

A Personalized System for Conversational Recommendations

Cynthia A. Thompson

School of Computing, University of Utah, 50 Central Campus Drive, Rm. 3190, Salt Lake City, UT 84112 USA

CINDI@CS.UTAH.EDU

Mehmet H. Göker

Kaidara Software Inc., 330 Distel Circle, Suite 150, Los Altos, CA 94022 USA

MGOKER@KAIDARA.COM

Pat Langley

Institute for the Study of Learning and Expertise, 2164 Staunton Court, Palo Alto, CA 94306 USA

LANGLEY@ISLE.ORG

Abstract

Increased computing power and the Web have made information widely accessible. In turn, this has encouraged the development of recommendation systems that help users find items of interest, such as books or restaurants. Such systems are more useful when they personalize themselves to each user's preferences, thus making the recommendation process more efficient and effective. In this paper, we present a new approach to recommendation systems: that of personalized conversational recommendation. This system uses the complementary strengths of recommendation systems and dialogue systems to overcome their respective deficiencies. We present a system – the Adaptive Place Advisor – implementing our ideas. This system treats item selection as an interactive, conversational process, with the program inquiring about item attributes and the user responding. The system incorporates a user model that contains item, attribute, and value preferences, which it updates during each conversation and maintains across sessions. The Place Advisor utilizes both the conversational context and the user model to retrieve candidate items from a database. It uses personalized heuristics to select which attribute to ask about in order to narrow down and rank possible choices. We report experimental results demonstrating the effectiveness of user modeling in reducing the time and number of interactions required to find a satisfactory item.

1. Introduction and Motivation

Recommendation systems help users find and select items (e.g., books, movies, restaurants) from the huge number available on the web or in other electronic information sources (Burke, Hammond, & Young, 1996; Resnick & Varian, 1997; Burke, 1999). Given a large set of items and a description of the user's needs, recommendation systems present to the user a small set of the items that are suited to the description. Recent work includes intelligent aides for filtering and choosing web sites (Pazzani, Muramatsu, & Billsus, 1996), news stories (Lang, 1995), TV listings (Cotter & Smyth, 2000), and other information.

The users of such systems often have diverse, conflicting needs. Differences in personal preferences, social and educational backgrounds, and private or professional interests are pervasive. As a result, it is desirable to have *personalized* intelligent systems that process, filter, and display available information in a manner that suits each individual using them. The need for personalization has led to the development of systems that adapt themselves by changing their behavior based on the inferred characteristics of user interacting with

them (Rich, 1979; Langley, 1999; Ardissono & Goy, 2000; Ferrario, Waters, & Smyth, 2000; Fiechter & Rogers, 2000).

The ability of computers to converse with users in natural language would arguably increase their usefulness and flexibility even further. Research in practical dialogue systems, while still in its infancy, has matured tremendously in recent years (Allen, Byron, Dzikovska, Ferguson, Galescu, & Stent, 2001; Dybkjær, Hasida, & Traum, 2000; Maier, Mast, & Luperfoy, 1996). Today's dialogue systems typically focus on helping users complete a specific task, such as information search, planning, event management, or diagnosis.

In this paper, we describe a personalized conversational recommendation system designed to help users choose an item from a large set of items of the same basic type. Our goal is to support conversations that become more efficient for individual users over time. Our sample system, the *Adaptive Place Advisor*, aims to help users select a destination (here, restaurants) that meets their preferences.

With the Adaptive Place Advisor, three novel contributions are presented. To our knowledge, this is the first personalized spoken dialogue system for recommendation. Second, we introduce a novel model for acquiring, utilizing, and representing user models. Third, we demonstrate the effectiveness of the system in reducing the number of questions and conversation time needed to find a satisfactory item.

The combination of dialogue systems with personalized recommendation addresses weaknesses on both sides of the combination. Most dialogue systems react similarly for each user interacting with them, and do not store information gained in one conversation for use in the future. Thus, interactions tend to be tedious and repetitive. By adding a personalized, long-term user model, the quality of these interactions can increase drastically. At the same time, collecting user preferences in recommendation systems typically requires form-filling or other explicit statements of preferences on the part of the user. This process is often difficult and time consuming. Collecting preferences in the course of the dialogue allows the user to immediately begin the task of item search.

