ABSTRACT

LEE, SUNYOUNG. Early Prediction of Student Goals and Affect in Narrative-Centered Learning Environments. (Under the direction of James C. Lester and Carla D. Savage.)

Recent years have seen a growing recognition of the role of goal and affect recognition in intelligent tutoring systems. Goal recognition is the task of inferring users' goals from a sequence of observations of their actions. Because of the uncertainty inherent in every facet of human computer interaction, goal recognition is challenging, particularly in contexts in which users can perform many actions in any order, as is the case with intelligent tutoring systems. Affect recognition is the task of identifying the emotional state of a user from a variety of physical cues, which are produced in response to affective changes in the individual. Accurately recognizing student goals and affect states could contribute to more effective and motivating interactions in intelligent tutoring systems. By exploiting knowledge of student goals and affect states, intelligent tutoring systems can dynamically modify their behavior to better support individual students.

To create effective interactions in intelligent tutoring systems, goal and affect recognition models should satisfy two key requirements. First, because incorrectly predicted goals and affect states could significantly diminish the effectiveness of interactive systems, goal and affect recognition models should provide accurate predictions of user goals and affect states. When observations of users' activities become available, recognizers should make accurate "early" predictions. Second, goal and affect recognition models should be highly efficient so they can operate in real time.

To address key issues, we present an inductive approach to recognizing student goals and affect states in intelligent tutoring systems by learning goals and affect recognition models. Our work focuses on goal and affect recognition in an important new class of intelligent tutoring systems, narrative-centered learning environments. We report the results of empirical studies of induced recognition models from observations of students' interactions in narrative-centered learning environments. Experimental results suggest that induced models can make accurate early predictions of student goals and affect states, and they are

sufficiently efficient to meet the real-time performance requirements of interactive learning environments.

Early Prediction of Student Goals and Affect in Narrative-Centered Learning Environments

by
Sunyoung Lee

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Science

Raleigh, North Carolina

August 2008

APPROVED BY:

Dedication

To my parents and grandmother

Biography

Sunyoung Lee was born and grew up in Seoul, South Korea where she attended the local elementary school, middle school, and high school. After graduating from high school, she attended Ewha Womans University and majored in Computer Science. After graduating in 1996 from Ewha Womans University in Seoul, she began her studies at the Graduate School of Pohang University of Science and Technology (POSTECH). After completing graduate school, she worked at Samsung SDS, one of the best IT companies in Korea, and ETRI, a top research center. She was involved in two projects in Samsung SDS, both of which were ERP consulting projects using SAP R/3. At ETRI, she analyzed anti-cyberterror technology, with an emphasis on computer viruses.

In 2003, she began her Ph.D. study in Computer Science at North Carolina State University. During her Ph.D. years, she worked as a Research Assistant in the area of intelligent learning environments under the direction of James Lester and in the area of Combinatorics under the direction of Carla Savage. For the past five years, she conducted research on the goals and affect recognition models for narrative-centered learning environments.

Acknowledgments

I would first thank to my co-advisors, James Lester and Carla Savage who have given me enormous support during my studies. James Lester has been a great advisor and friend throughout these years of collaboration. He gave me guidance to steer my research in the right direction and taught me how to be a researcher. Carla Savage has been a great mentor and friend during the past five years. She was always there to encourage me when I needed it and she has had a great influence on my life. I could not have accomplished this dissertation without them. I would also like to thank my thesis committee members, John Nietfeld, Munindar Singh, and Michael Young for their comments and suggestions.

Living in Raleigh, I have met great people. I was fortunate to study with and work with colleagues in the Intellimedia group at NCSU including Kristy Boyer, Cohan Carlos, Julius Goth, Eunyoung Ha, Seung Lee, Scott McQuiggan, Bradford Mott, Rob Phillips, Jonathan Rowe, and Michael Wallis. I am deeply indebted to my best friends, Kyoungeun Ahn, Matthew Nelson, and Youngjin Park for their encouragement and love. I would especially like to thank Matthew Nelson. He was always there to motivate me and gave me advice when I needed it.

Finally and most importantly, I would like to thank my parents, sisters, brother, and brother-in-law. Without them, I may have never finished. They have always supported me and encouraged me. I can never express how much I love them and how much I thank them for their endless love. To my grandmother, I love you and hope you find peace in heaven.

This research was supported by the National Science Foundation under Grants REC-0632450. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

It is widely believed that customizing users' interactions can significantly improve their effectiveness in performing tasks. User-adaptability can serve a broad range of functions in intelligent systems. For example, interactive applications can adapt their user interfaces to better suit individual users. They can help users focus their information-seeking tasks, provide information that takes into account users' interests in specific topics, and present information in a manner that is appropriate for users' visualization preferences (Jameson 2003). To create effective customized interactions, interactive systems need to reason about users' goals and plans. Goal recognition is the task of inferring a user's goals from a sequence of observations. By providing knowledge about users' goals, interactive systems can dynamically modify their behavior to better support individual users. Goal recognition has been widely used in a broad range of applications. Applications include natural language understanding (Carberry 1990a; Carberry 1990b), story understanding (Charniak and Goldman 1993), intelligent tutoring systems (Conati and VanLehn 1996; Conati *et al*. 2002; Mott *et al*. 2006a), adventure games (Albrecht *et al*., 1998), intrusion detections (Geib and Goldman, 2005), and multi-agent coordination (Huber *et al*. 1994).

Many goal recognition approaches have been proposed over the past thirty years. All of these approaches hypothesize goals that are consistent with a sequence of observations and then narrow down the hypotheses by using either logical methods or probabilistic inference. The source of information used by these approaches is a sequence of observations such as user actions and the states of the system with which the users are interacting.

Affective computing has emerged as a promising line of investigation in its own right. Affective computing is an active area of research that explores techniques for developing computational models of affect recognition and affect generation. Affect recognition is the task of identifying a user's affective state, and affect generation is the task of determining the most appropriate affective state that an interactive system should exhibit. Affect recognition can be cast as a pattern recognition problem and affect generation as a pattern synthesis problem (Picard 1997). Affect recognition can be used to customize interactions based on users' affective states, and affect generation can play an important role in creating synthetic agents for interactive environments.

In this thesis, we present inductive approaches to recognizing student goals and affect in an important new class of intelligent tutoring systems, narrative-centered learning environments. We propose a probabilistic framework for goals and affect recognition in which recognition models are induced from training corpora acquired from user interaction histories that include heart rate and galvanic skin response data streams.

1.1 Motivation

Recognizing users' goals and plans plays a central role in a variety of applications. Such applications range from software assistants and robotics to security and collaborative filtering. Four important categories of applications related to software assistants are adaptive systems, multi-agent systems, natural language processing, and intelligent tutoring systems. We briefly discuss each of these in turn.

- *User-adaptive systems*: User-adaptive systems depend on the information about users' goals, plans, beliefs, and preferences (Jameson 2003). For example, LUMIÈRE (Horvitz *et al*., 1998) and READY (Bohnenberger *et al*. 2002; Jameson *et al*. 1999) provide automated assistance by drawing inferences about users' current goals and plans. LUMIÈRE reasons about the user's task history, actions, and the set of documents she is accessing to determine whether she needs assistance. READY considers resource limitations such as time pressure and cognitive load to infer users' preferences. If the user requires information immediately, the system can infer her degree of time pressure. If time pressure is higher, information is "bundled" and presented concisely; if time pressure is lower, information is presented sequentially over time. PRIORITIES (Horvitz *et al.* 1999) is a user-adaptive system that transmits alerts when email arrives by considering the criticality of email and the user's focus of attention. LOOKOUT (Horvitz 1999) is a system for scheduling and meeting management on Microsoft Outlook. When the user opens a new email message, the system infers the user's goal (i.e., the user's goal for working with the scheduling system) and the system invokes the scheduling system based on the inference. More recently, adaptive systems have begun to explore how to take user affective states into account to create more effective interactive experiences. For example, SPECTER (Kleinbauer *et al*. 2003) is a personal assistant that senses the user's action and affective states based on data from a ubiquitous computing environment to provide personalized recommendations that are the most appropriate for the current context.

- *Multi-agent systems*: Plan recognition is also an important issue for the multi-agent systems community because agents can sometimes perform more effectively if they can take into account the plans of another agent or group of agents. To correctly predict other agents' behaviors, inferring other agents' goals and plans is an important task. The task is challenging in a cooperative setting in which agents are collaborating, and is even more difficult in a competitive setting, such as the RoboCup soccer competitions (Kaminka *et al*., 2002).

- *Natural language processing*: Natural language processing is one of the most active areas of plan recognition research. Many natural language applications have utilized plan recognition in the area of story understanding (Charniak and Goldman 1991; Charniak and Goldman 1993), natural language dialogue (Carberry 1988; Carberry 1990a; Carberry 1990b), and machine translation (Alexandersson 1995). More recently, there have been efforts to exploit affect to disambiguate dialogue acts (Bosma and André 2004). Bosma and André consider emotional inputs such as arousal and valence to estimate the probabilities of dialogue acts. The estimated probabilities are used to determine which interpretation is more likely to be correct.

- *Intelligent tutoring systems*: Intelligent tutoring systems (ITSs) comprise an important category of user-adaptive systems. ITSs employ student models, which are used to provide customized hints, generate explanations tailored to individual users, and select problems. Student modeling research has investigated a broad spectrum of issues in supporting customized pedagogical decisions. A central problem of student modeling is plan recognition, in which a system attempts to recognize students' intentions, their focus of attention, their current plan, and their current goal as they solve problems.

An important form of learning supported by intelligent tutoring systems is exploratory learning, in which students have great freedom to explore the physical environment and the problem-solving space (de Jong and van Joolingen 1998). However, this freedom poses a significant challenge to student modeling in general and plan recognition in particular. Goal and plan tracking are especially difficult when students have access to multiple techniques for solving a problem and when solution steps can have multiple (correct) orderings. When multiple goals and plans come into play, the student model must determine which is most likely given the current context.

Assessing the affective state of students is also an important aspect of student modeling. With information about the student's affective state, ITSs can interact more effectively with the student. Detecting frustration can enable ITSs to make intervention decisions. Knowledge of affective states may lead to more accurate and earlier detection of student difficulties. Pedagogical planners can take advantage of recognized goals, plans, and affective states to formulate appropriate tutorial strategy selection.

1.2 Problem Description

Plan recognition can be informally described as follows: Given a sequence of observations, plan recognition seeks to infer the plans that an agent is attempting to execute. Goal recognition is a special case of plan recognition. Goal recognition focuses on inferring a user's goals (Blaylock and Allen 2003), i.e., the specific objectives that the user is attempting to achieve. The solution to the goal recognition problem is a set of candidate goals that the user is attempting to achieve. The candidate goals should be consistent with the set of observed evidence. Thus, the inputs for plan and goal recognition are typically user actions, user locations, and the states of the application or interactive world. The solution to a plan recognition problem is a set of candidate plans that best explain the observed evidence. The candidate plans can be used to predict the user's future actions or goals.

Picard defines affect recognition as follows: "Affect recognition is inferring a user's emotional states from observations of emotional expressions and behavior, and through reasoning about an emotion-generation situation" (Picard 1997). In other words, because emotions are internal to the observed users, the recognizer has to infer the underlying affective states based on observable indicators of the affective states. For example, we can infer someone's emotional states from her gestures or the way she speaks. We can "wire" a person to measure her heart rates and skin conductivity, which can be indicators of her affective states.

1.3 Challenges

Plan recognition and affect recognition are difficult tasks. For example, drawing inferences about users' goals and plans inherently involves reasoning under uncertainty because users can follow multiple correct and incorrect paths to achieve their goals. Much useful information about the user is not directly observable, as users frequently do not or cannot explicitly express their goals and knowledge. In general, plan recognizers and affect recognizers should address the following requirements:

- *Scalability and Speed*: Plan recognition and goal recognition involve narrowing down possible hypotheses that best explain observed action sequences. Classic approaches to plan recognition employ plan libraries that represent goals, plans, and actions and use logic-based methods to exclude plans that are inconsistent with the observed actions. However, runtime performance is exponential in the number of goals and plans in the plan hierarchy. Probabilistic approaches to plan recognition also suffer from scalability issues. For example, Dempster-Shafer Theory (Bauer 1996a; Wilson 2000) is known to be exponential in general and exact inference in Bayesian networks is an NP-hard problem. Because most interactive applications require plan recognition results before the user completes all actions to accomplish a task or affect recognition results while a user is engaged in a task, ideally recognition should be performed quickly (on the order of a few milliseconds) to appropriately serve the user.

- *Early Prediction*: For interactive environments, goal recognizers and plan recognizers should make early accurate predictions and they should converge as quickly as possible on the most likely explanation (Albrecht *et al*. 1998; Blaylock and Allen 2003; Lee *et al*. 2007; McQuiggan *et al*. 2007; Mott *et al*. 2006a; Yin *et al*. 2004). Even if their computation is fast enough to address scalability issues, if they are not able to predict plans until they observe the final action of the user, they would be impractical for many interactive environments.
- *Partial Observability and Noisy Environments*: Users are not required (nor expected) to express their intentions or emotions, so goal and affect recognizers must infer users' intentions and affective states from observed activity. Furthermore, in some domains, users hide their intentions or emotions. For example, in intrusion detection, a malicious user could perform irrelevant actions to deceive the observing system, which could result in incorrect prediction of goals. Recently, there have been efforts to model partially observable domain. For example, Geib & Goldman propose a probabilistic approach to plan recognition in a partially observable domain (Geib and Goldman 2005), but the inference time for all examples with more than 12 observations exceeds 100 seconds. Handling partial observable domains can significantly increase the run-time cost of plan recognition.

Thus, an effective solution to the plan recognition problem must cope with the uncertainty, real-time requirements, scalability issues, and absence of direct access to the user's cognitive state.

1.4 Contributions

This dissertation reports on the following contributions:

- *Recognizing goals and affects for narrative-centered learning environments*: We propose a probabilistic goal recognition and affect recognition framework and introduce probabilistic recognition models such as *n*-grams, Bayesian networks, naïve Bayes and decision trees. The models exploit knowledge of students' actions, locations, temporal information, task structure, and physiological changes in physiological signals such as heart rate, skin conductivity. The models are used to predict students' goals and affective states early and accurately from a sequence of observations.
- *Inductive approach for creating goals and affect recognition models*: We devise an inductive approach to goal and affect recognition that learns models from training data. The framework learns goal and affect recognition models from action observation corpora augmented with physiological data. We adopt machine learning techniques to devise fast and efficient goals and affect recognizer. Machine learning techniques require a corpus, but in many domains, labeled training data are difficult to obtain. To gather corpora for goal recognition, we introduce a corpus tool which has a goal generator that generates goals that are expected to be achieved by training subjects. The corpus tool also asks training subjects to select the emotion, from a set of six emotions (excitement, fear, frustration, happiness, relaxation, and sadness), that is most closely related to their own feelings to gather labeled corpus for affect recognition.
- *Empirical studies*: We conduct empirical evaluations of probabilistic goal and affect recognition models in an implemented narrative-centered learning environment. Empirical results suggest that the induced models can make accurate early predictions of student goals and affective states as observations of students' activities become available over time. The models incrementally converge on correct interpretations. The induced goal and affect recognition models are efficient and meet the real-time requirements of interactive systems.

1.5 Dissertation Organization

This dissertation is organized as follows. Chapter 2 provides background on the problems studied by reviewing related work on plan recognition and affective reasoning. Chapter 3 introduces an inductive approach to automatically learning goals and affect recognition models from training data acquired from traces of users' activities in a narrative-centered learning environment. Chapter 4 reports the results of an empirical study in which *n*-gram models and Bayesian network goal recognition models were induced from observations of users' interactions in a narrative-centered learning environment. In Chapter 5 the empirical evaluation results of probabilistic affect recognition models are presented. In Chapter 6, we conclude by presenting a summary of the main contributions of the work. We also discuss limitations of the proposed approach to goal and affect recognition and explore promising directions for future work.

Chapter 2

Background

Plan recognition and affective computing have been the subject of increasing attention in recent years. Section 2.1 introduces user modeling, which is an important application area of both plan recognition and affect recognition. Section 2.2 provides background on goal and plan recognition. Finally, Section 2.3 discusses affective computing with an emphasis on affect recognition.

2.1 User Modeling

User modeling has emerged as an active area of research to provide the knowledge about individual users necessary for creating effective customized interactions. The task of user modeling is to dynamically model the goals, plans, beliefs, and preferences of users. Systems that employ user models can therefore provide customized assistance that is tailored to the specific needs of individual users in specific contexts. User models are employed by a broad range of applications such as proactively offering advice in mixed-initiative interfaces for problem solving and learning, facilitating natural language understanding, retrieving information, and making recommendations (Jameson 2003).

User modeling can be decomposed into three problems: representation, inference, and learning. First, user modeling requires a formalism for representing the goals, plans, beliefs, and preferences of users. A number of approaches have been taken to this problem: some are

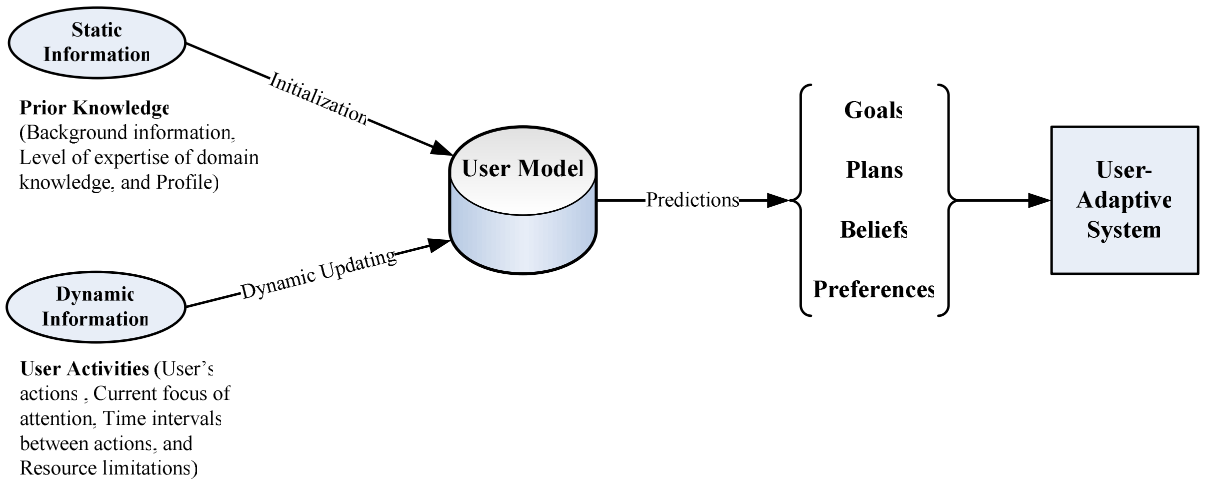

Static Information
Prior Knowledge (Background information, Level of expertise of domain knowledge, and Profile)
Initialization
Dynamic Information
User Activities (User's actions , Current focus of attention, Time intervals between actions, and Resource limitations)
Dynamic Updating
User Model
Predictions
Goals
Plans
Beliefs
Preferences
User-Adaptive System

Figure 2.1: Tasks of User Modeling

based on overlays of declarative representations, while others are plan-based. Second, the primary goal of user modeling is to dynamically update a user model for a user by drawing inferences about her behaviors at runtime. For example, when a student performs a series of (possibly incorrect) problem-solving actions, the student model should be updated to reflect the new knowledge about the student's understanding of the domain. The dynamically updated information in the user model can be used to predict the goals the user is attempting to achieve and the plans the user is attempting to execute to achieve the goals.

Goal and plan recognition are particularly challenging when there are different techniques for solving a problem and when solution steps can have different orderings. In complex situations where multiple goals and plans come into play, the user model must determine which is most likely given the current context. Third, to address scalability issues, it is desirable to adopt a formalism that can be learned. If user model acquisition can be partially automated, it will open the way towards a much broader range of user-adaptive applications.

User modeling occurs in two phases: (1) *initialization*, which uses static information about the user to "seed" the user model (initialization occurs prior to an interactive session), and (2) *dynamic updating*, which uses dynamic information to modify the user model so that it accurately reflects the changing state of the user's goals, plans, beliefs, and preferences.

Dynamic updating occurs on an ongoing basis throughout an interactive session (Figure 2.1). The resulting user model is used by a user-adaptive system to customize its interactions.

