prompt do 2pal-detail-explain2 sem prompt
  if 2pal-explain-6-9__and-preds-link-points-to-the-new-node
   kc +m5 sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else do abort

g 2pal-detail-explain2
  say 2pal-explain-6-9__and-preds-link-points-to-the-new-node
   kc +m5 sem correct
  if 2pal-response-prompt sem prompt
  if 2pal-i-see-so-it-inserts-after-pred kc +g5 sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort

g 2pal-detail-explain sem correct
  say 2pal-explain-6-9__in-lines-6-9-the-new-nodes-successor-is-
   kc +m5 sem correct
  if 2pal-response-prompt sem prompt
  if 2pal-i-see-so-it-inserts-after-pred kc +g5 sem correct
  if 2pal-explain6-9-init
   do "2pal-explain6-9-init-response abort" sem non-choice
  if 2pal-what-is-pred
   do "2pal-explain-pred abort" sem non-choice
  if 2pal-which-is-pred
   do "2pal-answer-which-is-pred abort" sem non-choice
  if 2pal-off-topic__we-should-draw
   do "2pal-respond-req-draw abort" sem non-choice
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem non-choice
  if 2pal-propose-single-case do 2pal-respond-single-case sem non-choice
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else do abort

g 2pal-why-pred-null sem correct diff agent-only
  say 2pal-explain-6-9__its-a-way-to-deal-with-inserting-the-fir
   kc +m2 sem correct
  do 2pal-what-you-think

g 2pal-why-pred-null sem hedge diff agent-only
  say 2pal-explain-6-9__its-something-to-do-with-inserting-the-f sem hedge
  do 2pal-what-you-think

g 2pal-why-pred-null sem partial-correct diff agent-only
  say 2pal-explain-6-9__if-pred-is-null-it-means-you-want-to-ins kc +g3
    sem partial-correct
  do 2pal-what-you-think
```

```
g 2pal-what-you-think diff agent-only
  say 2pal-off-topic__so-what-do-you-think
  if 2pal-explain-6-9__insert-as-next kc +g5
   say 2pal-response-agree sem correct
  if 2pal-explain-6-9__insert-after-pred kc +m4
   say 2pal-response-agree sem correct
  if 2pal-explain-6-9__pred-not-null kc +m3
   do 2pal-case-pred-not-null sem correct
  if 2pal-explain-6-9__insert-not-beginning kc +g5
   say 2pal-right-3-5-handle-insertion-at-the-front sem partial-correct
  if 2pal-explain-6-9__insert-end kc +g7
   say 2pal-response-agree sem correct
  if 2pal-some-wrong-answer do 2pal-detail-explain sem incorrect
  if 2pal-response-prompt do 2pal-detail-explain sem prompt
  if 2pal-off-topic__we-should-draw do "2pal-respond-req-draw abort"
  if 2pal-find-cases__case-middle-end do 2pal-respond-both-cases
  if 2pal-propose-single-case do 2pal-respond-single-case
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
     do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-unknown abort" sem non-choice
  if 2pal-unan-on-topic-propose-correct
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-explain6-9-init-response sem hedge diff agent-only
  say 2pal-response-hedge

g 2pal-explain6-9-init-response sem correct diff agent-only
  say 2pal-response-agree

g 2pal-explain-pred sem correct diff agent-only
  say 2pal-explain-6-9__i-think-that-pred-is-the-node-that-the-c kc +g1
  do 2pal-what-you-think

g 2pal-explain-pred sem hedge diff agent-only
```

```
  say 2pal-response-not-sure
  do 2pal-what-you-think

g 2pal-answer-which-is-pred sem correct
  say 2pal-explain-6-9__i-think-that-pred-can-be-any-node-in-the
   sem correct

g 2pal-answer-which-is-pred sem hedge diff agent-only
  say 2pal-response-not-sure

g 2pal-top-draw diff agent-only
  do 2pal-req-draw

g 2pal-finish-draw diff agent-only
  say 2pal-off-topic__we-need-to-finish-drawing
  if 2pal-off-topic__response-prompt
   do 2pal-decide-finish-draw sem prompt
  if 2pal-off-topic__ill-finish do 2pal-student-draw sem do-action
  if 2pal-off-topic__dont-know
   do 2pal-request-pencil-draw sem request-action
  if 2pal-d-insert-middle kc +m5
   do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5
   do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-pred do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-ins-element do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-elem-b do 2pal-clarify-draw-finish sem done-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-decide-finish-draw diff agent-only
  say 2pal-off-topic__why-dont-you-try
  if 2pal-off-topic__request-action-reject say
    2pal-off-topic__ok-ill-draw
     do 2pal-pencil-draw-insert-middle sem request-action
  if 2pal-off-topic__request-action-accept
   do 2pal-student-draw sem do-action
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-pred do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-ins-element do 2pal-clarify-draw-finish sem done-draw
  if 2pal-d-elem-b do 2pal-clarify-draw-finish sem done-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-req-draw sem prompt diff agent-only
  say 2pal-off-topic__were-supposed-to-draw-lines-6-9
  if 2pal-off-topic__ill-draw do 2pal-student-draw
  if 2pal-off-topic__you-draw do 2pal-request-pencil-draw
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-req-draw sem request-action diff agent-only
  say 2pal-off-topic__were-supposed-to-draw-this-why-
  if 2pal-off-topic__request-action-reject
   say 2pal-off-topic__ok-ill-draw
   do 2pal-request-pencil-draw sem request-action
  if 2pal-off-topic__request-action-accept
   do 2pal-student-draw sem do-action
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-req-draw sem do-action diff agent-only
  say 2pal-off-topic__were-supposed-to-draw-this-ill-
  if 2pal-off-topic__do-action-accept
   do 2pal-request-pencil-draw sem request-action
  if 2pal-off-topic__do-action-reject do 2pal-student-draw sem do-action
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-request-pencil-draw diff agent-only
  say 2pal-req-pencil
  if 2pal-rel-pencil do 2pal-draw
  else do abort

g 2pal-draw diff agent-only
  say 2pal-d-ins-element sem did-draw
  say 2pal-rel-pencil
  say 2pal-which-is-pred sem request-info
  if 2pal-pred-is-a do 2pal-pencil-draw-insert-middle sem correct1
  if 2pal-pred-is-b do 2pal-pencil-draw-insert-end sem correct2
  if 2pal-i-think-that-pred-is-any-node-in-the-lis
    do 2pal-pencil-draw-insert-middle sem correct1
  if 2pal-response-dont-know do 2pal-determine-pred sem prompt
  if 2pal-d-insert-middle kc +m5
   do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5
   do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  else do abort

g 2pal-pencil-draw-insert-middle diff agent-only
  say 2pal-req-pencil
  if 2pal-rel-pencil sem did-draw say 2pal-d-insert-middle
  else
  say 2pal-rel-pencil sem done-draw
  do 2pal-req-feedback-draw

g 2pal-pencil-draw-insert-end diff agent-only
  say 2pal-req-pencil
  if 2pal-rel-pencil say 2pal-d-insert-end
  say 2pal-rel-pencil sem done-draw
  do 2pal-req-feedback-draw

g 2pal-student-draw diff agent-only
  say 2pal-off-topic__go-for-it sem did-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-drawing sem did-draw
  if 2pal-d-pred do 2pal-done-drawing sem did-draw
  if 2pal-d-ins-element do 2pal-done-drawing sem did-draw
  if 2pal-d-elem-b do 2pal-done-drawing sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-req-feedback-draw
  say 2pal-d__request-feedback-on-action
  if 2pal-d__feedback-action-positive sem agree say 2pal-great
  if 2pal-feedback-action-negative sem disagree say 2pal-hmmm
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-find-cases__case-middle-end
   do 2pal-respond-both-cases sem propose-correct
  if 2pal-propose-single-case
   do 2pal-respond-single-case sem propose-partial-correct
  else do abort

g 2pal-give-feedback-draw sem agree diff agent-only
  say 2pal-d__feedback-action-positive

g 2pal-give-feedback-draw sem prompt diff agent-only
  say 2pal-feedback-action-prompt

g 2pal-clarify-draw-finish diff agent-only
  say 2pal-off-topic__can-i-finish
  if 2pal-off-topic__yes do 2pal-pencil-draw-insert-middle
  if 2pal-off-topic__no say 2pal-off-topic__ok-then-lets-move-on
  else do abort

g 2pal-done-drawing diff agent-only
  say 2pal-off-topic__done-with-drawing
  if 2pal-off-topic__yes do 2pal-pencil-draw-insert-middle
    say 2pal-off-topic__ill-give-it-a-shot
  if 2pal-off-topic__no do 2pal-student-finish-draw
  else

g 2pal-student-finish-draw diff agent-only
  say 2pal-off-topic__let-me-know-when-youre-done
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-drawing
  if 2pal-d-pred do 2pal-done-drawing
  if 2pal-d-ins-element do 2pal-done-drawing
  if 2pal-d-elem-b do 2pal-done-drawing
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  else

g 2pal-determine-pred diff agent-only
  say 2pal-d__i-gues-pred-since-pred-is-not-null-its-
  if 2pal-pred-is-a do 2pal-pencil-draw-insert-middle sem correct1
  if 2pal-pred-is-b do 2pal-pencil-draw-insert-end sem correct2
  if 2pal-response-prompt do 2pal-pencil-draw-insert-middle
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  else do abort

g 2pal-respond-where-draw diff agent-only
  say 2pal-off-topic__use-the-top-drawing
  if 2pal-d-insert-middle kc +m5
   do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-drawing sem did-draw
  if 2pal-d-pred do 2pal-done-drawing sem did-draw
  if 2pal-d-ins-element do 2pal-done-drawing sem did-draw
  if 2pal-d-elem-b do 2pal-done-drawing sem did-draw
  if 2pal-off-topic__response-prompt
  else do abort

g 2pal-respond-req-draw sem request-action diff agent-only
  say 2pal-off-topic__why-dont-you-try
  if 2pal-off-topic__request-action-reject
```