The interaction between conversation and personalized recommendation has also affected our choices for the acquisition, utilization, and representation of user models. We learn information about users unobtrusively, in the course of a normal conversation whose purpose is to find a satisfactory item. We store this information for use in future conversations with the same individual. Both acquisition and utilization occur not only when items are presented to and chosen by the user, but also during the process of searching for those items. Finally, our representation of models goes beyond item preferences to include both preferences about item characteristics and particular values of those characteristics. We believe that these ideas can extend to other types of preferences and other types of conversations.

The remainder of the paper is organized as follows. In the next section, we introduce personalized and conversational recommendation systems, presenting our design decisions as we go. Section 3 describes our system, while Section 4 presents our system evaluation. Sections 5 and 6 discuss related and future work, respectively, and Section 7 concludes and summarizes.

2. Personalized Conversational Recommendation Systems

Our goal is to improve the quality of the interaction with the recommendation system and the utility of the returned result by making the system user adaptive and conversational. We also want to increase the performance of the dialogue system by means of personalization.

As such, our goal for user modeling differs from the one commonly assumed in recommendation systems, that of improving accuracy or related measures like precision and recall. Our goal also differs from that of previous work in user modeling in dialogue systems (Kass, 1990; Carberry, 1990; Kobsa & Wahlster, 1998), which emphasizes the ability to track the user’s goals as a dialogue progresses, and which does not typically maintain models over multiple conversations.

Our hypothesis is that improvements in efficiency and effectiveness can be achieved by using an unobtrusively obtained user model to help direct the system’s conversational search for items to recommend. Our approach assumes that there is a large database of items from which to choose, and that a reasonably large number of attributes is needed to describe these items. Simpler techniques might suffice for situations where the database is small or items are simple to describe.

2.1 Personalization

Personalized, user adaptive systems obtain preferences through interactions with users, keep summaries of these preferences in a user model, and utilize this model to generate customized information or behavior. The goal of this customization is to increase the quality and appropriateness of both the interaction and the generated result(s) for each user.

The user models stored by personalized systems can represent stereotypical users (Chin, 1989) or individuals, they can be hand-crafted or learned (from questionnaires, ratings, or usage traces), and they can contain information about behavior such as previously selected items, preferences regarding item characteristics (such as location or price), or properties of the users themselves (such as age or occupation) (Rich, 1979; Kobsa & Wahlster, 1998). Finally, some systems store user models only for the duration of one interaction with a user, and others store them over the long term.

Our approach is to learn probabilistic, long-term, individual user models that contain information about preferences for items and item characteristics. We chose probabilistic models because of their flexibility: a single user can exhibit variable behavior, and their preferences are relative rather than absolute. Long-term models are important to allow their influence across multiple conversations. And, as already noticed, different users have different preferences, so individual models are required. Finally, preferences about items and item characteristics are needed to influence conversations and retrieval.

Another design decision relates to the method by which preferences are collected for subsequent input to the learning algorithm(s). Here, we can distinguish between two approaches. The *direct-feedback* approach places the burden on the user by soliciting preference information directly. For example, a system might ask the user to complete a form that asks them to classify or weight their interests using a range of categories. The problem with this approach is that users are typically irritated by the need to complete long questionnaires before they can even begin to enjoy a given service. In response, another form of

direct-feedback encourages the user to provide feedback as they continue to use a particular service.

The second approach to acquiring user models, and the one taken in the Adaptive Place Advisor, is to derive user preferences *unobtrusively*, by mining normal online behavior (Fiechter & Rogers, 2000; Rafter, Bradley, & Smyth, 2000). We feel that unobtrusive collection of preferences is advantageous, as it requires less effort from the user. Also, users often cannot articulate their preferences clearly until they learn more about the domain. A possible disadvantage to unobtrusive approaches is that users may not trust or understand the system’s actions when they change from one interaction to the next. This could be addressed by allowing the user to also view and modify the user model (Kay & Thomas, 2000).

