Computational Intelligence, Volume , Number ,

CONSTRUCTING AND UTILIZING A MODEL OF USER PREFERENCES IN COLLABORATIVE CONSULTATION DIALOGUES

SANDRA CARBERRY

Department of Computer Science

University of Delaware

Newark, Delaware 19716 USA

Email: carberry@cis.udel.edu

JENNIFER CHU-CARROLL

Bell Laboratories

Lucent Technologies

600 Mountain Avenue

Murray Hill, NJ 07974 USA

Email: jencc@bell-labs.com

STEPHANIE ELZER

AMP Inc.

PO Box 3608

Mail Stop 194-20

Harrisburg, PA 19105 USA

Email: selzer@amp.com

A natural language collaborative consultation system must take user preferences into account. A model of user preferences allows a system to appropriately evaluate alternatives using criteria of importance to the user. In addition, decision research suggests that an accurate model of user preferences could enable the system to improve a user's decision-making by ensuring that all important alternatives are considered, and that such a model of user preferences must be built dynamically by observing the user's actions while engaged in the decision-making process. This paper presents two strategies: one for dynamically recognizing user preferences during the course of a collaborative planning dialogue and the other for exploiting the model of user preferences to detect suboptimal solutions and suggest better alternatives. Our recognition strategy utilizes not only the utterances themselves but also characteristics of the dialogue in developing a model of user preferences. Our generation strategy takes into account both the strength of a preference and the closeness of a potential match in evaluating actions in the user's plan and suggesting better alternatives. By modeling and utilizing user preferences, our system is better able to fulfill its role as a collaborative agent.

Subject Categories: Natural Language (major), User Modeling (minor)

Key words: user preferences, dynamic user model acquisition, collaborative consultation dialogue, human-computer interaction, user model utilization

1. INTRODUCTION

We have been developing a natural language consultation system in which the system plays the role of an expert in the domain and collaborates with the user to construct a plan for achieving the user's domain goals. Many of the decisions that users make in these domains are preferential choices. That is, there are a wide variety of individual preferences that determine which solution will be considered best by a particular user. Kass and Finin classify these preferences under the category of attitudes, and observe that people "exhibit preferences and bias toward particular options or solutions. A natural language system may often need to recognize the bias and preferences a user has in order to communicate effectively."[Kass and Finin, 1988]

Consider the following conversation, based on a naturally-occurring dialogue taken from a corpus of travel agent dialogues [SRI transcripts, 1992]:

(1) A: I'd like TWA flight 323/751.

(2) B: Do you know that there's a nonstop United from Dulles to San Francisco? Instead of connecting through St. Louis.

(3) A: Yes, but on TWA I can get the business traveler seating.

In utterance (2), B makes the suggestion of a nonstop flight based on his model of A's preferences. More specifically, B believes that people stereotypically prefer direct flights over non-direct flights. However, in utterance (3), it becomes evident that A's preference for business traveler seating makes the non-direct flight the more desirable alternative.

This conversation clearly demonstrates the influence of preferences on the decision process, as well as the individuality of preferential choice. Other travelers might find the direct flight suggested by B to be a better alternative, or they might prefer the TWA flight for reasons other than the business traveler seating, such as the departure time. The dialogue also illustrates the importance of recognizing preferences. B could not be considered a cooperative agent if he did not recognize that A is expressing a preference for business traveler seating and subsequently take this preference into account in making further suggestions to A. This paper presents two strategies: one for dynamically recognizing user preferences during a collaborative planning dialogue and the other for exploiting the model of user preferences in order to detect suboptimal solutions and suggest better alternatives.

2. PREFERENCES VERSUS GOALS

One might argue that user preferences could be treated as user goals [van Beek, 1987]. For example, in the dialogue above, A could be said to have the goal of obtaining business traveler seating. However, preferences and goals play substantially different roles in plan construction. Users are conscious of their goals, have the intention of satisfying their goals, and use their goals to limit the search for a plan of action. Although a user may relinquish some of his goals if the entire goal set cannot be satisfied, the intent at the outset is to develop a plan that achieves all of the user's goals. Preferences, on the other hand, influence the selection and instantiation of the actions which achieve the user's goals. Stronger preferences have more of an effect on action choice than do weaker ones, and coming close to satisfying a preference may be regarded as better than ignoring it completely. In addition, users are often not conscious of their preferences at the outset of planning and only bring these preferences into play as they must evaluate alternative actions and choose among them. Thus, we contend that preferences should not be treated as goals, but as potential influencers [Bratman, 1990] held with respect to actions.

3. WHY MODEL USER PREFERENCES?

The ability of a natural language system to recognize user preferences is particularly important when involved in a collaborative consultation dialogue because of the different types of knowledge that the system and user bring to the dialogue. The system presumably has more extensive and accurate domain knowledge than the user, while the user has knowledge about his or her particular circumstances, intentions and preferences that are either restrictions on or potential influencers [Bratman, 1990] of the domain plan that is constructed. The system and the user must communicate with one another since neither has all of the relevant information for constructing the domain plan.

In a collaborative dialogue, each agent has the responsibility to contribute in every way possible to come up with the best possible solution to the problem at hand. It follows that if a collaborative system knows of a better way to solve the problem than the one currently under consideration, it has an obligation to inform the user of this solution. If the domain involves preferential choice, the system needs to be able to evaluate potential solutions with respect to the user's preferences in order to determine when one set of actions is more desirable than another. Thus the system must be able to recognize and make use of the user's preferences in order to fulfill its role as a collaborative participant.

Preferential choice has long been of interest in the area of psychology, and psychological research provides evidence that a natural language system's ability to recognize user preferences could lead to improved decision-making. Psychologists have studied the strategies that people use in making decisions, and have developed a wide range of models which attempt to explain the human decision process. These decision strategies capture a variety of possibilities ranging from heuristics which help to limit the amount of information that needs to be processed [Tversky, 1972, Tversky, 1969, Simon, 1955] to strategies which use all of the information available to come up with an overall rating of each alternative [Keeney and Raiffa, 1993, Dawes, 1979, Einhorn and Hogarth, 1975]. Research by Payne indicates that when faced with a large number of alternatives or a large quantity of information, people tend to use heuristics to quickly reduce the amount of information that needs to be considered [Payne, 1976]. This can lead to the choice of a less desirable option because of the possible elimination of better alternatives early on. This type of decision-making behavior alone would be enough to suggest that the system should make an independent evaluation of the alternatives using the user's criteria (i.e. preferences).

Note that there is an important distinction between the type of environment in which psychologists have been studying decision-making behavior and the type of domains which we have been considering. In most psychological studies, it is assumed that the user has all of the information necessary to make a decision. In the domains we are considering, the system is assumed to be the expert, and there may be alternatives or information which is unknown to the user. This means that a collaborative system must reason about the user's preferences not only to prevent the user from inadvertently discarding desirable options but also to suggest desirable alternatives of which the user is unaware.

Dynamically recognizing user preferences as they are expressed in the dialogue, rather than querying the user about those preferences beforehand, may be the only way to obtain an accurate model of the user's preferences. Decision research has shown that preferences and beliefs are frequently constructed while responding to a judgment or choice task [Slovic *et al.*, 1990]. This means that a user may not have an accurate model of her preferences until she is faced with an actual decision, so querying her preferences before a decision is made will be ineffectual.

4. OVERVIEW

In this paper, we present two strategies: one for dynamically recognizing user preferences during the course of a collaborative planning dialogue and the other for exploiting the model of user preferences to detect suboptimal solutions and suggest better alternatives. Our recognition strategy is the first to recognize preferences during the course of a natural dialogue, utilizing not only the utterance itself but also characteristics of the dialogue in developing a model of user preferences [Elzer *et al.*, 1994]. Our preference recognition strategies are designed to fit into a plan recognition framework in which the user's plans and goals are inferred by the system. Our generation strategy reflects a weighted additive model of human decision-making [Payne *et al.*, 1993] and takes into account both the strength of a preference and the closeness of a potential match. The examples in this paper have been run on our implemented system and are taken from a university domain of degrees, courses and requirements.

We are currently modeling *attribute-value preferences*, where an attribute-value preference is a preference that an object in an action have a particular attribute value. That is, if the user wants to execute an action *Take-Course(User, _course)*,[1] this action involves the parameter _course, and the user might have preferences involving the various attributes of the particular object (of type course) that will instantiate the _course parameter. Courses have attributes such as difficulty level, professor, meeting time, and content. The user might have attribute-value preferences for specific values of these attributes, such as preferring courses that meet in the morning or are taught by a certain professor. We are currently making the strong assumption that attribute preferences are independent of one another. For example, we do not consider situations in which a user's preference regarding the meeting time of a

[1]Variables will be represented as a string preceded by an underscore, where the string indicates that variable's type.

course is dependent on the type of course (music, computer science, etc.) Although such preferences can exist, it is questionable whether users expect them to be recognized unless they are explicitly stated. However, if the user does explicitly state such a dependency, our system could capture the dependency by adding a constraint to the preference which describes when that preference is applicable. Further research is needed to determine whether users expect such dependencies between preferences to be recognized when they have not been explicitly expressed.

Our system recognizes both preferences that are expressed by the user and preferences that can be detected through patterns in the user's acceptance or rejection of proposals. For example, a user may reject several solutions which have common characteristics without ever explicitly expressing a preference. By reasoning about this pattern of rejection, our system can detect a possible user preference that is only implicitly conveyed.

People often weigh their preferences when making a decision, and consider some preferences to be more important than others. Some users might, for example, consider the meeting time of a course to be much more important than who teaches it. Since a collaborative consultation system needs to be able to evaluate alternatives using the same criteria that a user would, it must be able to weigh preferences. Thus a complete representation of user preferences must capture the strength of each preference. Our system deduces the strength of a preference based on a combination of the conversational circumstances in which the preference is expressed and the semantics of the user's utterance. In addition, a consultant should have more confidence in some preferences that have been deduced than in others. For example, a consultant will have more confidence in beliefs about a user's preferences if they result from the user's explicit statement to that effect than if they are deduced from a pattern of rejections by the user. Our system captures this by using endorsements [Cohen, 1985] to reflect the reliability of evidence for each preference. It is important to note the difference between the strength of a preference and the strength of an endorsement of the preference: the strength of the preference represents how important the preference is to the user while the strength of the endorsement represents how strongly the system believes that the preference accurately models an attitude of the user.

Our system utilizes the recognized preferences to suggest better alternatives when an action is proposed by the user. We have developed a *ranking advisor* that is capable of rating different instantiations of an action based on the user's preferences. The ranking advisor selects preferences that are sufficiently reliable (as indicated by their endorsements) and calculates an overall rating for each possible instantiation using a combination of the strength of each preference and the closeness of a potential match. The instantiation with the highest rating will be considered the *best* choice for the user, and, if the best choice is different from and substantially better than the user's proposal, our system will generate the logical forms for natural language utterances to suggest this alternative to the user.

In this paper, we will first discuss methods for recognizing and modeling user preferences, capturing both the preference strength and the reliability of the evidence representing each preference. Methods for combining multiple and conflicting evidence will be presented. We will then discuss the utilization of the model of user preferences by the generation component in order to evaluate the user's proposals and suggest better alternatives. Finally, we will present the results of two studies conducted to provide a preliminary evaluation of our prototype system.