When a user first begins interacting with a system, the system has no knowledge about that particular user. Because users differ along many dimensions, a user model should be initialized for the user model, an operation which should be performed as efficiently as possible. Several approaches have been taken to user model initialization (Tsiriga and Virvou 2004). One approach is merely to assume that users have no domain knowledge or that all users have the same (pre-specified) level of competence at the beginning. Although this approach is straightforward and easy to implement, it may not be appropriate to ignore the differences of prior knowledge across users. Another simple way to initialize user models is to make the blanket assumption that the user model has no initial knowledge about users. In other words, the system does not assume anything about users. A third way to initialize a user model is to evaluate a user's prior knowledge by conducting a pre-test or to administer a questionnaire. Similarly, when a user begins interacting with the system, she can be asked to respond to a questionnaire where the system can assess her knowledge level or she can specify her preferences. However, this approach may not be applicable for broader domains because it may require excessive time for users to complete long questionnaires.

Once a user model is initialized, it must be updated dynamically since the user's plans, goals, and beliefs change over time. Dynamic updating to the user model may consider a variety of information about the user and her activities. For example, dynamic updates may consider information about the user's problem-solving actions, how many correct or incorrect problem-solving steps the student has performed, the physical environment (e.g., a user in a noisy environment may prefer text to audio), and her current focus of attention as indicated by the clickstream, mouse movement, or in more sophisticated multimodal environments, eye gaze direction. Temporal information may be especially useful. For example, extended time intervals between problem-solving actions may suggest that the user would benefit from a hint.

The dynamically updated information in the user model can be used to predict several properties about the user. First, they can be used to predict the goals the user is attempting to achieve and the plans the user is attempting to execute to achieve the goals. Second, user models can represent users' knowledge of the domain and of their current task. Conceptually, it can represent whether the user believes particular propositions are true, possible, impossible, or false. For example, it can represent what options or choices the user believes are possible for completing her current task. Third, user models can also represent users' preferences, e.g., presentation modality preferences for text or speech.

2.1.1 Classic User Modeling

Since the inception of the field, work on user modeling has explored a number of approaches to representing a system's beliefs about the user. Before turning to probabilistic approaches, we first introduce "classic" (non-probabilistic) frameworks for user modeling, which will be used for points of comparison in our discussion. There are four classic user modeling frameworks: overlays, stereotypes, bug libraries, and model tracing. Each of these frameworks provides the key functionality of user models: they enable systems to adapt to individual users. Each user has different characteristics, seeks to achieve different goals, and acts differently. If a user modeling system has insufficient information about the general behavior of users or an inadequate representation of users, it will not be able to support user adaptation. Therefore, the user model needs to be modified to characterize the user in order to provide personalized information about the user to user-adaptive systems. As we have noted, this is often referred to as dynamic updating of the user model. In addition to dynamic updating, it would also be beneficial to automatically induce "extensions" to the user model. Although there is no clear distinction between dynamic updating and learning, the emphasis in learning is on enabling the user model to generalize its representation (and its inference abilities) to apply to either the as-of-yet unseen behaviors of the current user or to unseen users. For each classical user modeling framework, we discuss its approach to representation, inference, and learning.

2.1.1.1 Overlay User Modeling

Overlay modeling is the simplest user modeling technique. It has been used in ITSs such as SCHOLAR (Carbonell 1970) and ELM-ART (Brusilovsky *et al*. 1996). A student's knowledge state is modeled implicitly by assuming it is a subset of (i.e., an "overlay" on) the expert's knowledge. When the student's actions indicate she has learned a concept, the student model is updated to represent the fact that she understands that concept. Overlay modeling makes the assumption that the knowledge that is in the expert model but not in the student model includes the concepts that the student has not explored, and the unexplored concepts are used to determine which topics to present next. To illustrate, the first intelligent tutoring system, SCHOLAR (Carbonell 1970; Wenger 1987), used a semantic network to represent the student model. SCHOLAR constructed the complete semantic network to model the perfect student, and it represented the student's knowledge by deleting and modifying nodes and links. To automatically extend overlay models, new concepts could be added. For example, the automatic construction of concept hierarchies in overlay models has been proposed for adaptive hypermedia (Njike *et al*. 2005). In the model, relationships of concepts are automatically discovered from corpora of hypermedia documents. Although overlays offer the advantage of simplicity, it is not possible to represent a student's misconceptions with overlay models, and they cannot represent the degree of confidence of its knowledge about the student.

2.1.1.2 Stereotype User Modeling

A frequently employed user modeling approach is that of stereotypes (Rich 1979). Stereotypes represent a set of common characteristics for a particular group of users in the application domain. In the stereotype approach, a system activates a stereotype which best fits a user when it observes certain events that serve as "triggers." For example, if the system observes the user employing an advanced functionality, then the "expert" stereotype can be activated. If the preconditions of the stereotype are satisfied, the stereotype can be applied to that user, and all of the assumptions associated with the stereotype are assigned to the user. In the stereotype approach, if a new user does not satisfy any precondition of existing stereotypes, the user modeling system would not be able to classify the user. Learning in the stereotype approach could involve adding new stereotypes automatically based on data. For example, Paliouras *et al*. use supervised machine learning techniques (C4.5) to induce stereotypes on data from questionnaires for a news filtering system (Paliouras *et al*. 1999). The stereotype approach is simple to implement and useful in domains in which users can be classified into groups. However, stereotypes are prone to inaccuracy because they assume that users within a group are likely to possess homogeneous characteristics. In contrast, we will see that the Bayesian framework can easily discriminate each user's differences because the probabilities in the Bayesian network are updated based on the user's actions or available evidence.

2.1.1.3 Bug Libraries

A significant disadvantage of overlay modeling is that overlay models cannot capture a student's misconceptions because they model users' knowledge only in terms of correct knowledge. An alternative to overlay models is the bug library. Bug libraries represent common misconceptions in a domain, i.e., they represent the faulty student knowledge in terms of a set of "bugs" or "misconceptions." By mapping a student's incorrect answer to bugs in the library, the system can determine the student's misunderstanding of a given concept. The initial work on bug libraries enumerated many observable bugs in the domain of algebra (Brown *et al*. 1975), which was later extended to include a diagnostic model (Burton 1982). In this work, studies were conducted on students solving algebra problems and representing their problem-solving activities with procedural networks. Another project investigating bug libraries is PROUST (Johnson and Soloway 1985), which analyzes novice's "buggy" Pascal programs and corrects them. The inference process of PROUST primarily consists of goal decomposition and plan analysis. To detect students' bugs, PROUST uses buggy plans. Because it is very difficult and labor-intensive task to enumerate all possible bugs, there have been several projects that investigate the automatic construction of bugs. For example, Sleeman *et al*. propose techniques for extending bug libraries (Sleeman *et al*. 1990). These techniques generate new faulty rules (mal-rules) when there are no available rules and mal-rules to reconstruct the student's solution process given a problem. Bug libraries can detect misconceptions of students. However, if there is more than one possible misconception that can explain students' errors, bug libraries have difficulty determining which misconception is most likely. In contrast, the Bayesian approach can determine the most probable misconception based on probabilities.

2.1.1.4 Model Tracing

The model tracing (or knowledge tracing) approach to user modeling, which has been used extensively in ITSs, was introduced in cognitive tutors (Anderson *et al*. 1995). Cognitive tutors are ITSs that are based on the ACT-R theory of cognition (Anderson 1983). According to ACT-R, a student can learn cognitive skills by employing declarative knowledge in the context of a problem-solving process to acquire procedural knowledge, and with increasing practice, the student is less likely to produce errors. Therefore, cognitive tutors present an initial brief declarative introduction to concepts and then provide guided practice. Typically, a cognitive tutor has a set of several hundreds production rules, which collectively represent correct solutions to problems. Cognitive tutors always attempt to guide students on the correct solution path during the problem-solving process. If a student's actions do not follow the ideal model, the tutor recognizes those actions as errors. In model tracing, plan recognition problems are circumvented by insisting that each action of the student be on an interpretable path so that the tutor can understand the student's plan within a reasonable amount of time. If there is an ambiguity in interpreting the student's action, the student herself is asked to identify the proper interpretation by selecting an item from a disambiguation menu presented by the system. Although cognitive tutors using model tracing have proven effective, this approach is only applicable to procedural learning and is not suitable for modeling conceptual knowledge (Shute 1995). Moreover, model tracing assumes that the system can always understand the student's intentions because it forces the student to follow the ideal student model. Therefore, it tends to restrict the student's actions and does not give the student freedom to explore the problem-solving space. Because cognitive tutors can also have buggy production rules that represent students' common misconceptions, some approaches for automatic construction of buggy rules could be applied to cognitive tutors. Although production rules of cognitive tutors are usually manually constructed, there has been a recent effort to learn production rules from examples of student problem solving (Matsuda *et al*. 2005).

2.1.2 Probabilistic User Modeling

Recent years have seen the successful introduction of probabilistic and statistical inference frameworks into a broad array of computational tasks. Bayesian Belief Networks, Dynamic Decision Networks, Hidden Markov Models, and Support Vectors Machines have gained widespread acceptance in almost every area of AI. Probabilistic approaches to user modeling offer much promise because of their ability to deal with the uncertainty inherent in human-computer interaction. There appears to be great potential for probabilistic inference frameworks to deal with challenging ITS problems such as plan recognition and misconception detection.

In this section we explore probabilistic approaches to user modeling. Our discussion emphasizes the role that Bayesian Belief Networks can play in user modeling. Bayesian user modeling has been employed to infer the user's focus of attention and whether or not the user desires assistance since she may not wish to be interrupted if she is working on a critical task. Example systems inferring the user's focus of attention include LUMIÈRE (Horvitz *et al*. 1998) and PRIORITIES (Horvitz *et al*. 1999). The LUMIÈRE (Horvitz *et al*. 1998) project at Microsoft Research investigated techniques for reasoning about the goals (target tasks) of a software user and the user's needs (information or automated actions to achieve the goal) to provide automated assistance. LUMIÈRE was deployed in Microsoft Office suite. In this project, the Bayesian network represents causal relationships among variables. Variables represent the user's background, her competency with using software, her task history, her actions (e.g., the user interacts with a mouse and keyboard, and her actions might include menu exploration), her goals (target tasks or subtasks at the focus of the user's attention), her acute needs (e.g., requiring immediate assistance), the set of documents on which she is currently working, and her explicit query. Since the purpose of the user model is to enable the system to provide automated assistance, the user modeling system weighs the benefit of providing assistance against the costs of such an action. LUMIÈRE uses utility nodes to represent, reason about, and optimize the expected utility of providing advice to the user.

PRIORITIES (Horvitz *et al*. 1999) is an alert-transmitting system that provides advice on the arrival of email messages based on their estimated criticality, the user's focus of attention, the informational benefits of alerts, and the costs of deferring alerts. Because alerts interrupt and distract users while they are working on tasks, the user modeling system needs to infer detailed information about these items. PRIORITIES employs a Bayesian network to infer the user's focus of attention. Variables that could influence the user's focus of attention are represented in its network. These include online calendar, user location, date, time, deadline status, application in focus, ambient acoustical signal, desktop activity, inspection interval (how often the user checks the email), and application usage pattern. Because the user's focus of attention changes over time, the user modeling system models two time slices, and a Bayesian network is built for each time slice with links between nodes at different time slices to model dependencies which are related to time.

A pioneering effort in Bayesian networks for student modeling is HYDRIVE (Mislevy and Gitomer 1995), an intelligent tutoring system for learning troubleshooting skills for a hydraulics system of the F-15 aircraft. HYDRIVE simulates important features of troubleshooting F-15 hydraulics systems. In HYDRIVE, a student can perform troubleshooting procedures by accessing video images of aircraft components and acting on those components. The student can also refer to on-line technical support material and make instructional selections at any time during troubleshooting. HYDRIVE's Bayesian student model is constructed in terms of the student's level of domain expertise such as system knowledge, strategic knowledge, and procedural knowledge such as a student's troubleshooting action sequences. The network has nodes for representing a student's general proficiency in the domain at the top level and nodes for actual troubleshooting actions at the bottom level.

While HYDRIVE represents students' declarative and procedural knowledge, ANDES (Conati *et al*. 2002) provides a more granular representation of these elements and supports explicit reasoning about students' performance over time. ANDES is a physics tutor for the domain of university level Newtonian physics that uses Bayesian networks for student modeling in open learning environments. In contrast to model tracing learning environments, in open learning environments, students are not constrained to follow a fixed order of problem-solving actions: they can pursue any strategy they wish. The Bayesian student model in ANDES is used to infer a student's goals and plans and to assess the student's knowledge in order to provide customized advice. ANDES' Bayesian network represents two kinds of knowledge: domain-specific knowledge and task-specific knowledge. Domain-specific knowledge includes student long-term knowledge which is represented as rule nodes and context-rule nodes in the Bayesian network. Every node has two values, mastered or unmastered. If a rule node has the value mastered, it indicates that the student can correctly apply the knowledge that the rule node represents in all possible contexts. A context-rule node represents whether a student has mastered a rule in a specific problem solving context. The Bayesian network represents task-specific knowledge via context-rule, fact, goal, rule-application, and strategy nodes. Goal and fact nodes represent information that is derived by applying rules during problem solving process. Rule-application nodes connect Context-rule nodes, proposition nodes (Goal or Fact nodes), and Strategy nodes to proposition nodes which are derived by applying the Context-rule. Strategy nodes address the case where a solution has more than one possible solution path. Moreover, to reason over time, ANDES uses a "rolleup" mechanism to summarize past evidence which is reflected in the Bayesian network for the current problem as prior probabilities.

Another approach to Bayesian student modeling is constraint-based modeling, which focuses on evaluation of the constraints that correct solutions should satisfy. Constraint-based modeling is based on Ohlsson's theory of learning from performance errors (Ohlsson 1996). Ohlsson's theory claims that humans make mistakes when they perform a task even if they have been taught the correct way to do it because the declarative knowledge has not been "compiled" into procedural knowledge. Constraint-based modeling assumes that all correct solutions satisfy all of the general principles of the domain and that incorrect solutions violate the principles of the domain (Mitrovic *et al*. 2003). In constraint-based modeling, constraints represent the conditions that all correct problem solutions have to satisfy. Therefore, if the student's action violates constraints, it indicates that the student has incomplete or incorrect knowledge (Mitrovic *et al*. 2001). An example of constraint-based student modeling is employed in the CAPIT project (Mayo and Mitrovic 2001). CAPIT (Capitalization And Punctuation Intelligent Tutor) teaches capitalization and punctuation for children. CAPIT represents constraints with a Bayesian network. One of the advantages of CAPIT is that it learns the structure of its Bayesian network from data so that it can be authored without an expert. CAPIT's Bayesian network represents only two variables, which are the outcome of the previous attempt to satisfy a solution constraint and the predicted outcome of the next attempt to satisfy that constraint. Therefore, it is much easier for CAPIT to learn the structure because the variables and the possible links are fixed.

Recently, there has been work on Bayesian networks for student modeling of affective factors such as enjoyment and morale. Conati *et al*. present a model of user affect for educational games (Conati 2002; Conati and Maclaren 2005). The techniques are illustrated with PRIME CLIMB, an educational game. Node's in PRIME CLIMB's network represent goals (e.g., have fun, learn math, avoid failing, succeed by myself, beat partner), goal satisfaction (e.g., have fun, learn math, avoid failing, succeed by myself, beat partner), and emotions (e.g., emotion for game-joy/distress, emotion for self-pride/shame, emotion for agent-admiration/reproach). During the interaction with users, PRIME CLIMB's pedagogical agent uses the assessment of user affect to direct its intervention, although it is not fully implemented in the current version. More recent work on this project is trying to refine the previous user affect model based on user studies (Conati and Maclaren 2005).

Some probabilistic user models employ a decision-theoretic approach with utility functions. While LOOKOUT (Horvitz 1999) is not based on a Bayesian network, it takes a decision-theoretic approach to user modeling for scheduling and meeting management with Microsoft Outlook. When a user opens a new email message, LOOKOUT parses the text of the email body and subject of the message and tries to identify the date and time of the appointment related to the email message. From this information, the system tries to infer whether the user will wish to schedule the appointment or not. To assist users with scheduling, LOOKOUT considers two expected utilities: the expected utility of taking autonomous actions to assist the user with an action, and the expected utility of not taking autonomous action to assist the user. Both of them are defined in terms of the weighted sum $p(G|E)$ (the probability that the user’s goal is to obtain the service given the observed evidence) and the utility for each case. If the expected utility of action is greater than the threshold (the point where two expected utility values are the same), the system then takes an autonomous action; otherwise, it does not. The system also considers the utilities for initiating dialogue about a goal when the user actually desires the goal and when the user does not desire the goal. Instead of taking action, the system engages the user in a dialogue about action, depending on whether the inferred probability that the user desires service is above the threshold for dialogue or action. LOOKOUT employs a probabilistic classification approach (a support vector machine for text classification) to identify the user's goal (e.g., whether the user wants to use the scheduling system). LOOKOUT constructs a linear support vector machine classifier by training on approximately 1000 messages to induce two classes (whether the user wishes to use the calendar for a message or not). However, LOOKOUT’s inferences are based solely on the content of email and do not consider other factors such as the user’s focus of attention or other characteristics of the work environment.

2.1.2.1 Inference in User Modeling

One of primary advantages of Bayesian networks is that data can be explicitly represented in Bayesian network variables and dependencies between variables can explicitly be represented in network structure. Together, these properties enable Bayesian networks to explicitly reason about how variables affect each other based on the available evidence. Reasoning (or inference) in Bayesian networks is the process of computing the posterior probability of the variable of interest given evidence based on Bayes’ theorem and conditional independence. Because exact inference in Bayesian network is NP-hard, approximation algorithms such as likelihood sampling or Monte Carlo simulation are often used.

A Bayesian user model is used for evaluating the student’s level of knowledge of the domain based on the observable student actions. Nodes at the bottom level of network represent observable student actions, the parents of the observable actions represent the knowledge of subsystem and strategic knowledge that the student should know to perform those actions, and the top level node represents overall proficiency. The Bayesian user model is used to evaluate the student's level of expertise by updating beliefs about the student's knowledge (strategic, procedural, or system knowledge) from observable actions using Bayesian inference and it was used for providing direct instruction to students.

A Bayesian user model can be used to create adaptive instruction, including providing customized hints, generating explanations tailored to individual users, and selecting problems. For example, the Bayesian network in ANDES is used to determine the topic of hints when a student requests help. To determine the topic of a hint, the user modeling system needs to know which goal the student is attempting to achieve and where the student will probably encounter difficulty. Since nodes of Bayesian network of ANDES represent goals, sub-goals, and rule nodes, the search procedure for finding hint topic starts from the most recently observed node and proceeds upward to find a goal node that the student is most likely to achieve (i.e., a goal node with highest probability). Then, it searches downward from that goal node to find a Rule-application node with a low probability and this Rule-application node becomes a hint topic. In ANDES, the Bayesian network for each problem is authored by an expert and the conditional probabilities of ANDES are defined on the basis of the knowledge of the domain expert. ANDES' top-down search-based mechanism enables it to find the most appropriate hint topic, although it cannot detect students' misconceptions.

In READY (Bohnenberger *et al*. 2002; Jameson *et al*. 1999), a dynamic Bayesian network is used to infer a user's resource limitations such as time pressure and cognitive load because time pressure and cognitive load may vary over time. Based on inference about resource limitations, the system can determine the most efficient interaction policy. The initial prior and conditional probabilities are learned from data acquired by off-line experiments with users. Then, during on-line interaction with users, the conditional probabilities are tuned for the particular user. READY utilized a Bayesian user model for making decisions about the type of assistance (e.g., step-wise or bundled instruction) to provide based on the inference about resource limitations. Because learning in READY uses data gathered from experiments conducted in a controlled environment, it can cope with inaccuracies that could otherwise be caused by sparse data.

Two variations on traditional Bayes networks have been used in user modeling: dynamic Bayesian networks, and dynamic decision networks. If variables in a Bayesian network change over time, the network should capture the dynamic properties of the variables. Dynamic Bayesian networks represent this time-varying property by maintaining multiple copies of the variables, one for each time slice. If a Bayesian network is extended with decision and utility nodes, the model is called a decision network and if the decision network represents temporal properties, then it is called a dynamic decision network (DDN). Some user models adopt static Bayesian networks and others adopt decision networks or DDN.