```
    say 2pal-off-topic__ok-ill-draw
    do 2pal-request-pencil-draw sem request-action
  if 2pal-off-topic__request-action-accept
   do 2pal-student-draw sem do-action
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-respond-req-draw sem do-action diff agent-only
  say 2pal-off-topic__ok-ill-draw-it
  if 2pal-off-topic__do-action-accept
   do 2pal-request-pencil-draw sem request-action
  if 2pal-off-topic__do-action-reject do 2pal-student-draw sem do-action
  if 2pal-off-topic__response-prompt do 2pal-student-draw
  if 2pal-d-insert-middle kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-insert-end kc +m5 do 2pal-give-feedback-draw sem done-draw
  if 2pal-d-unrecog do 2pal-done-draw sem did-draw
  if 2pal-d-pred do 2pal-done-draw sem did-draw
  if 2pal-d-ins-element do 2pal-done-draw sem did-draw
  if 2pal-d-elem-b do 2pal-done-draw sem did-draw
  if 2pal-off-topic__where-draw do 2pal-respond-where-draw
  else do abort

g 2pal-off-topic__we-should-draw
  say 2pal-off-topic__we-should-draw
  if 2pal-off-topic__go-for-it sem request-action
  if 2pal-im-not-sure sem prompt
  do 2pal-wait

g 2pal-draw-ins-elem
  say 2pal-d-ins-element kc +m1 sem did-draw
  if 2pal-feedback-action-prompt sem prompt

g 2pal-draw-pred
  say 2pal-d-pred kc +g1 sem did-draw
  if 2pal-feedback-action-prompt sem prompt
```

```
g 2pal-draw-elem-b
  say 2pal-d-elem-b kc +m4 sem did-draw
  if 2pal-feedback-action-prompt sem prompt

g 2pal-draw-insert-middle sem did-draw
  say 2pal-d-insert-middle kc +m5 sem done-draw
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-d__feedback-action-positive sem agree
  else

g 2pal-draw-insert-end sem did-draw
  say 2pal-d-insert-end kc +m5 sem done-draw
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-d__feedback-action-positive sem agree
  else

g 2pal-draw-unrecognizable
  say 2pal-d-unrecog sem did-draw
  if 2pal-feedback-action-prompt sem prompt

g 2pal-where-to-draw
  say 2pal-off-topic__where-draw
  if 2pal-off-topic__use-the-top-drawing sem correct
  say 2pal-off-topic__response-prompt

g 2pal-which-is-pred
  say 2pal-d__which-node-is-pred
  if 2pal-i-think-that-pred-is-node-a sem correct1
  if 2pal-i-think-that-pred-is-node-b sem correct2
  if 2pal-i-think-that-pred-is-any-node-in-the-lis sem correct3
  if 2pal-response-dont-know sem prompt

g 2pal-find-cases sem id-cases diff agent-only
  say 2pal-find-cases__so-what-do-you-think-are-the-two-cases
   sem request-info
  if 2pal-lines-6-9-are-for-when-pred-is-not-equal
    do 2pal-case-pred-not-null sem partial-correct
  if 2pal-response-dont-know do 2pal-propose-case sem prompt
  if 2pal-case-not-empty-list kc +mc2
    do 2pal-response-case-not-empty sem partial-correct
  if 2pal-find-cases__case-middle-end kc +g8
```

```
    do 2pal-response-both-cases sem correct
  if 2pal-find-cases__case-insert-middle kc +g6
    do 2pal-inquire-for-end-case sem partial-correct
  if 2pal-find-cases__case-insert-end kc +g7
    do 2pal-inquire-for-middle-case sem partial-correct
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-propose-correct abort" sem non-choice
  else

g 2pal-case-pred-not-null sem id-cases diff agent-only
  say 2pal-find-cases__hmmm-what-are-the-two-cases-when-pred-nu
   sem id-cases
  if 2pal-response-dont-know do 2pal-ask-pred-null sem prompt
  if 2pal-case-not-empty-list kc +mc2
    do 2pal-response-case-not-empty sem partial-correct
  if 2pal-find-cases__case-middle-end kc +g8
    do 2pal-response-both-cases sem correct
  if 2pal-find-cases__case-insert-middle kc +g6
    do 2pal-inquire-for-end-case sem partial-correct
  if 2pal-find-cases__case-insert-end kc +g7
    do 2pal-inquire-for-middle-case sem partial-correct
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-propose-case sem propose-correct1 diff agent-only
```

```
  say 2pal-find-cases__maybe-insertion-in-the-middle-of-the-lis kc +g6
   sem id-cases
  if 2pal-response-agree-proposal do 2pal-inquire-for-end-case sem agree
  if 2pal-response-disagree-proposal do 2pal-other-ideas sem disagree
  if 2pal-find-cases__case-insert-end do 2pal-response-both-cases
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-propose-case sem propose-correct2  diff agent-only
  say 2pal-find-cases__hmm-what-about-insertion-at-the-end-of-t
   kc +g7 sem id-cases
  if 2pal-response-agree-proposal
   do 2pal-inquire-for-middle-case sem agree
  if 2pal-response-disagree-proposal do 2pal-other-ideas
  if 2pal-find-cases__case-insert-middle do 2pal-response-both-cases
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
    do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-propose-case sem propose-partial-correct diff agent-only
  say 2pal-find-cases__hmmm-what-about-insertion-in-a-non-empt
   sem id-cases
  if 2pal-response-agree-proposal kc +mc2
   do 2pal-response-case-not-empty sem agree
  if 2pal-response-disagree-proposal kc -mc2
   do 2pal-other-ideas sem disagree
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
```

```
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else

g 2pal-propose-case sem propose-incorrect diff agent-only
  say 2pal-find-cases__how-about-when-pred-is-not-at-the-end-of
   kc -g7 sem id-cases
  if 2pal-response-agree-proposal
    do 2pal-remedy-not-end-cases sem prompt
  if 2pal-it-will-still-work-if-pred-is-at-the-end kc +g7
    do 2pal-other-ideas sem correct
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
     do "2pal-respond-to-counterpropose-to-incorrect abort" sem non-choice
  else

g 2pal-ask-pred-null diff agent-only
  say 2pal-find-cases__maybe-it-will-help-to-look-at-the-cases-
  say 2pal-what-are-the-cases-when-pred-is-null sem id-cases
  if 2pal-when-we-want-to-insert-at-the-front-of-t kc "+m2 +g4"
    do 2pal-propose-after-pred-null sem correct
  if 2pal-when-the-list-is-empty kc +g3
    do 2pal-propose-after-pred-null sem partial-correct
  if 2pal-when-we-want-to-insert-at-the-front-or-t kc "+m2 +g3 +g4"
    do 2pal-propose-after-pred-null sem correct
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-unknown abort" sem non-choice
  else
```

```
g 2pal-propose-after-pred-null diff agent-only
  say 2pal-find-cases__for-lines-6-9
  do 2pal-propose-case

g 2pal-inquire-for-middle-case sem correct diff agent-only
  say 2pal-find-cases__ok-one-case-is-insert-in-the-end-but-i-c
  if 2pal-find-cases__case-insert-middle kc +g6
    do 2pal-response-both-cases sem correct
  if 2pal-response-dont-know do 2pal-find-middle-case sem prompt
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-inquire-for-middle-case sem prompt diff agent-only
  say 2pal-find-cases__ok-one-case-is-insert-in-the-end-but-i-c
  if 2pal-find-cases__case-insert-middle kc +g6
    do 2pal-response-both-cases sem correct
  if 2pal-response-dont-know do 2pal-one-case
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-inquire-for-end-case sem correct diff agent-only
  say 2pal-find-cases__ok-one-case-is-insert-at-the-middle-but-
  if 2pal-find-cases__case-insert-end kc +g7
```

```
    do 2pal-response-both-cases sem correct
  if 2pal-response-dont-know do 2pal-find-end-case sem prompt
  if 2pal-unan-on-topic-clarification-correct
    do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
    do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
    do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
    do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-inquire-for-end-case sem prompt diff agent-only
  say 2pal-find-cases__ok-one-case-is-insert-at-the-middle-but-
  if 2pal-find-cases__case-insert-end kc +g7
   do 2pal-response-both-cases sem correct
  if 2pal-response-dont-know do 2pal-one-case
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
   do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
   do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-other-ideas diff agent-only
  say 2pal-find-cases__do-you-have-any-other-ideas sem id-cases
  if 2pal-find-cases__case-middle-end kc +g8
   do 2pal-response-both-cases sem correct
  if 2pal-find-cases__case-insert-middle kc +g6
   do 2pal-inquire-for-end-case sem partial-correct
  if 2pal-find-cases__case-insert-end kc +g7
   do 2pal-inquire-for-middle-case sem partial-correct
  if 2pal-response-prompt
   do 2pal-write-solution-one-case say 2pal-off-topic__lets-move-on
  if 2pal-unan-on-topic-clarification-correct
```