5. DETECTING AND MODELING USER PREFERENCES

The preference recognition component is responsible for dynamically building and updating a model of the user's preferences. To identify evidence suggesting particular preferences on the part of the user, the preference recognition component analyzes the user's utterances,

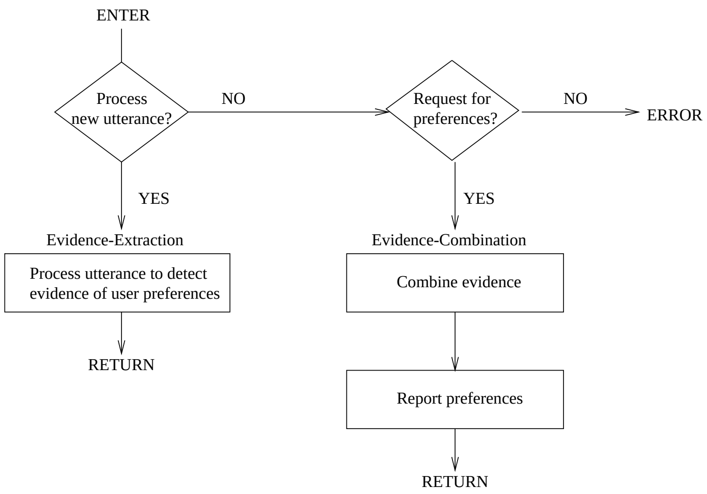

Figure 1. Overall Architecture of Module for Modelling User Preferences

derived actions (such as rejection of a system proposal), and the conversational circumstances, and exploits a knowledge base of stereotypes. As the dialogue progresses, each time a preference is deduced it will be recorded along with a strength-endorsement pair. If there is multiple evidence for a preference, that preference will have several strength-endorsement pairs. Thus the user model reflects the accumulation of evidence for individual preferences and the importance of those preferences to the user. When an application module requests information about the user's preferences, the recognition system combines the evidence and reports those preferences that exceed a specified reliability threshold.

The overall architecture of the preference recognition module is shown in Figure 1. The processes for extracting evidence (the Evidence-Extraction box in Figure 1) is outlined in Figure 2 and will be described in detail in Section 5.1 through Section 5.4. The process for combining evidence (the Evidence-Combination box in Figure 1) is outlined in Figure 9 on page 28 and is discussed in detail in Section 5.5.

```
Evidence-Extraction(_sm-utt, _dm, _user-model, _ph)
/* _sm-utt: semantic representation of utterance */
/* _dm: dialogue model containing domain and discourse plans */
/* _user-model: current evidence of user preferences */
/* _ph: proposal history */

If speaker(_sm-utt)=system
   /* Update proposal history to reflect new attribute values given to user */
   Then update-proposal-history(_ph, _sm-utt, _dm)
   Else Begin
         /* Use utterance and conversational circumstances to detect preferences */
         If detect-preference(_sm-utt,_dm)
            Then Begin
                  /* Extract evidence of user preference where _pref is a 3-tuple */
                  /* consisting of an object type, an attribute, and an attribute value */
                  _pref←infer-preference(_sm-utt,_dm)
                  _str←infer-preference-strength(_pref,_sm-utt,_dm)
                  _endmt←assign-endorsement(_pref,_sm-utt,_dm)
                  /* Attach preferences to _action and other domain actions in the dialogue model */
                  _evid-set← attach-to-actions(_action, _dm, (_pref,_str,_endmt))
                  /* Accumulate new evidence in user model */
                  _user-model←_user-model ∪ _evid-set ∪ indirect-propagation(_pref,_str,_endmt,Plan-Library)
                  End
         If accept-or-reject-proposal(_sm-utt,_dm)
            /* Extract evidence for user preferences from proposal history */
            Then Begin
                  _evid-set ← extract-and-propagate-preferences-proposal-history(_ph,_sm-utt,_dm)
                  _user-model←_user-model ∪ _evid-set
                  update-proposal-history(_ph, _sm-utt, _dm)
                  End
         End
```

FIGURE 2. Outline of Evidence Extraction

5.1. Preference Strength

Since some preferences are more important to the user than others, a system's ability to assess the strength of a preference is instrumental to its ability to reliably evaluate alternative actions and make useful suggestions to the user. We argue that an analysis of preference strength must take into account both the utterance in which the preference is conveyed and the *conversational circumstances* under which the preference is expressed.

Preferences Expressed in Utterances. From our analysis of naturally occurring dialogues, we have identified three ways in which a preference can be explicitly conveyed in an utterance. First, the preference can be expressed directly through a statement such as "I like AI courses." Utterances of this type represent the strongest surface expression of a preference. Second, a preference can be conveyed indirectly in the utterance. For example, suppose that the user requests objects that have a particular attribute value, as in the user query "Which math courses are available?" The fact that the user has chosen to ask about courses whose content area is math suggests a preference,[2] though a more moderate one than if he had directly stated that he likes math courses. Third, the user can convey an uncertain preference as in "I don't think I like AI courses" or "I might like to take an AI course." We call this *hedging* the preference and view it as conveying a weaker preference than the other two forms of expression.

We have also found that the *conversational circumstances*, or the situation in which the preference is expressed, contribute to determining the strength of the preference. For example, a preference is very strong if it is given by the user as the reason for rejecting a solution, since the user is effectively saying that potential solutions with (or, in some cases, without) this particular characteristic are undesirable. Similarly, a preference that is included by the user in the initial description of the problem to be solved is likely to be stronger than a preference which is expressed in response to a system query since in the first case the user was conscious of the preference at the outset and wanted it to be taken into account in formulating a plan. We use the *conversational circumstances* to determine a range of possible strength for the preference. We have identified four different conversational circumstances as

[2]The specified attribute value cannot be a constraint in the current plan. For example, if the user's current goal is to satisfy a math requirement, then asking which math courses are available does not convey a preference for taking courses whose content area is math.

follows:

Reject-Solution: The user gives the preference as a reason for rejecting a solution.

Example:

(4) S: CS360 is a networks course.

(5) U: I don't want to take CS360 because I don't like networks courses.

Volunteered-Background: The user includes certain preferences as part of the initial problem description.

Example:

(6) U: I want to satisfy my seminar course requirement.

(7) I'd like to take an AI course.

Volunteered: The user volunteers preference information (without prompting from the system) during the dialogue.

Example:

(8) S: CS360 meets at 6pm.

(9) U: Great. I like evening courses.

Question-and-Answer: The user provides preference information in response to a system query.

Example:

(10) S: What type of course are you interested in?

(11) U: I like networks courses.

We maintain six positive preference strengths and six negative preference strengths. Figure 3 illustrates how the conversational circumstances correlate with a range of possible preference strengths in our system.[3] Given the semantic representation of a user utterance, the system analyzes it to determine whether it expresses a preference; if it does, then the way the preference is conveyed determines the strength of the preference within the range established by the conversational circumstance in which the utterance occurs. If the preference is expressed as a direct statement, the strength will be the highest possible within the range. If the preference is conveyed indirectly (as in the utterance "Which AI courses are available?"), the strength will be the middle preference strength in the range. If the preference is hedged, the strength will be the lowest in the range. If, for example, the user begins a conversation by saying "I need to satisfy my seminar course requirement. I'd like to take an AI course" the preference strength will be *str-2*, the highest within the *Volunteered-Background* range. If the user had instead said "I need to satisfy my seminar course requirement. Which AI courses are there?" the preference strength would be *mod-1*, the middle within the *Volunteered-Background* range. If, however, the user had said "I want to satisfy my seminar course requirement. I think I'd like to take an AI course" the preference strength would be *mod-2*, the lowest within the *Volunteered-Background* range.

Other Ways of Detecting Preferences. Preferences can often be identified even though they are not expressed in an utterance. For example, a preference can be deduced by analyzing a pattern of rejections by the user for common features of the rejected solutions. Our system accomplishes this by maintaining a data structure called the *proposal history* which is explained in more detail in Section 5.2. Preferences detected in this manner are assigned a preference strength of *weak-2*, because we hypothesize that if the preference was more important to the user, then the user, being a collaborative agent, would explicitly convey

[3] Because positive and negative preference strengths are handled identically, Figure 3 can be viewed as depicting either the positive or negative direction. The direction is denoted by either a (pos) or (neg) following the preference strength.

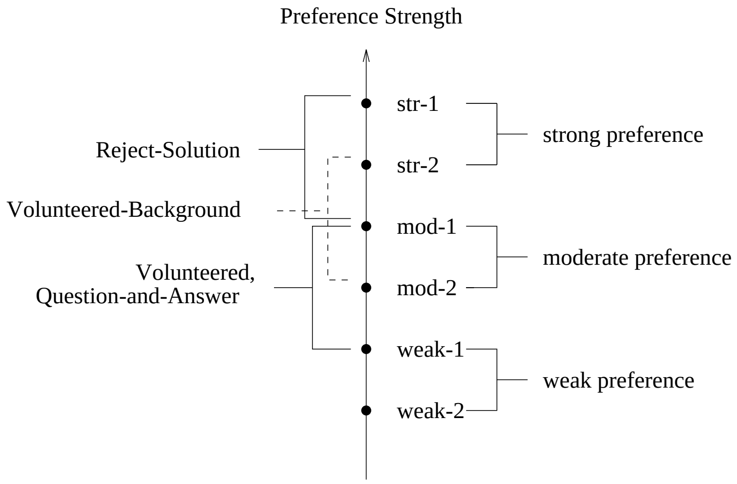

Figure 3. The Possible Ranges of Preference Strength

the preference to the system. This results in six (positive or negative) preference strengths, ranging from *str-1* to *weak-2* as shown in Figure 3.

In addition, a preference can be attributed to the user on the basis of a user stereotype or by generalizing a recognized preference to other related actions in addition to those actions for which it has been explicitly deduced. We call this generalization process *indirect propagation* and it is further explained in Section 5.3. Stereotypical and generalized preferences can be of any preference strength.

5.2. Endorsements

As stated earlier, a complete representation of a preference must contain two elements: 1) the strength of the preference, representing how important that preference is to the user, and 2) the system confidence in the preference, representing the evidence supporting the existence of that preference. In the previous section, we presented methods for modeling the strength of a preference, and in this section, we present methods for modeling the system confidence in a preference using endorsements. Endorsements are explicit records of factors that affect one's certainty in a hypothesis [Cohen, 1985]. Cohen introduced the endorsement-based approach to reasoning under uncertainty and endorsements have since been used in belief models [Galliers, 1992]. We use endorsements to represent the evidence that the system has accumulated to support its belief about the existence of a user preference. In our system, this evidence corresponds with how the preference was inferred; for example, preferences inferred from a stereotype are less reliable than preferences derived from a statement by the user expressing a preference at the start of a dialogue. This use of endorsements allows us to clearly evaluate the quality of the evidence for each preference.

The use of endorsements in the recognition of preferences gives us several advantages. First, the evidence for a preference can be allowed to accumulate and combine so that several weak pieces of evidence can combine to provide stronger cumulative evidence. Second, the system can better explain and justify its beliefs about the user's preferences should questions about its suggestions arise. Third, the generation component can better support its proposals by justifying them in terms of the user preferences with the most reliable evidence. Finally, the use of endorsements allows the generation component to explicitly choose what evidence is sufficient for various uses. For example, suppose that a user has little knowledge of the domain and requests guidance and advice from the system. In such situations, the system might take into account user preferences with a lower reliability rating than it would in suggesting a better alternative to the user's own proposal, since the system should have more confidence that its recommendation meets the user's preferences when interrupting with a counterproposal than when responding to the user's requests for a recommendation.