Dynamic Bayesian networks can be used for inferring the user's focus of attention. For example, PRIORITIES (Horvitz *et al*. 1999) uses a dynamic network to infer a probability distribution over the tasks competing for a user's focus of attention. To determine whether the system should transmit an alert immediately or defer it, PRIORITIES employs a decision-theoretic approach that considers a user's focus of attention and the estimated criticality of a message. Instead of including a utility node in the network, PRIORITIES defines utility functions in terms of the expected cost of transmitting an alert and the expected benefits of an alert. It assumes that the expected utility of transmitting an alert is the difference between the expected cost and benefits of the information provided by the alert. It computes the expected cost of immediate alerting and the expected cost of deferring an alert for some time *t*. The expected cost of deferring an alert is the difference between the expected utility of taking immediate ideal action at time t_0 and the expected utility of delaying the ideal action until some future time *t*. Although PRIORITIES infers the user's focus of attention base on high-level features such as the application in focus or user location, its accuracy might be improved if it considered lower-level features such as eye gaze.

2.1.2.2 Learning in User Modeling

Learning in Bayesian networks refers to learning conditional probabilities or the network structure from raw data. Very few user modeling projects have addressed learning network structure and conditional probabilities from data. In DT TUTOR (Murray *et al*. 2004), the structure is authored, most conditional probabilities are authored, but conditional probabilities for unobserved variables are learned from data. In DT TUTOR, data were collected from pretests, student interaction logs, and posttests. Pretest data was used to compute prior probabilities and many of DT TUTOR's conditional probability tables were constructed using so-called predefined rule-based techniques as in ANDES (Conati *et al*. 2002). However, DT TUTOR learned conditional probabilities for unobserved variables from data, but only basic techniques were used for learning. The probability for each outcome of the unobserved variable was calculated as the ratio of events with the outcome of the variable to the total number of similar events. Like DT TUTOR, READY (Bohnenberger *et al*. 2002) learns conditional probabilities from data, but the network structure is fixed.

CAPIT (Mayo and Mitrovic 2001) and WAYANG OUTPOST (Arroyo *et al*. 2004) and also learns the structure and conditional probabilities from data. Because CAPIT is a constraint-based tutor, it represents constraints for each problem solving attempt instead of representing skills and hints, an approach that is different from that used in WAYANG OUTPOST. However, CAPIT limits the possible links between variables so that structure learning can be accomplished efficiently because there is only limited number of possible links to be considered in structure learning. WAYANG OUTPOST, a web-based ITS for the mathematics section of the SAT (Arroyo *et al*. 2004), learns network structure and conditional probabilities from data. Because WAYANG OUTPOST focuses on the relationship between received hints and problem solving skills (e.g., "how seeing a hint is related to knowing a skill"), the learned network represents skills and hints and the network is used for predicting students' actions after seeing hints. Like CAPIT, structure learning can be accomplished efficiently because part of the structure (e.g., the number of layers or variables) is fixed. However, CAPIT only represents observable variables. In contrast, WAYANG OUTPOST has multiple layers which include hidden variables.

Zukerman's plan recognition system (Albrecht *et al*. 1998) has a fixed Bayesian network structure, but it learns conditional probabilities from data. The training data consists of a time stamp, the name of the player, the number of log-in sessions, the location where the action was performed by the player, and the name of the action. The learning process is based on a straightforward frequency counting approach, and if the system has not observed an action *A* given the location *B* and the current quest *C* during training, then the probability of an action *A* given *B* and *C* is simply set to zero even if the action *A* could occur in the same situation.

For parameter learning, missing values and sparse data should be handled properly because conditional probabilities learned from data with missing values could be inaccurate (probabilities of missing values could be zero even if one might observe those value). Moreover, if the network has hidden variables, learning could be even worse because hidden variables are not observable. Most projects dealing with structure learning make assumptions to make learning tractable. For example, they significantly reduce the scope of the problem by dealing only with constraints (CAPIT) or only with skills and hints (WAYANG OUTPOST). Furthermore, the initial structures are primarily constructed by a domain expert.

2.2 Intention Recognition

Plan recognition is typically divided into two types: keyhole plan recognition and intended plan recognition (Cohen *et al*. 1981). In *keyhole* plan recognition, the observed user is not aware of the observation. Most plan recognition applications perform keyhole plan recognition. In *intended* recognition, the observed user knows she is observed and consciously takes actions to make her intentions clear to the observer. For example, in natural language dialogue, speakers often form utterances in such a way that hearer can easily recognize the communicative goals she is attempting to achieve. Thus, understanding dialogue utterances is one form of intended recognition. The problem of goal recognition (Lesh 1997; Blaylock and Allen 2003) is a restricted form of the plan recognition problem. It should be noted that some use the term “intention recognition” to refer to a combination of goal and plan recognition (Mao and Gratch 2004b). We adopt this inclusive usage of the term.

2.2.1 Goal Recognition

Goal recognition is a special case of plan recognition. Goal recognition only seeks to predict a user’s goals from a series of observations. Goal recognition is not as informative as plan recognition, but it can be useful because performing goal recognition may be sufficiently efficient to meet the real-time requirements of interactive systems. Blaylock and Allen divide goal recognition into two types based on the target goal structure: flat goal recognition and hierarchical goal recognition (Blaylock and Allen 2005). While flat goal recognition focuses on predicting goals at a single level, typically the top-level, hierarchical goal recognition attempts to recognize active sub-goals as well as top-level goals.

Two fundamental approaches have been taken to goal recognition: logic-based goal recognition and probabilistic goal recognition. We first discuss logic-based goal recognition and then turn to probabilistic goal recognition.

An early logic-based goal recognition approach used graph analysis (Hong 2001; Lesh and Etzioni 1995). It employed a graph representation of the domain called a consistency graph. Nodes in a consistency graph represent actions and goals. Initially, nodes in a consistency graph are fully connected to each other. Inconsistent actions and goals are then pruned from the graph (Lesh and Etzioni 1995). The experiment was conducted in the Unix domain. Subjects were given goal descriptions and they were asked to achieve each goal by executing Unix commands. The results showed that their recognizer successfully detected every inconsistency between goals and observations. Their algorithm runs in time that is polynomial in the size of inputs.

Hong (2001) extended Lesh & Etzioni’s work. His approach to goal recognitions consists of two stages. First, a graph structure called a goal graph is constructed to represent actions, goals, and the states of the world. In a goal graph, action nodes represent observed actions at each time steps, proposition nodes represent the states of the world at each time steps, and goal nodes represents the partially or fully achieved goals at each time steps. Edges in a goal graph represent the relationships between action nodes and proposition nodes and relationships between proposition nodes and goal nodes. For example, the action nodes at level *i* are connected to proposition nodes at level *i* which are preconditions of the actions, and the action nodes are also connected to proposition nodes at level *i*+1, which are effects of the action. After a goal graph is constructed, the algorithm uses the constructed graph to recognize consistent goals and valid plans. The goal recognizer was also tested in the Unix domain with a corpus collected at the University of Washington. Subjects were given goal descriptions and then were asked to achieve the goals by executing Unix commands. The algorithm successfully recognized 13 goals out of 14 given goals, and it is computationally efficient (polynomial-time and polynomial-space).

However, these logic-based approaches suffer from several problems. First, they are not able to distinguish between logically consistent goals and are not able to deal with uncertainty. Second, noisy observations make the recognizers predict erroneous goals because they prune away logically inconsistent goals every time they observe an action which may in fact be noise. We have recently begun to see attempts to solve these problems with probabilistic approaches such as Dempster-Shafer theory, *n*-gram models, Bayesian networks, and probabilistic grammars.

A landmark project in Bayesian goal recognition is Zukerman's Bayesian keyhole goal recognition model (Albrecht *et al*. 1998). Here, a Bayesian goal recognition model infers a user's long term goals in the context of a text-based adventure game where users compete for limited resources to achieve various goals. The experiments were conducted in game with more than 4,700 locations and 20 different quests (goals); players could perform more than 7,000 actions. The Bayesian model is used to predict a user's current quest (goals), next action, and next location on the basis of the previous quest and the series of previous actions and locations.

The conditional probabilities of the model are learned from data gathered from actual user interactions with the system. However, the system does not learn the structure of Bayesian networks; rather, Albrecht *et al*. propose three different types of dynamic Bayesian network structures that represent different conditional dependencies of the variables. Efficiency is important for runtime operation of the model because the user's plans must be predicted in real-time. The inference time for Bayesian networks depend on the possible values of each node and the size of data for computing conditional probabilities. To increase the runtime performance of the model, the approach used here reduces the size of representation of locations and actions by screening non-significant actions using classification techniques and abstracting locations (e.g., represent a specific location as a larger location which includes the specific location). Together, these speed up the runtime inference. One drawback is that their recognizer only recognizes atomic goals, and cannot recognize goal schemas.

Blaylock and Allen devised probabilistic approaches (Unigram, Bigram, and Hidden Markov Models) for flat goal recognition and hierarchical goal recognition (Blaylock and Allen 2003; Blaylock and Allen 2004; Blaylock and Allen 2005; Blaylock and Allen 2006a; Blaylock and Allen 2006b). For empirical evaluations of "flat goal recognition", they use Lesh's Unix plan corpus (Lesh 1997). The results show 55.4 % accuracy and 78% of cases are converging. The hierarchical goal recognition approach was evaluated with the Monroe corpus (an artificially generated corpus for the emergency response domain). For top-level goal recognition, precision was 94.3% for the 1-best case. For other-level goal recognition, precision was high 90's for the 1-best case. Recognition for both flat and hierarchical goal recognition are fast: polynomial time for flat goal recognition and quadratic in the number of possible goals and the number of observations for hierarchical goal recognition. Machine learning techniques are used to train the goal recognizer on a domain given a corpus. Because their recognizer learns users' behaviors from a corpus, it does not require a manually constructed plan library. In addition, because it supports goal schema recognition and action schema recognition, it can make partial predictions if not all parameter values are available.

Yin *et al.* (Yin *et al.* 2004) propose two-level goal recognition architecture which uses a Dynamic Bayesian Network (DBN) to infer a user's actions from raw sensory inputs provided by wireless networks and *n*-gram models to infer a user's goal from a predicted user's action. In the domain, there are 11 actions (e.g., Walk-in-HW4) that a user can perform and 19 goals (e.g., Seminar-in-Room1, Print-at-Room1, Return-to-Office) that a user can achieve. Low-level DBN models start from raw sensory inputs to infer a user's action and then high-level *n*-gram models were used to infer a user's goal from a user's actions. Given a sequence of observations, the DBN infers the most probable action sequences. Then, given the estimated sequence of actions, the *n*-gram model infers the most likely goal. They compared the DBN-only model (which uses a DBN for both layers) and a DBN+*n*-gram model on accuracy and convergence rate. The DBN+*N*-gram showed a similar performance as the DBN-only model. For the experiment, they collected about 570 traces for 19 goals of a professor's behavior in an office area. They report 90.5 % of accuracy for DBN+*N*-gram model and 89.5% accuracy for the DBN-only model. Like Zukerman's model, their recognizer can only predict a single goal and it is unable to recognize multiple goals and goal schema.

2.2.2 Plan Recognition

Plan recognition approaches can be divided into two types: logic-based approaches and probabilistic approaches. In the logic-based approach, a plan library is used. Each plan operator in the library specifies an action's preconditions, which are true in the current world state or can be achieved by sub-goaling, and the effects of executing the actions. For example, plan recognition for user modeling begins with a set of goals that a user might be attempting to achieve and an observed action by the user. To infer the user's goal from the observed actions, the plan inference system connects the observed action to one of possible goals which could be achieved by the action. From these goals, the system finds actions for which the goal is a precondition or sub-goals, and performs the same process from these actions to the goals, and so on. The resulting inference path is an alternating path of actions and goals which start with an action and ends with a goal (Carberry 2001). Since there could be many possible plans or goals that are achieved by one observed action, the plan inference system often produces multiple hypotheses about the user's plan. Therefore, plan recognition research has focused on the techniques for narrowing the space of possible hypotheses. Logic-base approaches use logical methods to choose more plausible hypothesis.

One of the first plan recognition systems is BELIEVER (Schmidt *et al*. 1978). BELIEVER was based on psychological studies that show how humans do plan recognition. The experiments showed that humans make the best guess based on observations instead of having the list of possible plans that explain the observations. BELIEVER endeavored to imitate these human plan recognition processes. BELIEVER was given observations and knowledge about the world and attempted to match observations with the expected plan structure, which is a graph of actions. If the expected plan structure is not matched with observations, the expected plan structure is revised.

Kautz presents a formal theory of plan recognition (Kautz and Allen 1986; Kautz 1987). In this plan recognition framework, the possible plans are represented by a set of first-order statements called an event hierarchy, which includes the abstraction and decomposition relationships between events. This framework assumes that an event hierarchy is complete, i.e., it is an exhaustive description of the ways in which actions can be performed. It also assumes that users perform the minimum number of consistent actions to achieve goals and have correct plans and complete knowledge about the domain. Thus, plan recognition becomes the problem of finding the minimal set of events covering the observed actions given an event hierarchy. The runtime performance of the recognizer is exponential in the size of the event hierarchy so that it is not scalable to large domain, and the assumptions are too strong for any realistic domain although it can handle partial-order and interleaved plans.

Carberry (Carberry 1988; Carberry 1990a) presents a plan recognition approach for a natural language dialogue system. Her system first analyzes an utterance without considering the preceding dialogue to hypothesize a set of domain-dependent goals (candidate focused goals) and associated plans (candidate focused plans) by using plan-identification heuristics. Secondly, the system relates an utterance to the context by considering the preceding dialogue. The system uses a tree structure called a context model to represent users' goals and plans inferred from preceding dialogue. As each new utterance occurs, the context model is expanded to include the most likely relationship between one of the hypothesized candidate plans and the context model. Because the plan recognition process is based on the communication between users and systems and users directly communicate their intended actions and goals, the approach cannot be directly applied to keyhole plan recognition.

Logic-based approaches to plan recognition suffer from two problems. First, they do not perform well with noisy data. Noisy data can cause classic plan recognition to infer incorrect plans. Second, to work effectively, plan libraries must have a complete plan library. However, it is unrealistic for a plan library to represent all possible way for a user to achieve a goal. Together, these concerns prevent a classic approach to plan recognition from being robust. A probabilistic approach to plan recognition addresses these issues. Probabilistic approaches do not require a complete plan library. For example, complete plan libraries do not need to be built in probabilistic approaches because probabilistic models can derive plans by interpreting probability values.

Probabilistic approaches are based on probabilistic models of planning and employ abductive reasoning techniques such as Bayesian inference or Dempster-Shafer theory (DST) to infer the underlying plans from a sequence of observations. For example, Bauer uses Dempster-Shafer theory to initialize and update the probability of goals and plans given observations (Bauer 1996a). He explains the technique in the domain of email (read/delete email). Bauer's recognizer employs a hierarchical plan representation to represent abstract plans, decomposed plans, and actions. The recognizer uses the relative frequencies of performed plans to obtain a numerical estimation of the user's future behavior, which is used for a DST basic probability assignment. Bauer also attempts to integrate machine learning techniques (decision-tree classification) into Dempster-Shafer theory in order to handle the situation where a user's action is deviated from her typical behavior (Bauer 1996b). To assess the hypotheses, he uses results from decision-tree classification and then combines this result with the existing hypotheses assessment using Dempster-Shafer rule.

Charniak and Goldman propose a formal plan recognition model which consists of the knowledge-base of facts about the world expressed in a first-order notation and rules to construct a Bayesian network by using the knowledge-base (Charniak and Goldman 1991; Charniak and Goldman 1993). Based on the formal model, they built a system (Wimp3) that creates a Bayesian network directly from English narratives. Wimp3 consists of three components: a parser, a network constructor, and a network evaluator. Words from a narrative are given to the parser and the parser produce a syntactic parse which is given to the network constructor to produce the Bayesian network. The Bayesian network evaluates the conditional probability of the competing hypotheses given the evidence. Although their model can predict the most probable hypothesis, it cannot handle situations in which the recognizer fails to observe some actions that were in fact performed by the agent; it also cannot handle dynamic plan recognition problems in which the observed propositions change over time.

More recently, Geib *et al*. propose a probabilistic plan recognition framework to handle partially-ordered plans and multiple, interleaved plans (Goldman *et al*. 1999; Geib and Goldman 2005). They built PHATT (Probabilistic Hostile Agent Task Tracker), a system based on their framework that uses Bayes rule to compute the conditional probability of an explanation given observations. To handle partial observability, they quantified the probability that the observed agent actually performed some action, but it was not observed (probability of not observing given an action). For each possible observed action, they learn the long-term false negative rate of the observed stream and use this as a prior probability that action might not observed when it actually happens. This prior probability was also combined into the computation of the conditional probability of the explanation given a sequence of observation. They explored the run time cost of their algorithm. Without considering unobserved cases, the running time was under a second. Considering unobserved cases, all of the examples with more than 12 observations exceed 100 seconds. Accuracy was 75% (correctly identifying the plans) for all of the test cases, but their test cases did not exhibit a high degree of ambiguity.

None of the work discussed above takes into account the expected utility of user actions. In real-world environments, when a user performs actions, it is natural for users to adopt plans that maximize the expected utility of their actions. Mao and Gratch propose a decision-theoretic approach to plan recognition that takes into account outcome utilities (Mao and Gratch 2004a; Mao and Gratch 2004b). In their approach, they adopt a probabilistic approach to plan representation in which each action has a set of precondition and effects. Probability values are associated with preconditions and effects to represent their likelihood. Utility is also associated with action effects to represent the desirability of action outcomes. To disambiguate the competing plans, they compute the expected utility of the observed user's plans and predict that the observed user is pursing the plan which has the maximum expected utility.

2.3 Affective Computing

Because affect plays a central role in human cognition, it is widely believed that affect modeling can contribute to a broad range of computational tasks (Picard 1997). Affective computing investigates techniques for giving computers the ability to effectively recognize, understand, and express emotion. Incorporating affective computing into interactive applications holds much appeal. For example, affect-informed educational, training, and entertainment systems may create more effective, interesting, customized experiences for users.

Recognizing and expressing affect have been studied extensively in the context of synthetic agents. The task of affect generation (affect synthesis) is to give computers the ability to synthesize or generate emotions. Work on affect synthesis has focused on virtual humans. For example, the Oz project explored techniques for creating computational models of believable agents (Bates 1994). Based on the OCC model of emotions (Ortony *et al.*, 1988), the Oz group presented self-animating creatures called “Woggles”. They used a goal-directed and behavior-based architecture to direct the actions of the Woggles and connect the architecture of the action to a component for representing, generating, and expressing emotions based on OCC model. For example, a Woggle exhibits the behavior of anger when it experiences an important goal failure and judges that it was caused by another woggle.

Gratch and Marsella proposed an alternative general computational model of affect (Gratch and Marsella 2004). The model was implemented in Mission Research Exercise (MRE), a training system for teaching decision-making skills in social interactions. The conceptual basis of their model is Smith and Lazarus’ appraisal theory of emotion. Appraisal theory explains the human behavior as two basic cognitive processes: appraisal and coping (Smith and Lazarus 1990). Appraisal refers to a person’s interpreted relationship of their physical and social environment. Coping determines strategies for repairing, altering, and maintaining the relationship. The model constructs and maintains a causal interpretation of an agent’s current mental state which includes past, present, and future states and actions, their likelihood and desirability and casual relationships between them. The features of causal interpretation are represented as multiple appraisal frames and each appraisal frame are mapped into individual instances of emotion. The model adopts a coping strategy to respond the current emotional state.

Affect recognition is the task of identifying the emotional state of a user from a variety of signals which are produced in response to affective changes in the individual. These include facial expressions, posture, vocal intonation and changes in physiological signals such as muscle tension, blood pressure, galvanic skin response, and respiration (Picard 1997). Affect recognition work has investigated emotion classification from student self reports of motivation and mood (Beal and Lee 2005), video recording of tutorial interactions (de Vicente and Pain 2002), physiological signals (Conati 2002; Prendinger *et al*. 2003; McQuiggan *et al*. 2006), and from combinations of visual cues (e.g., facial expression, eye movement, and posture) and physiological signals (Burleson and Picard 2004).

