Rules for the Implicit Acquisition of Knowledge About the User

Robert Kass and Tim Finin[1]
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389

Abstract

A major problem with incorporating a user model into an application has been the difficulty of acquiring the information for the user model. To make the user model effective, past approaches have relied heavily upon the explicit encoding of a large amount of information about potential system users. This paper discusses techniques for acquiring knowledge about the user *implicitly* (as the interaction with the user proceeds) in interactions between users and cooperative advisory systems. These techniques were obtained by analyzing transcripts of a large number of interactions between advice-seekers and a human expert, and have been encoded as a set of user model acquisition rules. Furthermore, the rules are domain independent, supporting the feasibility of building a *general user modelling module*.

I. Introduction

With the development of knowledge-based systems, computers are now being used for tasks that previously required significant human intelligence. As computers assume these tasks, expectations about their behavior have evolved as well. Systems that exhibit human-like reasoning abilities are expected to interact in an intelligent manner. Thus, humans might expect a system to (among other things) understand natural language, be able to infer intentions that are not explicitly stated, and tailor system responses to the individual user. One feature important to systems that support intelligent interaction is the ability to maintain information about their users—such systems are said to have models of their users.

A *user model* can be loosely described as a collection of assumptions or beliefs the system holds about the user. In this sense, all computer programs have some implicit user model, since they make assumptions about how the user will interact with the program. Of more interest are systems that keep *explicit* information about each individual user, using this information to tailor their communication with the user. Information that a system might keep about the user includes: the user's goals and plans, the user's beliefs or knowledge about the domain of discourse, objective properties about the user such as age or name, and the user's beliefs about other agents (such as the system itself).

[1]This work was supported by grants from the Army Research Office and the Digital Equipment Corporation

User modelling systems built in recent years[2] have demonstrated two major problems. First, acquiring knowledge about the user is very difficult. Second, user models seem to be restricted to the specific system for which they were created. Thus, developing a new system requires the development of a new user model. A solution to these problems enhances the feasibility of building a *general user model* [Finin and Drager, 1986] that can be used for multiple systems.

The research described in this paper addresses both general user modelling problems. In fact, solving the first problem goes a long way towards solving the second as well. This paper presents a group of user model acquisition rules that can be used to build a model of the user during an interaction. These rules were developed after study of an extensive collection of transcripts of conversations between advice-seekers and a human expert. The rules are domain independent, thus the user modelling portion of the system can handle different applications that have a similar form of interaction. Sections II. and III. briefly discuss the user model acquisition problem and general user modelling, while the following four sections present some of the model acquisition rules; section VIII. discusses future work planned in this area. A fuller treatment of the topics in this paper can be found in [Kass, 1987].

II. User Model Acquisition

In most existing user modelling systems, knowledge about the user is acquired *explicitly*, with information about the users directly asserted by the system designers. The most common method of asserting this information is to pre-encode the contents of the user model. Pre-encoding may take several forms: (1) a range of possible beliefs about the user may be listed in the model, (2) assumptions about all users may be collected into a *generic* model, or (3) assumptions may be collected into *stereotypes* [Rich, 1979] reflecting the beliefs of classes of users. When a new user interacts with the system, the user modelling process consists of identifying *which* pre-encoded information most accurately explains the observed behavior of the user.

Most user modelling systems rely on the user model during the course of the interaction, hence a robust user model must be developed quickly. Generic and stereotype modelling

[2][Kass and Finin, 1987] presents a survey of user modelling for natural language systems, while [Kass, 1986] surveys user modelling in intelligent tutoring systems.

approaches are particularly attractive because they can rapidly develop a large set of beliefs about a particular user. A generic model provides an initial set of assumptions about the user, while a stereotype approach will also provide a large set of beliefs once a stereotype (or several stereotypes) is triggered.

Unfortunately, the amount of information that must be explicitly pre-encoded can be prohibitive. In fact, for many systems, building the user model can be much more time consuming than building the domain knowledge base, making the implementation of a user modelling system very unattractive. Furthermore, specific user models must be built for each application.

The user model acquisition techniques discussed in this paper take a different approach to acquiring the user model. These techniques build the user model *implicitly*—as the system interacts with the user. Implicit user model acquisition minimizes (or even eliminates) the need for explicit coding of user model information. Thus, effective implicit user model acquisition can greatly reduce the development effort required to implement a user modelling system.