Figure 4 contains a list of the endorsements that we use in our system. Each endorsement corresponds with how a preference might be inferred, such as through the conversational circumstances, user stereotypes, processing of the proposal history, or indirect propagation. The

Reliability Rating	**Endorsements**
Very-Strong	Reject-Solution, Volunteered-Background
Strong	Volunteered, Question-and-Answer
Moderate	PH-Deduction-Strong
Weak	PH-Deduction-Weak, Stereotypical, Indirect-Propagation
Very-Weak	PH-Deduction-Initial

Key:

PH: refers to Proposal History

FIGURE 4. Reliability Ratings of Endorsements

endorsements are grouped into classes according to their estimated reliability. For example, the system has much more confidence in the existence of preferences expressed as the reason for rejecting a solution than in the existence of preferences hypothesized by user stereotypes. These classes are similar to Murray's endorsement reliability classes [Murray, 1991], although our methods for combining endorsements differ from his.

Because our endorsements correspond with how the preferences are detected, each conversational circumstance is represented by an endorsement. The ranking of these endorsements also corresponds to their preference strength ranges. This correlation exists because of the collaborative nature of consultation dialogues. If a preference is strong, then it is important to the collaborative process and the user will be more likely to clearly express it, thus leading to more reliable evidence. As discussed earlier, the endorsements which do not match the conversational circumstances (*PH-Deduction-Strong*, *PH-Deduction-Weak*, *Stereotypical*, *Indirect-Propagation*, and *PH-Deduction-Initial*) represent ways of detecting preferences that are not explicitly expressed in the dialogue. The following sections describe two of these, while the third (*Indirect-Propagation*) is described in Section 5.3.

Proposal History. As the problem solving process progresses, both the system and the user will propose various actions. These proposals can be made explicitly as in "You should take CS360" or "I would like to take CS882." They can also be made implicitly. For example, relating a user's query such as "Who is teaching CS360?" to the existing domain plan might require inferring a chain of domain actions that are not already part of the plan, including actions such as *Take-Course(User, CS360).* These inferred actions explain *why* the user asked the question and are actions that the user is implicitly proposing be added to the plan.

We maintain a data structure called a *Proposal History* (PH) which keeps track of proposals by recording the status of the proposal (accepted, rejected, current, or suspended), and the information that the system has explicitly given the user about the proposal (such as the instructor and meeting time of a course for a proposed *Take-Course* action). The PH allows the system to record trends in the user's processing of proposed solutions so that it can detect preferences which are not explicitly expressed. If, for instance, the user has rejected several proposals, the objects involved in these proposals may have common factors which could reflect a preference. For example, the user may have rejected several *Take-Course* actions, each of which involved a course that is known to meet early in the morning or to be very difficult. This checking is done not only with the rejected proposals in the PH but also with the accepted proposals.[4] The preferences detected by examining rejected proposals will be recorded as negative preferences and those indicated by accepted proposals as positive preferences.

Processing the PH enables us to capitalize on our use of endorsements. Associated with each proposed action in the PH is the set of attributes of objects involved in the action[5]

[4]There may be more than one accepted proposal in some domains — including the course advisement domain.

[5]These objects are the parameters of the action.

whose values have been conveyed during the dialogue.[6] Each time a proposal is accepted or rejected, these known characteristics will be recorded as possible preferences with the endorsement *PH-Deduction-Initial*. This is an extremely weak endorsement since little can be deduced from a single instance. Consider, for example, the following dialogue in which the user is considering taking CS360:

(12) U: Who is teaching CS360?

(13) S: Dr. Brown.

(14) U: What is the difficulty level?

(15) S: CS360 has a moderate difficulty level.

(16) U: I don't think I want to take CS360.

As the dialogue progresses and the user is informed of the professor and the meeting time of the course, the values of these attributes are recorded in the PH as known to the user. Once the user rejects CS360 as an alternative, these attribute values are recorded as possible negative preferences. The preferences recognized by processing the PH after this dialogue segment are shown in Figure 5. Note that in this dialogue, even though it is tempting to conclude that the moderate difficulty level of CS360 is the reason for rejecting the course, there is not enough evidence to warrant this conclusion. It might be the case that the user genuinely dislikes Dr. Brown but asks about the difficulty level anyway figuring that if the course is very easy, he might take it. While we feel that the structure of the dialogue does reveal preference information, more research is needed to determine precisely what can be deduced about user preferences from the order of the questions asked by the user and the point at which the user rejects an alternative.

A second *PH-Deduction-Initial* endorsement can combine with the first to result in a *PH-*

[6]In the future, stereotypical knowledge about attribute values will also be used.

preference	**preference strength**	**endorsement**
professor(_course,brown)	weak-2 (neg)	PH-Deduction-Initial
difficulty-level(_course,moderate)	weak-2 (neg)	PH-Deduction-Initial

FIGURE 5. PH Preferences for *Take-Course(User, _course)* for Utterances (12)-(16)

Deduction-Weak endorsement. This process of combining evidence occurs when the generation component accesses the model of user preferences for information about the preferences for a particular action, and is described in more detail in Section 5.5. Suppose that at a later point in the dialogue, the user is considering taking CS410 and the following exchange occurs:

(17) U: What is the difficulty level?

(18) S: CS410 has a moderate difficulty level.

(19) U: Who is teaching CS410?

(20) S: Dr. Jones.

(21) U: I don't think I want to take CS410 either.

As before, the attribute values for professor and difficulty level are recorded in the PH as being known to the user. When the user rejects CS410 as an alternative, the evidence of two potential negative preferences is recorded in the model of user preferences. The negative preference for courses with a moderate difficulty level now has two pieces of evidence, both with the endorsement *PH-Deduction-Initial*, which will later combine to result in a single *PH-Deduction-Weak* endorsement. The preferences detected by processing the PH on both dialogue segments are shown in Figure 6. Note that the two PH-Deduction-Initial endorsements on difficulty-level both appear in Figure 6, since evidence is not combined until

preference	preference strength	endorsement
professor(_course,brown)	weak-2 (neg)	PH-Deduction-Initial
professor(_course,jones)	weak-2 (neg)	PH-Deduction-Initial
difficulty-level(_course,moderate)	weak-2 (neg)	PH-Deduction-Initial
	weak-2 (neg)	PH-Deduction-Initial

Figure 6. PH Preferences for *Take-Course(User, _course)* for Utterances (12)-(21)

an application module requests the user's preferences.

If the user later rejects a third course which he has been informed has a moderate difficulty level, the negative preference for courses with a moderate difficulty level would have three *PH-Deduction-Initial* endorsements. When the evidence is eventually combined, the resulting endorsement will be a *PH-Deduction-Strong*. Thus the number of proposals demonstrating the pattern will determine the endorsement on the preference when it is used to evaluate alternative actions that might be included in the user's plan.

Default Stereotypical Information. In order to capture preferences that are typical of a class of users, we maintain a set of stereotypical preferences. These preferences can be of any strength, but all will receive the endorsement *Stereotypical* when attributed to the user. Although stereotypical information can be useful, it is not as reliable as information that is detected through actual discourse. This is because stereotypes ascribe characteristics based on group membership as opposed to individual observation, and thus exceptions can occur. For example, while the stereotypical student has a strong preference against early morning classes, a substantial number of students are exceptions and like early classes. Thus the endorsement *Stereotypical* is low in our reliability ranking and can be overridden by preferences deduced from the dialogue.

5.3. Attaching Preferences to Actions

As mentioned in Section 2, preferences are not held in isolation but rather with respect to particular goals or actions. In our previous travel agent example, the preference for business traveler seating is only relevant when the action being considered is a *Take-Flight* action. In fact, a user may prefer different values for the same attribute when considering different actions. For example, an individual might prefer a moderately difficult course when they are going to take the course for credit, while they prefer difficult courses when they are going to audit the course. For this reason, our system attaches the recognized user preferences to relevant actions.

The relevant actions are those that are under consideration by the user at the time when the preference is expressed. They consist of all actions on the path from the root node of the domain plan (which is represented as a tree) to the node marked as the current focus of attention. Therefore, if the user is building a sub-plan to take a course and we recognize a preference for early morning courses, we attach this preference to the *Take-Course* action. When a preference is recognized, it is attached to all of the relevant domain actions in the user's inferred plan that contain the relevant object (in this case, _course) as a parameter. We call this direct attachment of the preference.

However, it is also the case that many preferences apply to several different actions. For example, the user might always prefer evening courses, regardless of whether they are auditing the course or taking it for credit. If the user has been building a plan for taking a course and we recognize a preference for evening courses, and then the user begins to discuss auditing a course, it would seem unnatural if the system simply discarded all of the preferences that had been recognized while discussing the *Take-Course* action. However, it is possible that some of the preferences are not the same for the *Audit-Course* action, as pointed out earlier. Therefore, we want to generalize the preferences as much as possible without making inappropriate assumptions about their applicability. To accomplish this, we make a distinction between the direct attachment and the indirect propagation of preferences.

We use indirect propagation to generalize the preference to actions that are not currently being considered in the user's inferred plan but to which the preference might apply. Indirect propagation attaches the preference to all of the actions in our action library which contain as a parameter the relevant object. In the example we have been discussing in which the user is building a plan which includes the *Take-Course* action, direct attachment attaches the recognized preferences to the *Take-Course* action, while indirect propagation would attach recognized preferences to other actions in the action library which contain _course as a parameter such as *Audit-Course*. Note that while we have reason to believe that this preference may apply to these additional actions, it is far from certain.

From the perspective of *Audit-Course*, indirect propagation represents another way of recognizing preferences, separate from the ones originally used to detect a preference from the dialogue. The *Indirect-Propagation* endorsement captures the kind of evidence used to deduce this preference. Because indirect propagation is not an extremely reliable way of recognizing preferences, the *Indirect-Propagation* endorsement has a reliability rating of *weak*. The weaker reliability rating of the *Indirect-Propagation* endorsement reflects our weaker confidence in the existence of the preference, and so the preference can be easily overridden by subsequent information. The strength of the indirectly propagated preference is the preference strength that was assigned to the preference when it was initially detected.

preference	**preference strength**
difficulty-level(_course,moderate)	mod-1 (pos)
meeting-time(_course,<10am)	mod-2 (neg)

FIGURE 7. Stereotypical Preferences for *Take-Course(_user, _course)*

5.4. Example of Preference Detection

The following example demonstrates how our preference recognition process detects preferences.[7] Suppose that the stereotypical user preferences for the action *Take-Course(_user,_course)* are those in Figure 7, and the following dialogue takes place.

(22) U: I need to satisfy my seminar course requirement.

(23) Which AI courses are there?

(24) S: CS882, CS883, and CS889.

(25) U: What is CS882?

(26) S: Natural Language Processing.

(27) U: Who is teaching CS882?

(28) S: Dr. Smith.

(29) U: Great. What is the difficulty level?

(30) S: CS882 has a moderate difficulty level.

(31) U: Good. What time does CS882 meet?

(32) S: 6pm.

(33) U: Then I don't want to take CS882 since I don't like evening courses.

[7]Section 7 describes the input given to the system.

(34) I prefer courses which meet before 10am.