Beal and Lee present an approach to assessing student's motivation and mood from students' self reports for an ITS for learning secondary school mathematics. Before students use the system, they complete a brief mood report by answering questions such as, "I am having a <great day, OK day, bad day>. From students' self-reports of mood, the ITS makes pedagogical decisions about the difficulty of the problems, types of the problems, and types of multimedia help. For example, if a student reports that she is having a bad day, the system might suggest a tutorial of the material that the student has already mastered. Instead of recognizing motivational states, their work focused on determining pedagogical actions based on the student's motivational state and mood which were self-reported by the students.

An interesting approach to recognizing students' motivational states from their observed actions has been proposed (de Vicente and Pain 2002). The work is based on an empirical study for diagnosing students' motivational states in the tutoring system for learning Japanese numbers. In the study, a tutor is asked to infer a student's motivational state from actions recorded from students' interacting with a tutoring system. The tutor can only watch prerecorded interactions of a student with the ITS. When the tutor inferred the student's perceived motivational state, she was asked to comment on the student's motivational states and verbalize the reasoning behind her inference. Based on the inference made by the participants, they elicit 85 motivation diagnosis rules.

Some groups have explored the use of physiological response to detect users' emotional states (Prendinger *et al*. 2003). Prendinger and his team utilized bio-signals to investigate the effects of a like-like synthetic agent with affective behavior on users' emotional states which were derived from user physiological data. They reported the result of an empirical study showing that affective feedback of a synthetic agent may reduce user stress. Conati *et al*. present a model of user affect for educational games (Conati 2002; Conati and Maclaren 2005). The techniques are illustrated with PRIME CLIMB, an educational game in which a pedagogical agent uses the assessment of student affect to direct its intervention. More recent work on this project is exploring techniques for refining the previous user affect model based by utilizing user physiological data such as galvanic skin response (Conati and Maclaren 2005). In a similar vein, the Affective Learning Companion project at MIT senses students' affective states through various channels (e.g., facial expression analysis, pressure mouse, and skin conductivity) to assess students' progress, which could be used to determine appropriate interactions and interventions of the affective learning companion (Burleson and Picard 2004).

Chapter 3

Goal and Affect Recognition

In this chapter, we first define the tasks of goal recognition and affect recognition. We present an affect-goal recognition architecture that consists of two components: a Goal Recognizer and an Affect Recognizer. Manually constructing goal and affect recognition models is labor intensive and for many domains impractical. We therefore present the adoption of an inductive approach to model acquisition. After describing the task of model induction, we discuss issues of training corpus acquisition and then turn to learning issues.

3.1 Task Definition

The tasks of goal recognition and affect recognition can be defined as follows.

- *Goal Recognition*: The task of plan recognition is to identify the plan P^* from a set of candidate plan trees p_1, p_2, …, pm given an action sequence A_1, A_2, …, A_n. The plan recognition result P^* is a plan tree that explains the observation sequence A_1, A_2, …, A_n. The task of goal recognition (a special case of plan recognition) is to identify the goal G^* from a set of candidate goals g_1, g_2, …, g_m given an action sequence A_1, A_2, …, A_n. Note that while we use the terms "plan recognition" and "goal recognition," what will be studied in practice is a combination of goal recognition and plan recognition, termed by some "intention recognition" (Mao and Gratch 2004b).

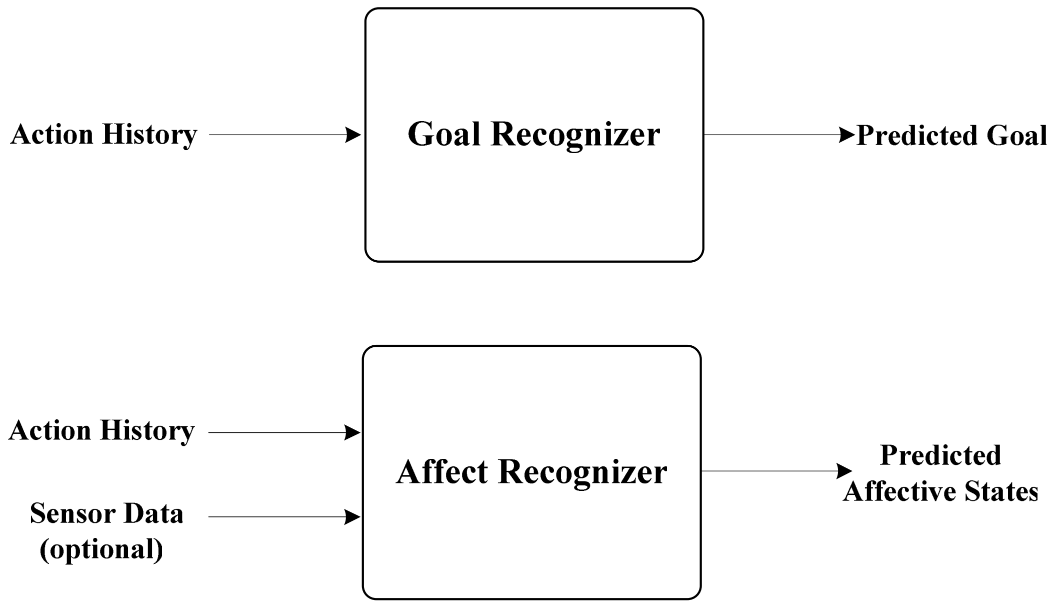


Figure 3.1: Goal Recognition and Affect Recognition

- *Affect Recognition*: The task of affect recognition is to identify the affective state S^* from a set of candidate affective states s_1, s_2, …, s_m given an observation sequence D_1, D_2, …, D_n. Each D_i represents observable attribute vector that the underlying affective states may be inferred by observing the attributes. This input attribute vector includes various sensor data such as physiological response, eye movement, voice intonation, and bodily movement

The task of goal and affect recognition is to identify both the goal G^* and affective state S^* given an observation sequence O_1, O_2, …, O_n. O_i represents the attribute vector. The choice of input attributes depends on the application. We consider the following input attribute vector:

- *User Actions* A_1, A_2, …, A_n: An action sequence that a user has performed so far. An affect-plan recognizer can observe users' actions in the world; recognizers also have access to auxiliary information about the interactions, e.g., any artifacts manipulated such as which objects have been picked up or which doors have been opened, as well as the characters with which users have interacted.

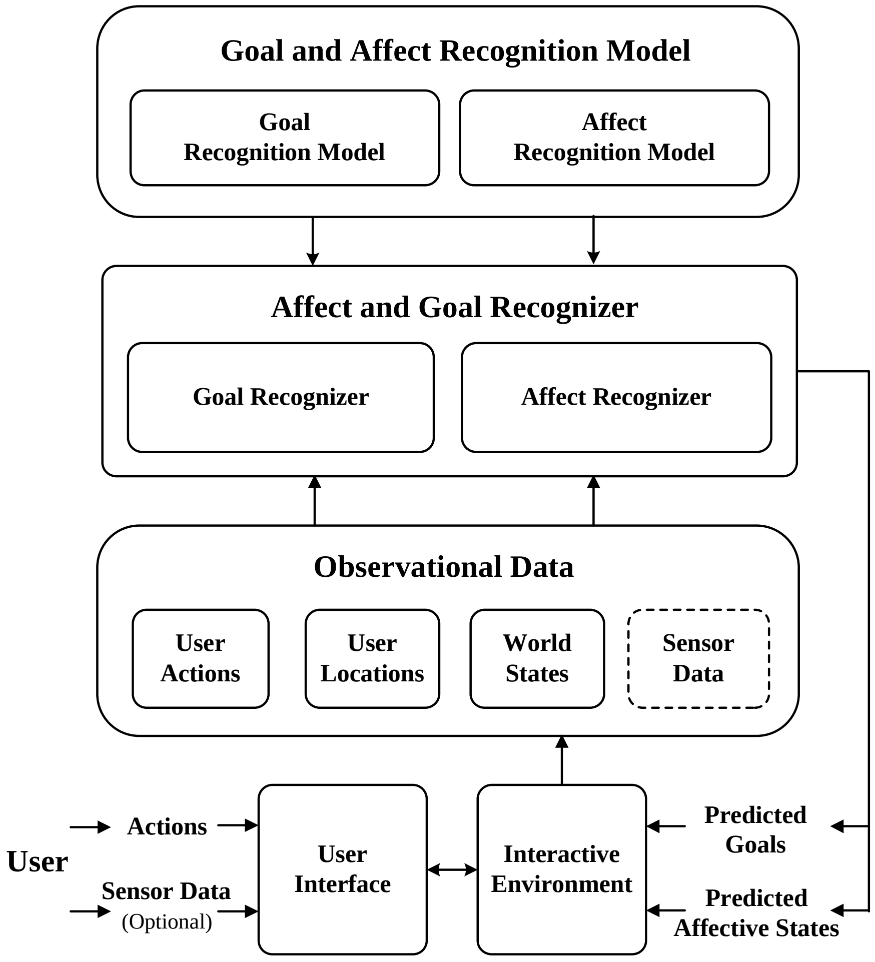


Figure 3.2: Goal and Affect Recognition Architecture

- *User Locations* L_1, L_2, …, L_n: A sequence of locations in which a user has performed actions in virtual environments. Goal and affect recognizers can bring to bear a broad range of knowledge about the location in which users' actions are performed in virtual environments. In contrast to activity recognition in physical environments where recognizers must cope with noise and errors in sensors and perception (e.g., vision and speech), goal and affect recognition has access to precise locational information.
- *Sensor Data* D_1, D_2, …, D_n: A sequence of sensor data (e.g., galvanic skin conductivity or heart rate) from which user affective states can be inferred. Goal and affect recognizers have access to sensor data. In contrast to user location information, there

can be noise in sensor data, so goal and affect recognizers must cope with noisy sensor data streams.

- *The State of the World* W_1, W_2, …, W_n: Other miscellaneous information such as information about virtual agents or objects in the environments. Goal and affect recognizers track all information in the world such as other character's actions and goals and narrative states.

Figure 3.2 depicts the goal and affect recognition architecture. In runtime operation, the user interacts with a virtual environment via a user interface. The user navigates the virtual 3D world to accomplish problem-solving tasks. Throughout the interaction, the interactive environment tracks all activities in the world and monitors the observational data such as user actions, user locations, states of the world, and sensor data. User actions include navigation actions (e.g., move to the location), manipulation actions (e.g., pick up objects, stack objects, open a door, test an object), communication actions (e.g., talk to a person), and information-seeking actions (e.g., read a book). User locations represent the location in which user actions are performed. World states represent the current state of the world (the state of other character, narrative states, the focus of user attention). Various sensor data can be used to detect user emotions such as users' physiological state changes, user body posture, eye movements, and voice signal changes. Note that the task we propose to explore includes both the "tethered" and "untethered" versions of the problem, i.e., at runtime, sensor data may or may not be available. These observational data are given to the Goal and Affect Recognizers.

Predicted user goals and affective states are given to the interactive environment to create customize interactions for users. For example, based on the predicted affective states, an Intelligent Tutoring System could determine when to present a hint or how to present a hint, including modality choices. Interactive environments can also take advantage of the predicted goals to assess the knowledge state of a user.

3.2 Learning Goal and Affect Recognition Models

The task of model learning is to induce a model from observations of training data such as the interaction traces between a user and an environment. The central goal of model induction is to generate a model that makes accurate predictions for new (unseen) data. Ideally, the gathered training data should be representative of all possible situations in which users will be encountered at runtime. However, this assumption rarely holds in practice, so the induction process must generalize over the training data to unseen data. The induced model should therefore represent the underlying systematic characteristics of the data rather capturing specific details of the particular training data.

Two fundamental approaches to learning can be adopted: supervised learning and unsupervised learning. In supervised learning, for each data item, the value of the target output is specified. Thus, supervised learning is the task of learning an input-output mapping (the target function). In unsupervised learning, for each input data, the value of target output is not specified. Instead of learning a target function, unsupervised learning may model the probability distribution of the input data or discover clusters or other structure in the input data. We adopt a supervised learning approach in which labeled classifications in the form of goals, plans, or affective states are provided by a trainer. The task of model learning will therefore occur in two phases: a training phase, which is followed by a learning phase. In the training phase, training data will be collected by recording detailed logs of the interactions. In the learning phase, the model will be induced from the training data.

Probabilistic frameworks are well suited to learning. Learning in Bayesian networks often refers to learning conditional probabilities or the structure from raw data. Usually, the network structure is manually constructed because structure learning is much more difficult than parameter learning. Moreover, if data is missing or some of the nodes are hidden (not observable), then learning is much more difficult. Structure learning is useful when prior knowledge is unavailable and we want to discover underlying causal or informational relationships to infer knowledge about the domain. Most Bayesian network structures of user models are manually constructed by experts because the experts know what kinds of variables are needed and the relationships between them. However, it is much more difficult to capture the relationships between variables beforehand for unrestricted environments (e.g., those that are common in interactive learning environments to support exploratory learning) in which users can adopt many approaches to accomplish tasks. Structure learning is the model selection process of choosing among possible models or hypotheses based on an objective function. An objective function measures how well the model can fit the data. Because structure learning is inducing a directed acyclic graph that best explains the given data, the number of possible acyclic graphs given N variables is super-exponential in N (Murphy 2002). Thus, designers of Bayesian networks often use heuristics to avoid examining all possible structures or they begin with an initial proposed structure.

Determining prior and conditional probabilities are crucial steps in constructing Bayesian networks. If one can observe all possible configurations of variables and there are no hidden variables in the network, then computing probability tables is merely counting the number of occurrences of each configuration. However, because all possible configurations of variables typically cannot be observed, conditional probabilities of some variables cannot be easily computed from data. For example, if a network has N nodes (variables) and even if each node can have discrete binary values, then, at worst, the total number of possible configuration is 2^N. Therefore, it is often the case that sufficient data is not available to learning the conditional probabilities. Techniques for learning parameters for incomplete data in a Bayesian network include Gibbs sampling (a Monte-Carlo method), Gaussian approximation, and the EM algorithm. Monte-Carlo methods give accurate results, but when the sample size is large, they are often intractable (Heckerman 1999). The Gaussian approximation approach is more efficient than Monte-Carlo methods and is accurate for relatively large samples (Heckerman 1999).

N-gram models constitute a special case of dynamic Bayesian networks, and the same learning techniques for Bayesian networks can be used to estimate conditional probabilities. *N*-gram models are Markov models of order n-1 (Murphy 2002). If $n = 2$ then, we have bigram models. Note that n-gram models do not have hidden states. Thus, learning n-gram models is the task of computing conditional probabilities from observational data by counting the number of occurrences of each possible configuration in the observational data.

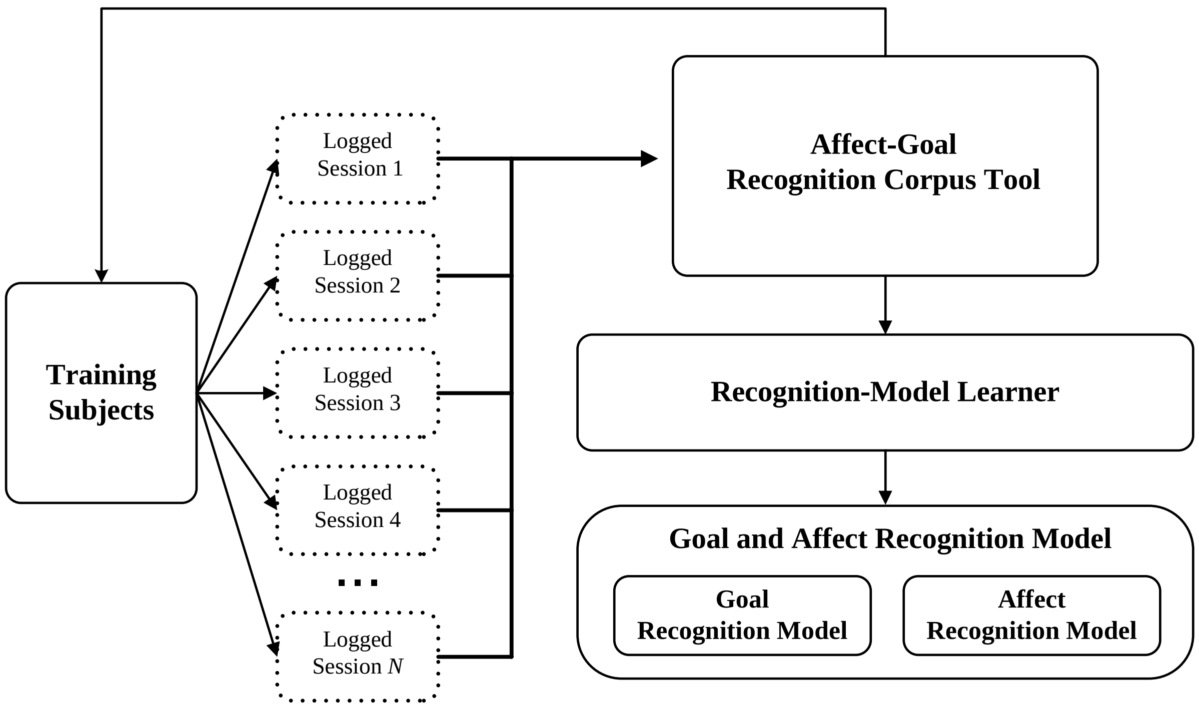


Figure 3.3: Training Corpus Acquisition Data Flow

3.2.1 Corpus Acquisition

To induce goal and affect recognition models, training data can be collected from training subjects interacting with a test bed environment. Figure 3.3 depicts data flow during training data acquisition. An affect-goal recognition corpus tool is responsible for collecting training data. The corpus tool has a goal generator that generates goals that are expected to be achieved by training subjects. In the training phase, training subjects are first situated in an interactive learning environment and given an overview of the kinds of activities they could perform. In order to collect sensor data, training subjects are “wired” to biofeedback equipment, eye-tracking equipment, or other types of sensors. Next, subjects are told how their character could be controlled when they interact with the virtual environment. Then, the goal generator successively gives training subjects goals they are expected to achieve. During the training phase, training subjects are asked to select the emotion from a set of emotions that is most closely related to their own feelings at that particular juncture. Each training instance is labeled with goals and affective states. Training subjects complete each goal, and detailed quest traces are recorded of all sequences of actions, goals, locations, sensor data, physiological changes, and world states. In the following phase, the recognition model learner induces goal and affect recognition models from the collected training data.

As noted above, training data should be as representative as possible of that which will be encountered by users at runtime. Thus, training sessions should satisfy the following requirements.

1. The representation of training data must be sufficiently expressive to support assessment of affective state changes and observations of complex plans that users might follow, and it must be encoded with features that are easily observable at runtime.
2. The training subjects should be faced with goals of various levels of difficulty so that Affect-Goal Recognition Corpus tool can monitor the changes of user affective states and can gather a rich training plan corpus. For example, some goals should be very easy to achieve and others should be very challenging so that training users could exhibit affective states during the interactions. By posing goals of various levels of difficulty, we can collect the training data as representative as possible.
3. The training session should be long enough to produce a large number of records. This allows training subjects to exhibit affective states because users sometime require extended periods of time to become emotionally engaged. The large number of records also enables the model to learn statistically significant probability values for inducing model of prediction user plans.
4. Training data should be recorded at least as often as significant events occur. The events are likely to be significant if they potentially affect user emotions and actions which are critical to executing plans.