```
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
   do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
   do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-one-case diff agent-only
  say 2pal-find-cases__thats-only-one-case-but-i-cant-think-of- sem id-cases
  if 2pal-me-neither say 2pal-off-topic__lets-move-on
   do 2pal-write-solution-one-case sem prompt
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
    do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-propose-correct
   do "2pal-respond-to-propose-correct abort" sem non-choice
  if 2pal-unan-on-topic-propose-incorrect
   do "2pal-respond-to-propose-incorrect abort" sem non-choice
  else

g 2pal-remedy-not-end-cases diff agent-only
  say 2pal-find-cases__no-i-was-wrong-it-will-work
  if 2pal-response-prompt do 2pal-remedy-not-end-cases1 sem prompt
  if 2pal-find-cases__elem-would-point-to-null kc +g7
   do 2pal-inquire-for-middle-case sem correct
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else
```

```
g 2pal-remedy-not-end-cases1 diff agent-only
  say 2pal-find-cases__elem-would-point-to-null
  if 2pal-response-prompt do 2pal-other-ideas sem prompt
  if 2pal-b-would-point-to-elem-and-elem-would-poi kc +g7
   do 2pal-inquire-for-middle-case sem correct
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-response-case-not-empty sem correct diff agent-only
  say 2pal-find-cases__no-i-dont-think-thats-what-they-mean kc -mc2
  say 2pal-find-cases__case-insert-middle
  if 2pal-response-agree-proposal kc "-mc2 +g6"
   do 2pal-inquire-for-end-case sem agree
  if 2pal-find-cases__case-insert-end kc +g8
   do 2pal-response-both-cases sem correct
  if 2pal-response-disagree-proposal kc -g6 do 2pal-other-ideas sem disagree
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice

g 2pal-response-case-not-empty sem partial-correct diff agent-only
  do 2pal-one-case

g 2pal-response-case-not-empty sem hedge diff agent-only
  say 2pal-find-cases__no-i-dont-think-thats-what-they-mean
  do 2pal-other-ideas

g 2pal-response-both-cases sem prompt diff agent-only
  say 2pal-response-prompt sem id-cases
  do 2pal-write-solution-both-cases
```

```
g 2pal-response-both-cases sem agree diff agent-only
  say 2pal-find-cases__right-it-works-both-in-the-middle-and-at
   kc +g8 sem id-cases
  say 2pal-off-topic__lets-move-on
  do 2pal-write-solution-both-cases

g 2pal-find-end-case sem correct diff agent-only
  say 2pal-find-cases__how-about-insertion-at-the-end-of-the-li
  if 2pal-response-agree-proposal do 2pal-response-both-cases sem agree
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-find-end-case sem prompt diff agent-only
  do 2pal-one-case

g 2pal-find-middle-case sem correct diff agent-only
  say 2pal-find-cases__how-about-insertion-in-the-middle-of-the
  if 2pal-response-agree-proposal do 2pal-response-both-cases sem agree
  if 2pal-unan-on-topic-clarification-correct
   do "2pal-respond-to-clarification-correct abort" sem non-choice
  if 2pal-unan-on-topic-clarification-incorrect
   do "2pal-respond-to-clarification-incorrect abort" sem non-choice
  if 2pal-unan-on-topic-question
   do 2pal-respond-to-question-general sem non-choice
  if 2pal-unan-on-topic-counterpropose
   do "2pal-respond-to-counterpropose-to-correct abort" sem non-choice
  else

g 2pal-find-middle-case sem prompt diff agent-only
  do 2pal-one-case

g 2pal-respond-both-cases sem agree diff agent-only
  say 2pal-response-agree-proposal kc +g8 sem id-cases
```

```
g 2pal-respond-both-cases sem prompt diff agent-only
  say 2pal-response-prompt kc +g8 sem id-cases

g 2pal-respond-single-case diff agent-only
  say 2pal-response-prompt

g 2pal-propose-case-middle-end sem id-cases
  say 2pal-find-cases__case-middle-end kc +g8 sem id-cases
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-prompt sem prompt
  else

g 2pal-propose-case-not-empty-list sem partial-correct
  say 2pal-find-cases__how-about-insertion-into-a-not-empty-lis
   kc +mc2 sem id-cases
  if 2pal-find-cases__no-i-dont-think-thats-what-they-mean sem disagree
  if 2pal-response-agree-proposal sem agree
  else

g 2pal-propose-case-insert-middle
  say 2pal-find-cases__case-insert-middle kc +g6 sem id-cases
  if 2pal-response-prompt sem prompt
  if 2pal-response-agree-proposal sem agree
  else

g 2pal-propose-case-insert-end
  say 2pal-find-cases__case-insert-end kc +g7 sem id-cases
  if 2pal-response-prompt sem prompt
  if 2pal-response-agree-proposal sem agree
  else

g 2pal-req-write-solution sem request-action diff agent-only
  say 2pal-off-topic__you-write-explanation
  if 2pal-off-topic__request-action-accept
   do 2pal-student-explanation sem do-action
  if 2pal-off-topic__request-action-reject
   do 2pal-agent-explanation sem request-action
  if 2pal-write-soln__solution-desc-only sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-correct kc +g8
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-middle kc +g6
```

```
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-incomplete
   do 2pal-remedy-explanation sem solution
  if 2pal-write-soln__solution-wrong
   do 2pal-remedy-explanation sem solution

g 2pal-req-write-solution sem propose diff agent-only
  say 2pal-off-topic__someone-write-explanation
  if 2pal-off-topic__request-action-accept
   do 2pal-student-explanation sem do-action
  if 2pal-off-topic__request-action-reject
   do 2pal-agent-explanation sem request-action
  if 2pal-write-soln__solution-desc-only sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-correct kc +g8
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-incomplete
   do 2pal-remedy-explanation sem solution
  if 2pal-write-soln__solution-wrong
   do 2pal-remedy-explanation sem solution

g 2pal-req-write-solution sem do-action
  say 2pal-off-topic__me-write-explanation sem solution
  if 2pal-off-topic__do-action-accept
   do 2pal-agent-explanation sem request-action
  if 2pal-off-topic__do-action-reject
   do 2pal-student-explanation sem do-action

g 2pal-student-explanation diff agent-only
  say 2pal-off-topic__go-for-it
  if 2pal-write-soln__solution-desc-only sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-correct kc +g8
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-incomplete
   do 2pal-remedy-explanation sem solution
  if 2pal-write-soln__solution-wrong
   do 2pal-remedy-explanation sem solution

g 2pal-remedy-explanation sem correct diff agent-only
  say 2pal-write-soln__that-doesnt-look-right-i-think-youre-sup
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-desc-only
   do 2pal-respond-add-a-case sem solution
  if 2pal-write-soln__solution-incomplete say 2pal-ok-i-guess
  if 2pal-write-soln__solution-wrong say 2pal-ok-i-guess
  if 2pal-response-disagree-proposal say 2pal-ok-i-guess

g 2pal-remedy-explanation sem prompt diff agent-only
  say 2pal-feedback-action-prompt

g 2pal-agent-explanation diff agent-only
  say 2pal-req-pencil
  if 2pal-rel-pencil do 2pal-write-expl
  say 2pal-rel-pencil
  say 2pal-req-feedback-expl
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-disagree-proposal
   do 2pal-you-fix-expl sem disagree
  else

g 2pal-write-solution-both-cases sem request-action diff agent-only
  say 2pal-off-topic__you-write-explanation
  if 2pal-off-topic__request-action-accept
   do 2pal-student-expl-both-cases sem do-action
  if 2pal-off-topic__request-action-reject
   do 2pal-agent-expl-both-cases sem request-action
  else

g 2pal-write-solution-both-cases sem do-action diff agent-only
  say 2pal-off-topic__me-write-explanation
  if 2pal-off-topic__do-action-accept
   do 2pal-agent-expl-both-cases sem request-action
  if 2pal-off-topic__do-action-reject
   do 2pal-student-expl-both-cases sem do-action
  else

g 2pal-student-expl-both-cases diff agent-only
  say 2pal-off-topic__go-for-it
  if 2pal-write-soln__solution-desc-only
   do 2pal-respond-add-both-cases sem solution
  if 2pal-write-soln__solution-correct kc +g8
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-incomplete do 2pal-remedy-both-cases
  if 2pal-write-soln__solution-wrong do 2pal-rememdy-both-cases
  else

g 2pal-respond-add-both-cases diff agent-only
  say 2pal-write-soln__i-think-were-supposed-to-put-the-cases-i
  if 2pal-write-soln__solution-correct kc +g8
   say 2pal-response-agree-proposal sem correct
  if 2pal-response-disagree-proposal say 2pal-ok-i-guess

g 2pal-remedy-both-cases diff agent-only
  say 2pal-write-soln__i-think-were-supposed-to-describe-what-t
  if 2pal-write-soln__solution-correct sem correct
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-desc-only
   do 2pal-respond-add-both-cases sem partial-correct
  if 2pal-response-disagree-proposal say 2pal-ok-i-guess

g 2pal-agent-expl-both-cases
  say 2pal-req-pencil
  if 2pal-rel-pencil do 2pal-write-expl-both-cases
  say 2pal-rel-pencil
  say 2pal-req-feedback-expl
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-disagree-proposal
   do 2pal-you-fix-expl sem disagree

g 2pal-write-expl-both-cases sem correct diff agent-only
  say 2pal-write-soln__solution-correct sem solution
  say 2pal-pause-10