In this dialogue, the first preference expressed is in Utterance (23) when the user asks about courses whose content is AI. Because this positive preference for courses whose content is AI is expressed by the user in the initial problem description, the conversational circumstance is *Volunteered-Background.* Since the preference is conveyed indirectly by the user in the form of a question the preference strength is *mod-1*, the middle strength in the *Volunteered-Background* range. (The ranges of possible preference strengths corresponding to the different conversational circumstances were shown in Figure 3). The endorsement is *Volunteered-Background.*

In Utterance (28) the system informs the user that the professor of CS882 is Dr. Smith, and in Utterance (29) the user comments positively on this information, thus expressing a positive preference for Dr. Smith as a professor. Because the user volunteered this information without prompting from the system, the conversational circumstance is *Volunteered.* Since the positive preference was directly stated by the user the preference strength is *mod-1*, the highest within the *Volunteered* range. The endorsement is *Volunteered.* Similarly, a positive preference for courses with a moderate difficulty level is recognized in Utterance (31) when the user comments on the information given by the system about the difficulty level of CS882. This evidence for a preference for courses with moderate difficulty levels is added to the *Stereotypical* evidence (from Figure 7) already in the model of user preferences.

In Utterance (33) the user rejects the proposal of taking CS882 and expresses a negative preference for courses that meet in the evening. Because this negative preference for evening courses is expressed as a reason for rejecting a solution, the conversational circumstance is *Reject-Solution.* Since the user has directly expressed this preference in a statement, the preference strength is *str-1*, the highest within the *Reject-Solution* range. The endorsement is *Reject-Solution.* In Utterance (34), the user continues to support his decision to reject CS882

preference	**preference strength**	**endorsement**	**Utterance**
content(_course,AI)	mod-1 (pos)	Volunteered-Background	(23)
professor(_course,smith)	mod-1 (pos)	Volunteered	(29)
difficulty-level(_course,moderate)	mod-1 (pos)	Stereotypical	
	mod-1 (pos)	Volunteered	(31)
meeting-time(_course,>6pm)	str-1 (neg)	Reject-Solution	(33)
meeting-time(_course,<10am)	mod-2 (neg)	Stereotypical	
	str-1 (pos)	Reject-Solution	(34)

Figure 8. Evidence for Preferences Associated with *Take-Course(User, _course)*

by conveying a positive preference for courses that meet before 10am. Since this positive preference is expressed as additional support for rejecting a solution, the conversational circumstance is again *Reject-Solution.* The preference strength is again *str-1* (although this time it has a positive strength) and the endorsement is *Reject-Solution.* This new piece of evidence for a preference for courses which meet before 10am is added to the *Stereotypical* evidence already stored in the model.

Figure 8 summarizes the preferences that are detected for the *Take-Course* action, the strength of the preferences, the endorsements attached to the preferences, and the utterance in which each preference was conveyed. This example does not illustrate use of the Proposal History since it only comes into play in longer dialogues.

5.5. Combining the Evidence

One of the problems that must be addressed when using endorsements is how to combine the various pieces of evidence that are accumulated during the course of a dialogue. When the generation component requests information about the user's preferences, all of the evidence about each preference must be combined into a single *strength-reliability* tuple (consisting of a preference strength and a reliability rating) and any conflicts resolved. While our algorithm provides for all possible combinations of evidence, truly conflicting evidence is rare. This process of combining evidence (the Evidence-Combination box in Figure 1) on page 10 is outlined in Figure 9 and described in the rest of this section.

In our algorithm for combining evidence and resolving conflicts, there are three basic steps. First, there may be multiple pieces of evidence resulting in multiple endorsements for a single attribute value and preference strength. For example, in Figure 8 there are two pieces of evidence for a user preference for courses with a moderate difficulty level with a *mod-1(pos)* preference strength but with different endorsements. This multiple evidence must be combined into an overall reliability rating for the preference strength. Second, there may be evidence for different preference strengths for a single attribute value. For example, in Figure 8 there is evidence for two different preference strengths for the user preference for courses which meet before 10am. Finally, there may be evidence for different preferred values for the same attribute. In Figure 8, there is both evidence of a preference with respect to a meeting time after 6pm and evidence of a preference with respect to a meeting time before 10am.

The algorithm for combining evidence is invoked each time the generation component calls upon the model of user preferences for information about the user's preferences pertaining to a particular action. While combining the evidence, a working copy of the model of user preferences is used so that all of the original evidence is retained. This is done so that additional evidence provided by subsequent dialogue will be allowed to interact with all of the evidence extracted previously. To better explain our algorithm, we will continue our example from Section 5.4 and also work with an example containing a different set of preferences as we explain each of the three steps in more detail.

```
Combine-evidence(_user-model, _action, _param)
/* _user-model: current evidence of user preferences */
/* _action: the domain action under consideration */
/* _param: a parameter of _action for which preferences are requested */

/* Get evidence set of triples of form (_pref,_str,_endmt) where _str is a strength, _endmt is an */
/* endorsement, and _pref is itself a triple consisting of _param, an attribute, and an attribute value */
_evid-set ← retrieve-evidence(_user-model, _action, _param)

_evid-set ← combine-proposal-history-endorsements(_evid-set)

/* Combine evidence for a single strength of an attribute-value preference */
_evid-set-1 ← nil
_pref-set ← {(_attr,_attr-value,_str) | ∃(_x1, _x2, _x3) ∈ _evid-set ∧ _attr = get-attribute(_x1) ∧
                                         _attr-value = get-attribute-value(_x1) ∧ _str = _x2}
For each _prefstr ∈ _pref-set, repeat
      _evid-set-1 ← _evid-set-1 ∪ combine-elements(_evid-set,_prefstr)

/* Resolve evidence for different strengths of an attribute-value preference */
_evid-set-2 ← nil
_value-set ← {(_attr,_attr-value) | ∃(_x1, _x2, _x3) ∈ _evid-set-1 ∧ _attr = get-attribute(_x1) ∧
                                         _attr-value = get-attribute-value(_x1) }
For each _value ∈ _value-set, repeat
      _evid-set-2 ← _evid-set-2 ∪ combine-evidence-for-same-attribute-value(_evid-set-1,_value)

/* Resolve evidence of preferences for different values of an attribute */
_evid-set-3 ← nil
_attr-set ← {_attr | ∃(_x1, _x2, _x3) ∈ _evid-set-2 ∧ _attr = get-attribute(_x1)}
For each _attr ∈ _attr-set
      Begin
      If only-one-occurrence(_attr,_evid-set-2) ∨ disjoint-attribute(_attr)
         Then _evid-set-3 ← _evid-set-3 ∪ collect-evidence(_attr, _evid-set-2)
         Else _evid-set-3 ← _evid-set-3 ∪ resolve-different-preferred-values(_attr,_evid-set-2)

/* Return hypotheses about user preferences */
Return _evid-set-3
```

FIGURE 9. Outline of Evidence Combination

preference	preference strength	endorsement
professor(_course,brown)	mod-1 (pos)	Question-and-Answer
professor(_course,jones)	weak-1 (neg)	Volunteered
meeting-time(_course,<10am)	mod-1 (pos)	Volunteered-Background
	mod-2 (pos)	Volunteered
difficulty-level(_course,moderate)	mod-1 (pos)	Volunteered-Background
difficulty-level(_course,easy)	weak-2 (pos)	PH-Deduction-Initial
	weak-2 (pos)	PH-Deduction-Initial
	weak-2 (pos)	Stereotypical

Figure 10. Example(A) Preferences for *Take-Course(U,_course)*

Combining Multiple Endorsements for a Single Preference Strength. The first step of our algorithm combines multiple endorsements for a single attribute value and preference strength into a single reliability rating. It is at this stage of processing that several weaker pieces of evidence can combine to provide stronger cumulative evidence. It is also at this point that the PH-Deduction endorsements can combine as described in Section 5.2. After the PH-Deduction endorsements have been combined, the specific endorsements are replaced by their reliability ratings since we are concerned not with how the preference was detected, but only with how reliable the individual pieces of evidence are. If there are two pieces of evidence with the same reliability rating, they combine in strength to result in one reliability rating of the next higher level. This process is continued until there are no pieces of evidence with the same reliability rating. The highest reliability rating is then retained since it best represents the cumulative evidence for this preference strength.

In order to demonstrate the manner in which several weak pieces of evidence can combine to provide stronger cumulative evidence in this step, assume that the preferences in Figure 10

preference	**preference strength**	**reliability rating**
professor(_course,brown)	mod-1 (pos)	Strong
professor(_course,jones)	weak-1 (neg)	Strong
meeting-time(_course,<10am)	mod-1 (pos)	Very-Strong
	mod-2 (pos)	Strong
difficulty-level(_course,moderate)	mod-1 (pos)	Very-Strong
difficulty-level(_course,easy)	weak-2 (pos)	Moderate

Figure 11. Example(A) Preferences after Performing Step(1)

have been detected. There is multiple evidence for a user's preference for courses with a difficulty level of *easy*, all with the strength *weak-2*. Because the preference strength for each piece of evidence is equal, the endorsements are allowed to combine. First, the PH-Deduction endorsements are combined. The two *PH-Deduction-Initial* endorsements combine to result in a *PH-Deduction-Weak* endorsement. The endorsements are then replaced by their reliability ratings. Both the *PH-Deduction-Weak* endorsement and the *Stereotypical* endorsement have reliability ratings of *Weak*. These two pieces of evidence combine, and the result is an overall reliability rating of *Moderate*. The result of performing the first step of the algorithm on the preferences in Figure 10 is shown in Figure 11.

Our second example, example(B), utilizes the preferences shown in Figure 8. Each specific endorsement is first replaced by its corresponding reliability rating. The only preference in Figure 8 with multiple evidence for the same preference strength is the preference for courses with a moderate difficulty level, which has two pieces of evidence. The first piece of evidence is a *Stereotypical* endorsement with a reliability rating of *Weak* and the second is a *Volunteered* endorsement which has a reliability rating of *Strong*. Because these endorsements do not

preference	**preference strength**	**reliability rating**
content(_course,AI)	mod-1 (pos)	Very-Strong
professor(_course,smith)	mod-1 (pos)	Strong
difficulty-level(_course,moderate)	mod-1 (pos)	Strong
meeting-time(_course,>6pm)	str-1 (neg)	Very-Strong
meeting-time(_course,<10am)	mod-2(neg)	Weak
	str-1 (pos)	Very-Strong

Figure 12. Example(B) Preferences after Performing Step(1)

have the same reliability rating, they cannot combine to yield a stronger rating. Instead, only the highest reliability rating is retained – in this case, the resulting reliability rating for the preference is *Strong*. The result of performing the first step of the algorithm on the preferences in Figure 8 is shown in Figure 12.

Resolving Evidence for Multiple Preference Strengths. The second step of our algorithm for combining evidence involves resolving evidence for different preference strengths for the same preferred attribute value. After the completion of the first step, there is only one reliability rating for each preference strength, but there still may be multiple strength-reliability tuples for the same attribute-value preference. This second step is concerned with resolving the evidence for different preference strengths so that there will be only one strength-reliability tuple associated with a particular attribute value preference.

For each attribute-value preference, the various strength-reliability tuples are sorted by their reliability rating and then the conflicts are resolved pair-wise beginning with those whose reliability rating is greatest, until there is only one strength-reliability tuple remaining. In resolving a pair of strength-reliability tuples, the reliabilities are considered first. If the reliability ratings differ by two or more levels of reliability, the tuple with the higher reliability is retained. This is done because the evidence for the higher reliability tuple strongly outweighs the evidence for the lower reliability tuple.