Systems typically take one of two approaches to preference determination. The *content-based* approach recommends items similar to the items that the user has liked in the past (Pazzani et al., 1996; Lang, 1995; Shardanand & Maes, 1995; Segal & Kephart, 1999). In contrast, the *collaborative* approach selects and recommends items to the current user that similar users have liked in previous interactions (Konstan, Miller, Maltz, Herlocker, Gordon, & Riedl, 1997; Billsus & Pazzani, 1998; Cotter & Smyth, 2000). Because collaborative filtering bases recommendations on previous selections of other users, it is not suitable for new or one-off items (latency) or for users with uncommon preferences. The content-based approach, on the other hand, uses the item description itself for recommendation, and is therefore not prone to the latency problem. This makes it suitable for use in the Adaptive Place Advisor.

Ultimately, personalization techniques are all about how one can utilize a learned user profile in order to search for, identify, and present relevant information to the right user in the right way at the right time. User models have been utilized in recommendation systems for content processing and selection (information filtering), navigation support in web browsers, and choice of modality and style of presentation and interaction (Brusilovsky & Maybury, 2002). During information filtering, user preferences can serve as a basis for removing irrelevant choices from the results returned by a search or retrieval algorithm. Navigation can be aided by modifying a page to highlight information (Pazzani et al., 1996) or even by changing the link structure of a site. Finally, presentation modality can shift between a mostly textual description of information, a graphical presentation, and speech. Also, the algorithms for deciding what to do next in the interaction can be influenced by a user model. This can help streamline the interaction by eliminating behavior (e.g., popping up a help window) that has been unsatisfactory to the user in the past. The Adaptive Place Advisor adapts its information filtering and interaction behavior, since these are most relevant for our application and the majority of the interaction is through natural language.

2.2 Recommendation

The goal of recommendation systems is to help users search for items of interest. Thus, they must eliminate some choices from consideration while keeping others. To achieve this, a recommendation system needs to determine the interests of the current user.

One of the most widely used frameworks for information acquisition and presentation in recommendation systems is the *form based / ranked list* approach. In this scheme, the user specifies their needs in a form and the system presents a usually long list of results, ordered by their predicted relevance to the user. This technique originated in information retrieval, where it was used to help find documents or reports during library research. The basic method is more general, however, and it underlies many of the simpler search engines for the World Wide Web, which are used regularly for many different selection tasks.

In contrast, the Adaptive Place Advisor supports *conversations* as its mode of interaction. In this approach, one person, that we call the inquirer, asks a second person, that we call the advisor, for assistance in deciding on an item on which the inquirer is planning to take some action. For example, if the deciding on a restaurant at which to have dinner, the conversation may proceed as follows:

1. Inquirer: Where do you think I should eat tonight?
2. Advisor: What type of food would you like?
3. Inquirer: What types are there?
4. Advisor: You can say things like Chinese, Indian, and Mediterranean.
5. Inquirer: Oh, maybe a cheap Indian place.
6. Advisor: What kind of parking would you like?
7. Inquirer: I don't care, as long as it's in Palo Alto.
8. Advisor: I'm sorry, I don't know of any restaurants like that, would you like to search for any price?
9. Inquirer: No, I think I'd like Chinese instead.
10. Advisor: I know three inexpensive Chinese restaurants in Palo Alto. One is Mandarin Gourmet on 420 Ramona. How does that sound?¹
11. Inquirer: No, what else do you have?
12. Advisor: How about Jing-Jing Szechwan Hunan Gourmet on 443 Emerson Street?
13. Inquirer: Sure, that sounds fine.

We will refer to systems that mimic this approach to recommendation as *conversational recommendation systems* (Aha, Breslow & Muñoz-Avila, 2001).

The interaction supported by conversational recommendation systems is quite different from that found in the form based / ranked list approach. The most important distinction is that the interaction takes the form of a sequence of questions, most designed to eliminate some items from consideration. Answering these questions plays a similar role to giving keywords with the form based scheme, but the aim is to remove alternatives rather than to only order the results in the list. This conversational process can also help the inquirer better understand his own desires, since thinking about possible questions and answers may clarify goals in ways that keyword entry in a form does not encourage. In addition, a conversation can support user variability by allowing the user to control item and attribute choices in each conversation. Finally, the inquirer never hears about a complete item until only one, or at most a few, choices remain. This avoids presenting a long list of items, a tedious process in a spoken interface, and lets the user narrow down the choices in an iterative, manageable fashion.