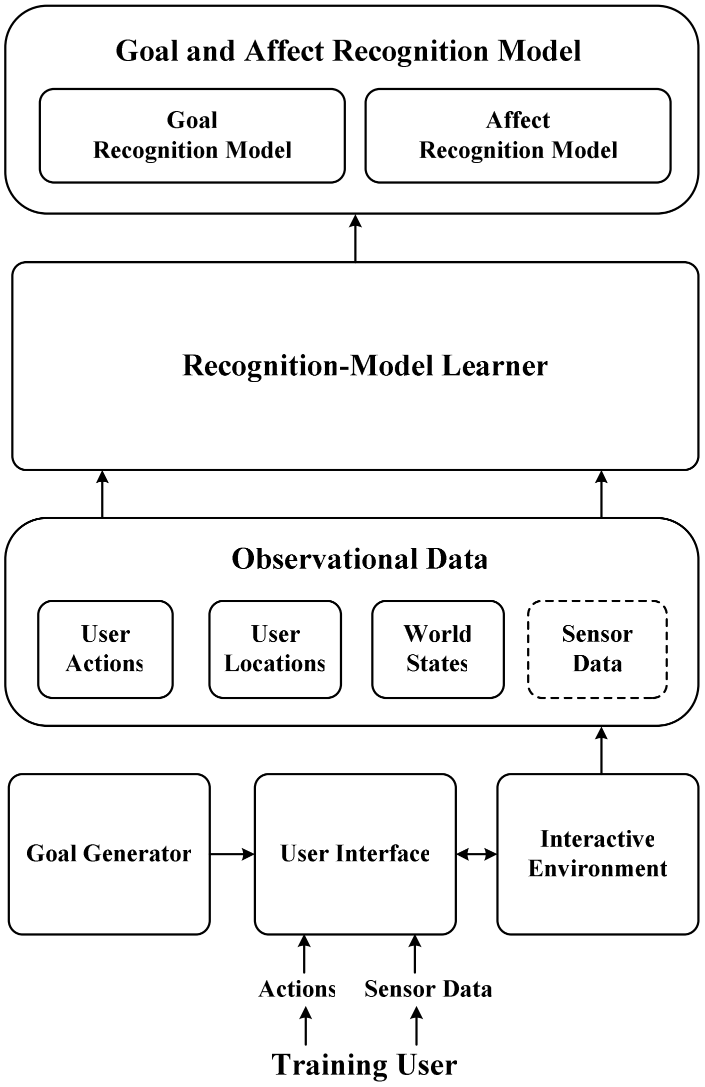


Figure 3.4: Goal and Affect Recognition Model Induction

3.2.2 Model Induction

The data in the training phase yields a set of the observations of user actions, user locations, sensor data, and world states. During model induction, the observational data are passed to the recognition model learner (Figure 3.4). Many types of models can be learned. Possible probabilistic models include *n*-grams, Bayesian networks, and Decision trees. As noted above, if all variables are observable (i.e., there are no hidden variables in the world), then learning conditional probabilities for both models (*n*-gram and Bayesian network) is basically counting the number of occurrences of each configuration. However, because training data are necessarily sparse, i.e., we are unlikely to observe all actions, locations, and goals, *n*-gram and Bayesian models will employ smoothing techniques to reevaluate zero-probability and low-probability.

To avoid zero-probabilities, adding a small number into each cell of sparse conditional probability tables is often employed (Hu 1999). Often called a *flattening constant* (denoted by α), the flattening constant can be added to only empty cells or all cells in the table. After adding α to cells (either only empty cells or all cells), conditional probabilities are recomputed. Different choices of α have been proposed (e.g., adding 1/2 to all cells or adding 1/*D* to empty cells where *D* is the total number of cells).

To summarize, student goal and affect recognizers play a central role in user modeling in narrative-centered learning environments. Providing student models with the ability to make accurate early predictions is particularly important for techniques that are to provide feedback to pedagogical planners in real-time. In this chapter, we introduced an inductive approach to automatically learning goal and affect recognition models rather than manually creating the models. In the following chapters, we present implemented goal and affect recognition models that can make early, accurate predictions. The models have been empirically evaluated in CRYSTAL ISLAND, a narrative-centered learning environment.

Chapter 4

Narrative-Centered Goal Recognition

As with most tasks to be performed in the context of interactive systems, an empirical approach to evaluation holds much appeal. We adopt the following 3-phase evaluation methodology to study the probabilistic goal recognizers introduced above.

1. *Corpus Acquisition*: In a narrative-centered learning environment, collect traces of users' performing narrative quests. Quest traces encode extensive sequences of narrative states and user goals, locations, and actions.
2. *Goal Recognizer Induction*: Learn narrative goal recognizers from the quest traces by using the narrative state sequences, user goals, user location sequences, and user action sequences to induce goal classifiers.
3. *Predictive Recognition Evaluation*: With a cross validation approach, determine the accuracy and the incremental recognition abilities of each goal recognizer.

This methodology has been used to study the unigram, bigram, and Bayesian network narrative goal recognition models. We briefly describe the narrative-centered learning environment testbed in which the experiments were carried out. After discussing the challenges of empirically evaluating affect and goal recognition models, we describe the evaluation methodology we propose to adopt for our work and report the experimental results.

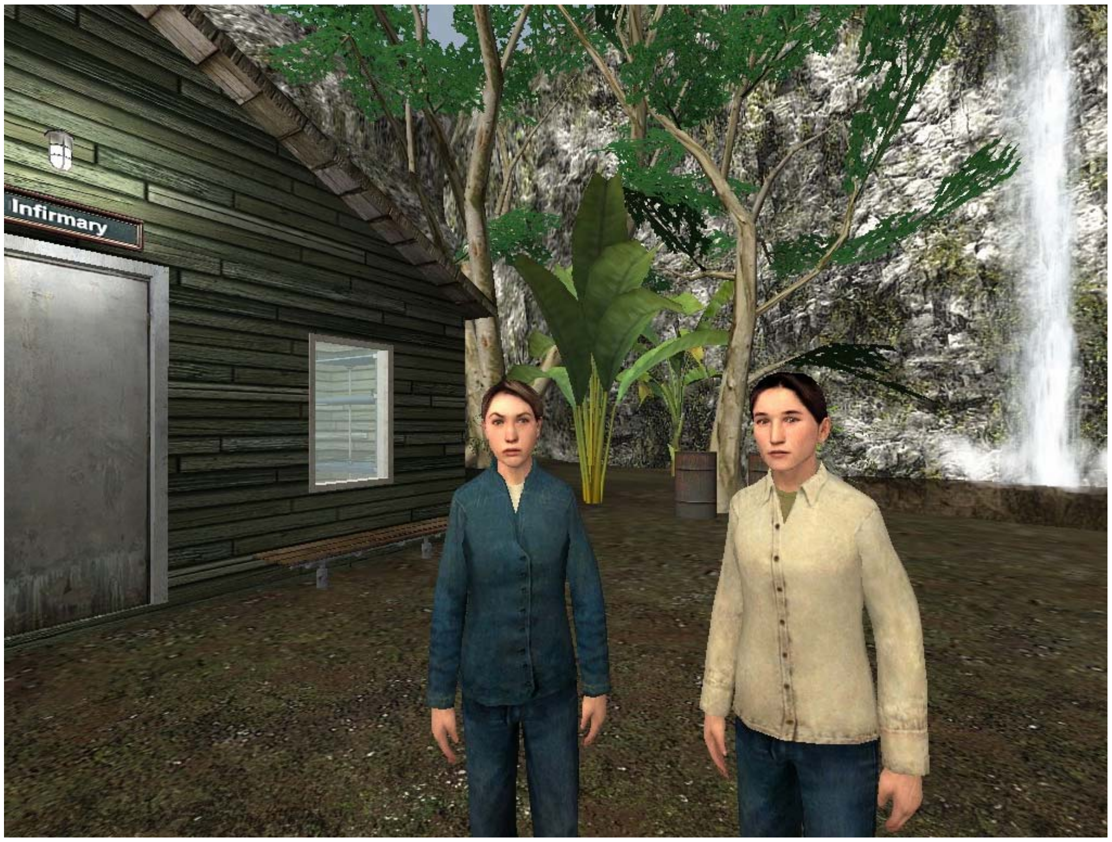


Figure 4.1: The CRYSTAL ISLAND Testbed

4.1 Testbed: Crystal Island

To serve as an effective "laboratory" for studying user goals and affect recognition in a narrative-centered learning environment, a testbed should pose the same kinds of challenges that goal and affect recognition modelers are likely to encounter in future runtime environments. It should offer users a broad range of actions to perform and provide a rich set of tasks and goals in a nontrivial narrative-centered learning environment. The goals should exhibit some complexity, and the environment should be populated by manipulable artifacts and be inhabited by multiple characters. To this end, we have devised CYRSTAL ISLAND (Mott *et al*. 2006b), a narrative-centered learning environment testbed featuring a science mystery (Figure 4.1). The mystery is set on a recently discovered volcanic island where a research station has been established to study the unique flora and fauna. The user finds herself as the daughter of a visiting scientist who is attempting to discover the origins of an unidentified illness at the research station. The environment begins by introducing her to the island and the members of the research team for which her father serves as the lead scientist. As members of the research team fall ill, it is her task to discover the cause of the outbreak. She is free to explore the world to collect physical evidence and interact with other characters. Through the course of her adventure she must gather enough evidence to correctly choose among candidate diagnoses including botulism, cholera, salmonellosis, and tick paralysis as well as identify the source of the disease.

The narrative learning environment of CYRSTAL ISLAND, the semiautonomous characters that inhabit it, and the user interface were implemented with Valve Software's Source™ engine, the 3D game platform for Half-Life 2. The Source engine also provides much of the low-level (reactive) character behavior control. In CYRSTAL ISLAND, the user can perform a broad range of actions including performing experiments in the laboratory, interacting with other characters, reading "virtual books" to obtain background information on diseases, and collecting data about the food recently eaten by the members of the research team. Throughout the mystery, users can walk around the island and visit the infirmary, the lab, the dining hall, and the living quarters of each member of the team. In the current testbed, there are twenty goals the users can achieve, three hundred unique actions the user can carry out, and over fifty unique locations in which the actions can be performed.

4.2 Challenges of Empirical Evaluation

Effectively evaluating the goal and affect recognition framework requires dealing with several challenging issues. First, we need to determine the granularity of the evaluation. Every time a student performs an action, she might or might not change her current goal. Students' goals or affective states could be influenced by other events (e.g., other character's actions) in the environments. We might evaluate the predictive power of the model every time a student performs an action or an important event occurs, or it could be done at larger time intervals. Second, we are presented with the problem of when a student interacts with a system multiple times. When the student first interacts with the system, it is likely that the sequence of student actions is not optimal. However, when the student interacts with the system on subsequent occasions, the actions are more likely to be optimal. Thus, the test data collected from the former case may not be representative of the interactions that repeat users have, nor will test data collected from the latter case be representative of the interactions that first-time users will have. When we evaluate the model, we need to treat two cases differently. Third, an important issue to investigate is how well the recognition system generalizes to different tasks, domains, or populations of students. Because most goal recognition systems have been concerned with domain specific actions and goals and test data are also collected from the interactions with a specific system, it is difficult to judge the generalizability of the model.

For goal and affect recognition, there are no standard data sets that are used. Unlike the machine learning community which has its own repository of data sets, with the possible exception of the Unix domain, there are no similar data sets for evaluating goal recognition, nor are there for affect recognition. Moreover, there is currently no standard, commonly accepted approach to evaluating goal recognition frameworks. As researchers in the goal recognition community have noted, there are no standard evaluation metrics for goal recognition (Blaylock and Allen 2003). This is also true for evaluating affect recognition. Although there is a lack of the agreement on evaluation metrics, some metrics have emerged for evaluating goal recognition in interactive environments (Blaylock and Allen 2003; Blaylock and Allen 2005). Model performance has been evaluated in terms of two criteria: accuracy and convergence. Accuracy measures are concerned with how correctly the model can predict users' goals. Convergence measures are concerned with "early prediction."

Accuracy measures

- *Accuracy*: The number of correct predictions divided by the total number of actions observed.
- *Precision*: the number of correct predictions divided by the total number of predictions made.

Convergence measures

- *Converged*: The percentage of observation sequences in which the plan recognizer's final prediction is correct.
- *Convergence Point*: For observation sequences which converged, the point within the sequence when the plan recognizer started making the correct prediction and continued to make the correct prediction for the remainder of the sequence.
- *Average Actions of Converged*: The average number of actions within observation sequences which converged.

4.3 Narrative-Centered Goal Recognition Study

In this section, we discuss the problem of narrative goal recognition, introduce a solution to the narrative goal recognition problem that utilizes *n*-gram models and Bayesian network models, and present the results of an empirical evaluation. The models were induced from training data acquired from interactive sessions.

4.3.1 The Narrative-Centered Goal Recognition Model

Narrative-centered learning environments dynamically craft engaging story-based experiences for users, who are themselves active participants in unfolding stories. A key challenge posed by interactive narrative is recognizing users' goals so that narrative planners can dynamically orchestrate plot elements and character actions to create rich, customized stories. In this section, we discuss an inductive approach to predicting users' goals by learning probabilistic goal recognition models.

Goal recognition for interactive narrative should satisfy three requirements. First, because incorrectly predicting goals could significantly diminish the effectiveness of narrative planners, narrative goal recognizers should accurately infer users' goals. Moreover, as observations of users' activities become available, recognizers should make accurate "early" predictions (Blaylock and Allen 2003) ideally these would be *k*-best predictions rather than a single predicted goal and they should converge as quickly as possible on the most likely interpretation. Second, the real-time requirements of interactive narrative call for extraordinarily efficient recognizers. Any approach that depends on computations spanning more than a few milliseconds could be infeasible. Third, users in most narrative environments should not be interrogated about their current goal; their actions must speak for themselves. Because interrupting users to pose questions about their goals could interfere with the flow of the narrative and cause users to forgo their suspension of disbelief, a narrative goal recognizer should perform "keyhole goal recognition," i.e., it should unobtrusively observe a user as she interacts with the environment as discussed in Chapter 3.

To address the requirements for narrative goal recognition set forth above (accuracy, incremental recognition, and efficiency), and to cope with the uncertainty inherent in recognizing users' goals in interactive narrative environments, we investigate two families of probabilistic approaches to user goal recognition: *n*-gram models and Bayesian networks.

Narrative goal recognizers can exploit three sources of information to infer users' goals: narrative states, user actions, user locations. Narrative states represent the plot (typically represented in a plot graph (a partially ordered graph of plot elements) (Weyhrauch 1997)) or narrative plan (Riedl *et al*. 2003), the current focus of the story arc and its episodic structure, and the plans and goals of the synthetic agents who serve as (the other) characters in the story. User actions represent the actions that a user can perform (e.g., open the door, talk to a person, examine a book, and etc.), and user locations represent the locations in which actions have been performed.

We define narrative goal recognition as follows: Given a sequence of n observed user actions $a_1, a_2, \ldots, a_n$ in a narrative environment, their associated narrative states $n_1, n_2, \ldots, n_n$ and user locations $l_1, l_2, \ldots, l_n$, identify the most likely goal G^* from a set of candidate goals $g_1, g_2, \ldots, g_m$ that accounts for the action sequence in the given context.

For *n*-gram, narrative goal recognition can be formally defined as follows. Given an observation sequence $O_1, O_2, \ldots, O_n$, the objective of narrative goal recognition is to identify the most likely goal G^* such that:

$$\begin{aligned} G^* &= \arg\max P(G \mid O_1, O_2, O_3, \ldots, O_n) \\ &= \arg\max P(G \mid O_{1:n}) \end{aligned}$$

where each O_i is an observation encoding the current narrative state, the user's action, and the location at which her action was performed. The observation sequence O_1, O_2, …, O_n is denoted by $O_{1:n}$. Applying Bayes' rule yields:

$$G^* = \arg\max \frac{P(O_{1:n} \mid G)P(G)}{P(O_{1:n})}$$

which can be simplified by eliminating the constant term $\mathrm{P}(O_{1:n})$ to obtain:

$$G^* = \arg\max P(O_{1:n} \mid G)P(G)$$

Applying the Chain Rule, the equation becomes:

$$\begin{aligned} G^* = \arg\max P(O_n \mid O_{1:n-1}, G)P(O_{n-1} \mid O_{1:n-2}, G) \\ \cdot P(O_{n-2} \mid O_{1:n-3}, G)\ldots P(O_1 \mid G)P(G) \end{aligned}$$

However, estimating these conditional probabilities is impractical and it would require exponentially large training data sets, so we make a Markov assumption that an observation O_i depends only on the goal G and a limited window of preceding observations. Following an approach initially proposed for goal recognition in natural language dialogue (Blaylock and Allen 2003), we explore two *n*-gram narrative goal recognition models, a unigram model and a bigram model. The unigram model is based on the assumption that, given the goal G, O_i is conditionally independent of all other observations. Thus, the goal recognition formula for the unigram model can be simplified to:

$$G^* = \arg\max P(G)\prod_{i=1}^{n} P(O_i \mid G)$$

The bigram model is based on the assumption that, given the goal G and the preceding observation O_{i-1}, O_i is conditionally independent of all other observations. Thus, the goal recognition formula for the bigram model can be simplified to:

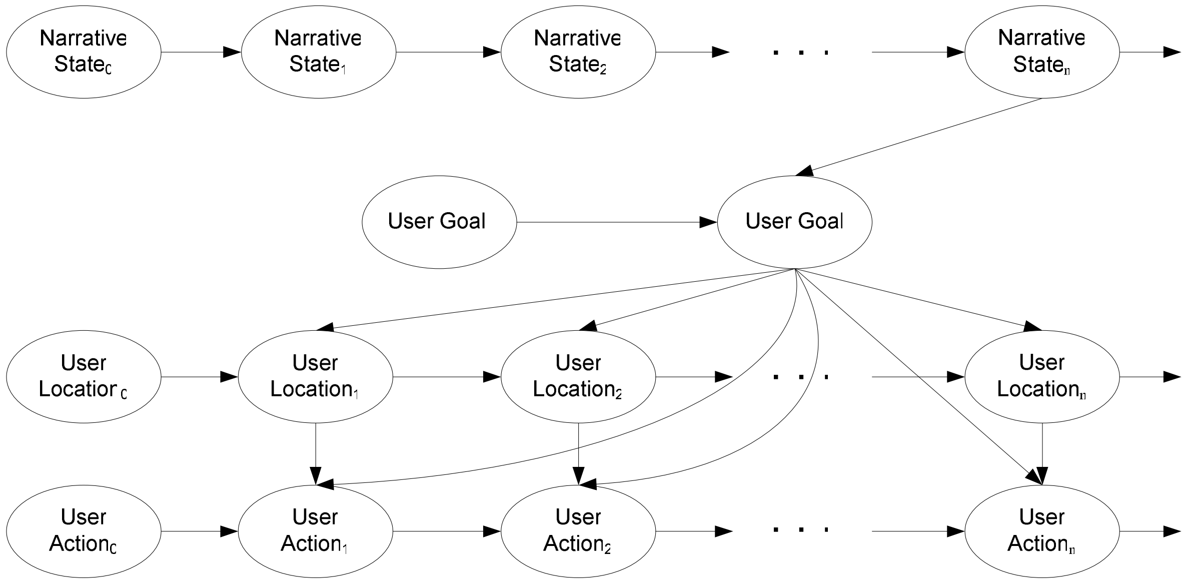


Figure 4.2: Bayesian Network Goal Recognition Model

$$G^* = \arg\max P(G)\prod_{i=1}^{n} P(O_i \mid O_{i-1}, G)$$

For the bigram model, O_0 is taken to be the null observation when a narrative begins and the previous narrative state, user action, and user location are all null. The resulting formulae for the unigram and bigram models are very efficient because updating the goal prediction for each new observation only requires computing the product of the probability returned by the previous prediction and the current conditional probability.

Narrative goal recognition can alternatively be modeled with Bayesian networks. Following an approach initially proposed for keyhole plan recognition in a text-based adventure game (Albrecht *et al*. 1998), we explore a Bayesian network model for narrative goal recognition (Figure 4.2). In contrast to the aggregate observation variables O_i of the *n*-gram models, the Bayesian network goal recognizer explicitly models dependencies between the constituent variables, i.e., between narrative state, user action, and user location. Thus, it represents the influences of the following variables on the user's goal *G*: the user's previous goal *G'*, the sequence of narrative states n_1, n_2, …, n_n, the sequence of user actions a_1, a_2, …,

a_n, and the sequence of user locations l_1, l_2, …, l_n. While the narrative states influence G because the plot elements and story arc affect the goals of the user in the story, G itself influences the locations where the user performs her actions, as well as the actions that she performs there. The locations also influence the user actions directly. Locations are modeled as influencing actions because particular locations afford particular types of actions. However, the converse model in which actions influence locations is also plausible because a user might journey to a location in the storyworld to perform an intended action there. Since Bayes nets are by definition acyclic, one of these directions of causality must be selected. While the former is chosen here, the latter offers an interesting direction for future work. As with the bigram model, N_0, A_0, and L_0 are taken be null when a narrative begins.