Implicit user model acquisition techniques have not been used extensively in the past because they have not performed well. It has been generally believed that the content of the communication between user and system is too limited to quickly build a robust model of a new user. The goal of this paper is to show that implicit acquisition techniques *can* quickly produce a robust model. In doing so, the acquisition rules rely on certain features of human behavior, using information obtained from user and system behavior (as well as the domain model of the underlying application and the current model of the user) as clues to infer more general information about the beliefs and knowledge of the user. In fact, the rules are capable of producing a model that can support a substantial portion of the behavior of the expert participant in the transcripts studied.

III. General User Modelling

There are three senses of generality that apply to user modelling. User models may be general with respect to the *range of users* they can handle. Most user modelling systems have this form of generality. User models may also be general with respect to the *form of interaction* with the user. Such a user model could effectively deal with interactions that might include menus, graphics, or natural language. Finally, user models may be general with respect to the *domain* of the interaction. A domain-general user model could be used in systems covering a diverse range of applications.

Completely general user modelling allows the user modelling portion of the system to be an independent module that collects and maintains information about the users, and communicates with other modules in the system via a well-defined interface. Such a user modelling module would use four sources of information for making inferences about the user: the behavior of the user observed by the system, the behavior of the system observable by the user, the domain knowledge of the underlying application, and the current model of

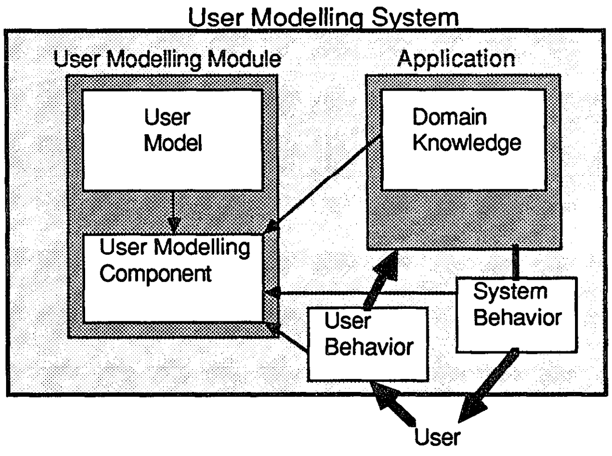

Figure 1: User modelling module sources of information

the user. The organization of a system incorporating a user modelling module is illustrated in figure 1.

This paper will focus on user modelling that is general with respect to the domain. (User generality is assumed to be a requirement of any user modelling system.) There are two reasons for this limitation. First, there is an existing trend towards building domain-independent systems, such as expert system shells. An expert system shell provides the reasoning and control structures for a system, and is capable of reasoning with knowledge bases from a variety of domains. A domain-independent user modelling module can thus be used in conjunction with other domain-independent modules to enhance the capabilities of such systems.

The second reason for focusing on domain generality is that building user models that are general with respect to the form of interaction is very difficult. Many of the implicit acquisition rules assume particular interaction characteristics. Shifting the form of interaction can affect not only these assumptions, but even the methods used to access the interaction between user and system. (Consider the difference between natural language interaction and interaction using graphics and a mouse.) Restricting the form of interaction thus constrains the model acquisition problem, enabling useful assumptions about the behavior of user and system.

In this work, the form of interaction is limited to *cooperative advisory systems*. A cooperative advisory system has a substantial body of knowledge about a particular domain, using this knowledge to give advice to users. Since it is cooperative, the system will try to anticipate the user's needs and goals, tailoring its interaction to be as helpful as possible. Although not a requirement of cooperative advisory systems, this work also assumes that the user and system communicate with each other through a natural language interface. The knowledge of the advisory system consists of factual knowledge about concepts and things in the world, and the reasoning rules it uses to give advice. The information to be modelled is primarily *long term*, such as the user's knowledge about the domain that tends to