g 2pal-write-expl-both-cases sem partial-correct diff agent-only
  say 2pal-write-soln__solution-desc-only sem solution
  say 2pal-pause-10

g 2pal-you-fix-expl diff agent-only
  say 2pal-off-topic__then-you-fix-it
  if 2pal-write-soln__solution-correct kc +g8 say 2pal-response-prompt
  if 2pal-write-soln__solution-desc-only say 2pal-response-prompt
  if 2pal-write-soln__solution-desc-not-empty say 2pal-response-prompt
  if 2pal-write-soln__solution-incomplete say 2pal-response-prompt
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-prompt
  if 2pal-write-soln__solution-desc-end kc +g7 say 2pal-response-prompt
  if 2pal-write-soln__solution-wrong say 2pal-response-prompt

g 2pal-write-solution-one-case sem request-action diff agent-only
  say 2pal-off-topic__you-write-explanation
  if 2pal-off-topic__request-action-accept
   do 2pal-student-expl-one-case sem do-action
  if 2pal-off-topic__request-action-reject
   do 2pal-agent-expl-one-case sem request-action
  else

g 2pal-write-solution-one-case sem do-action diff agent-only
  say 2pal-off-topic__me-write-explanation
  if 2pal-off-topic__do-action-accept
   do 2pal-agent-expl-one-case sem request-action
  if 2pal-off-topic__do-action-reject
   do 2pal-student-expl-one-case sem do-action
  else

g 2pal-student-expl-one-case diff agent-only
  say 2pal-off-topic__go-for-it
  if 2pal-write-soln__solution-desc-only d
   o 2pal-respond-add-a-case sem solution
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-incomplete
   do 2pal-remedy-one-case sem solution
  if 2pal-write-soln__solution-wrong
   do 2pal-rememdy-one-case sem solution
  else

g 2pal-respond-add-a-case sem correct diff agent-only
  say 2pal-write-soln__you-should-probably-add-the-case-we-foun
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-response-disagree-proposal say 2pal-ok-i-guess
  else

g 2pal-remedy-one-case sem correct diff agent-only
  say 2pal-write-soln__you-should-describe-what-the-code-does-a
  if 2pal-write-soln__solution-desc-middle kc +g6
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-end kc +g7
   say 2pal-response-agree-proposal sem solution
  if 2pal-write-soln__solution-desc-not-empty sem solution
   say 2pal-response-agree-proposal
  if 2pal-write-soln__solution-desc-only
   do 2pal-respond-add-a-case sem solution
  if 2pal-response-disagree-proposal say 2pal-ok-i-guess
  else

g 2pal-agent-expl-one-case diff agent-only
  say 2pal-req-pencil
  if 2pal-rel-pencil do 2pal-write-expl
  else
  say 2pal-rel-pencil
  say 2pal-req-feedback-expl
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-disagree-proposal do 2pal-you-fix-expl sem disagree
  else

g 2pal-write-expl sem partial-correct diff agent-only
  say 2pal-write-soln__solution-desc-only sem solution
  say 2pal-pause-10

g 2pal-sol-correct
  say 2pal-write-soln__solution-correct kc +g8 sem solution
  if 2pal-response-agree-proposal sem agree
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-response-disagree-proposal sem disagree

g 2pal-sol-middle
  say 2pal-write-soln__solution-desc-middle kc +g6 sem solution
  if 2pal-response-agree-proposal sem agree
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-response-disagree-proposal sem disagree

g 2pal-sol-end
  say 2pal-write-soln__solution-desc-end kc +g7 sem solution
  if 2pal-response-agree-proposal sem agree
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-response-disagree-proposal sem disagree

g 2pal-sol-not-empty
  say 2pal-write-soln__solution-desc-not-empty sem solution
  if 2pal-response-agree-proposal sem agree
  if 2pal-feedback-action-prompt sem prompt
  if 2pal-response-disagree-proposal sem disagree

g 2pal-sol-incomplete
  say 2pal-write-soln__solution-incomplete
  if 2pal-req-correct-solution sem disagree
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-prompt sem prompt

g 2pal-sol-incomplete
  say 2pal-write-soln__solution-wrong
  if 2pal-req-correct-solution sem disagree
  if 2pal-response-agree-proposal sem agree
  if 2pal-response-prompt sem prompt

g 2pal-finish-problem diff agent-only
  say 2pal-off-topic__so-are-we-done
  if 2pal-submit-solution do logout
  if 2pal-off-topic__response-prompt do 2pal-submit-solution
  else

g 2pal-submit-solution sem request-action diff agent-only
  say 2pal-off-topic__why-dont-you-press-the-done-but
  if 2pal-off-topic__request-action-accept d
   o 2pal-student-submit sem do-action
  if 2pal-submit-solution do logout
  if 2pal-off-topic__request-action-reject
   do 2pal-press-done sem request-action
  else

g 2pal-submit-solution sem do-action diff agent-only
  say 2pal-off-topic__ill-press-the-done-button
  if 2pal-off-topic__do-action-accept
   do 2pal-press-done sem request-action
  if 2pal-off-topic__do-action-reject
   do 2pal-student-submit sem do-action
  else

g 2pal-student-submit diff agent-only
  say 2pal-wait__do-nothing
  if 2pal-submit-solution
  else do 2pal-press-done

g 2pal-press-done
  say 2pal-submit-solution

g 2pal-unrecognized-on-task
  say 2pal-something-unrecognized-on-task
  if 2pal-response-dont-know do "2pal-wait abort" sem prompt

g 2pal-unanticipated-on-task
  say 2pal-something-unanticipated-on-task
  if 2pal-uh-huh do "2pal-wait abort" sem prompt
```

```
g 2pal-unrecognized-off-task
  say 2pal-off-topic__unrecog
  if 2pal-off-topic__lets-talk-about-the-problem
   do "2pal-wait abort" sem prompt

g 2pal-done-yet
  say 2pal-off-topic__are-we-done
  if 2pal-off-topic__-we-need-to-come-up-with-two-ca
   do "2pal-wait abort"

g 2pal-student-request-feedback
  say 2pal-write-soln__what-do-you-think-of-my-drawingsolution
  if 2pal-response-not-sure do "2pal-wait abort"

g 2pal-clarification sem correct
  say 2pal-new-topic__unan-clarification-correct
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-clarification sem incorrect
  say 2pal-new-topic__unan-clarification-incorrect
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-question
  say 2pal-new-topic__unan-question
  if 2pal-respond-to-question__i-dont-know do "2pal-wait abort"

g 2pal-propose sem correct
  say 2pal-new-topic__unan-propose-correct
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-propose sem incorrect
  say 2pal-new-topic__unan-propose-incorrect
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-counterpropose sem correct
  say 2pal-new-topic__unan-counterpropose-correct
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-counterpropose sem incorrect
  say 2pal-new-topic__unan-counterpropose-incorrect
  if 2pal-response-disagree do "2pal-wait abort" sem disagree
  if 2pal-response-hedge do "2pal-wait abort" sem hedge
  if 2pal-response-agree do "2pal-wait abort" sem agree
  if 2pal-response-prompt do "2pal-wait abort" sem prompt

g 2pal-request-write-expl-not-ready
  say 2pal-off-topic__req-write-expl-not-ready
  if 2pal-off-topic_finish-problem

g 2pal-request-write-expl-ready
  say 2pal-off-topic__req-write-expl-ready
  if 2pal-ok-why-dont-you-try
  say 2pal-write-soln__solution-correct sem solution
  if 2pal-response-agree-proposal

g 2pal-procedure-question
  say 2pal-off-topic__procedure-question
  if 2pal-off-topic__read-prob-description
```

D.2 Problem 2 - Control Condition

```
g prob2palc diff agent-only
  say 2palc-new-problem-2
  if 2palc-set-up do prob__2 do logout
  else do logout

g prob__2 diff agent-only
```

```
  do 2palc-greeting
  do 2palc-explain-6-9 opt sem explain6-9
  do 2palc-top-draw opt sem did-draw
  do 2palc-finish-draw opt sem done-draw
  do 2palc-find-cases opt sem id-cases
  do 2palc-write-solution-both-cases opt sem solution
  do 2palc-finish-problem
  do end__

g 2palc-greeting diff agent-only
  say 2palc-off-topic__lets-start

g 2palc-explain-6-9 sem correct
  say 2palc-explain-6-9__insert-as-next kc +g5 sem explain6-9
  if 2palc-response-prompt
  if 2palc-explain6-9-init
    do "2palc-explain6-9-init-response abort" sem non-choice
  if 2palc-what-is-pred do "2palc-explain-pred abort" sem non-choice
  if 2palc-which-is-pred
    do "2palc-answer-which-is-pred abort" sem non-choice
  if 2palc-off-topic__we-should-draw
    do "2palc-respond-req-draw abort" sem non-choice
  if 2palc-find-cases__case-middle-end
    do 2palc-respond-both-cases sem non-choice
  if 2palc-propose-single-case
    do 2palc-respond-single-case sem non-choice
  else do abort

g 2palc-explain-6-9 sem correct
  say 2palc-explain-6-9__insert-after-pred kc +m4 sem explain6-9
  if 2palc-response-prompt do 2palc-explain-6-9-cont sem prompt
  if 2palc-explain-6-9__and-preds-link-points-to-the-new-node
    kc +m5 sem correct
  if 2palc-explain6-9-init
    do "2palc-explain6-9-init-response abort" sem non-choice
  if 2palc-what-is-pred do "2palc-explain-pred abort" sem non-choice
  if 2palc-which-is-pred
    do "2palc-answer-which-is-pred abort" sem non-choice
  if 2palc-off-topic__we-should-draw
    do "2palc-respond-req-draw abort" sem non-choice
  if 2palc-find-cases__case-middle-end
     do 2palc-respond-both-cases sem non-choice
  if 2palc-propose-single-case
     do 2palc-respond-single-case sem non-choice
  else do abort