If the tuples differ by less than two levels of reliability, the quality of evidence for each preference strength is comparable, and the preference strengths themselves must be taken into account. If the preference strengths differ by one or two degrees, the evidence is not really in strong conflict. If the preference strengths differ by only one degree, the stronger reliability rating is paired with the weaker preference strength – this allows our system to maintain the reliability of the evidence (since the two pieces of evidence are quite similar) without overstating the importance of the preference to the user. If the preference strengths differ by two degrees, then there is more conflict between the two pieces of evidence and so the weaker reliability rating is retained along with the average of the preference strengths. If, however, the preference strengths differ by more than two degrees, the evidence is in conflict, and since both pieces of evidence are of approximately the same reliability, it is difficult to determine the actual strength of the user's preference. In this case, nothing about this preferred value is returned to the generation component at this time. Notice, however, that all of the original evidence is maintained in the preference model so that evidence could be detected at a later point in the dialogue which could resolve the conflict.

The second step of the algorithm will first be illustrated with the example(A) preferences in Figure 11. There was evidence for two different preference strengths for the user's preference for courses that meet before 10am. Because the reliability ratings differ by only one degree, the algorithm takes into account the preference strengths. The preference strengths also differ by one degree, and so the conflict can be resolved. The stronger reliability rating, *Very-Strong*, is paired with the weaker preference strength *mod-2 (pos)*. The result of the second step of the algorithm is shown in Figure 13.

preference	preference strength	reliability rating
professor(_course,brown)	mod-1 (pos)	Strong
professor(_course,jones)	weak-1 (neg)	Strong
meeting-time(_course,<10am)	mod-2 (pos)	Very-Strong
difficulty-level(_course,moderate)	mod-1 (pos)	Very-Strong
difficulty-level(_course,easy)	weak-2 (pos)	Moderate

Figure 13. Example(A) Preferences after Performing Step(2)

preference	preference strength	reliability rating
content(_course,AI)	mod-1 (pos)	Very-Strong
professor(_course,smith)	mod-1 (pos)	Strong
difficulty-level(_course,moderate)	mod-1 (pos)	Strong
meeting-time(_course,>6pm)	str-1 (neg)	Very-Strong
meeting-time(_course,<10am)	str-1 (pos)	Very-Strong

Figure 14. Example(B) Resolved Model of User Preferences

Next we will apply the second step of the algorithm to the example(B) preferences in Figure 12. The only preference that has evidence for multiple preference strengths is the preference for courses that meet before 10am. This preference has two strength-reliability tuples – a (mod-2(neg),Weak) and a (str-1(pos),Very-Strong). Since the tuples differ by more than two degrees of reliability, the (str-1(pos),Very-Strong) tuple is retained. The result after this second step of the algorithm is shown in Figure 14.

Resolving Different Preferred Values. Next we must resolve evidence for different preferred values for the same attribute of an object. We have differentiated three different types of attributes: disjoint, scalar, and complex. A disjoint attribute is an attribute for which preferences for different values of the attribute do not conflict. A preference for a specific professor is an example of a disjoint attribute preference. Preferences of this type do not conflict at all. A user might have positive preferences for two professors, and a negative preference for another, and the various preferred values should all be retained. Scalar attributes, on the other hand, cannot have multiple preferred values. A scalar attribute is an attribute whose values can be viewed as a scale and preferences for values at different points on the scale conflict. A preference for the difficulty level of a course is an example of a scalar attribute preference. It does not make sense to prefer both very difficult and very easy courses, and so different preferred values for scalar preferences must be resolved. The third type of attribute, complex, is also an attribute whose values can be arranged on a scale. However, preferences for values of a complex attribute may fall at different points on the scale without conflicting. The meeting time of a course is an example of a complex attribute preference. In this case, different preferred values may or may not conflict. For example, a user might like evening courses as well as liking early morning courses. As long as the preferred values for this type of preference do not intersect, they do not conflict.

The methods for resolving conflicts between preferred values differ for each of the three types of attributes. Since differing preferred values for disjoint attributes do not conflict, no resolution is necessary. Only the basics of the conflict resolution for the scalar and complex attributes will be described here, since much of the detail of the algorithm deals with conditions which rarely (if ever) occur in this type of collaborative communication.

For complex attributes, preferred values can be represented as ranges of values. For example, a user might like courses which meet after 6pm, before 10am and/or between

1pm and 3pm. The various preferred values do not conflict unless these ranges intersect in some manner. In resolving conflicting ranges, the reliability of the strength-reliability tuples is considered first. As before, if the reliabilities differ by two or more levels, the higher reliability evidence overrides the lower reliability evidence. If the reliabilities are comparable, the methods for resolution depend on the types of relations involved, and all of the possible combinations of before (denoted by $<$), between, and after (denoted by $>$) are handled. The algorithm concentrates on separating, where possible, the conflicting ranges into two or more non-conflicting ranges. In cases where conflicts cannot be resolved, no information is returned to the generation component about the conflicting preferred values.

The conflict resolution for scalar attributes is somewhat simpler, since the preferred values are points on a scale rather than ranges. However, the principles behind the resolution remain the same as in the case of complex attributes. The reliability of the evidence is considered first, and strength-reliability tuples which differ by two or more degrees of reliability are resolved by retaining only the tuple with the higher reliability evidence. If, however, the strength-reliability tuples differ by less than two degrees of reliability, the attribute values themselves are taken into consideration. If the two values are relatively close[8] on the scale and their preference strengths are similar, an average preferred attribute value is computed as well as a conservative estimate of the preference strength and reliability.[9] If, however, the preferred values are further apart on the scale, the fact that the reliability of the evidence is comparable indicates a conflict, and no information about this preference is reported to the generation component at this time.

This third step of the algorithm will first be illustrated with the example(A) preferences

[8] "Closeness" is determined relative to the size of the scale – we are currently treating values differing by .4 or less of the total number of values on the scale as close.

[9] The new strength-reliability tuple is constructed as described in the previous section entitled *Resolving Different Preferred Values*.

preference	**preference strength**	**reliability rating**
professor(_course,brown)	mod-1 (pos)	Strong
professor(_course,jones)	weak-1 (neg)	Strong
meeting-time(_course,<10am)	mod-2 (pos)	Very-Strong
difficulty-level(_course,moderate)	mod-1 (pos)	Very-Strong

Figure 15. Example(A) Resolved Model of User Preferences

shown in Figure 13. There is evidence for a positive preference for Dr. Brown as the professor of a course and a negative preference for Dr. Jones as the professor of a course. Because the professor of a course is a disjoint attribute, these preferences do not conflict and no resolution is necessary. There is also evidence for two different preferred attribute values for the difficulty level of a course. Because the difficulty level of a course is a scalar attribute, the conflict must be resolved. The two pieces of evidence have reliability ratings of *Very-Strong* and *Moderate*. Because these reliability ratings differ by two degrees, the conflict is resolved by retaining the preferred value with the highest reliability rating along with its strength-reliability tuple – in this case, the preference for courses with a moderate difficulty level along with its (mod-1(pos),Very-Strong) tuple. This produces the final resolved model of user preferences shown in Figure 15.

In example(B), for which the preferences are shown in Figure 14, there is evidence for a positive preference for courses which meet before 10am as well as evidence for a negative preference for courses which meet after 6pm. Preferences for the meeting time of a course are classified as complex preferences, and since the two ranges of <10am and >6pm do not intersect, no resolution is necessary.

Recall that evidence is combined only when the generation component (or some other application program) requests information about the user's preferences. Once the recognition component has combined the evidence to produce a model of the user's preferences, it extracts those preferences whose reliability rating exceeds a threshold specified by the generation component and passes them to that subsystem for use in generating helpful responses.

5.6. Discussion

Our system recognizes preferences by analyzing each user utterance and the conversational circumstances in which it occurs and by detecting patterns in the user's acceptance or rejection of proposals. Endorsements are used to represent the accumulated evidence for each hypothesized preference, and heuristics are used to combine individual pieces of evidence in order to produce an overall estimate of the reliability of a hypothesis.

Machine learning is a powerful tool for constructing classification rules. If we had a large body of data regarding the user's evaluation (perfect, good, ok, etc.) of different instantiated actions given the attributes of the objects used to instantiate the parameters, then a machine learning procedure such as C4.5 [Quinlan, 1993] could be used to induce a decision tree for classifying action instances based on the user's preferences of attribute values. However, since such training data is not readily available for individual users, this use of machine learning would require a user model acquisition phase in which the user is asked to explicitly evaluate different possible actions. One objective of our research was to avoid such an explicit acquisition phase. On the other hand, machine learning techniques might be applied to learning from the data collected in the proposal history. However, since machine learning typically works best when there is a large amount of data, we chose instead to rely on heuristics.

A number of approaches have been developed for reasoning under uncertainty, including certainty factors[Shortliffe, 1976], Dempster-Shafer[Shafer, 1976], and Bayesian systems[Pearl, 1988]. We chose to follow Cohen[Cohen, 1985] and adopt an endorsement-based approach to avoid assigning numerical values to evidence; instead we use heuristics to combine symbolic endorsements. Recently, Jameson[Jameson *et al.*, 1995] has discussed how our representations could be captured in a Bayesian network. Further experimentation is needed to assess the implications of doing so and to identify the best mechanism for representing and reasoning about user preferences.

6. USING USER PREFERENCES IN RESPONSE GENERATION

In a collaborative environment, if the consultant knows of a substantially superior alternative to an action proposed by the user, but does not suggest it to the user, she cannot be said to have fulfilled her responsibility as a collaborative agent. Thus, an intelligent consultation system must use its beliefs about the user's preferences to identify user proposals that it believes are suboptimal and to suggest better alternatives to the user. When a user has proposed an action, either explicitly (e.g., "Can I take CS360?") or implicitly (e.g., by asking a question such as "Who is teaching CS360?", from which our plan recognition component [Lambert and Carberry, 1991] would infer a *Take-Course(User,CS360)* action), our response generation system [Chu-Carroll and Carberry, 1994] will evaluate, for each action proposed by the user, all the possible instantiations of its parameters, and determine whether a substantially better alternative to the proposal exists. To accomplish this, the system invokes the *ranking advisor*, outlined in Figure 16 and described in the next section, to rate different instantiations of a proposed action that will accomplish the user's goal. If a substantially better alternative is identified, the system will propose the alternative action and support its proposal with the attribute values that it believes make the alternative preferable to the user. In the future, we will extend our system to consider alternative generic actions that will accomplish the user's goal, rather than just alternative instantiations of the actions proposed

Ranking-advisor(_param,_obj-user,_action)
/* *_param: a parameter of _action that is instantiated by the user's query* */
/* *_obj-user: the user's instantiation of _param* */
/* *_action: the domain action under consideration* */

_attr-set ← get-attributes(_param)
_obj-set ← get-possible-instantiations(_param,_action)
/* *Obtain user preferences from preference modelling module* */
_pref-set ← request-reliable-preferences(_param,_action,Reliability-Threshold)

/* *evaluate each possible instantiation of _param in _action* */
_obj-tuples ← {(_obj,_rating) | _obj ∈ _obj-set ∧

$$\text{\_rating} = \sum_{\_attr \in \_attr-set} [\text{get-pref-strength(\_attr)} \times \text{closeness-of-match(\_attr,value(\_attr,\_obj),\_pref-set)}]\}$$

/* *Compare object that has the best rating with object proposed by user* */
_best-obj ← get-tuple-of-max-rating(_obj-tuples)

$$\text{If threshold} < \text{get-rating(\_best-obj)} - \sum_{\_attr \in \_attr-set} [\text{get-pref-strength(\_attr)} \times \text{closeness-of-match(\_attr,value(\_attr,\_obj-user),\_pref-set)}]$$

  Then Return get-object(_best-obj)
  Else Return nil

Figure 16. Outline of Mechanism for Identifying a Better Alternative

by the user.