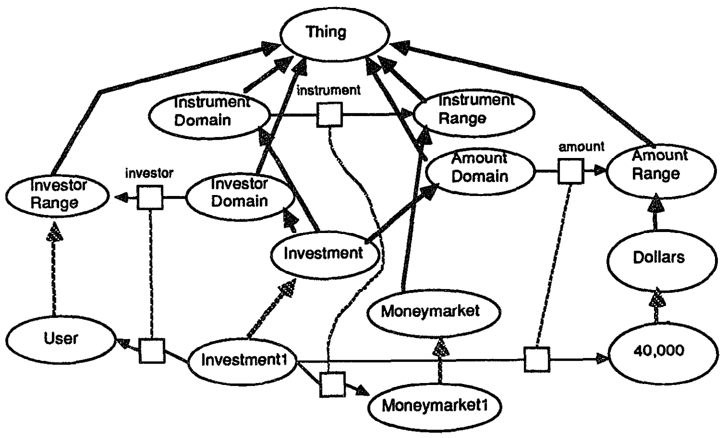

Figure 2: KL-ONE representation of concepts derived from "I have $40,000 in moneymarket"

persist over many interactions.

IV. The Acquisition Rules

The rules presented in this paper have been developed by analyzing interactions between human experts and their clients. The data examined includes transcripts of approximately 100 interactions from a radio talk show entitled "Harry Gross: Speaking about Your Money."[3] The examples used in this paper are taken from these transcripts. These conversations are appropriate for analysis since they represent a situation similar to what might occur in a cooperative advisory system: the form of interaction is quite limited (the participants communicate via telephone), the callers vary in their knowledge of the domain, and the expert has no pre-defined model of the caller.

The user model acquisition rules in the following sections should be considered to be *reasonable* rules—they are not absolute. Exceptions (which are sometimes quite easy to find) exist for each rule. This does not detract from the effectiveness of the rules, since the acquisition rules are intended to draw conclusions a human would reasonably make. Sometimes these conclusions will be over-ridden as new information arrives; sometimes the rules will draw conclusions that are not correct—humans have the same problem. It might be convenient to think of the acquisition rules as *default rules* [Reiter, 1980], but other approaches, such as evidential reasoning methods, could be used as well.

The rules can be loosely partitioned into three categories: communicative rules, model-based rules, and human behavior rules. Communicative rules focus on the communication between the system and the user. Model-based rules depend on certain relationships in the structure of information between the domain model and the current model of the user. Human behavior rules depend on features of human behavior that are typical or in some sense universal. The following sections look at these classes of rules, and discuss several rules in detail. A complete description of all of the rules can be found in [Kass, 1987].

V. Communicative Rules

The communicative rules are triggered by statements made by the user or the system, deriving information about the user based on the conventions governing normal discourse between cooperative agents. The class of communicative rules can be further divided into *direct inference* rules and *implicature* rules.

Direct inference rules are concerned solely with the information contained in a statement. When the user makes a statement, the user modelling module can assume the user believes that statement. This can result in a number of assertions to the user model concerning the user's beliefs about the concepts and attributes mentioned in the statement. For example, the statement "I have $40,000 in money market" could produce the following output from the parser:

$$\begin{aligned}\exists x_1 \quad & (\text{investment}(x_1) \wedge \\ & \exists x_2(\text{investor}(x_1, x_2) \wedge x_2 = \text{user}) \wedge \\ & \exists x_3(\text{instrument}(x_1, x_3) \wedge \text{moneymarket}(x_3)) \wedge \\ & \exists x_4(\text{amount}(x_1, x_4) \wedge \text{dollar}(x_4) \wedge x_4 = 40,000))\end{aligned}$$

This, in turn, can be decomposed to produce 15 simple assertions used to generate the concept and role definitions for a KL-ONE knowledge base depicted in figure 2.[4] Furthermore, statements may have presuppositions, which the user must believe as well. Kaplan [Kaplan, 1982] and Kobsa [Kobsa, 1984] have presented methods for computing these presuppositions, which may be asserted to the user model.

The implicature rules are inspired by Grice's maxims for cooperative communication [Grice, 1975]. The assumption

[3]The transcripts were made by Martha Pollack and Julia Hirschberg from shows originally broadcast on station WCAU in Philadelphia between February 1 and February 5, 1982.

[4]These steps have been implemented in a Prolog program that takes a first-order logic representation of a statement and builds a NIKL (New Implementation of KL-ONE) [Moser, 1983] knowledge base.

that the user is striving to be cooperative provides the system with certain expectations about user behavior. These expectations can be exploited to draw inferences about what the user does and does not know. The remainder of this section presents three rules inspired by the maxims of relation, quantity, and manner; illustrating them with examples from the transcripts.