g 2palc-explain-6-9-cont
  say 2palc-explain-6-9__and-preds-link-points-to-the-new-node
      kc +m5 sem correct
  if 2palc-ok sem hedge
  if 2palc-i-see-so-it-inserts-after-pred kc +g5 do abort sem correct
  if 2palc-explain6-9-init
      do "2palc-explain6-9-init-response abort" sem non-choice
  if 2palc-what-is-pred do "2palc-explain-pred abort" sem non-choice
  if 2palc-which-is-pred
      do "2palc-answer-which-is-pred abort" sem non-choice
  if 2palc-off-topic__we-should-draw
      do "2palc-respond-req-draw abort" sem non-choice
  if 2palc-find-cases__case-middle-end
      do 2palc-respond-both-cases sem non-choice
  if 2palc-propose-single-case
      do 2palc-respond-single-case sem non-choice
  else do abort

g 2palc-why-pred-null sem correct
  say 2palc-explain-6-9__its-a-way-to-deal-with-inserting-the-fir
      kc +m2 sem correct

g 2palc-explain6-9-init-response sem correct
  say 2palc-response-agree

g 2palc-explain-pred sem correct
  say 2palc-explain-6-9__i-think-that-pred-is-the-node-that-the-c kc +g1

g 2palc-answer-which-is-pred sem correct
  say 2palc-explain-6-9__i-think-that-pred-can-be-any-node-in-the
      sem correct

g 2palc-top-draw diff agent-only
  do 2palc-req-draw

g 2palc-finish-draw diff agent-only
  say 2palc-off-topic__why-dont-you-finish
  if 2palc-off-topic__request-action-reject
```

```
      say 2palc-off-topic__ok-ill-draw do 2palc-pencil-draw-insert-middle
       sem request-action
  if 2palc-off-topic__request-action-accept
       do 2palc-student-draw sem do-action
  if 2palc-d-insert-middle kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-clarify-draw-finish sem done-draw
  if 2palc-d-pred do 2palc-clarify-draw-finish sem done-draw
  if 2palc-d-ins-element do 2palc-clarify-draw-finish sem done-draw
  if 2palc-d-elem-b do 2palc-clarify-draw-finish sem done-draw
  if 2palc-off-topic__where-draw do 2palc-respond-where-draw
  else do abort

g 2palc-req-draw sem request-action diff agent-only
  say 2palc-off-topic__were-supposed-to-draw-this-why-
  if 2palc-off-topic__request-action-reject
       say 2palc-off-topic__ok-ill-draw do 2palc-request-pencil-draw
       sem request-action
  if 2palc-off-topic__request-action-accept
       do 2palc-student-draw sem do-action
  if 2palc-off-topic__response-prompt do 2palc-student-draw
  if 2palc-d-insert-middle kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-draw sem did-draw
  if 2palc-d-pred do 2palc-done-draw sem did-draw
  if 2palc-d-ins-element do 2palc-done-draw sem did-draw
  if 2palc-d-elem-b do 2palc-done-draw sem did-draw
  if 2palc-off-topic__where-draw do 2palc-respond-where-draw
  else do abort

g 2palc-request-pencil-draw diff agent-only
  say 2palc-req-pencil
  if 2palc-rel-pencil do 2palc-draw
  else do abort

g 2palc-draw diff agent-only
  say 2palc-d-ins-element sem did-draw
  say 2palc-rel-pencil
  say 2palc-which-is-pred sem request-info
  if 2palc-pred-is-a do 2palc-pencil-draw-insert-middle sem correct1
  if 2palc-pred-is-b do 2palc-pencil-draw-insert-end sem correct2
  if 2palc-i-think-that-pred-is-any-node-in-the-lis
        do 2palc-pencil-draw-insert-middle sem correct1
  if 2palc-response-dont-know do 2palc-determine-pred sem prompt
  if 2palc-d-insert-middle kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-draw sem did-draw
  if 2palc-d-pred do 2palc-done-draw sem did-draw
  if 2palc-d-ins-element do 2palc-done-draw sem did-draw
  if 2palc-d-elem-b do 2palc-done-draw sem did-draw
  else do abort

g 2palc-pencil-draw-insert-middle diff agent-only
  say 2palc-req-pencil
  if 2palc-rel-pencil sem did-draw say 2palc-d-insert-middle
  else
  say 2palc-rel-pencil sem done-draw
  say 2palc-off-topic__i-done-drawing

g 2palc-pencil-draw-insert-end diff agent-only
  say 2palc-req-pencil
  if 2palc-rel-pencil say 2palc-d-insert-end
  say 2palc-rel-pencil sem done-draw
  say 2palc-off-topic__i-done-drawing

g 2palc-student-draw diff agent-only
  say 2palc-off-topic__go-for-it sem did-draw
  if 2palc-d-insert-middle kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-drawing sem did-draw
  if 2palc-d-pred do 2palc-done-drawing sem did-draw
  if 2palc-d-ins-element do 2palc-done-drawing sem did-draw
  if 2palc-d-elem-b do 2palc-done-drawing sem did-draw
  if 2palc-off-topic__where-draw do 2palc-respond-where-draw
  else do abort

g 2palc-give-feedback-draw sem agree
  say 2palc-d__feedback-action-positive

g 2palc-clarify-draw-finish diff agent-only
```

```
  say 2palc-off-topic__can-i-finish
  if 2palc-off-topic__yes do 2palc-pencil-draw-insert-middle
  if 2palc-off-topic__no say 2palc-off-topic__ok-then-lets-move-on
  else do abort

g 2palc-done-drawing diff agent-only
  say 2palc-off-topic__done-with-drawing
  if 2palc-off-topic__yes do 2palc-pencil-draw-insert-middle
        say 2palc-off-topic__ill-give-it-a-shot
  if 2palc-off-topic__no do 2palc-student-finish-draw
  else

g 2palc-student-finish-draw diff agent-only
  say 2palc-off-topic__let-me-know-when-youre-done
  if 2palc-d-insert-middle kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5
        do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-drawing
  if 2palc-d-pred do 2palc-done-drawing
  if 2palc-d-ins-element do 2palc-done-drawing
  if 2palc-d-elem-b do 2palc-done-drawing
  if 2palc-off-topic__response-prompt do 2palc-student-draw
  else

g 2palc-determine-pred diff agent-only
  say 2palc-d__i-gues-pred-since-pred-is-not-null-its-
  if 2palc-pred-is-a do 2palc-pencil-draw-insert-middle sem correct1
  if 2palc-pred-is-b do 2palc-pencil-draw-insert-end sem correct2
  if 2palc-response-prompt do 2palc-pencil-draw-insert-middle
  if 2palc-d-insert-middle kc +m5
   do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5
   do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-draw sem did-draw
  if 2palc-d-pred do 2palc-done-draw sem did-draw
  if 2palc-d-ins-element do 2palc-done-draw sem did-draw
  if 2palc-d-elem-b do 2palc-done-draw sem did-draw
  else do abort

g 2palc-respond-where-draw diff agent-only
  say 2palc-off-topic__use-the-top-drawing
  if 2palc-d-insert-middle kc +m5
   do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5
   do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-drawing sem did-draw
  if 2palc-d-pred do 2palc-done-drawing sem did-draw
  if 2palc-d-ins-element do 2palc-done-drawing sem did-draw
  if 2palc-d-elem-b do 2palc-done-drawing sem did-draw
  if 2palc-off-topic__response-prompt
  else do abort

g 2palc-respond-req-draw sem request-action diff agent-only
  say 2palc-off-topic__why-dont-you-try
  if 2palc-off-topic__request-action-reject
   say 2palc-off-topic__ok-ill-draw do 2palc-request-pencil-draw
   sem request-action
  if 2palc-off-topic__request-action-accept
   do 2palc-student-draw sem do-action
  if 2palc-off-topic__response-prompt do 2palc-student-draw
  if 2palc-d-insert-middle kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-draw sem did-draw
  if 2palc-d-pred do 2palc-done-draw sem did-draw
  if 2palc-d-ins-element do 2palc-done-draw sem did-draw
  if 2palc-d-elem-b do 2palc-done-draw sem did-draw
  if 2palc-off-topic__where-draw do 2palc-respond-where-draw
  else do abort

g 2palc-respond-req-draw sem do-action diff agent-only
  say 2palc-off-topic__ok-ill-draw-it
  if 2palc-off-topic__do-action-accept
   do 2palc-request-pencil-draw sem request-action
  if 2palc-off-topic__do-action-reject do 2palc-student-draw sem do-action
  if 2palc-off-topic__response-prompt do 2palc-student-draw
  if 2palc-d-insert-middle kc +m5
   do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-insert-end kc +m5 do 2palc-give-feedback-draw sem done-draw
  if 2palc-d-unrecog do 2palc-done-draw sem did-draw
  if 2palc-d-pred do 2palc-done-draw sem did-draw
  if 2palc-d-ins-element do 2palc-done-draw sem did-draw
  if 2palc-d-elem-b do 2palc-done-draw sem did-draw
  if 2palc-off-topic__where-draw do 2palc-respond-where-draw
  else do abort