From the Bayesian network, we have the following

$$G^* = \arg\max P(G \mid G', A_{0:n}, L_{0:n}, N_{0:n})$$

Given a sequence of actions A_0, A_1, A_2, …, A_n, a sequence of locations L_0, L_1, L_2, …, L_n, and a sequence of narrative states N_0, N_1, N_2, …, N_n, the current action An depends only on the previous action $A_{n\text{-}1}$, the current location L_n, and the current goal G; the current location L_n depends only on the previous location $L_{n\text{-}1}$ and the current goal G; the current narrative state N_n depends only on the previous narrative state $N_{n\text{-}1}$; and the current goal G only depends on the previous goal *G'* and the current narrative state N_n. Therefore:

$$\begin{aligned}
&P(A_n \mid G, G', A_{0:n-1}, L_{0:n}, N_{0:n}) = P(A_n \mid G, A_{n-1}, L_n)\\
&P(L_n \mid G, G', A_{0:n-1}, L_{0:n-1}, N_{0:n}) = P(L_n \mid G, L_{n-1})\\
&P(N_n \mid G', A_{0:n-1}, L_{0:n-1}, N_{0:n-1}) = P(N_n \mid N_{n-1})\\
&P(G \mid G', A_{0:n-1}, L_{0:n-1}, N_{0:n}) = P(G \mid G', N_n)
\end{aligned}$$

where $n \geq 1$. By applying the Chain Rule and the above equations, the goal recognition formula for the Bayesian network becomes:

$$\begin{aligned} G^* &= \arg\max P(G \mid G', A_{0:n}, L_{0:n}, N_{0:n}) \\ &= \arg\max P(G')P(G \mid G', N_n)P(A_0)P(L_0)P(N_0) \\ &\quad \cdot \prod_{i=i}^{n} P(A_i \mid G, A_{i-1}, L_i)P(L_i \mid G, L_{i-1})P(N_i \mid N_{i-1}) \end{aligned}$$

As with the *n*-gram models, the resulting formula is very efficient because updating the goal prediction for each new observation requires only noting the previous prediction, looking up a few CPT entries, and computing their product. During training, we estimate P(A_i|G, $A_{i\text{-}1}$, L_i), P(L_i|G, $L_{i\text{-}1}$), P(N_i|$N_{i\text{-}1}$), P(G|G', N_i), and P(G) using training data acquired with a narrative environment as described in the below.

4.3.2 Narrative Goal Recognition Model Induction

In a model induction, more than fifteen hours of narrative trace data were collected from forty subjects interacting with the testbed environment. Subjects were first situated in the narrative world and given an overview of the kinds of activities they could perform. Next, they were told how their character could be controlled, and once they entered the virtual environment, they were successively given goals they were expected to achieve via an onscreen message. Based on the narrative structure, the order of the goals differed from session to session. Users completed each goal, and upon completing the final goal, they were complimented on their successful performance.

In the current testbed, there are 20 goals that user can achieve, three hundreds unique actions the user can perform, over fifty unique locations in which actions can be performed. Detailed quest traces were recorded of all sequences of actions, goals, locations, and narrative states. Narrative states were represented with the episodic structure of the unfolding story and the narrative arc in which it was situated. There were eighty training sessions collected (two sessions per subject), which generated just over twenty thousand training records. The number of training records was high because of the frequency of sampling and the length of action sequences per goal. Unigram, bigram, and Bayesian network goal recognition models were learned from the collected quest traces. For the Bayesian network model, the structure was fixed (as shown in Figure 4.2), but the

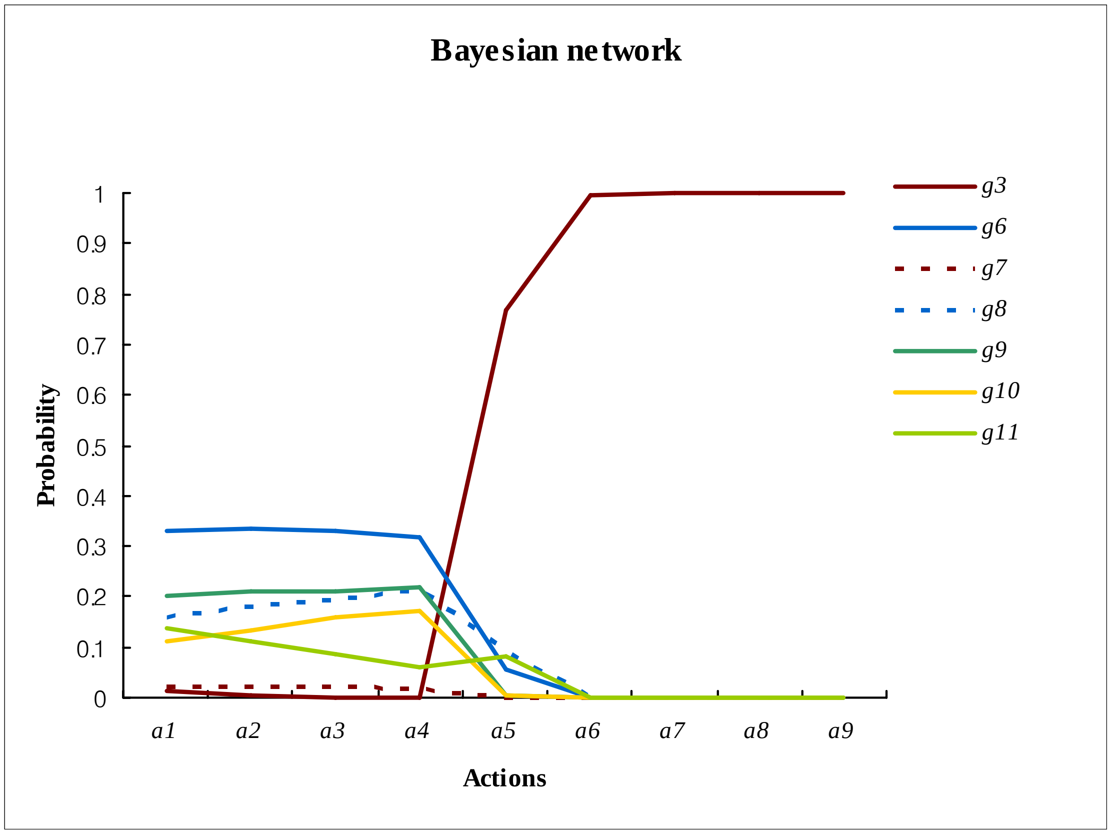


Figure 4.3: Bayesian Network Convergence Graph

conditional probabilities were learned. Because of the high dimensionality of collected data, we add a flattening constant into all cells to avoid zero probabilities.

4.3.3 Empirical Evaluation Results

In evaluation of narrative goal recognition models, we studied several properties of accuracy and convergence. Unigram, bigram, and Bayesian goal recognition models were evaluated using the following criteria (Blaylock and Allen 2003): accuracy, converged, convergence point, and average actions of converged. The induced models were tested using a 10-fold cross validation. (In each fold, nine segments are used for training and one, which was not used for training, is used for testing.) The results are shown in Table 4-1. Each of the models performed at a reasonable level. Although the 51% to 54% accuracy may at first appear low, the recognizers performed significantly better than chance, which would be 5%.

(There were 20 candidate goals.) The resulting accuracies are promising and are consistent with the results seen in other statistical approaches, e.g., (Blaylock and Allen 2003). In addition to accuracy, the models also exhibited the ability to make early predictions, i.e., predictions based on a few observations, and to converge reasonably quickly to the correct goal. Figure 4.3 shows the top seven predictions of the Bayesian goal recognition model as it converges to g3 after five actions. Because all of the models are probabilistic, they can provide the *k*-best predictions.

Table 4-1: Goal Recognition Evaluation Results

	Unigram	**Bigram**	**Bayesian**
Accuracy	54.8 %	51.5 %	53.7%
Converged	83.7 %	79.3 %	67.2%
Convergence Point	50.5 %	48.5 %	43.9%
Average Actions of Converged	16.3	16.9	17.1

All three models show approximately similar performance. For accuracy, unigram is significantly different from bigram and unigram is different from Bayesian with a small significance. For converged, all three models are significantly different from one another. However, there is no significant difference between all three models for convergence point.

In this chapter, we have presented a probabilistic goal recognizer that uses machine learning techniques to induce a goal recognition model from an activity corpus gathered from users' interactions with the CRYSTAL ISLAND learning environment. Both the prior and conditional probabilities are learned from the corpus. The recognizer is fast and efficient, and it supports "early" prediction. The evaluated recognizer represents a first step towards goal recognition for interactive narrative environments. In the next chapter, we describe an implemented probabilistic affect recognizer and report its performance on training data gathered in the CRYSTAL ISLAND learning environment.

Chapter 5

Narrative-Centered Affect Recognition

Recent years have seen a growing recognition of the importance of affective reasoning in human-computer interaction. Foundational work on affect in intelligent tutoring systems has yielded advances in affective student modeling (Conati and Mclaren 2005), detecting frustration and stress (Burleson and Picard 2004; McQuiggan *et al*. 2007; Prendinger and Ishizuka 2005), gauging student motivation (de Vicente and Pain 2002), and modeling students' levels of self-efficacy (Beal and Lee 2005; McQuiggan and Lester 2006a). Complementing these results are efforts to devise affect-based models of social interaction for virtual agents focusing on politeness (Johnson and Rizzo 2004; Porayska-Pomsta and Pain 2004) and empathy (McQuiggan and Lester 2006b; Paiva *et al*. 2005), as well as techniques for modeling their emotional states (André and Muller 2003; Gratch and Marsella 2004). Collectively, this work seeks to increase the effectiveness of user experiences by recognizing user affect and supporting more productive and enjoyable affect-informed interactions.

Affect recognition is the task of identifying the emotional state of a user from a variety of physical cues, which are produced in response to affective changes in the individual. These include visually observable cues such as body and head posture, facial expressions, and posture, and changes in physiological signals such as heart rate, skin conductivity, temperature, and respiration (Frijda 1986). However, because affect is fundamentally a cognitive process in which the user appraises the relationship between herself and her environment (Gratch and Marsella 2004; Smith and Lazarus 1990), affect recognition models should take into account both physiological and environmental information. For narrative-centered learning environments, affect recognition models can leverage knowledge of task structure and user goals to effectively reason about users' affective states. In particular, for narrative-centered learning environments, affect recognition models can use appraisal theory (Lazarus 1991) to recognize users' emotions generated in response to their assessment of how their actions and events in the environment relate to their goals.

To effectively recognize affect states in narrative-centered learning environments, affect recognition models should satisfy two key requirements. First, they should provide accurate predictions of users' affective states. Incorrectly predicting affective states could jeopardize the performance of affect-informed systems, so affect recognition models should correctly classify users' most probable affective states. As observations of users' activities become available, recognizers should make accurate "early" predictions. Second, they should be highly efficient. Because users' affective states can quickly change, particularly when interacting in highly dynamic environments, affect recognition models should be realized in computational frameworks that address the real-time demands of interactive systems.

In this chapter, we present an inductive approach to recognizing users' affective states in narrative-centered learning environments by learning affect recognition models. The models, which exploit task structure as well as physiological and environmental information, are induced from training data acquired from traces of users performing tasks in rich virtual environments. We report on an empirical evaluation of induced affect recognition models in a narrative-centered learning environment in which users solve a science mystery in the domain of microbiology. Experimental results suggest that induced models can accurately predict users' affect states, and they are sufficiently efficient to meet the real-time performance requirements of narrative-centered learning environments.

The chapter is structured as follows. First, it provides the inductive approach to affect recognition models in Section 1. Section 2 reports on an evaluation of prediction of six affective states. In section 3, we report the results of prediction of student frustration.

5.1 Inducing Affect Recognition Models

Affect recognition work has explored emotion classification from self reports (Beal and Lee 2005), post-hoc reports (de Vicente and Pain 2002), physiological signals (Conati and Mclaren 2005; Picard *et al*. 2001; Prendinger and Ishizuka 2005), combinations of visual cues and physiological signals (Burleson and Picard 2004; Kliensmith *et al*. 2005), and from world state feature representations of temporal, locational and intentional information (McQuiggan *et al*. 2006). This body of research serves as the springboard for the work described in this chapter, which reports on techniques for recognizing users' affective states from both physiological and task structure information. This work focuses on learning models of affect recognition that are grounded in appraisal theory for rich, narrative-centered learning environments.

The family of affect recognition models we explore are founded on appraisal theory (Lazarus 1991), which offers a motivational-emotive account of human cognition. Of particular interest here are the notions of appraisal and coping. Appraisal refers to the assessment of one's relationship with the surrounding environment, taking into account constructs such as goals and plans. Coping refers to one's manipulation of the environment to promote change in this relationship or to maintain it. Coping outcomes may take the form of external actions and behaviors that change the physical environment or internal cognitive transformations (e.g., modification of goals or plans) that result in changes in the next appraisal. This continuous cycle of appraisal-coping is clearly evident in narrative-centered learning environments where users constantly assess the relationship between their previous, current, and planned actions and their goals. Thus, being able to represent knowledge related to actions and goals in the virtual environment should allow computational mechanisms to accurately model user appraisal and coping in narrative-centered learning environments.

Further, since appraisals often result in internalized affective states, such computational mechanisms should contribute directly to affect recognition.

5.1.1 Training

Users interacting in narrative-centered learning environments naturally form goals. In the case of some environments such as the test bed environment presented in Section 4.1, goals are explicitly given by the environment. Users perform actions to achieve these goals while continually appraising the relationship between their action sequences (past, present and future actions) and the goal. Often these appraisals result in emotional states. Thus, affect recognition models can exploit several sources of information to infer users' affective states. The affect recognition model learner employs an expressive representation of the events occurring in the task-oriented environment by monitoring observable attributes pertaining to the following, interrelated categories:

- *User Actions*: Affect recognition models can observe users' actions in the world and their relationship to achieving particular goals; affect recognition models also have access to auxiliary information about the interactions, e.g., any artifacts manipulated such as which objects have been picked up or which doors have been opened, as well as the characters with which users have interacted.
- *User Locations*: Affect recognition models have access to a variety of information pertaining to the users' precise location in the environment and the location's relationship to achieving particular goals. For example, some goals may only be achievable in exact locations, while other actions and goal achievements may occur anywhere in the environment.
- *Temporal Information*: Affect recognition models can observe the time users' spend on task, the time spent in particular abstract locations (e.g., particular rooms of the environment), and the time carrying out particular actions.
- *Task Structure*: Affect recognition models can observe the users' task progression, i.e., whether the user is completing actions that will or will not help achieve certain goals. The affect recognition model also has access to knowledge of the explicitly stated goal

in the training environment. At runtime, this knowledge would be acquired through goal recognition mechanisms, e.g., (Blaylock and Allen 2003; Mott *et al*. 2006a). In the case of the CRYSTAL ISLAND narrative-centered learning environment, goals are explicitly communicated to users.

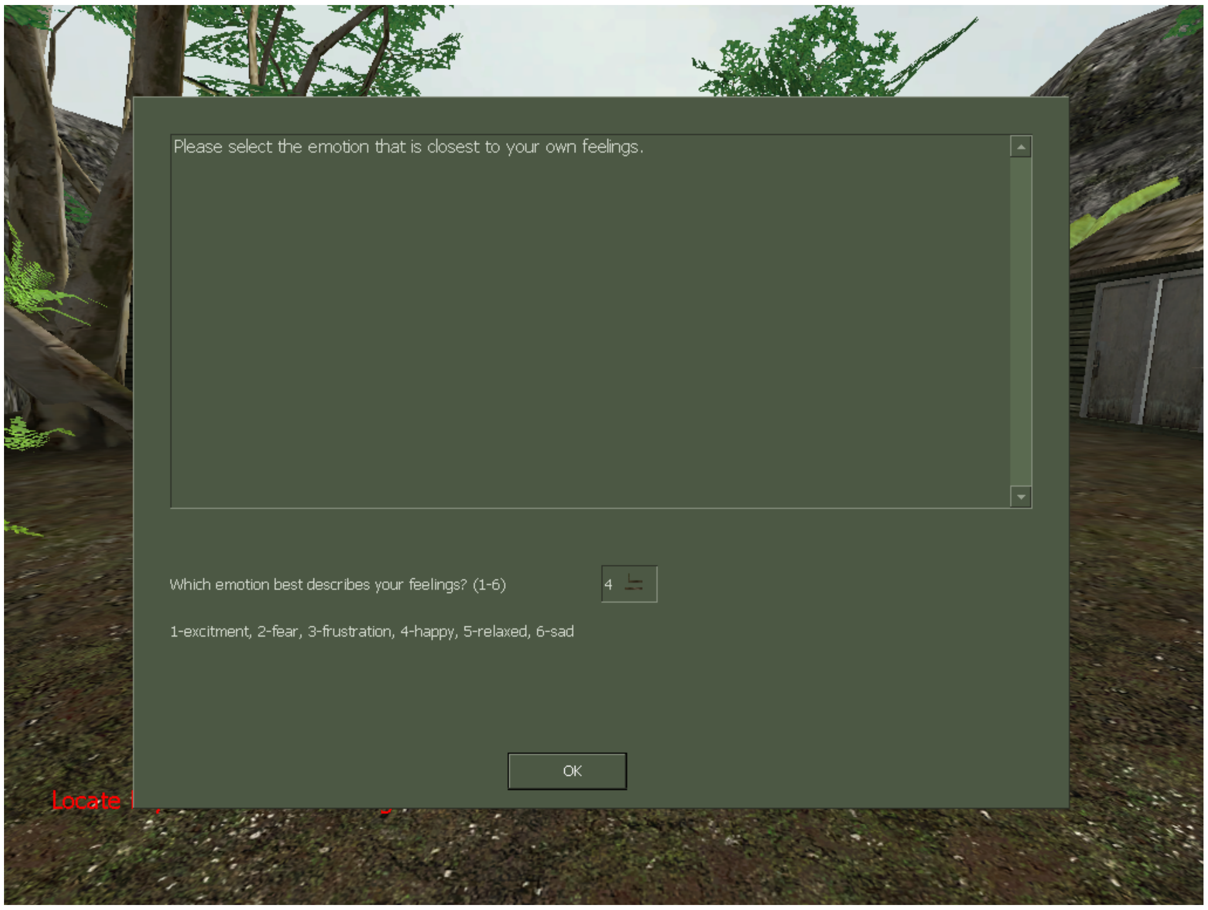
- *Physiological Response*: Affect recognition models can observe users' physiological changes in response to events in the environment, such as carrying out an action, goal achievement, or interacting with a particular agent in the environment. Because physiological responses are directly triggered by changes in affect, biofeedback data streams such as heart rate and galvanic skin response can be useful for inferring user affect.

To address the requirements for affect recognition set forth above (accuracy and efficiency), we investigate an inductive approach to generating user affect recognition models.

5.1.2 Learning

Many types of affect recognition models can be learned. Work to date has investigated two families: rule-based models (decision trees) and probabilistic models (naïve Bayes). Naïve Bayes and decision tree classifiers are effective machine learning techniques for generating preliminary predictive models. They are efficient mechanisms for embedding into runtime environments. Naïve Bayes classification approaches produce probability tables that can be incorporated into runtime systems and used to continually update probabilities for predicting user affective states. Decision trees provide interpretable rules that support runtime decision control components. With both naïve Bayes and decision tree classifiers, runtime control components can efficiently monitor the state of observable attributes in the probability tables (for naïve Bayes) or rules (for decision trees) to determine when conditions are met for predicting particular affective states (e.g., happiness or frustration). Both naïve Bayes and decision tree classification techniques are useful for preliminary predictive model induction for large multidimensional data, such as the observational attribute vector used here to induce affect recognition models. Two approaches can be distinguished in learning techniques: those that are completely automated, and those that require the knowledge provided by a domain expert. The experiment reported below focuses on fully automated learning approaches. Model induction proceeds in four phases:

- *Data Construction*: Each training log is first translated into a full observational attribute vector. For example, blood volume pulse (BVP) and galvanic skin response (GSR) readings were taken nearly 30 times every second reflecting changes in both heart rate and skin conductivity. Attributes observed directly from the environment were combined with physiological response attributes and self-reported affective states. While self-reporting mechanisms can often be intrusive and obtaining information regarding user affect is often difficult and can be unreliable inducing models of self-reported affect removes the requirement of the self-reporting mechanism in the runtime environment.
- *Data Cleansing*: First, data are converted into an attribute vector format. Second, a dataset is generated that contains only instances in which the biofeedback equipment was able to successfully monitor BVP and GSR throughout the entire learning session. For example, data from two sessions had to be removed for this reason: BVP (used for monitoring heart rate) readings were difficult to obtain from these participants.
- *Naïve Bayes Classifier and Decision Tree Learning*: Once the dataset is prepared, it is passed to the learning systems. The affect data were loaded into the WEKA machine learning tool (Witten and Frank 2005), a naïve Bayes classifier and decision tree were learned, and tenfold cross-validation analyses were run on the resulting models. The entire dataset was used to generate several types of affect recognition models. These included models that considered different sets of observed attributes (e.g., datasets with and without goal knowledge).