Relevancy Rule

Grice's maxim of relation tells a speaker to make the contents of an utterance relevant. Assuming the user obeys this maxim, the user modelling module can assume what the user says is relevant. The rule can be stated as follows:

Rule 1 *If the user says P, the user modelling module can assume that the user believes that P, in its entirety, is used in reasoning about the current goal or goals of the interaction.*

In addition to claiming that the user believes what is said is relevant, the relevancy rule states that the user believes that *everything* in the statement is relevant. This can be illustrated with an example (the client's statements are preceded by a "C" and the expert's by an "E").

C. I just retired December first, and in addition to my pension and social security I have a supplemental annuity which I contributed to while I was employed from the state of New Jersey mutual fund. I'm entitled to a lump sum settlement which would be between $16,800 and $17,800, or a lesser life annuity and the choices of the annuity would be $125.45 per month. That would be the maximum with no beneficiaries.

E. You can stop right there, take your money.

In this example the caller believes a large amount of information is required to answer her question, thus she proceeds to talk about her recent retirement and source of income. The expert recognizes that the only information relevant in determining how she should take her supplemental annuity is the value of the annuity if taken as a lump sum versus the monthly payments that could be received. Once the expert has this information (and has identified the caller's goal) he interrupts and provides the answer. Meanwhile, in modelling the caller, the expert can conclude that she has little knowledge of the reasoning involved in making the decision. She feels that all the information she has listed is important to the reasoning process, while in fact it is not. Thus the relevancy rule can be used to acquire information about incorrect reasoning a user may perform.

Sufficiency Rule

The sufficiency rule is inspired by the maxim of quantity. The system can reason as follows: if the user were completely knowledgeable about the domain, he would provide information sufficient for the system to satisfy the user's goal. Suppose what the user says turns out to be insufficient? In this case the user must lack some knowledge that the system has. A user may have three types of knowledge about entities in the domain knowledge base: *knowledge of* an entity, knowledge of the *relevance* of an entity, and knowledge of the *value* of an entity. When the user is cooperative, yet omits a piece of information that the system knows is relevant, it is due to a lack of knowledge of one of these three types.

The sufficiency rule says:

Rule 2 *If the user omits a relevant piece of information from a statement, then either the user does not know of that piece of information, does not know whether that information is relevant to his current goal or goals, or does not know the value for the piece of information.*

Once again an example will illustrate this rule.

C. I've got $2250 to invest right now in an 18 month certificate and I don't know whether to go the variable rate or the fixed rate now or the fixed rate later.

E. Have you any money invested now?

C. Yes, I do.

E. In what?

C. I've got $5000 in a money market fund.

E. Have you anything in certificates or anything else?

C. I've got three stocks.

E. Three separate stocks?

C. Yes sir.

In this conversation, the caller believes his initial statement of the problem is sufficient for the expert to make a decision. Instead, the expert realizes a lot more information about the caller's investments are needed. The expert proceeds to ask questions to obtain this information. Even then, the caller provides minimal answers because he does not know what additional information is relevant until the expert specifically asks for it. In this case it seems obvious that the user knows the additional information, he just does not realize it is relevant.

In using the sufficiency rule, the user modelling module must be able to "turn around" the reasoning rules in the domain model, in order to identify properties that are relevant. This collection of relevant properties creates an expectation of the information the user should provide. Information in the set of expectations that is not provided thus must be information the user lacks knowledge of.

The sufficiency rule might be strengthened further. If the user is being fully cooperative he will try to be as helpful as possible. Suppose the user knows a piece of information, but does not know its value. For example, the user might know that the due date of a money market certificate is relevant information, but not know the actual due date. A truly cooperative user would tell the system that he does not know the due date. Thus the sufficiency rule might be limited to conclude that either the user does not know of the information, or does not know that it is relevant. Furthermore, if the user does not know of the information, he certainly cannot believe it is relevant, so the sufficiency rule could make a definite conclusion in this case.

Although the strengthened sufficiency rule seems attractive, that level of cooperation by the user does not seem likely. People are reluctant to display their ignorance. Thus, when they don't know something, they avoid mentioning it, even when they believe it is relevant.