g 2palc-off-topic__we-should-draw
  say 2palc-off-topic__we-should-draw
  if 2palc-off-topic__go-for-it sem request-action

g 2palc-draw-ins-elem
  say 2palc-d-ins-element kc +m1 sem did-draw
  if 2palc-feedback-action-prompt sem prompt

g 2palc-draw-pred
  say 2palc-d-pred kc +g1 sem did-draw
  if 2palc-feedback-action-prompt sem prompt

g 2palc-draw-elem-b
  say 2palc-d-elem-b kc +m4 sem did-draw
  if 2palc-feedback-action-prompt sem prompt

g 2palc-draw-insert-middle sem did-draw
  say 2palc-d-insert-middle kc +m5 sem done-draw
  if 2palc-feedback-action-prompt sem prompt
  if 2palc-d__feedback-action-positive sem agree
  else

g 2palc-draw-insert-end sem did-draw
  say 2palc-d-insert-end kc +m5 sem done-draw
  if 2palc-feedback-action-prompt sem prompt
  if 2palc-d__feedback-action-positive sem agree
  else

g 2palc-draw-unrecognizable
  say 2palc-d-unrecog sem did-draw
  if 2palc-feedback-action-prompt sem prompt

g 2palc-where-to-draw
  say 2palc-off-topic__where-draw
  if 2palc-off-topic__use-the-top-drawing sem correct
  say 2palc-off-topic__response-prompt

g 2palc-which-is-pred
  say 2palc-d__which-node-is-pred
  if 2palc-i-think-that-pred-is-node-a sem correct1
  if 2palc-i-think-that-pred-is-node-b sem correct2
  if 2palc-i-think-that-pred-is-any-node-in-the-lis sem correct3

g 2palc-find-cases sem id-cases diff agent-only
  say 2palc-find-cases__so-what-do-you-think-are-the-two-cases
   sem request-info
  if 2palc-lines-6-9-are-for-when-pred-is-not-equal
   do 2palc-case-pred-not-null sem partial-correct
  if 2palc-response-dont-know do 2palc-propose-case sem prompt
  if 2palc-case-not-empty-list kc +mc2
   do 2palc-response-case-not-empty sem partial-correct
  if 2palc-find-cases__case-middle-end kc +g8
   do 2palc-response-both-cases sem correct
  if 2palc-find-cases__case-insert-middle kc +g6
   do 2palc-inquire-for-end-case sem partial-correct
  if 2palc-find-cases__case-insert-end kc +g7
   do 2palc-inquire-for-middle-case sem partial-correct
  if 2palc-off-topic__what-cases do 2palc-propose-case
   say 2palc-off-topic__respond-what-cases
  else

g 2palc-case-pred-not-null sem id-cases diff agent-only
  say 2palc-find-cases__what-are-the-two-cases-when-pred-nu sem id-cases
  if 2palc-response-dont-know do 2palc-ask-pred-null sem prompt
  if 2palc-case-not-empty-list kc +mc2
   do 2palc-response-case-not-empty sem partial-correct
  if 2palc-find-cases__case-middle-end kc +g8
   do 2palc-response-both-cases sem correct
  if 2palc-find-cases__case-insert-middle kc +g6
   do 2palc-inquire-for-end-case sem partial-correct
  if 2palc-find-cases__case-insert-end kc +g7
   do 2palc-inquire-for-middle-case sem partial-correct
  else

g 2palc-propose-case sem propose-correct1 diff agent-only
  say 2palc-find-cases__maybe-insertion-in-the-middle-of-the-lis
   kc +g6 sem id-cases
  if 2palc-response-agree-proposal
   do 2palc-inquire-for-end-case sem agree
  if 2palc-response-disagree-proposal say 2palc-i-think-its-right
   do 2palc-inquire-for-end-case sem disagree
  if 2palc-find-cases__case-insert-end do 2palc-response-both-cases
  else

g 2palc-propose-case sem propose-correct2 diff agent-only
  say 2palc-find-cases__what-about-insertion-at-the-end-of-t
   kc +g7 sem id-cases
  if 2palc-response-agree-proposal
   do 2palc-inquire-for-middle-case sem agree
  if 2palc-response-disagree-proposal
   do 2palc-inquire-for-middle-case say 2palc-i-think-its-right
  if 2palc-find-cases__case-insert-middle do 2palc-response-both-cases
  else

g 2palc-ask-pred-null diff agent-only
  say 2palc-find-cases__maybe-it-will-help-to-look-at-the-cases-
  say 2palc-what-are-the-cases-when-pred-is-null sem id-cases
  if 2palc-when-we-want-to-insert-at-the-front-of-t kc "+m2 +g4"
   do 2palc-propose-after-pred-null sem correct
  if 2palc-when-the-list-is-empty kc +g3
   do 2palc-propose-after-pred-null sem partial-correct
  if 2palc-when-we-want-to-insert-at-the-front-or-t kc "+m2 +g3 +g4"
   do 2palc-propose-after-pred-null sem correct
  else

g 2palc-propose-after-pred-null diff agent-only
  say 2palc-find-cases__for-lines-6-9
  do 2palc-propose-case

g 2palc-response-case-not-empty sem correct diff agent-only
  say 2palc-find-cases__no-i-dont-think-thats-what-they-mean kc -mc2
  do 2palc-propose-case

g 2palc-inquire-for-middle-case sem correct diff agent-only
  say 2palc-find-cases__ok-one-case-is-insert-in-the-end-but-i-c
  if 2palc-find-cases__case-insert-middle kc +g6
   do 2palc-response-both-cases sem correct
  if 2palc-response-dont-know do 2palc-find-middle-case sem prompt
  else

g 2palc-inquire-for-end-case sem correct diff agent-only
  say 2palc-find-cases__ok-one-case-is-insert-at-the-middle-but-
  if 2palc-find-cases__case-insert-end kc +g7
   do 2palc-response-both-cases sem correct
  if 2palc-response-dont-know do 2palc-find-end-case sem prompt
  else

g 2palc-response-both-cases sem agree diff agent-only
  say 2palc-find-cases__right-it-works-both-in-the-middle-and-at
   kc +g8 sem id-cases
  say 2palc-off-topic__lets-move-on
  do 2palc-write-solution-both-cases

g 2palc-find-end-case sem correct diff agent-only
  say 2palc-find-cases__how-about-insertion-at-the-end-of-the-li
  if 2palc-response-agree-proposal do 2palc-response-both-cases sem agree
  if 2palc-response-disagree-proposal say 2palc-i-think-its-right
   do 2palc-response-both-cases sem disagree
  if 2palc-response-prompt do 2palc-response-both-cases
  else do abort

g 2palc-find-middle-case sem correct diff agent-only
  say 2palc-find-cases__how-about-insertion-in-the-middle-of-the
  if 2palc-response-agree-proposal do 2palc-response-both-cases sem agree
  if 2palc-response-disagree-proposal
   say 2palc-i-think-its-right do 2palc-response-both-cases sem disagree
  if 2palc-response-prompt do 2palc-response-both-cases
  else do abort

g 2palc-respond-both-cases sem agree
  say 2palc-response-agree-proposal kc +g8 sem id-cases

g 2palc-respond-single-case
  say 2palc-response-single-case

g 2palc-propose-case-middle-end sem id-cases
  say 2palc-find-cases__case-middle-end kc +g8 sem id-cases
  if 2palc-response-agree-proposal sem agree
  else

g 2palc-propose-case-not-empty-list sem partial-correct
  say 2palc-find-cases__how-about-insertion-into-a-not-empty-lis
   kc +mc2 sem id-cases
  if 2palc-find-cases__no-i-dont-think-thats-what-they-mean sem disagree
  else
```

```
g 2palc-propose-case-insert-middle
  say 2palc-find-cases__case-insert-middle kc +g6 sem id-cases
  if 2palc-response-single-case sem agree
  else

g 2palc-propose-case-insert-end
  say 2palc-find-cases__case-insert-end kc +g7 sem id-cases
  if 2palc-response-single-case sem agree
  else


g 2palc-write-solution-both-cases sem request-action diff agent-only
  say 2palc-off-topic__you-write-explanation
  if 2palc-off-topic__request-action-accept
   do 2palc-student-expl-both-cases sem do-action
  if 2palc-off-topic__request-action-reject
   do 2palc-agent-expl-both-cases sem request-action
  else

g 2palc-student-expl-both-cases diff agent-only
  say 2palc-off-topic__go-for-it
  if 2palc-write-soln__solution-desc-only
   do 2palc-respond-add-both-cases sem solution
  if 2palc-write-soln__solution-correct kc +g8
   say 2palc-response-agree-proposal sem solution
  if 2palc-write-soln__solution-incomplete do 2palc-remedy-both-cases
  if 2palc-write-soln__solution-wrong do 2palc-rememdy-both-cases
  else

g 2palc-respond-add-both-cases diff agent-only
  say 2palc-write-soln__i-think-were-supposed-to-put-the-cases-i
  if 2palc-write-soln__solution-correct kc +g8
   say 2palc-response-agree-proposal sem correct
  if 2palc-response-disagree-proposal say 2palc-ok-i-guess

g 2palc-remedy-both-cases diff agent-only
  say 2palc-write-soln__i-think-were-supposed-to-describe-what-t
  if 2palc-write-soln__solution-correct sem correct
   say 2palc-response-agree-proposal
  if 2palc-write-soln__solution-desc-only
   do 2palc-respond-add-both-cases sem partial-correct
  if 2palc-response-disagree-proposal say 2palc-ok-i-guess
```