Please select the emotion that is closest to your own feelings.

Which emotion best describes your feelings? (1-6) 4

1-excitment, 2-fear, 3-frustration, 4-happy, 5-relaxed, 6-sad

Figure 5.1: Self Report Emotion Dialog

5.2 Evaluation

In a formal evaluation, data was gathered from thirty-six subjects. There were 5 female and 31 male participants varying in age, race, and marriage status. Approximately 44% of the participants were Asian, 50% were Caucasian, and 6% were of other ethnicities. Participants' average age was 26.0 (SD=5.4).

After filling out a consent form and demographic survey, participants began training sessions by first completing a practice task. The practice task allowed them to become familiar with the keyboard and mouse controls as well as interacting in a 3D virtual environment. Following the practice task, participants were presented a controlled backstory for CRYSTAL ISLAND situating them on the island and providing details about their task. For reference, participants had access to a cast of agents found in the CRYSTAL ISLAND environment as well as an overview map. Participants then interacted with the environment to solve the science mystery. The training testbed provided them with specific goals to focus on, guiding them through the solution to the mystery. Periodically a “self-report emotion dialog” box would appear after significant events (talking to characters, testing objects) and every 75 seconds in between. Participants were asked to select the emotion, from a set of six emotions (excitement, fear, frustration, happiness, relaxation, and sadness), that was most closely related to their own feelings at that particular juncture (Figure 5.1). This set of emotions was chosen to effectively cover the affect space so that most subjects would easily be able to relate their feelings during interaction to one of the six affective states. In addition to periodic reports, participants had the ability to trigger the self-report emotion dialog if they felt compelled to report a change in their affective state. This functionality proved to be used sparingly. After solving the science mystery participants completed a post-experiment survey before exiting the training session.

5.2.1 Results

Both naïve Bayes and decision tree models were induced from data collected in the training sessions described above. Models were evaluated using tenfold cross-validation (Witten and Frank 2005). Table 5-1 below reports the accuracy of naïve Bayes and decision tree affect recognition models. The percentages refer to correctly classified instances. The highest performing model is a decision tree affect recognition model induced from representations of user actions, locations, task structure, temporal information and physiological response. The model accuracy is significantly better than a decision tree model induced from representations of user actions, locations, task structure and temporal information (likelihood ratio, $\chi^2 = 379.247$, $p < 0.0001$*, and Pearson, $\chi^2 = 374.311$, $p < 0.0001$*).

Table 5-1: Affect Recognition Classification Results

	Physiological Data Only	Goals, Actions, Locations, Temporal Information & Task Structure	All Attributes
Decision Tree	56.39 %	95.39 %	97.42%
Naïve Bayes	38.48 %	52.18 %	55.79%

Tables 5-2 and 5-3 report the early prediction results of naïve Bayes and decision tree affect recognition models. The decision tree model induced from all attributes converged after 12.63% of observations contained in a defined window. The decision tree model induced from representation of user actions, locations, temporal information and task structure has 90.18% of the convergence rate and it is not significantly better than the decision tree model induced from all attribute (89.20 % of the convergence rate). Since participants choose from a selection of six affective states chance is 16.7%. An additional baseline to consider is selecting the most common affective state, frustration, which appeared in 34.4% of self-reported affective states.

Table 5-2: Early Prediction Results for Naïve Bayes

Naïve Bayes	Physiological Data Only	Goals, Actions, Locations, Temporal Information & Task Structure	All Attributes
Converged	34.66%	50.41%	52.92%
Convergence Point	59.13%	34.33%	42.45%
Average Observations of Converged	56.04	52.40	53.27

Table 5-3: Early Prediction Results for Decision Tree

Decision Tree	Physiological Data Only	Goals, Actions, Locations, Temporal Information & Task Structure	All Attributes
Converged	46.51%	90.18%	89.20%
Convergence Point	78.31%	11.92%	12.63%
Average Observations of Converged	52.36	51.84	51.91

Table 5-4 drills down to further analyze the performance of the best affect recognition model reporting precision and recall analysis of individual affective states. *Precision* refers to the percentage of instances correctly classified as particular value of all instances classified as the same value. For instance, more than 97% of the instances recognized as "happiness" were actually instances in which the model predicted an affective state of "happiness." *Recall* refers to the percentage of instances correctly classified as a particular value of all instances recorded to be the same value (e.g., the percentage of instances correctly classified as "frustration" that were actually reported as "frustration" in self-reports during training sessions.)

Table 5-4: Precision and Recall Analysis for the Decision Tree Affect Recognition Model (Induced from Non-physiological Data).

	Precision	Recall
Excitement	0.963	0.968
Fear	0.966	0.957
Frustration	0.977	0.982
Happiness	0.976	0.963
Relaxation	0.977	0.977
Sadness	0.978	0.95

The results suggest that an approach affect recognition based on appraisal theory can be effective in narrative-centered learning environments, and that representations of user action, location, task structure and temporal information can be used to realize it in a computational model. The performance of decision tree models likely stems from an apparently broad and deep relationship between self-reported affective states and appraisal factors that were explicitly represented in the model. The affect recognition models reported on here seem to be able to capture the relationship between user actions and goals that are assessed during users' appraisal periods.

The CRYSTAL ISLAND narrative-centered learning environment was specifically designed to feature goals with many possible action sequences to achieve the goal. Particular goals were designed to be challenging, i.e., certain artifacts were difficult to find in the environment and even when found required problem-solving skills to access the object (e.g., having to climb on boxes to reach a book on the top of a high shelf). A variety of tasks and goals ranging from easy to difficult to complete were presented to each participant. The varying degree of difficulty seemed to elicit a wider range of emotions. Many participants began difficult tasks in positive states (e.g., happy, relaxed) and found themselves feeling frustrated or sad after spending a significant amount of time on particular goal without making progress.

5.3 Early Prediction of Student Frustration

Frustration occurs when something or someone impedes a student's progress towards a particular goal. As an emotional response, frustration is not fundamentally different from another negative affective response common to a variety of situations, anxiety. Anxiety is often more than merely an emotional response; it also consists of behavioral, cognitive, and physiological responses (Seligman *et al.* 2001). However, since our work focuses on interactive narrative-centered learning environments (Mott *et al.* 2006b), where the construction and achievement of goals is critical to student learning episodes, we also consider frustration. Both anxiety and frustration can lead students to fixate on the impeding source of frustration, diverting attention from, and in some cases causing students to ignore, the task at hand (Goleman 2001). Anxiety particularly arises when students affectively respond to their focus on planning contingencies for potential future events. Detecting situations that will likely lead to student anxiety or frustration that in turn may eventually lead to student impasses would allow learning environments to intervene early, i.e., before the emotion is fully realized as the student approaches her threshold for the particular emotion.

Several strategies can be employed to identify levels of anxiety and frustration that are not detrimental to learning. Setting realistic expectations based on a student's abilities and observed past performance can contribute to student successes. Encouragement, and specific feedback directed at particular behaviors, not merely global performance assessments, may help motivate students and provide them with guidance so that they can improve their self-assessment and help them cope with frustration and anxiety (Ormrod 2002). The central questions that must be answered are, "How can we detect and monitor anxiety and frustration levels so that our learning environments have sufficient time to plan and execute appropriate scaffolding?" and, "With what computational mechanisms can we draw inferences about the student, the task, and the environment to accurately predict student frustration?"

5.3.1 Modeling Frustration

To create models that make accurate predictions of student frustration as early as possible, we first collect training data by observing students interacting with an intelligent tutoring system. From this training data, we then induce *n*-gram models to make early predictions of student frustration. *N*-gram models are useful for early prediction because they are induced from sequences of observations, making predictions with each new observation until they arrive at the final observation of the sequence. In the final observation, concrete evidence of student affect (used as the class label) is obtained. Each prediction from an *n*-gram model is attempting to determine the affective state of the student recorded in this final observation of the sequence. In many cases, *n*-gram model predictions will converge on the correct affective state early in a sequence of observations. The point at which an *n*-gram model first begins making the correct prediction and then continues to make a correct prediction for the remainder of the sequence is known as the convergence point.

While sequential models such as *n*-grams allow us to make early predictions, they are not computationally well suited to large multidimensional data. To address this issue, we investigate three non-sequential modeling techniques: naïve Bayes and decision trees.

5.3.1.1 N-gram models for Early Prediction of Frustration

Given an observation sequence O_1, O_2, …, O_n, the objective of affect recognition is to identify the student's most likely affective state E^* (i.e., frustrated or not frustrated) such that:

$$\begin{aligned} E^* &= \arg\max P(E \mid O_1, O_2, O_3, \ldots, O_n) \\ &= \arg\max P(E \mid O_{1:n}) \end{aligned}$$

where each O_i is an observation encoding the user's goals, user's action, the location at which action was performed, and physiological responses such as heart rate and galvanic skin responses. The observation sequence O_1, O_2, …, O_n, is denoted by $O_{1:n}$. Applying Bayes rule and the Chain Rule, the equation becomes:

$$\begin{aligned} E^* = \arg\max\, & P(O_n \mid O_{1:n-1}, E) P(O_{n-1} \mid O_{1:n-2}, E) \\ & \cdot P(O_{n-2} \mid O_{1:n-3}, E) \ldots P(O_1 \mid E) P(E) \end{aligned}$$

However, estimating these conditional probabilities is impractical – it would require exponentially large training data sets – so we make a Markov assumption that an observation Oi depends only on the affective state *E* and a limited window of the preceding observations.

We explore two *n*-gram affect recognition models for detecting student frustration, a unigram model and a bigram model. The unigram model is based on the assumption that, given the affective state *E*, O_i is conditionally independent of all other observations. Thus, the affect recognition formula for the unigram model can be simplified to:

$$E^* = \arg\max P(E) \prod_{i=1}^{n} P(O_i \mid E)$$

The bigram model is based on the assumption that, given the affective state E and the preceding observation O_{i-1}, O_i is conditionally independent of all other observations. Thus, the affect recognition formula for the bigram model can be simplified to:

$$E^* = \arg\max P(E)\prod_{i=1}^{n} P(O_i \mid O_{i-1}, E)$$

The resulting formulae for the unigram and bigram models are very efficient because updating the affect prediction for each new observation only requires computing the product of the probability returned by the previous prediction and the current conditional probability.

During training, we estimate P(E), P(O_i|E), and P(O_i,| O_{i-1}, E) using training data acquired with an interactive learning environment as described below. Because training data is necessarily sparse, i.e., we are unlikely to observe all possible combinations of actions, locations, goals, and physiological response levels, the unigram and bigram models employ a standard smoothing technique (a flattening constant and simple Good-Turing frequency estimation (Gale and Sampson 1995) to re-evaluate zero-probability and low-probability *n*-grams.

5.3.1.2 Naïve Bayes and Decision Tree for Modeling Frustration

Naïve Bayes and decision trees are effective machine learning techniques for generating preliminary predictive models. Bayes classification approaches produce probability tables that can be implemented in runtime systems and used to continually update probabilities for predicting student affective states, and, in the approach proposed here, for predicting whether students are frustrated or not. Decision trees provide interpretable rules that support runtime decision making. The runtime system monitors the condition of the attributes in the rules to determine when conditions are met for diagnosing particular student emotions.

These classification techniques are particularly useful for inducing models with large multidimensional data, such as the data gathered in the user study described below. Because it is unclear precisely which runtime variables are likely to be the most predictive, naïve Bayes and decision tree modeling provide useful analyses that can inform more expressive machine learning techniques (e.g., Bayesian networks) that also leverage domain experts'

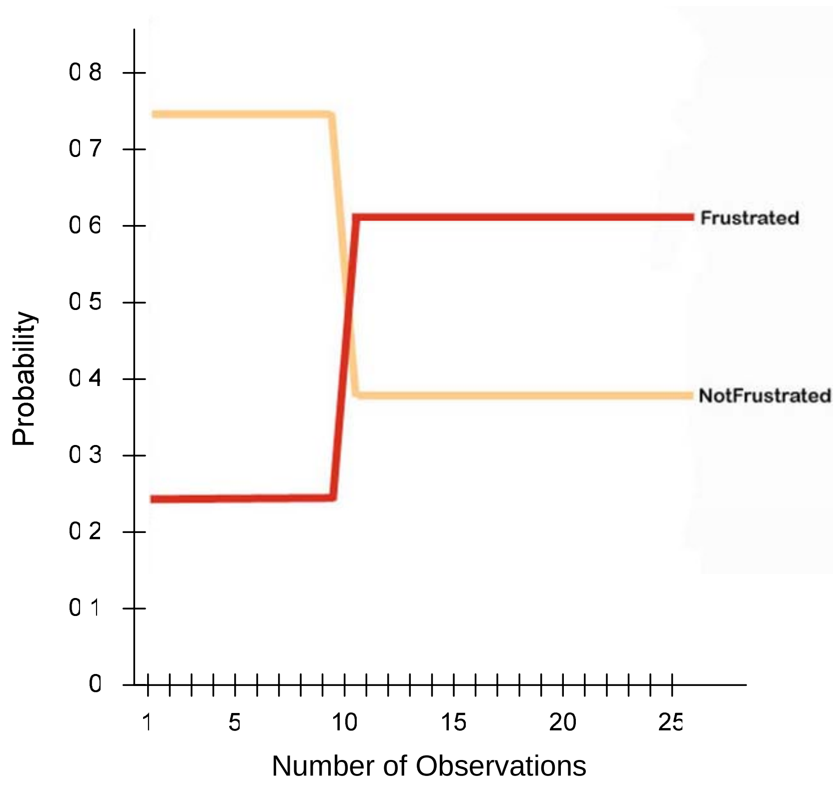


Figure 5.2: Bigram Convergence Graph

knowledge. We have used the WEKA machine learning toolkit (Witten and Frank 2005) to analyze naive Bayes and decision tree approaches for generating models of student affect to predict student frustration as early as possible.

5.3.2 Results

Unigram, bigram, naïve Bayes and decision tree affect recognition models for detecting student frustration were learned from the datasets collected in the study. The induced *n*-gram models were tested using a tenfold cross validation. (As noted above, in each fold, nine segments are used for training and one, which is held out of training, is used for testing.) The results of *n*-gram, the naïve Bayes, and decision tree affect recognition models are presented in Table 5-5. Figure 5.2 shows a bigram convergence graph depicting the amount of data (actions) required by the model to converge on the correct affective state and the associated probability of that emotion classification. Note that the bigram model utilizing a flattening constant converged after consuming 6.5% of the records leading up to the student self-reported affective state (the class label). In instances where *n*-gram models converged, the models were able to correctly classify whether the student was frustrated, on average, 35 seconds prior to the self-report. All induced models are able to predict student affective states (i.e., whether students are frustrated or not) early, i.e., long before we receive confirmation of the student's affective state from self-reports.

Table 5-5: Frustration Recognition Evaluation Results

	Accuracy	**Converged**	**Convergence Point**	**Average Observations of Converged**
Unigram (flattening Constant)	68.5%	39.7%	22.6%	54.3
Unigram (Good Turing)	73.4%	67.1%	7.1%	51.7
Bigram (flattening Constant)	73.6%	67.8%	6.5%	51.8
Bigram (Good Turing)	73.5%	67.2%	6.9%	51.8
Naïve Bayes	75.7%	75.7%	25.2%	51.3
Decision Tree	98.5%	93.6%	9.0%	51.2

Below ANOVA statistics are presented for results that are statistically significant. Because the tests reported here were performed on discrete data, we report Chi-square test statistics (χ^2), including both likelihood ratio Chi-square and the Pearson Chi-square values. To analyze the performance of induced models we first establish a baseline level. Because six affective states were reduced to a two-class predictive classifier (frustrated vs. not frustrated), we consider chance as a baseline measure of performance. If our baseline model were to predict the most frequent classifier (not frustrated, n=3859), then the baseline model would correctly predict a student's frustration state 65% of the time. Using this model as a baseline, we observe that all induced models outperform the baseline model. The lowest performing induced model, a unigram model using a flattening constant accurately predicted 68.5% of instances correctly in testing. This performance is statistically significantly better than the baseline (likelihood ratio, $\chi^2 = 16.075$, $p = 6.089 \times 10\text{-}5$, and Pearson, $\chi^2 = 16.067$, $p = 6.1 \times 10\text{-}5$, $df = 1$). Thus, the performance of all induced models is a statistically significant improvement over the baseline.

The experiment has two important implications for the design of runtime student frustration modeling. First, by monitoring student physiological response, the student's learning task, and events unfolding in the learning environment, induced models can make early, accurate predictions of forthcoming student frustration. Second, using models that can make early predictions of student frustration creates a significant window of opportunity for the learning environment to take corrective action; early-prediction models offer an improvement over traditional approaches that predict affective states and self-reports on a moment-by-moment basis.

Chapter 6

Conclusion

Goal recognition and affect recognition are critical problems in interactive learning environments. Providing intelligent learning environments with the ability to infer students' intent and affect states on a moment-by-moment basis could contribute to their ability to craft personalized experiences that are all the more engaging. With effective goal recognition, interactive learning environments might be able to more accurately assess students' domain and task knowledge. Being able to detect negative affective states early, i.e., before they lead students to abandon learning tasks, could permit interactive learning environments sufficient time to adequately prepare for, plan, and enact affective tutorial support strategies. To support effective, enjoyable interactions, affect-informed systems must be able to accurately and efficiently recognize user affect from available resources. Following appraisal theory, representations of users' actions and goals enable affect recognition models to consider the same relationship that users continually assess in order to predict their affective states.

This dissertation has introduced an inductive approach to generating goal and affect recognition models, which are foundational components of student models in narrative-centered learning environments. The induced models can cope with the uncertainty inherent in the task and offer the advantage of being automatically acquired rather than being manually constructed. In this approach, goal and affect recognition model-learners observe "training users" in a narrative-centered learning environment in which user actions, locations, goals, temporal information, and changes in physiological signals are monitored. After problem-solving traces have been recorded, goal and affect recognition models are induced.

We have employed several probabilistic frameworks such as *n*-grams, naïve Bayes, Bayesian networks, and decision trees to devise efficient goal and affect recognizers. In order to gather corpora for goal and affect recognition models, we introduced a corpus tool that generates goals that are expected to be achieved by training subjects. To obtain labeled corpora for affect recognition, the corpus tool also asks training subjects to select their current emotional states. Conditional probability tables were learned from the corpora. The models were used to predict students' goals and affective states early and accurately from a sequence of observations.

We conducted empirical studies of the models in the CRYSTAL ISLAND learning environment. Empirical studies of probabilistic goal and affect recognizers suggest that probabilistic approaches can accurately perform keyhole user goal recognition and affect recognition in a manner that is incrementally converging. The goal and affect recognizers are efficient and address the real-time requirements of interactive systems.

6.1 Limitations

Although the induced goal and affect recognition models make accurate early predictions, there are several limitations of the models. First, the presented goal and affect recognizers were tested only on corpora from the CRYSTAL ISLAND. Thus, the recognizers need to be evaluated on different corpora or different domains such as dialog systems. Due to the computational and space complexities of Bayesian networks, the Bayesian goal recognizer might not be applicable to certain domains. Second, the current goal recognizers only predict top-level goals. The recognizers might require a long sequence of observations to make an accurate prediction, especially when goals are difficult to achieve. It could be useful to predict chains of sub-goals for achieving the hard goal. Recognizing such sub-goals also could provide valuable information about how users are achieving goals. This can also allow the recognizers to predict goals much earlier than they can predict top-level goals. Third, to avoid zero probabilities, we adopt two smoothing methods (adding a flattening constant and simple Good-Turing estimation). Simple smoothing methods may not be appropriate for applications where optimal performance is critical. Finally, our models assume that all variables are observable. Learning probabilities with hidden variables should be addressed.