6.1. The Ranking Advisor

The ranking advisor's task is to determine how the parameters of an action can best be instantiated, based on the user's preferences. For each object that can instantiate a parameter of an action (such as CS883 and CS889 for instantiating _course in *Take-Course(User,_course)*), the ranking advisor extracts the values of its attributes from the knowledge base and queries the preference recognition component for the user's preferences relevant to the action. A threshold is used to eliminate preferences with low reliability ratings so that only preferences with reliability ratings above a certain level will be allowed to affect the system's decision.

Since suggesting better alternatives requires rejecting actions proposed by the user and also interrupts the flow of an on-going discourse, a collaborative system should not attempt to do so unless it is reasonably certain that the alternative will indeed be preferred by the user; therefore for the purpose of suggesting better alternatives, our system uses a threshold of *Moderate* so that only preferences with reliability ratings of at least *Moderate* are considered.

The ranking advisor takes into account two factors when ranking the candidate instantiations: the *strength* of a user preference for an attribute value and the *closeness of a match*. The preference strength, as discussed earlier, represents the importance the user attaches to a preference. The closeness of the match, which can have a value of *exact, strong, weak*, or *none*, measures how well the actual and the preferred values of an attribute match. It is measured based on the *distance* between the two values where the unit of measurement differs depending on the type of the attribute. For example, for disjoint type attributes (such as the professor of a course), the closeness of the match is either *exact* or *none* since a partial match is not possible for disjoint attributes. For scalar attributes (such as the difficulty of a course), the unit of measurement is proportional to the distance between the endpoints of the scale: if the distance between two values is less than or equal to 1/4 of the maximum distance, the two values have a *strong* match; if the distance is between 1/4 and 1/2 of the maximum distance, they have a *weak* match; and if the distance is greater than 1/2 of the maximum distance, the closeness of the match between them will be *none*. For instance, the distance between *moderate* and *very-difficult*[10] is two, which is 1/2 of the maximum distance; thus the match between them is *weak*. The closeness of the match must be modeled in order to capture the fact that if the user prefers difficult courses, a moderate course will be considered preferable to an easy one, even though neither of them exactly satisfies the user's preference.

For each candidate instantiation, the ranking advisor records the preference strength for

[10]Based on a scale of *very-difficult*, *difficult*, *moderate*, *easy*, and *very-easy*.

the relevant attributes and computes the closeness of each match. A rating is computed for each candidate instantiation by summing the products of corresponding terms of the preference strength and the closeness of the match, which corresponds to a weighted *additive* model of human decision-making [Keeney and Raiffa, 1993, Payne *et al.*, 1993]. The instantiation with the highest rating is considered the *best* instantiation for the action under consideration for this particular user. If the best instantiation is different from the one proposed by the user and substantially better, the system will suggest the better alternative to the user's proposal.

Example of Ranking Two Objects. Suppose the dialogue shown in utterances (22) - (34) is continued as follows:

(35) U: What is CS883?

Our plan recognition component would infer *Take-Course(User,CS883)* as a domain action that the user is considering.

We demonstrate the ranking advisor by showing how the two remaining candidates from the dialogue, CS883 and CS889, of the *Take-Course* action are ranked. The ranking advisor obtains user preferences for the *Take-Course* action, shown in Figure 14, from the preference recognition component, and extracts the attribute values for the two instantiations (Figure 17) from the knowledge base.

The ranking advisor matches the user's preferences against the domain knowledge for each candidate instantiation. For each attribute that the user has indicated a preferred value, the advisor records its preference strength and the closeness of the match for each instantiation (in this case, CS883 and CS889). For instance, in considering the preference *meeting-time(_course, <10am)*, the preference strength will be *str-1(pos)*, and the closeness

Domain Knowledge:

CS883:	**CS889:**
professor(CS883,brown)	professor(CS889,smith)
difficulty-level(CS883,moderate)	difficulty-level(CS889,easy)
workload(CS883,light)	workload(CS889,moderate)
meeting-time(CS883,2-3:15pm)	meeting-time(CS889,10:30-11:45am)
content(CS883,{expert systems, AI})	content(CS889,{planning, AI})

Figure 17. Domain Knowledge for the Ranking Advisor

of the match will be *none* and *strong* for CS883 and CS889, respectively.[11] Table 1 shows a summary of the preference strength and the closeness of the match for the attribute values for both candidates.

The ranking advisor then assigns numerical values to each preference strength and closeness of match: values from 6 to -6 are assigned to preference strengths from *str-1(pos)* to *str-1(neg)*, and values from 3 to 0 are assigned to closeness of matches from *exact* to *none*. The preference strength and the closeness of the match for each attribute are then multiplied to attain a weight for the attribute, and these attribute weights are summed to attain an overall evaluation for each candidate instantiation. A normalized evaluation for each candidate is then calculated by dividing its overall evaluation by the maximum possible evaluation.[12] In this example, the normalized weight for CS883 is 24/54 and that for CS889 is 44/54; thus taking CS889 is considered a substantially superior alternative to taking CS883 for this

[11]The unit of measurement for the complex attribute type *meeting-time* is one-hour intervals.

[12]The maximum possible evaluation is the evaluation that will be assigned to an alternative when the closeness of matches for all attributes with positive preferences are *exact*, and those for attributes with negative preferences are *none*.

CS883	Preference Strength		Closeness of Match		
Content	mod-1 (pos)	4	exact	3	12
Professor	mod-1 (pos)	4	none	0	0
Difficulty-level	mod-1 (pos)	4	exact	3	12
Meeting-time	str-1 (neg)	-6	none	0	0
Meeting-time	str-1 (pos)	6	none	0	0
					24

CS889	Preference Strength		Closeness of Match		
Content	mod-1 (pos)	4	exact	3	12
Professor	mod-1 (pos)	4	exact	3	12
Difficulty-level	mod-1 (pos)	4	strong	2	8
Meeting-time	str-1 (neg)	-6	none	0	0
Meeting-time	str-1 (pos)	6	strong	2	12
					44

Table 1. Preference Strengths and Closeness of Matches

particular user.

6.2. Suggesting Better Alternatives

If the ranking advisor finds a substantially superior alternative to the action proposed by the user, our generation system will notify the user of this, suggest its proposed alternative, and justify this proposal with evidence. The supporting evidence is selected by examining the information used by the ranking advisor (Table 1) and selecting the attributes that make

CS889 substantially better than CS883. In this example, the attributes that cause CS889 to be preferred over CS883 are *professor* and *meeting-time*; thus these two attributes and their values are used as justification for the suggested alternative. This process causes the system to generate the logical form of the following utterances:

(36) S: Taking CS889 is a better alternative than taking CS883.
(37) The professor of CS889 is Dr. Smith.
(38) The meeting time of CS889 is 10:30-11:45am.

These utterances initiate a negotiation with the user concerning the desirability of the user's proposed plan. Whereas Cawsey *et al.*'s system [1993] initiates negotiation dialogues with the user based on discrepancies between the system's and the user's domain-related beliefs and intentions, our system further takes into account detected better alternatives based on user preferences.[13]

Note that our system supports its suggested alternative by presenting the attribute values that make it substantially better than the user's proposal. This implicitly conveys the system's belief that the user would prefer these attribute values. Several questions arise regarding what information to convey to the user. The first is whether the system should more explicitly convey the preferences that prompted its decision. We have chosen to implicitly convey them in order to reduce the verbosity of the response, but further research is needed to determine the effectiveness of different approaches. In addition, we have chosen not to provide the user with a model of the preferences that the system is using in its decision-making. Although human agents appear to make use of an inferred model of agent preferences, they do not share this model with the other agent except to use aspects of it as

[13]For details on how our system initiates negotiation to resolve conflicting beliefs between the system and the user, see [Chu-Carroll and Carberry, 1998].

support for their proposals. Moreover, since the system's model changes over the course of the dialogue, keeping the user apprised of such changes would cause serious digressions from the focus of the dialogue. Future research will investigate how much of the preference model to convey to the user if the user questions the system's suggestions.

The second question is whether the system should provide an extended comparison between the suggested alternative and the user's proposal. Our preliminary evaluation, discussed in the next section, suggests that if the user's proposal has some highly desirable attribute values that are not shared with the system's alternative suggestion, then a comparison is warranted in which these attribute values are noted. Lastly, since preferences may change over time or be influenced by external factors of the current situation that the system is unaware of, the question arises as to whether the system should notify the user of other alternatives. Our current system only informs the user of its suggested alternative since there are often many other potential choices. However, one possibility that could be explored in future research is to suggest additional alternatives that match stereotypical preferences if these have not been overridden by preferences extracted from the current dialogue.

7. IMPLEMENTATION AND EVALUATION

Both the preference recognition module and the response generation module have been implemented in Common Lisp on a Sun Sparcstation. The recognition system maintains a representation of the proposal history and of evidence for possible user preferences. It is invoked either with a request to process a new utterance or a request to report the system's current hypothesis about the user's preferences. When asked to process a new utterance, the system is given as input the semantic representation of the utterance along with a dialogue model containing the current discourse and domain plans.[14] The semantic representation captures the logical meaning of the utterance as a surface speech act. From this semantic representation, the system can classify utterances as needed for recognizing user preferences. For example, a WH-question from the user to the system is represented as (Surface-WH-question USER SYSTEM _item _prop) where _item is one of the arguments of _prop and the user is requesting the values of _item that satisfy _prop. _prop is a proposition consisting of an attribute and two arguments, where the first argument specifies an object and the second is the attribute's value for that object. So for example, the utterance "Which AI courses are there?" would have a semantic representation in which the user specifies the value of the second argument (AI) and requests the value of the first argument; thus the user is requesting all objects with a specific attribute value and the system considers the possibility of a user preference for this attribute value. On the other hand, the utterance "When does CS882 meet?" has a semantic representation in which the user requests the value of the second argument (the attribute value) for CS882, and no attribute preference is hypothesized at this time. Utterances such as "Good" are represented as a surface speech act designating approval.

The recognition system analyzes each new utterance and the dialogue model to detect any evidence of user preferences and stores each piece of evidence as a preference strength (for or against an attribute value) with an associated endorsement. Only when an application module, such as the ranking advisor, invokes the preference recognition component with a request for information about the user's preferences does the recognition module combine the evidence and report a set of user preferences. The ranking advisor uses the reported

[14]The domain plan captures the task-related actions under consideration (such as *Take-Course*) and the discourse plan captures the communicative actions being performed (such as *Obtain-Information, Inform,* or *Reject*).

preferences to determine whether there is a better alternative to the user's proposal; if a better alternative is found, the generation system produces the logical form for a response that informs the user of this alternative and supports it by giving the features of the suggested alternative that make it better than the user's proposal. Since at most two types of speech acts are produced, one that provides a recommendation and one that provides support for the recommendation, our generation system adopts the claim-first strategy (what Cohen refers to as a pre-order transmission[Cohen, 1987]) by first presenting the recommendation followed by the evidence supporting it. We leave for future work the investigation of principled mechanisms for selecting among this and other possible organizational strategies, as well as realization of speech acts in natural language utterances.