```
g 2palc-agent-expl-both-cases diff agent-only
  say 2palc-req-pencil
  if 2palc-rel-pencil do 2palc-write-expl-both-cases
  say 2palc-rel-pencil
  say 2palc-req-feedback-expl
  if 2palc-response-agree-proposal sem agree
  if 2palc-response-disagree-proposal say 2palc-i-think-its-right

g 2palc-write-expl-both-cases sem correct diff agent-only
  say 2palc-write-soln__solution-correct sem solution
  say 2palc-pause-10

g 2palc-sol-correct
  say 2palc-write-soln__solution-correct kc +g8 sem solution
  if 2palc-response-agree-proposal sem agree

g 2palc-sol-middle
  say 2palc-write-soln__solution-desc-middle kc +g6
  if 2palc-response-agree-proposal sem agree

g 2palc-sol-end
  say 2palc-write-soln__solution-desc-end kc +g7
  if 2palc-response-agree-proposal sem agree

g 2palc-sol-not-empty
  say 2palc-write-soln__solution-desc-not-empty
  if 2palc-response-disagree-proposal sem disagree

g 2palc-sol-incomplete
  say 2palc-write-soln__solution-incomplete
  if 2palc-req-correct-solution sem disagree

g 2palc-sol-wrong
  say 2palc-write-soln__solution-wrong
  if 2palc-req-correct-solution sem disagree

g 2palc-finish-problem diff agent-only
  say 2palc-off-topic__so-are-we-done
  if 2palc-submit-solution do logout
  if 2palc-off-topic__response-prompt do 2palc-submit-solution
  else
```

```
g 2palc-submit-solution sem request-action diff agent-only
  say 2palc-off-topic__why-dont-you-press-the-done-but
  if 2palc-off-topic__request-action-accept
   do 2palc-student-submit sem do-action
  if 2palc-submit-solution do logout
  if 2palc-off-topic__request-action-reject
    do 2palc-press-done-complete sem request-action
  else

g 2palc-student-submit diff agent-only
  say 2palc-wait__do-nothing
  if 2palc-submit-solution
  else do 2palc-press-done-complete

g 2palc-unrecognized-on-task
  say 2palc-something-unrecognized-on-task
  if 2palc-response-dont-know do "2palc-wait abort" sem prompt

g 2palc-unanticipated-on-task
  say 2palc-something-unanticipated-on-task
  if 2palc-uh-huh do "2palc-wait abort" sem prompt

g 2palc-unrecognized-off-task
  say 2palc-off-topic__unrecog
  if 2palc-off-topic__lets-talk-about-the-problem
     do "2palc-wait abort" sem prompt

g 2palc-done-yet
  say 2palc-off-topic__are-we-done
  if 2palc-off-topic__-we-need-to-come-up-with-two-ca
     do "2palc-wait abort"

g 2palc-student-request-feedback
  say 2palc-write-soln__what-do-you-think-of-my-drawingsolution
  if 2palc-response-not-sure do "2palc-wait abort"

g 2palc-clarification sem correct
  say 2palc-new-topic__unan-clarification-correct
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-clarification sem incorrect
  say 2palc-new-topic__unan-clarification-incorrect
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-question
  say 2palc-new-topic__unan-question
  if 2palc-respond-to-question__i-dont-know do "2palc-wait abort"

g 2palc-propose sem correct
  say 2palc-new-topic__unan-propose-correct
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-propose sem incorrect
  say 2palc-new-topic__unan-propose-incorrect
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-counterpropose sem correct
  say 2palc-new-topic__unan-counterpropose-correct
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-counterpropose sem incorrect
  say 2palc-new-topic__unan-counterpropose-incorrect
  if 2palc-response-disagree do "2palc-wait abort" sem disagree
  if 2palc-response-hedge do "2palc-wait abort" sem hedge
  if 2palc-response-agree do "2palc-wait abort" sem agree
  if 2palc-response-prompt do "2palc-wait abort" sem prompt

g 2palc-request-write-expl-not-ready
```

```
  say 2palc-off-topic__req-write-expl-not-ready
  if 2palc-off-topic_finish-problem

g 2palc-request-write-expl-ready
  say 2palc-off-topic__req-write-expl-ready
  if 2palc-ok-why-dont-you-try
  say 2palc-write-soln__solution-correct sem solution
  if 2palc-response-agree-proposal

g 2palc-procedure-question
  say 2palc-off-topic__procedure-question
  if 2palc-off-topic__read-prob-description