1. At this point in a conversation our system actually displays the information about the restaurant on the screen.

Such dialogues seem better for recommendations that must be delivered by speech rather than visually, for example those engaged in while the inquirer is driving. They also seem ideal, independent of modality, for tasks like destination selection or help-desk support (Aha & Breslow, 1997; Göker & Roth-Berghofer, 1999), in which the user needs to converge on at most a few items. On the other hand, form-based keyword entry and ranked list methods seem more appropriate in situations where the user prefers to provide all requirements at once, where information can be presented visually, and in situations where the user may want to examine many options.

By eliminating options, conversational recommendation systems will ultimately direct their users to a suitable solution. However, such a conversation can become tiring and the quality of the first result returned may not be acceptable for each user. Just as interactions with a friend who knows your concerns can be more directed and produce better results than those with a stranger, dialogues with the Adaptive Place Advisor become more efficient and effective over time. Our goals for user modeling include improvement of the subjective quality and effectiveness of both the results (found items) and the conversation that leads to these results. For example, after several conversations with the inquirer above, a new interaction may proceed as follows, where the question about parking has been eliminated and the item presentation order has changed:

1. Inquirer: Where do you think I should eat tonight?
2. Advisor: What type of food would you like?
3. Inquirer: Oh, maybe a Chinese place.²
4. Advisor: What city do you prefer?
5. Inquirer: Do you have something in Palo Alto?
6. Advisor: How does Jing-Jing Szechuan Gourmet on 443 Emerson sound?
7. Inquirer: Sure, that sounds fine.

2.3 Conversation via Dialogue Management

Dialogue systems carry out conversations with users in natural language, whether spoken or typed. The main tasks performed by dialogue systems are natural language interpretation and generation, and dialogue management. Natural language interpretation and generation is a topic onto itself and we do not cover all alternatives here; for two introductory texts, see Allen (1995) and Jurafsky and Martin (2000). To allow a focus on user modeling, our system allows moderately complex user utterances but has a pre-coded set of allowable outputs, as discussed further in Section 3.3.

The simplest dialogue managers are based on finite-state automata in which the states correspond to questions and arcs correspond to actions, dependent on a provided response (Winograd & Flores, 1986; Stent, Dowding, Gawron, Bratt, & Moore, 1999). These systems support what are called fixed- or system-initiative conversations, in which only one of the participants controls the actions, whether it be the system helping the user or the user asking questions of the system (Bobrow et al., 1977; Pieraccini, Levin, & Eckert, 1997). Next in complexity are frame- or template-based systems in which questions can be asked and answered in any order. Finally, true mixed-initiative (Allen, 1999; Haller & McRoy, 1998)

2. This shows that the Inquirer will have learned how to use the system more efficiently as well.

systems allow either dialogue participant to contribute to the interaction as appropriate to their knowledge. Thus, the conversational focus can change at any time due to the user’s (or system’s) initiative of that change. For example, plan-based systems (Cohen & Perrault, 1979; Allen et al., 1995) and those that use models of rational interaction (Sadek, Bretier, & Panaget, 1997) support complex and robust dialogues.

To allow reasonably complex conversations without introducing major system complexity, we chose a frame-based approach to dialogue management. Thus, the Adaptive Place Advisor is more flexible than a fully system-initiative paradigm. Users can fill in attributes other than or in addition to those suggested by the system. However, the user cannot force the system to transition to new subtasks, nor can the system and user negotiate to determine which participant should take the initiative.

2.4 Interactive Constraint-Satisfaction Search

Constraint-satisfaction problems provide a general framework for defining problems of interest in many areas of Artificial Intelligence, such as scheduling and satisfiability (Kumar, 1992). In their most general form, constraint-satisfaction problems are stated as follows: given is a set of variables whose domains are finite and discrete, and a set of constraints. The constraints are defined over some subset of the variables and limit the value combinations that those variables can take. The goal is to find an assignment of values to variables that satisfy the given constraints.