In order to obtain an initial assessment of our techniques and to identify areas for further work, we performed two studies evaluating our preference recognition and response generation strategies. However, the studies were not intended to be a full-scale evaluation such as would be provided by integrating our algorithms into an actual interactive advisement system.

The first study focused on the recognition of user preferences. Six human subjects, who were unfamiliar with our system, were presented with a set of hypothetical dialogues between a student and an advisor, along with a set of stereotypical student preferences (such as that students typically have a strong preference against courses that meet before 10am). At various points during each dialogue, the subjects were given a model of the advisor's beliefs about the student's preferences and were asked to determine whether these beliefs were reasonable and whether alternative preferences should be hypothesized. The dialogues were selected so that they exercised the features of our preference recognition algorithm.

An analysis of the responses and extensive commentary provided by the human subjects showed that the main disagreement between the human subjects and the system concerned the system's extraction of preferences from the proposal history.[15] Two of the subjects argued that an accepted proposal indicates *no dislike* for the attributes of the proposal rather than a *preference for* these attributes. However, we only derive a very weak preference in such cases, and we associate with it an endorsement that is so weak that the preference is not used in response generation until it is supported by further accepted proposals or other evidence.

Recall from Section 5.2 that we take a very conservative approach to deriving preferences from a sequence of rejected proposals. In particular, our current approach is to extract only very weak preferences from the proposal history and to treat all attributes of the proposal equally in assigning blame for the rejection. Two of the human subjects argued that a rejected proposal conveys a negative preference only for the attribute whose value was requested in the utterance immediately preceding the rejection. A third subject suggested that a stronger negative preference should be assigned to the last requested attribute than to the other queried attributes. Thus although the subjects agreed with extracting preferences from rejected proposals, there was disagreement about how the relative order of queried attributes should affect the derived preferences. In addition, three of the subjects argued that if a queried attribute is part of several rejected proposals, then a stronger negative preference should be recorded. Our current approach is to record a very weak negative preference but increase the system's confidence in this preference. The disagreement among the subjects indicates the need for further research on extracting preferences from the proposal history, which will be part of our future work.

The second study focused on the response generation component. A set of scenarios were formulated showing responses produced by our system along with one scenario that was a control question whose response had been altered to be different from what the system

[15] In addition, two of the subjects suggested that the system should decrease the strength it assigns to the preference conveyed indirectly via questions such as "Which AI courses are available?"

	Mean	Median	Standard Deviation
Direct Answer	4.9	5	0.28
Alternative 1	4.2	5	1.1
Alternative 2	4.1	4	1.0
Alternative 3	3.3	3	1.2
Alternative 4	3.9	4	1.1
Alternative 5	4.0	4	1.0
Alternative 6	4.2	5	0.99
Alternative 7	4.2	5	0.99
Control Question	1.8	1	1.0

Table 2. Response Generation Scenarios

actually produced. Of the eight scenarios that represented responses produced by our system, one was a situation in which the user's question was answered (because there was no better alternative) and the others were situations in which the system suggested a better alternative. In all of the scenarios, the following query was used:

U: *I want to satisfy my seminar course requirement.*

Who is teaching CS883?

For each scenario, the preferences of a hypothetical user with respect to the *Take-Course* action were shown along with the system's knowledge regarding characteristics of each alternative course. Based on this information, twelve human subjects were asked to rate the system's responses on a scale of *strongly agree, agree, ambivalent, disagree* and *strongly disagree*. Each rating was assigned a numerical value from 1 to 5, with 1 being *strongly disagree* and 5 being *strongly agree*, and the means, medians, and standard deviations were computed.

Table 2 presents the results of this study. The example in which the system directly answered the user's question (because there was no better alternative) received a near-perfect score, while the control question received a mean score of 1.8 and a median of 1. In the case of the seven examples in which the system suggested a better alternative, the subjects commented that they would have preferred that the system include in its response an answer to the user's question along with the suggested alternative. One of the examples in which the system suggested a better alternative to the user's proposal received a particularly low rating, with a mean of 3.3 and a median of 3. The system's response for this example was the following:

S: *Taking CS852 is a better alternative than taking CS883.*

The professor of CS852 is Dr. White.

For this example, several subjects suggested that a comparison between the suggested alternative and the user's original proposal should have been given, such as "Although CS883 has a moderate difficulty level, you might prefer CS852 because the professor of CS852 is Dr. White." The subjects suggested that such a comparison would be better in this instance since CS883 was preferable to CS852 in terms of its difficulty level and the user had a fairly significant preference for courses that were easy although he had a stronger preference for Dr. White being the professor. Thus the subjects agreed with the system's selection of a better alternative, although they suggested that the system call attention to the highly desirable attributes of the user's proposal that were not shared by the system's proposal.

Despite the aforementioned problems, in the seven examples where the system suggested better alternatives to the user's proposal, three examples received a median of 5, three received a median of 4, and one received a median of 3. This indicates that in most cases the majority of the subjects considered the system's responses to be cooperative and coherent.

8. RELATED WORK

Research on acquiring a user model has focused on recognizing a user's goals [Carberry, 1987, Allen and Perrault, 1980, Litman and Allen, 1987], level of expertise [Chin, 1989, Chin, 1988] and knowledge [Kass, 1991]. Research in response generation has concentrated on tailoring explanations and responses based on different aspects of the user model such as by taking into consideration user goals [McKeown, 1988, Sarner and Carberry, 1992, Cohen and Jones, 1989, Cohen *et al.*, 1989], the user's level of expertise, or background knowledge [Paris, 1988, Paris, 1985, Chin, 1989, Chin, 1988, Cohen and Jones, 1989, Cohen *et al.*, 1989, Mittal and Paris, 1993].

There have been several systems which take user preferences into account. However, these systems either obtain the information by explicit user model acquisition [Rich, 1979, Sycara and Miyashita, 1994, Morik, 1989], by repeatedly observing certain actions during performance of a task[Bauer, 1994, Bauer, 1995], or assume that the preferences are provided to them [van Beek, 1987, Ardissono and Sestero, 1995]. The Grundy system [Rich, 1979] utilizes user stereotypes in order to play the role of a librarian in making suggestions about particular books that a user might enjoy. Grundy begins by asking the user to select words to describe himself and then uses stereotypes to infer information about other facets of the user including preferred characteristics of books. Grundy then compares the user's preferences to its knowledge about particular books to propose a book to the user and give a description of the book. If the user doesn't like the suggestion, Grundy begins to ask questions to determine which aspects of the user model are incorrect. The user model acquisition in this system is system-driven and explicit, with the system asking the user questions in order to obtain the desired information.

Sycara and Miyashita [1994] developed a case-based system for capturing user preferences about optimization criteria in job shop scheduling. During the knowledge acquisition phase, the user evaluates the outcome of the system's revision of suboptimal schedules. Whenever the user deems a revised schedule to be unacceptable, the system asks the user for an explanation in terms of the importance of individual features of the revised schedule to the unsatisfactory evaluation. This information is captured as cases and is subsequently utilized by the system to identify acceptable schedules without the user's intervention.

Morik's HAM-ANS system [1989] is designed to play the role of a hotel clerk and to make recommendations about hotel rooms based on the user's inferred criteria or preferences. The "specification" or user model acquisition phase is system-driven. The system asks the user questions about her planned stay at the hotel, uses the explicit specifications given by the user to activate user stereotypes, then creates evaluation criteria from these stereotypes. Based on the evaluation criteria, the system makes a recommendation to the user about the user's best suited room category. After this recommendation has been made, the user is allowed to ask questions of the system about the room category. The system utilizes the evaluation criteria to appropriately answer the user queries. As a step towards dynamic user model acquisition during the consultation dialogue, Morik does propose using the notion of "isotopy" or the co-occurrence of semantic features to infer new evaluation criteria from the user's questions as the dialogue progresses. However, while this idea might result in the inference of some of the user's criteria, it seems unlikely that the use of isotopy alone could result in a sufficient model. In surveying user modeling systems, Morik notes that "integrating the acquisition of the user image into the normal processing of the dialog is lacking for almost all systems" and that "the revision of user models is another weak point of current systems." [Morik, 1989, page 376] Our system addresses both of these points with its dynamic recognition of user preferences, and its continual updating of the model throughout the dialogue in combination with our methods for resolving conflicting evidence.

Van Beek [van Beek and Cohen, 1986, van Beek, 1987, van Beek *et al.*, 1993] developed a system that produces responses by taking into account the overall plans and goals hypothesized for a user, based on [Joshi *et al.*, 1984]. His user goals include what our system treats as preferences; thus instead of evaluating a user plan with respect to user preferences, van Beek's system evaluates a plan based on the number of goals[16] it achieves. However, we contended in Section 2 that preferences should not be treated as goals. Furthermore, in treating preferences as goals, van Beek's system does not take into account the strength of the preferences and closeness of the matches in evaluating user proposals. We argue that this does not accurately model the way humans evaluate alternative plans. In addition, van Beek's system assumes that user preferences and goals are given; thus it is not capable of dynamically recognizing user preferences during the course of a dialogue.

Bauer[Bauer, 1994, Bauer, 1995] and Ardissono and Sestero[Ardissono and Sestero, 1995] utilize agent preferences to distinguish among otherwise mutually plausible hypotheses about the user's plans and goals. Bauer's system monitors a user working in an application domain and identifies preferences from the user's repetition of particular action sequences and the co-occurrence of certain actions with particular plans. Bauer's approach for recognizing preferences is well-suited to applications such as his domain of providing help for email, where it is common for a goal to arise repeatedly and to be addressed in a similar manner each time that it occurs. However, it is unclear whether more complex consultation domains, such as student advisement, would exhibit this feature of repeated action sequences. Ardissono and Sestero's system used preferences extracted from explicit statements by the user or from user stereotypes. Further work by Ardissono and Cohen[Ardissono and Cohen, 1996] extended van Beek and Cohen's algorithm[van Beek and Cohen, 1991] for clarification subdialogues to a wider range of advice-seeking questions; instead of selecting clarifying questions based on the faults distinguishing the possible plans, their system could select questions based on the

[16]These goals include goals that represent user preferences.

user's presumed preferences. This continued until the user's intentions were limited to a plan subset for which the same general advice-giving response was appropriate.

A mixed initiative system is one in which both user and system can take control of the problem-solving or the communication between the participants[Haller and McRoy, 1998]. Many of the above systems exhibit some form of system initiative, and current research is investigating more powerful mixed-initiative architectures[User Modeling: Special Issue, 1998]. One such research effort is COLLAGEN[Rich and Sidner, 1998], a toolkit for constructing mixed-initiative collaborative planning systems. Although COLLAGEN includes tools for maintaining a coherent mixed-initiative interaction, it does not include a means whereby user preferences can affect system initiative. In our work, the system's mechanism for deciding whether to take the initiative is its rating of possible actions with respect to the user's preferences, and the system exhibits initiative when it suggests alternatives to the user's proposed action.