g 2palc-press-done-complete
  say 2palc-submit-solution
  do logout

g 2palc-press-done-notcomplete
  say 2palc-submit-solution-notdone
  if 2palc-not-done-yet
```

CITED LITERATURE

AA. VV.: Computer Science, Final Report, The Joint Task Force on Computing Curricula. IEEE Computer Society and Association for Computing Machinery, IEEE Computer Society., 2001.

AA. VV.: US bureau of labor statistics http://www.bls.gov/oco/oco20016.htm. 2006.

AA. VV.: Chat slang dictionary http://www.pulpchat.com. 2007.

AA. VV.: Chat slang http://www.computerhope.com. 2007.

Baker, M., Hansen, T., Joiner, R., and Traum, D.: The role of grounding in collaborative learning tasks. Collaborative learning: Cognitive and computational approaches, pages 31–63, 1999.

Baker, M. and Lund, K.: Promoting reflective interactions in a CSCL environment. Journal of Computer Assisted Learning, 13(3):175–193, 1997.

Barron, B.: Problem solving in video-based microworlds: Collaborative and individual outcomes of high-achieving sixth-grade students. Journal of Educational Psychology, 92(2):391–398, June 2000.

Ben-Ari, M.: Constructivism in computer science education. SIGCSE Bull., 30(1):257–261, 1998.

Bhatt, K., Evens, M., and Argamon, S.: Hedged responses and expressions of affect in human/human and human computer tutorial interactions. In Proceedings Cognitive Science, 2004.

Birtz, M. W., Dixon, J., and McLaughlin, T. F.: The effects of peer tutoring on mathematics performance: A recent review. B. C. Journal of Special Education, 13(1):17–33, 1989.

Brown, A. L. and Palincsar, A. S.: Guided, cooperative learning and individual knowledge acquisition, pages 307–226. Hillsdale, NJ, Lawrence Erlbaum Associates, 1989.

Burton, M., Brna, P., and Pilkington, R.: Clarissa: a laboratory for the modelling of collaboration. International Journal of Artificial Intelligence in Education, 11, 2000.

Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Computational Linguistics, 22(2):249–254, 1996.

Chan, T.-W. and Baskin, A.: Studying with the prince. In Proceedings of the ITS-88 Conference, pages 194–200, 1988.

Chi, M., De Leeuw, N., Chiu, M., and LaVancher, C.: Eliciting self-explanations improves understanding. Cognitive Science, 18(3):439–477, 1994.

Chou, C.-Y., Chan, T.-W., and Lin, C.-J.: Redefining the learning companion: the past, present, and future of educational agents. Computers & Education, 40(3):255–269, 2003.

Chou, C.-Y., Lin, C.-J., and Chan, T.-W.: An approach to developing computational supports for reciprocal tutoring. Knowledge-Based Systems, 15(7):407–412, September 2002.

Chu-Carroll, J. and Brown, M. K.: An evidential model for tracking initiative in collaborative dialogue interactions. User Modeling and User-Adapted Interaction, 8(3–4):215–253, September 1998.

Chu-Carroll, J. and Carberry, S.: Collaborative response generation in planning dialogues. Computational Linguistics, 24(3):355–400, 1998.

Cleary, J. G. and Trigg, L. E.: K*: An instance-based learner using an entropic distance measure. In Proceedings of the 12th International Conference on Machine Learning, pages 108–114, 1995.

Cohen, P., Kulik, J., and Kulik, C.: Education outcomes of tutoring: A meta-analysis of findings. American Education Research Journal, 19(2):237–248, 1982.

Cohen, W. W.: Fast effective rule induction. In Machine Learning: Proceedings of the Twelve International Conference, 1995.

Conati, C., Gertner, A., and VanLehn, K.: Using Bayesian networks to manage uncertainty in student modeling. User Modeling and User-Adapted Interaction, 12(4):371–417, 2002.

Constantino-González, M. and Suthers, D. D.: A coached collaborative learning environment for entity-relationship modeling. Intelligent Tutoring Systems, pages 324–333, 2000.

Corbett, A. T. and Anderson, J. R.: The effect of feedback control on learning to program with the LISP tutor. In Proceedings of the Twelfth Annual Conference of the Cognitive Science Society, pages 796–803, 1990.

Core, M. G., Moore, J. D., and Zinn, C.: The role of initiative in tutorial dialogue. In EACL '03: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, pages 67–74, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

Di Eugenio, B., Jordan, P., Thomason, R., and Moore, J.: The agreement process: An empirical investigation of human-human computer-mediated collaborative dialogues. International Journal of Human-Computer Studies, 53(6):1017–1076, 2000.

Dillenbourg, P. and Traum, D.: The long road from a shared screen to a shared understanding. In Proceedings of the 3rd conference on computer supported collaborative learning, Stanford, 1999.

Evens, M. W., Chang, R.-C., Lee, Y. H., Shim, L. S., Woo, C. W., Zhang, Y., Michael, J. A., and Rovick, A. A.: CIRCSIM-Tutor: an intelligent tutoring system using natural language dialogue. In Proceedings of the fifth conference on Applied natural language processing: Descriptions of system demonstrations and videos, pages 13–14, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

Fisher, E.: Distinctive features of pupil-pupil classroom talk and their relationship to learning: How discursive exploration might be encouraged. Language and Education, 7:239–257, 1993.

Forbus, K., Usher, J., Lovett, A., Lockwood, K., and Wetzel, J.: Cogsketch: Open-domain sketch understanding for cognitive science research and for education. In Proceedings of the Eurographics Workshop on Sketch-Based Interfaces and Modeling, 2008.

Goodman, B. A., Linton, F. N., Gaimari, R. D., Hitzeman, J. M., Ross, H. J., and Zarrella, G.: Using dialogue features to predict trouble during collaborative learning. User Modeling and User-Adapted Interaction, 15(1):85–134, March 2005.

Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A., and Louwerse, M. M.: Autotutor: A tutor with dialogue in natural language. Behavior Research Methods, Instruments, & Computers, 36:180–192(13), May 2004.

Guinn, C. I.: An analysis of initiative selection in collaborative task-oriented discourse. User Modeling and User-Adapted Interaction, 8(3-4):255–314, 1998.

Harrer, A., McLaren, B. M., Walker, E., Bollen, L., and Sewall, J.: Creating cognitive tutors for collaborative learning: steps toward realization. User Modeling and User-Adapted Interaction, 16(3-4):175–209, 2006.

Hausmann, R. G.: Elaborative and Critical Dialog: Two Potentially Effective Problem-Solving and Learning Interactions. Doctoral dissertation, University of Pittsburgh, 2005.

Hausmann, R. G., Chi, M. T., and Roy, M.: Learning from collaborative problem solving: An analysis of three hypothesized mechanisms. In 26th Annual Conference of the Cognitive Science Society, eds. K. Forbus, D. Gentner, and T. Regier, pages 547–552, Mahwah, NJ, 2004.

Johnson, D. W. and Johnson, R. T.: Learning Together and Alone. Allyn and Bacon, 1999.

Jordan, P. and Siler, S.: Student initiative and questioning strategies in computer-mediated human tutoring dialogues. In Proceedings of ITS 2002 Workshop on Empirical Methods for Tutorial Dialogue Systems, 2002.

Jordan, P. W. and Di Eugenio, B.: Control and initiative in collaborative problem solving dialogues. In Working Notes of the AAAI Spring Symposium on Computational Models for Mixed Initiative, pages 81–84, Menlo Park, CA, 1997.

Jordan, P. W., Hall, B., Ringenberg, M. A., Cue, Y., and Rosé, C. P.: Tools for authoring a dialogue agent that participates in learning studies. In Artificial Intelligence in Education, AIED 2007, pages 43–50, 2007.

Katz, S., Aronis, J., Allbritton, D., Wilson, C., and Soffa, M.: Gender and race in predicting achievement in computer science. Technology and Society Magazine, IEEE, 22(3):20–27, 2003.

Lane, H. C. and VanLehn, K.: Coached program planning: dialogue-based support for novice program design. SIGCSE Bull., 35(1):148–152, 2003.

Lochbaum, K. E. and Sidner, C. L.: Models of plans to support communication: An initial report. In Proceedings of the Eighth National Conference on Artificial Intelligence, pages 485–490. AAAI Press, 1990.

Mitrović, A., Suraweera, P., Martin, B., and Weerasinghe, A.: DB-Suite: Experiences with Three Intelligent, Web-Based Database Tutors. Journal of Interactive Learning Research, 15(4):409–433, 2004.

Ploetzner, R., Dillenbourg, P., Preier, M., and Traum, D.: Learning by explaining to oneself and to others. Collaborative learning: Cognitive and computational approaches, pages 103–121, 1999.

Rekrut, M. D.: Teaching to learn: Cross-age tutoring to enhance strategy instruction. American Education Research Association, 1992.

Resnick, L. B. (Ed.): Knowing, Learning, and Instruction: Essays in Honor of Rober Glaser. Hillsdale, NJ, Erlbaum, 1989.

Roscoe, R. D. and Chi, M. T. H.: Understanding tutor learning: Knowledge-building and knowledge-telling in peer tutors' explanations and questions. Review of Educational Research, 77(4):534–574, 2007.

Sidner, C. L.: An artificial discourse language for collaborative negotiation. In AAAI '94: Proceedings of the twelfth national conference on artificial intelligence (vol. 1), pages 814–819, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.

Soller, A.: Computational modeling and analysis of knowledge sharing in collaborative distance learning. User Modeling and User-Adapted Interaction, Volume 14(4):351–381, January 2004.

Soller, A.: Understanding knowledge-sharing breakdowns: A meeting of the quantitative and qualitative minds. Journal of Computer Assisted Learning,, 20(3):212–223, June 2004.

Soller, A. and Lesgold, A.: A computational approach to analyzing online knowledge sharing interaction. In Proceedings of Artificial Intelligence in Education, Sydney, Australia, pages 253–260, 2003.

Strayer, S. E. and Heeman, P. A.: Reconciling initiative and discourse structure. In Proceedings of the Second SIGdial Workshop on Discourse and Dialogue, pages 1–9, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

Tin, T. B.: Does talking with peers help learning? the role of expertise and talk in convergent group discussion tasks. Journal of English for Academic Purposes, 2(1):53–66, 2003.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pages 173–180, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

Uresti, J. A. R.: Should I teach my computer peer? some issues in teaching a learning companion. In Proceedings of the 5th International Conference on Intelligent Tutoring Systems, pages 103–112, London, UK, 2000. Springer-Verlag.

VanLehn, K., Jordan, P. W., Rosé, C. P., Bhembe, D., Böttner, M., Gaydos, A., Makatchev, M., Pappuswamy, U., Ringenberg, M. A., Roque, A., Siler, S., and Srivastava, R.: The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In ITS '02: Proceedings of the 6th International Conference on Intelligent Tutoring Systems, pages 158–167, London, UK, 2002. Springer-Verlag.

Vizcaíno, A.: A simulated student can improve collaborative learning. International Journal of Artificial Intelligence in Education, 15(1):3–40, 2005.

von Mayrhauser, A. and Vans, A. M.: Program comprehension during software maintenance and evolution. Computer, 28(8):44–55, 1995.

Walker, M. and Whittaker, S.: Mixed initiative in dialogue: an investigation into discourse segmentation. In Proceedings of the 28th annual meeting on Association for Computational Linguistics, pages 70–78, Morristown, NJ, USA, 1990. Association for Computational Linguistics.

Warendorf, K. and Tan, C.: ADIS-an animated data structure intelligent tutoring system or putting an interactive tutor on the WWW. In Intelligent Educational Systems on the World Wide Web (Workshop Proceedings), at the Eight International Conference on Artficial Intellignece in Education, 1997.

Whittaker, S. and Stenton, P.: Cues and control in expert-client dialogues. In Proceedings of the 26th annual meeting on Association for Computational Linguistics, pages 123–130, Morristown, NJ, USA, 1988. Association for Computational Linguistics.

Witten, I. H. and Frank, E.: Data Mining: Practical machine learning tools and techniques. San Francisco, Morgan Kaufmann, 2005.

VITA

NAME:	Cynthia Kersey
EDUCATION:	Ph.D., Computer Science University of Illinois at Chicago, 2009
	M.S., Computer Science Governors State University, 2001
	B.B.A., Computer Science University of Illinois at Chicago, 1985
WORK EXPERIENCE:	Research Assistant Computer Science Department University of Illinois at Chicago, 1/2006–Present
	Computer Science Lecturer Governors State University, 8/2001–Present
RESEARCH GRANTS:	National Science Foundation, 2006 ITR-Broadening Participation. Collaborative and Integrative Environment for Computer Science Programs, Co-PI, $238,628
PUBLICATIONS:	Cynthia Kersey, Barbara Di Eugenio, Pamela Jordan, and Sandra Katz. Knowledge Co-construction and Initiative in Peer Learning Interactions . AIED 2009, The 14th International Conference on Artificial Intelligence in Education. Brighton, UK. July 2009.
	Cynthia Kersey, Barbara Di Eugenio, Pamela Jordan, and Sandra Katz. KSC-PaL: A Peer Learning Agent that Encourages Students to take the Initiative . NAACL-HLT 2009 Workshops, The 4th Workshop on Innovative Use of NLP for Building Educational Applications. Boulder, Co. June 2009.

Cynthia Kersey. Impact of Initiative on Collaborative Problem Solving. ACL 2008, The 46th Annual Meeting of the Association for Computational Linguistics, Student Research Workshop. Columbus, OH. 2008.

Cynthia Kersey, Barbara Di Eugenio, Pamela Jordan, and Sandra Katz. Modeling Knowledge Co-Construction for Peer Learning Interactions. ITS 2008, The 9th International Conference on Intelligent Tutoring Systems, Student Research Workshop. Montreal, Canada. 2008.

Cynthia Kersey, Barbara Di Eugenio, Pamela Jordan, and Sandra Katz. Collaboration in Peer Learning Dialogues . Decalog: The 2007 Workshop on the Semantics and Pragmatics of Dialogue, Trento, Italy, May 2007 (Poster).

Cynthia Kersey, Zhenwei Yu, and Jeffrey J.P. Tsai. Intrusion Detection for Wireless Network. Wireless Ad Hoc Networking. Eds. Shin-Lin Wu and Yu-Chee Tseng. Aurberach Publications. 2007. 505-533.