Recently, Cucchiara, Lamma, Mello, and Milano (1997) presented *interactive* constraint-satisfaction problems. They described three extensions to a standard constraint-satisfaction problem. First, a variable’s domain can include both a defined and undefined portion and during constraint acquisition new values can be added to the defined portion of the space. Second, new constraints can be incrementally added to the problem being solved. Third, the search process can include incremental update of a partial solution to the problem, based on the domain and constraint updates.

We use a simplified version of this model to describe part of the conversational search carried out by the Adaptive Place Advisor. The item presentation portion of the conversation is not included in this model, but the item search portion is. In our setting, constraints are simply attribute-value specifications, such as *cuisine = Chinese*. We do not incorporate the notion of undefined portions of domains, but include the other two characteristics of interactive constraint-satisfaction. New constraints are added by the user’s specifications during a conversation, and the system incrementally updates solutions in response.

3. The Adaptive Place Advisor

In this section, we first present an overview of the Adaptive Place Advisor’s functioning, then follow with details about its components.³ The Adaptive Place Advisor has to carry out the following main tasks to support a personalized interaction (conversation and recommendation) with the user:

3. Our approach to destination advice draws on an earlier analysis of the task by Elio and Haddadi (1998, 1999). The main distinctions from that work are that their approach does not include personalization, that they distinguish between search through a task space and through a discourse space, while we combine the two, and that they place a greater emphasis on user intentions.

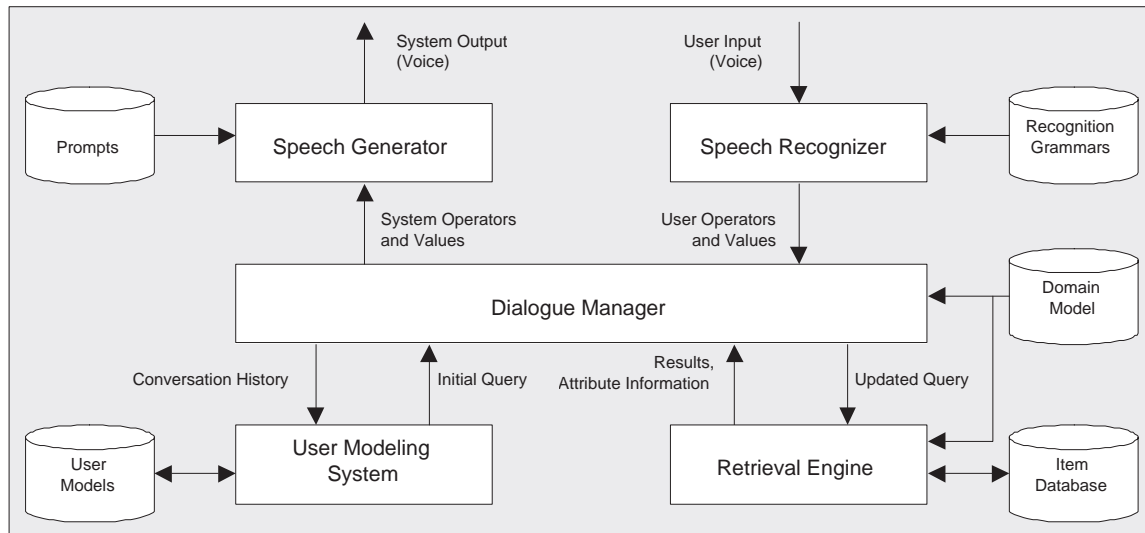


Figure 1: Components of the Adaptive Place Advisor and their interactions

- utilizing the user model, initialize a probabilistic item representation as a *query*,
- generate context-appropriate utterances,
- understand the user’s responses,
- refine the query with the explicit requirements (constraints) obtained from the user during the conversation,
- retrieve items matching the explicitly specified part of the query from a database
- calculate the similarity of the retrieved items to the user’s preferences,
- select the next attribute to be constrained or relaxed during a conversation when the number of highly similar items is not acceptable,
- present suitable items when the number of items is acceptable, and
- acquire and update the user model based on these interactions.