Rule 7 *If the user is the agent of an action, then the user modelling module can attribute to the user knowledge about the action, the substeps of the action, and the factual information related to the action.*

For example, if the user says "I just rolled over two CD's," the user modelling module can recognize that the user is the agent of the "roll over" action. The agent rule will thus conclude that the user knows about the steps involved in rolling over a CD. Furthermore, the agent rule will also assert that the user knows about related facts, such as: that CD's have a due date, that money from a CD can be reinvested, that CD's are obtained from banks, and so on. Thus the agent rule can be particularly powerful in contributing information about the user.

Evaluation Rule

The evaluation rule uses the domain model's reasoning knowledge to make conclusions about the reasoning knowledge the user has. Many times in advisory interactions the expert must evaluate beliefs held by the user or actions performed by the user in light of the extensive knowledge the expert has. If the expert's evaluation differs from that of the user, this indicates that the user does not know some of the reasoning that the expert used. The rule is stated as follows:

Rule 8 *If the system is able to evaluate actions taken by the user given a certain situation, and those actions do not conform to the actions the system would have taken, then the user modelling module can identify portions of the reasoning done by the system that the user does not know about.*

An example of the use of the evaluation rule can be seen in the following conversation.

C. I have $10,000 in stocks and $10,000 in a savings fund

E. Why so much in a savings account?

In this conversation the caller makes a statement that leads the expert to an immediate evaluation: she has too much money in her savings account.[5] Thus, in modelling the caller, the expert can conclude that she does not recognize the reasoning that implies that a lot of money in a savings account will result in a relatively low return.

VIII. Conclusion

The long term goal of this research is the development of a general user modelling system that can act as a repository of knowledge about individual users, and service the needs of a number of applications. The primary obstacle to this goal is the difficulty of explicitly acquiring and building the user models, motivating the development of a set of rules (currently 18, 8 of which are discussed in this paper) that can be used to acquire knowledge about the user implicitly from his interaction with the system.

[5]These conversations were transcribed in February, 1982, when the U.S. inflation rate was near its peak. At that time the interest rate on a savings account was considerably less than what could be earned in a money market fund, so having a lot of money in a savings account was always a bad idea.

The next step is to implement the model acquisition rules in a user modelling system to test their effectiveness. This implementation will comprise one portion of a general user modelling module. This module will assume that the user communicates with the underlying application via a natural language interface, having access to the output of a parser encoded in a *meaning representation language* (MRL). An underlying domain model for investment securities has been built and will be used to test the acquisition rules. This model uses the KL-ONE-like language NIKL [Moser, 1983], and currently consists of over 150 concepts.

There are a number of issues not discussed in this paper that will be addressed in the implementation. These issues include the use of a truth maintenance system to manage the non-monotonic nature of user modelling, and the development of rules to arbitrate between the acquisition rules when they conflict regarding the user's knowledge of a particular item. A more complete description of current research and future plans can be found in [Kass, 1987].

References

[Finin and Drager, 1986] T. Finin and D. Drager. GUMS$_1$: a general user modelling system. In *Proceedings of the 1986 Conference of the Canadian Society for Computational Studies of Intelligence*, pages 24–30, 1986.

[Grice, 1975] H. P. Grice. Logic and conversation. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics*, Academic Press, New York, 1975.

[Kaplan, 1982] S. J. Kaplan. Cooperative responses from a portable natural language database query system. *Artificial Intelligence*, 19(2):165–188, 1982.

[Kass, 1986] R. Kass. *The Role of User Modelling in Intelligent Tutoring Systems.* Technical Report MS-CIS-86-58 (Linc Lab 41), Department of Computer and Information Science, University of Pennsylvania, 1986.

[Kass, 1987] R. Kass. *Implicit Acquisition of User Models in Cooperative Advisory Systems.* Technical Report MS-CIS-87-05, Department of Computer and Information Science, University of Pennsylvania, 1987.

[Kass and Finin, 1987] R. Kass and T. Finin. Modelling the user in natural language systems. *Computational Linguistics*, Special Issue on User Modelling, 1987.

[Kobsa, 1984] A. Kobsa. Three steps in constructing mutual belief models from user assertions. In *Proceedings of the 6th European Conference on Artificial Intelligence*, pages 423–427, 1984.

[Moser, 1983] M. G. Moser. *An Overview of NIKL, The New Implementation of KL-ONE.* Technical Report 5421, Bolt, Beranek and Newman, 1983.

[Reiter, 1980] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1):81–132, 1980.

[Rich, 1979] E. Rich. User modelling via stereotypes. *Cognitive Science*, 3:329–354, 1979.