9. FUTURE WORK

Our implemented system currently detects only preferences for attribute values of objects involved in an action. While such attribute-value preferences appear to account for many of the preferences conveyed in naturally occurring dialogue, we plan to extend our algorithm to detect preferences for the generic actions themselves. We also plan to investigate other ways by which an agent might convey a preference, such as by selecting an object from a list presented by the system. In addition, it seems likely that the order in which questions are asked by the user, and the point in the dialogue at which the user chooses to reject or accept a particular solution, indicate something about the strength of the user's preferences. However, it is unclear how reliable this evidence might be in relation to the other dialogue factors. These questions certainly merit further research.

We also plan to explore the recognition of the user's decision-making strategy. For example, in the strategy of elimination-by-aspects, the user considers the attributes of the alternatives in the order of importance, beginning with the most important attribute [Tversky, 1972]. By recognizing that the user is using this strategy, the system would know which preferences were most important to the user.

10. CONCLUSION

This paper has addressed the recognition and exploitation of user preferences. Preferences and goals play substantially different roles in plan construction. Users are conscious of their goals but are often not conscious of their preferences, at least at the outset of the dialogue. In addition, while goals must be satisfied (unless they are relinquished), preferences only influence the selection of alternatives; stronger preferences have more of an impact on action choice than do weaker ones, and coming close to satisfying a preference is regarded as better than ignoring it completely.

We have presented a mechanism for dynamically recognizing user preferences during the course of a collaborative task-oriented dialogue. Motivated by psychological research, our mechanism utilizes both the utterance conveying a preference and the conversational circumstances in which the utterance occurs. In addition, it can detect preferences that are communicated implicitly through a pattern of accepted or rejected proposals. We have also presented a strategy for exploiting the model of user preferences to detect suboptimal solutions and suggest better alternatives. By taking user preferences into account, our system is better able to fulfill its role as a collaborative agent.

ACKNOWLEDGMENTS

The authors would like to thank Rachel Sacher for her help with the system development and evaluation and the anonymous reviewers for their helpful comments. This material is based upon work supported by the National Science Foundation under Grant No. IRI-9122026.

REFERENCES

[Allen and Perrault, 1980] James F. Allen and C. Raymond Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15:143–178, 1980.

[Ardissono and Cohen, 1996] Liliana Ardissono and Robin Cohen. Extending the Role of User Feedback in Plan Recognition and Response Generation for Advice-Giving Systems: An Initial Report. In *Proceedings of the Canadian Conference on Artificial Intelligence*, pages 109–119, 1996.

[Ardissono and Sestero, 1995] Liliana Ardissono and Dario Sestero. Using Dynamic User Models in the Recognition of the Plans of the User. *User Modeling and User-Adapted Interaction*, 5:157–190, 1995.

[Bauer, 1994] Mathias Bauer. Quantitative Modeling of User Preferences for Plan Recognition. In *Proceedings of the Fourth International Conference on User Modeling*, pages 73–78, 1994.

[Bauer, 1995] Mathias Bauer. A Demster-Shafer Approach to Modeling Agent Preferences for Plan Recognition. *User Modeling and User-Adapted Interaction*, 5:317–348,1995.

[Bratman, 1990] Michael Bratman. What is Intention? In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 15–31. MIT Press, 1990.

[Carberry, 1987] Sandra Carberry. Pragmatic Modeling: Toward a Robust Natural Language Interface. *Computational Intelligence*, 3:117–136, 1987.

[Cawsey *et al.*, 1993] Alison Cawsey, Julia Galliers, Brian Logan, Steven Reece, and Karen Sparck Jones. Revising beliefs and intentions: A unified framework for agent interaction. In *The Ninth Biennial Conference of the Society for the Study of Artificial Intelligence and Simulation of Behaviour*, pages 130–139, 1993.

[Chin, 1988] David N. Chin. Exploiting User Expertise in Answer Expression. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 756–760, 1988.

[Chin, 1989] David N. Chin. KNOME: Modeling What the User Knows in UC. In Alfred Kobsa and Wolfgang Wahlster, editors, *User Models in Dialog Systems*, pages 74–107. Springer-Verlag, 1989.

[Chu-Carroll and Carberry, 1994] Jennifer Chu-Carroll and Sandra Carberry. A plan-based model for response generation in collaborative task-oriented dialogues. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 799–805, 1994.

[Chu-Carroll and Carberry, 1998] Jennifer Chu-Carroll and Sandra Carberry. Response generation in planning dialogues. *Computational Linguistics*, to appear, 1998.

[Cohen and Jones, 1989] Robin Cohen and Marlene Jones. Incorporating User Models into Expert Systems for Educational Diagnosis. In Alfred Kobsa and Wolfgang Wahlster, editors, *User Models in Dialogue Systems*, pages 313–333. Springer-Verlag, Berlin, West Germany, 1989.

[Cohen, 1987] Robin Cohen. Analyzing the Structure of Argumentative Discourse. *Computational Linguistics*, 13(1-2):11–24, 1987.

[Cohen *et al.*, 1989] Robin Cohen, Marlene Jones, Amar Sanmugasunderam, Bruce Spencer, and Lisa Dent. Providing responses specific to a user's goals and background. *International Journal of Expert Systems*, 2(2):135–162, 1989.

[Cohen, 1985] Paul R. Cohen. *Heuristic Reasoning about Uncertainty: An Artificial Intelligence Approach*. Pitman Publishing Company, 1985.

[Dawes, 1979] Robyn M. Dawes. The robust beauty of improper linear models in decision making. *American Psychologist*, 34:571–582, 1979.

[Einhorn and Hogarth, 1975] Hillel J. Einhorn and Robin M. Hogarth. Unit weighting schemes for decision making. *Organizational Behavior and Human Performance*, 13:171–192, 1975.

[Elzer *et al.*, 1994] Stephanie Elzer, Jennifer Chu-Carroll, and Sandra Carberry. Recognizing and utilizing user preferences in collaborative consultation dialogues. In *Proceedings of the Fourth International Conference on User Modeling*, pages 19–24, 1994.

[Galliers, 1992] Julia Rose Galliers. Autonomous belief revision and communication. In Gardensfors, editor, *Belief Revision*. Cambridge University Press, 1992.

[Haller and McRoy, 1998] Susan Haller and Susan McRoy. Preface to Special Issue on Computational Models for Mixed-initiative Interaction. *User Modeling and User-Adapted Interaction*, 1998.

[Jameson *et al.*, 1995] Anthony Jameson, Ralph Schafer, Joep Simons, and Thomas Weis. Adaptive provision of evaluation-oriented information: Tasks and techniques. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pages 1886–1893, 1995.

[Joshi *et al.*, 1984] Aravind Joshi, Bonnie Webber, and Ralph M. Weischedel. Living up to expectations: Computing expert responses. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, pages 169–175, 1984.

[Kass and Finin, 1988] Robert Kass and Tim Finin. Modelling the User in Natural Language Systems. *Computational Linguistics*, pages 5–22, 1988.

[Kass, 1991] Robert Kass. Building a User Model Implicitly from a Cooperative Advisory Dialog. *User Modeling and User-Adapted Interaction*, 1(3):203–258, 1991.

[Keeney and Raiffa, 1993] Ralph L. Keeney and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* Cambridge University Press, 1993.

[Lambert and Carberry, 1991] Lynn Lambert and Sandra Carberry. A tripartite plan-based model of dialogue. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 47–54, 1991.

[Litman and Allen, 1987] Diane Litman and James Allen. A plan recognition model for subdialogues in conversation. *Cognitive Science*, 11:163–200, 1987.

[McKeown, 1988] Kathleen R. McKeown. Generating Goal-Oriented Explanations. *International Journal of Expert Systems*, 1(4):377–395, 1988.

[Mittal and Paris, 1993] V. Mittal and C. Paris. Automatic Documentation Generation: The Interaction of Text and Examples. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1158–1163, 1993.

[Morik, 1989] Katharina Morik. User Models and Conversational Settings: Modeling the User's Wants. In Alfred Kobsa and Wolfgang Wahlster, editors, *User Models in Dialog Systems*, pages 364–385. Springer-Verlag, Berlin,New York, 1989.

[Murray, 1991] William R. Murray. An endorsement-based approach to student modelling for planner-controlled tutors. In *Proceedings of the 12th International Joint Conference on Artificial Intel-*

ligence, pages 1100–1106, 1991.

[Paris, 1985] Cecile Paris. Description strategies for naive and expert users. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 238–245, 1985.

[Paris, 1988] Cecile L. Paris. Tailoring Object Descriptions to a User's Level of Expertise. *Computational Linguistics*, 14(3):64–78, 1988.

[Payne *et al.*, 1993] John W. Payne, James R. Bettman, and Eric J. Johnson. *The Adaptive Decision Maker.* Cambridge University Press, 1993.

[Payne, 1976] John W. Payne. Task complexity and contingent processing in decision making: An information search and protocol analysis. *Organizational Behavior and Human Performance*, 16:366–387, 1976.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufman, 1988.

[Quinlan, 1993] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufman, 1993.

[Rich, 1979] Elaine Rich. User Modelling via Stereotypes. *Cognitive Science*, 3(4):329–354, 1979.

[Rich and Sidner, 1998] Charles Rich and Candace Sidner. COLLAGEN: A Toolkit for Building Collaborative Interface Agents. *User Modeling and User-Adapted Interaction*, 1998.

[Sarner and Carberry, 1992] Margaret Sarner and Sandra Carberry. Generating tailored definitions using a multifaceted user model. *User Modeling and User-Adapted Interaction*, 2(3):181–210, 1992.

[Shafer, 1976] G. Shafer. *A Mathematical Theory of Evidence.* Princeton University Press, 1976.

[Shortliffe, 1976] E. Shortliffe. *Computer-based Medical Consultations: MYCIN.* Elsevier, 1976.

[Simon, 1955] Herbert A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69:99–118, 1955.

[Slovic *et al.*, 1990] Paul Slovic, Dale Griffin, and Amos Tversky. Compatibility effects in judgment and choice. In R. M. Hogarth, editor, *Insights in Decision Making: A Tribute to Hillel J. Einhorn*, pages 5–27. University of Chicago Press, 1990.

[SRI transcripts, 1992] Transcripts derived from audiotape conversations made at SRI International, Menlo Park, CA, 1992. Prepared by Jacqueline Kowtko under the direction of Patti Price.

[Sycara and Miyashita, 1994] Katia Sycara and Kazuo Miyashita. Case-based acquisition of user preferences for solution improvement in ill-structured domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 44–49, 1994.

[Tversky, 1969] Amos Tversky. Intransitivity of preferences. *Psychological Review*, 76:31–48, 1969.

[Tversky, 1972] Amos Tversky. Elimination by aspects: A theory of choice. *Psychological Review*, 79:281–299, 1972.

[User Modeling: Special Issue, 1998] *User Modeling and User-Adapted Interaction*: Special Issue on Computational Models for Mixed-initiative Interaction. 1998–1999.

[van Beek and Cohen, 1986] Peter van Beek and Robin Cohen. Towards user specific explanations from expert systems. In *Proceedings of the Sixth Canadian Conference on Artificial Intelligence*, pages 194–198, 1986.

[van Beek and Cohen, 1991] Peter van Beek and Robin Cohen. Resolving Plan Ambiguity for Cooperative Response Generation. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 938–944, 1991.

[van Beek *et al.*, 1993] Peter van Beek, Robin Cohen, and Ken Schmidt. From plan critiquing to clarification dialogue for cooperative response generation. *Computational Intelligence*, 9(2):132–154, 1993.

[van Beek, 1987] Peter G. van Beek. A model for generating better explanations. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pages 215–220, 1987.

List of Figures