6.2 Future Work

Several directions for future research appear promising. First, the goal recognition models investigated here are based on the simplifying assumption that the user is pursuing a single goal. Users often pursue more than one goal at the same time, so models accommodating multiple simultaneous goals need to be studied. Second, it will be interesting to develop techniques for dynamically relaxing the keyhole requirement. In some narrative situations, it is appropriate for a character to approach the user and ask her what she is doing. It will be important to create techniques for identifying such situations and integrating the resulting information into the goal monitoring system. Third, while goal recognition provides an important source of information about the user to narrative-centered learning environments, plan recognition would also be beneficial, particularly in domains with large plan spaces.

Affect recognition offers much promise for narrative-centered learning environments. Early detection of negative emotions could play an important role in tutoring. In the future, it will be important to investigate affect recognition models that will enable affect-informed systems to inform runtime components of possible undesired user emotions. Determining frustration thresholds may allow learning environments to monitor students' persistence, intervening only when necessary, so that students exhibit affective states that best support effective learning. Evaluating the resulting models as runtime control components in narrative-centered learning environments is a critical next step in the investigation of affect-informed interactivity. It will also be useful to investigate other probabilistic affect recognition models. For example, there has been limited work in employing Bayesian user models for affective reasoning, although a few exploratory projects are underway (Conati 2002; Conati and Maclaren 2005). Because of the uncertainty involved, Bayesian affect modeling frameworks appear promising.

6.3 Concluding Remarks

We have seen inductive approaches to recognizing student goals and affect for narrative-centered learning environments. Accurately recognizing users' goals and affective states could contribute to more productive and enjoyable interactions, particularly for narrative-centered learning environments. We believe that it is important to investigate goal and affect recognition models that will enable goal and affect informed systems to make "early" predictions of user goals and affect, perhaps informing runtime components of possible undesired user emotions or sub-optimal paths to the goals. Early detection would allow systems adequate time to prepare for particular goals and affective states or to take action in an effort to ward off states such as high levels of frustration.

This dissertation represents a first step toward goal and affect recognition in narrative-centered learning environments. Systematically exploring the space of goal and affect recognition modeling techniques will contribute to the design of systems that can create increasingly effective and motivating interactive learning experiences.

References

Albrecht, D., Zukerman, I., Nicholson, A. 1998. Bayesian Models for Keyhole Plan Recognition in an Adventure Game. *User Modeling and User-Adapted Interaction* 8(1-2):5-47.

Alexandersson, J. 1995. Plan Recognition in VERBMOBIL. In *Proceedings of IJCAI 95 Workshop on The Next Generation of Plan Recognition Systems: Challenges for and Insight from Related Areas of AI*, 2-7.

Anderson, J. 1983. *The Architecture of Cognition*. Harvard University Press, Cambridge, MA.

Anderson, J., Corbett, A., Koedinger, K., and Pelletier, R. 1995. Cognitive Tutors: Lessons Learned. *The Journal of the Learning Sciences* 4:167–207.

André, E., and Mueller, M. 2003. Learning Affective Behavior. In *Proceedings of the Tenth International Conference on Human-Computer Interaction*, 512-516, Lawrence Erlbaum, Mahwah, NJ.

Arroyo, I., Beal, C., Murray, T., Walles, R., and Woolf, B. 2004. Web-based Intelligent Multimedia Tutoring for High Stakes Achievement Tests. In James C. Lester, Rosa Maria Vicari, and Fábio Paraguaçu, editors, *Intelligent Tutoring Systems* 3220:468–477.

Bates, J. 1994. *The Role of Emotion in Believable Agents.* Report CMU-CS-94-136, Carnegie Mellon University.

Bauer, M. 1996. A Demster-Shafer Approach to Modeling Agent Preferences for Plan Recognition. *User Modeling and User-Adapted Interaction* 5(3-4):317-348.

Bauer, M. 1996. Acquisition of User Preferences for Plan Recognition. In *Proceedings of the Fifth International Conference on User Modeling*, 105-112. Hawaii.

Beal, C. and Lee, H. 2005. Creating a Pedagogical Model that Uses Student Self Reports of Motivation and Mood to Adapt ITS Instruction. *AIED-05 Workshop on Motivation and Affect in Educational Software*, 18-22. Amsterdam, The Netherlands.

Blaylock, N. and Allen, J. 2003. Corpus-based, Statistical Goal Recognition. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 1303-1308, Acapulco, Mexico.

Blaylock, N. and Allen, J. 2004. Statistical Goal Parameter Recognition. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling*, 297-304. Whistler, British Columbia, Canada.

Blaylock, N. and Allen, J. 2005. Recognizing Instantiated Goals Using Statistical Methods. In *Proceedings of IJCAI Workshop on Modeling Others from Observations*, 79-86. Edinburgh, Scotland, UK.

Blaylock, N. and Allen, J. 2006. Hierarchical Instantiated Goal Recognition. In *Proceedings of AAAI Workshop on Modeling Others from Observations*, Boston, MA.

Blaylock, N and J. Allen, J. 2006. Fast Hierarchical Goal Schema Recognition. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 786-801, Boston, MA.

Bohnenberger, T., Brandherm, B., Großmann-Hutter, B., Heckmann, D., and Wittig, F. 2002. Empirically Grounded Decision-theoretic Adaptation to Situation-Dependent Resource Limitations. *Künstliche Intelligenz* 16(3):10-16.

Bosma, W. and André, E. 2004. Exploiting Emotions to Disambiguate Dialogue Acts. In *Proceedings of the 9th International Conference on Intelligent User Interfaces*, 85-92.

Brown, J., Burton, R., Miller, M., de Kleer, J., Purcell, S., Hausman, C., and Bobrow, R. 1975. *Steps toward a Theoretical Foundation for Complex Knowledge-Based CAI*. BBN Report 3135 (ICAI Report 2). Bolt, Beranek and Newman, Inc., Cambridge, MA.

Brusilovsky, P., Schwarz, E., and Weber, G. 1996. ELM-ART: An Intelligent Tutoring System on World Wide Web. In *Proceedings of the Third International Conference on Intelligent Tutoring Systems*, 261–269, London, UK.

Burleson, W. and Picard, R. 2004. Affective Agents: Sustaining Motivation to Learn through Failure and a State of Stuck. *ITS-04 Workshop of Social and Emotional Intelligence in Learning Environments*, Maceiò, Alagoas, Brazil.

Burton, R. 1982. Diagnosing Bugs in a Simple Procedural Skill. In Derek H. Sleeman and J. S. Brown, editors, *Intelligent Tutoring Systems*, Chapter 8. Academic Press, London.

Carberry, S. 1988. Modeling the User's Plans and Goals. *Computational Linguistics* 14(3):23-37.

Carberry, S. 1990. Incorporating Default Inferences into Plan Recognition. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, 471–478.

Carberry, S. 1990. *Plan Recognition in Natural Language Dialogue*. Cambridge, MA: The MIT Press.

Carberry, S. 2001. Techniques for Plan Recognition. *User Modeling and User-Adapted Interaction* 11(1-2):31-48.

Carbonell, J. 1970. AI in CAI: An Artificial Intelligence Approach to Computer Aided Instruction. *IEEE Transactions on Man-Machine Systems* 11(4): 190-202.

Charniak, E. and R. Goldman, R. 1991. A Probabilistic Model of Plan Recognition. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, 160-165.

Charniak, E. and Goldman, R. 1993. A Bayesian Model of Plan Recognition. *Artificial Intelligence* 64(1):53-79.

Cohen, P, Perrault, C., and Allen, J. 1981. Beyond Question Answering. In: *W. Lehnert and M. Ringle (eds.): Strategies for Natural Language Processing*, 245-274.

Conati, C. 2002. Probabilistic Assessment of User’s Emotions in Educational Games. *Journal of Applied Artificial Intelligence*, Special Issue on Merging Cognition and Affect in HCI, 16(7-8):555-575.

Conati, C. and Maclaren, H. 2005. Data-Driven Refinement of a Probabilistic Model of User Affect. In *Proceedings of the Tenth International Conference on User Modeling*, 40-49. Edinburgh, UK.

Conati, C and VanLehn, K. 1996. Probabilistic Plan Recognition for Cognitive Apprenticeship. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*, 403-408.

Conati, C., Gertner, A., and VanLehn, K. 2002. Using Bayesian Networks to Manage Uncertainty in Student Modeling. *Journal of User Modeling and User-Adapted Interaction*, 12(4):371-417.

Frijda, N. 1986. *The Emotions*. Cambridge University Press.

Gale, A. and Sampson, G. 1995. Good-Turing Frequency Estimation Without Tears. *Journal of Quantitative Linguistics*, 2(3): 217-237.

Geib, C. and Goldman, R. 2005. Partial Observability and Probabilistic Plan/Goal Recognition. In *Proceedings of IJCAI Workshop on Modeling Others from Observations*, Edinburgh, Scotland.

Goldman, R., Geib, C., and Miller, C. 1999. A New Model of Plan Recognition. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 245-254. Stockholm, Sweden.

Goleman, D. 2001. *Emotional Intelligence: Why It Can Matter More Than IQ*. Bantam Books, New York, NY.

Gratch, J. and Marsella, S. 2004. A Domain-Independent Framework for Modeling Emotion. *Journal of Cognitive Systems Research* 5(4):269-306.

Heckerman, D. 1999. *A Tutorial on Learning with Bayesian Networks. Learning in Graphical Models*, M. Jordan, ed., MIT Press, Cambridge, MA.

Hong, J. 2001. Goal Recognition Through Goal Graph Analysis. *Journal of Artificial Intelligence Research* 15:1-30.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D., and Rommelse, K. 1998. The Lumière Project: Bayesian User Modeling for Inferring the Goals and Needs of Software users. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 256–265, Madison, WI.

Horvitz, E. 1999. Principles of Mixed-Initiative User Interfaces. In *Proceeding of the CHI 99 Conference on Human Factors in Computing Systems*, 159–166.

Horvitz, E., Jacobs, A., and Hovel, D. 1999. Attention-Sensitive Alerting. In Kathryn B. Laskey and Henri Prade, editors, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 305–313, Morgan Kaufmann.

Hu, M. 1999. *Model Checking for Incomplete High Dimensional Categorical Data*. Ph.D. Thesis, University of California, Los Angeles.

Huber, M., Durfee, E., and Wellman, M. 1994. The Automated Mapping of Plans for Plan Recognition. In *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence,* 344–351. San Francisco, CA.

Jameson, A., Schäfer, R., Weis, T., Berthold, A, and Weyrath, T. 1999. Making Systems Sensitive to the User’s Changing Resource Limitations. *Knowledge-Based Systems*, 12:413–425.

Jameson, A. 2003. Adaptive Interfaces and Agents. In Julie A. Jacko and Andrew Sears, editors, *Human-Computer Interaction Handbook*, 305-330. Erlbaum, Mahwah, NJ.

Johnson, L., and Rizzo, P. 2004. Politeness in Tutoring Dialogs: “Run the Factory, That’s What I’d Do”. In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems*. Springer, 67-76.

Johnson, W and Soloway, E. 1985. PROUST: Knowledge-Based Program Understanding, *IEEE Transactions on Software Engineering* 11(2):267-275.

de Jong, T. and van Joolingen, W. 1998. Scientific Discovery Learning with Computer Simulations of Conceptual Domains. *Review of Educational Research* 68(2):179-201.

Kaminka, G., Chang, M., and Veloso, M.. 2002. Learning the Sequence Behavior of Teams from Observations. In *Proceedings of the 2002 RoboCup Symposium*.

Kautz, H. 1987. *A Formal Theory of Plan Recognition*. Ph.D. Thesis, University of Rochester.

Kautz, H. and Allen, J. 1986. Generalized Plan Recognition. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, 32–37. Philadelphia, PA.

Kleinbauer, T., Bauer, B. and Jameson, A. 2003. Specter – A User-Centered View on Ubiquitous Computing. In *Proceedings of the Eleventh GI-Workshop "Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen"*, Karlsruhe, Germany.

Kliensmith, A., De Silva, R., and Bianchi-Berthouze, N. 2005. Recognizing Emotion from Postures: Cross-Cultural Differences in User Modeling. In *Proceedings of the Tenth International Conference on User Modeling*, 50-59, Springer-Verlag, New York, NY.

Lazarus, R. 1991. *Emotion and Adaptation*. Oxford University Press, New York, NY.

Lee S., McQuiggan, S., and Lester, J. 2007. Inducing User Affect Recognition Models for Task-Oriented Environments. In *Proceedings of the Eleventh International Conference on User Modeling*, 380-384, Corfu, Greece.

Lesh, N. 1997. Adaptive Goal Recognition. In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, 1280-1214, Nagoya, Japan.

Lesh, N. and Etzioni, O. 1995. A Sound and Fast Goal Recognizer. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1704-1710, Montréal, Québec, Canada.

Mao, W. and Gratch, J. 2004. *Decision-Theoretic Approaches to Plan Recognition*. USC/ICT Technical Report #ICT-TR-01-2004.

Mao, W. and Gratch, J. 2004. A Utility-Based Approach to Intention Recognition. In *Proceedings of AAMAS 2004 Workshop on Agent Tracking: Modeling Other Agents from Observations*, NY City, NY.

Matsuda, N., Cohen, W., and Koedinger, K. 2005. An Intelligent Authoring System with Programming by Demonstration. In P*roceedings of the Japan National Conference on Information and Systems in Education*, Kanazawa, Japan.

Mayo, M. and Mitrovic, A. 2001. Optimising Its Behavior with Bayeisan Networks and Decision Theory. *International Journal of Artificial Intelligence in Education* 12:124–153.

McQuiggan. S., Lee, S., and Lester, J. 2006. Predicting User Physiological Response for Interactive Environments: An Inductive Approach. In *Proceedings of the Second Conference on Artificial Intelligence and Interactive Entertainment*, 60-65, Marina del Rey, CA.

McQuiggan, S., Lee, S., and Lester, J. 2007. Early Prediction of Student Frustration. In *Proceedings of the Second International Conference on Affective Computing and Intelligent Interactions*, 698-709, Lisbon, Portugal.

McQuiggan, S., and Lester, J. 2006a. Learning Empathy: A Data-Driven Framework for Modeling Empathetic Companion Agents. In *Proceedings of the Fifth International Conference on Autonomous Agents and Multi-Agent Systems*, 961-968, ACM Press, New York, NY.

McQuiggan, S., and Lester, J. 2006b. Diagnosing Self-Efficacy in Intelligent Tutoring Systems: an Empirical Evaluation. In *Proceedings of the Eighth International Conference on Intelligent Tutoring Systems*, 565-574, Springer-Verlag, New York, NY.

Mislevy, R. and Gitomer, D. 1995. The Role of Probability-based Inference in an Intelligent Tutoring System. *User Modeling and User-Adapted Interaction* 5(4):253–282.

Mitrovic, A., Koedinger, K., and Martin, B. 2003. A Comparative Analysis of Cognitive Tutoring and Constraint-based Modeling. In Peter Brusilovsky, Albert T. Corbett, and Fiorella de Rosis, editors, *Lecture Notes in Computer Science* 2702:313–322, Springer.

Mitrovic, A., Mayo, M., Suraweera, P., and Martin, B. 2001. Constraint-based Tutors: A Success Story. In *IEA/AIE '01: Proceedings of the 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, 931–940, London, UK.

Mott, B., Lee, S., and Lester, J. 2006a. Probabilistic Goal Recognition in Interactive Narrative Environments. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 187-192.

Mott, B., McQuiggan, S., Lee, S., Lee, S., and Lester, J. 2006b Narrative-Centered Environments for Guided Discovery Learning, *Workshop on Agent-Based Systems for Human Learning in conjunction with fifth International Conference on Autonomous Agents and Multi-Agent Systems*, Hakodate, Japan.

Murpy, K. 2002. *Dynamic Bayesian Networks: Representation, Learning and Inference*. Ph.D. Thesis, University of California, Berkeley.

Murray, R., VanLehn, K., and Mostow, J. 2004. Looking Ahead to Select Tutorial Actions: A Decision-Theoretic Approach. *International Journal of Artificial Intelligence in Education* 14(3-4):235-278.

Njike, H., Artières, T., Gallinari, P., Blanchard, J., and Letellier, G. 2005. Automatic Learning of Domain Model for Personalized Hypermedia Applications. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, Edinburgh, Scotland, UK.

Ohlsson, S. 1996. Learning from Performance Errors. *Psychological Review* 103:241–262.

Ormrod, J. 2002. *Educational Psychology: Developing Learners*, (4th ed.). Prentice Hall, Upper Saddle River, NJ.

Ortony, A., Clores, G., and Collins, A. 1988. *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge, MA.

Paiva, A., Dias, J., Sobral, D., Aylett, R., Woods, S., Hall, L., and Zoll, C. 2005. Learning by Feeling: Evoking Empathy with Synthetic Characters. *Applied Artificial Intelligence* 19:235-266.

Paliouras, G., Karkaletsis, V., Papatheodorou, C., and Spyropoulos, C. 1999. Exploiting Learning Techniques for the Acquisition of User Stereotypes and Communities. In *Proceedings of the International Conference on User Modelling*, 169-178.

Picard, R. 1997. *Affective Computing*. MIT Press, Cambridge, MA.

Picard, R., Vyzas, E., and Healey, J. 2001. Toward Machine Emotional Intelligence: Analysis of Affective Physiological State. *IEEE Transactions Pattern Analysis and Machine Intelligence* 23(10): 1185-1191.

Porayska-Pomsta, K. and Pain, H. 2004. Providing Cognitive and Affective Scaffolding through Teaching Strategies. In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems*, 77-86, Springer-Verlag, New York, NY.

Prendinger, H., and Ishizuka, M. 2005. The Empathic Companion: A Character-Based Interface that Addresses Users' Affective States. *Applied Artificial Intelligence*, 19:267-285.

Prendinger, H., Mayer, S., Mori, J., and Ishizuka, M. 2003. Persona Effect Revisited: Using Bio-Signals to Measure and Reflect the Impact of Character-Based Interfaces. In *Proceedings of the fourth International Conference on Intelligent Virtual Agents*, 283-291. Kloster Irsee, Germany.

Rich, E. 1979. User Modeling via Stereotypes. *Cognitive Science,* 3:329-354.

Riedl, M., Saretto, C., and Young, M. 2003. Managing Interaction Between Users and Agents in a Multi-agent Storytelling Environment. In *Proceedings of the Second International Conference on Autonomous Agents and Multi-Agent Systems*, 741-748. Melbourne, Australia.

Schmidt, C., Sridharan, N., and Goodson, J. 1978. The Plan Recognition Problem: An Intersection of Psychology and Artificial Intelligence. *Artificial Intelligence*, 11:45-83.

Seligman, M., Walker, E. and Rosenhan, D. 2001. *Abnormal Psychology*, (4th ed.) New York: W.W. Norton & Company, Inc.

Shute, V. 1995. Smart: Student Modeling Approach for Responsive Tutoring. *User Modeling and User-Adapted Interaction*, 5(1):1–44.

Sleeman, D., Hirsh, H., Ellery, I., and Kim, I. 1990. Extending Domain Theories: Two Case Studies in Student Modeling. *Machine Learning*, 5:11-37.

Smith, C., and Lazarus, R. 1990. Emotion and Adaptation. In Pervin (Ed.), *Handbook of Personality: Theory and Research*, 609-637, Guilford Press, New York.

Tsiriga, V and Virvou, M. 2004. A Framework for the Initialization of Student Models in Web-based Intelligent Tutoring Systems. *User Modeling and User-Adapted Interaction*, 14(4):289–316.

de Vicente, A. and Pain, H. 2002. Informing the Detection of the Students' Motivational State: An Empirical Study. In *Proceeding of the Sixth International Conference on Intelligent Tutoring Systems*. Biarritz, France and San Sebastian, Spain.

Wenger, E. 1987. *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Weyhrauch, P. 1997. *Guiding Interactive Drama*. Ph.D. Thesis., Dept. of Computer Science, Carnegie Mellon Univ.

Wilson, N. 2000. Algorithms for Dempster-Shafer Theory. *Algorithms of Handbook of Defeasible Reasoning and Uncertainty Management* 5: 421-475.

Witten, I., and Frank, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, San Francisco, CA: Morgan Kaufman.

Yin, J., Chai, X., and Yang, Q. 2004. High-Level Goal Recognition in a Wireless Lan. In *Proceedings of the Nineteenth National Conference in Artificial Intelligence*, 578-584. San Jose, CA.