The responsibilities for these tasks are distributed among various modules of the system, as shown in Figure 1. The Dialogue Manager generates, interprets, and processes conversations. It also updates the query after each user interaction. The Retrieval Engine is a case-based system that uses the query to retrieve items from the database and measures their similarity to the user’s preferences. The User Modeling System generates the initial (probabilistic) query and updates the long term user model based on the conversation history. The Speech Recognizer and the Speech Generator handle the user’s input and control the system’s output, respectively.

To find items to recommend to the user, the Place Advisor carries out an augmented interactive constraint-satisfaction search. The goal of the entire conversation is to present an item that will be acceptable to the user. During the constraint-satisfaction portion, the system carries out a conversation to find such an item. During the search phase, two situations determine the system’s search operators and thus its questions. First, an under-constrained specification means that many items match the constraints, and more information must be obtained from the user. A similarity computation is used to rank

the items that satisfy the constraints. Second, if there are no matching items, a constraint must be relaxed, thus allowing items to contain any domain value for the relaxed attribute.⁴ The search phase is ended when only a small number of items match the constraints and are highly similar (based on a similarity threshold) to the user’s preferences. Then, item presentation (in order of similarity) begins.

The search and item presentation process is also influenced by the User Modeling System and thus is *personalized*. The main mechanism for personalization is through the query, a probabilistic representation of the user’s preferences, both long-term (over many conversations) and short-term (within a conversation). In a sense, the initial query represents what constraints the system thinks the user will “probably want”. This query is incrementally refined in the course of the conversation with the user and explicit, firm constraints are set as assumptions are verified or disconfirmed. For the long-term, the User Modeling System updates the user model based on the user’s responses to the attributes and items offered during a conversation.

The Retrieval Engine searches the database for items that match the firm constraints in the query. It then computes the similarity of the retrieved items to the user’s preferences as reflected in the remainder of the query and in the user model. Depending on the number of highly similar results, the Retrieval Engine also determines which attribute should be constrained or relaxed.

In sum, the system directs the conversation in a manner similar to slot-filling, retrieves and ranks items using a case-based reasoning paradigm (Aamodt & Plaza, 1994), and adapts the weights in its similarity calculation based on past conversations with a user, thereby personalizing the retrieval and the conversation.

3.1 The User Model

Our focus on personalized conversation suggests modeling user preferences at a fine-grained level, focusing on the questions a user prefers to answer and the responses he tends to give, in addition to preferences about entire items. In later sections, we will describe how this user model influences item ranking, which in turn determines how quickly the system can stop asking questions and start presenting items.

In general, a user may have preferences about:

- the relative importance of attributes,
- values for an attribute,
- specific items,
- the combination of certain item characteristics, and
- the diversity of the suggested items and values.

The preferences regarding an attribute represent the relative importance a user places on the attribute while selecting an item (e.g., how important is cuisine vs. price). Preferred values show the user’s bias towards certain item characteristics (e.g., Italian restaurants vs. French restaurants). Item preferences manifest themselves in the user having a bias for or against a certain item, independent of its characteristics. Combination preferences repre-

4. Because other constraints can later be modified, we allow the user to later specify any value, even the one that caused the over-constrained situation.

Table 1: Example of a user model

User Name	Homer						
Attributes	w_i	Values and probabilities					
Cuisine	0.4	Italian	French	Turkish	Chinese	German	English
		0.35	0.2	0.25	0.1	0.1	0.0
Price Range	0.2	one	two	three	four	five	
		0.2	0.3	0.3	0.1	0.1	
...					
Parking	0.1	Valet		Street		Lot	
		0.5		0.4		0.1	
Item Nbr.	0815	5372	7638	...			6399
Accept/Present	23 / 25	10 / 19	33 / 36	...			12 / 23

sent constraints with respect to the combined occurrence of characteristics in an item (e.g., accepts restaurants in San Francisco only if they have valet parking). Diversity preferences model the time that needs to pass between an item or characteristic being suggested again. While item preferences are related to single items, attribute, value and combination preferences are applicable to the search for those items in general. Diversification preferences relate to both the items and the search.

Currently, the Adaptive Place Advisor models preferences that the user may have about attributes, values and items, but not combination or diversity preferences.

Attribute, value, and item preferences are easily captured by either probability distributions or counts, as illustrated in Table 1. The Place Advisor maintains a probability distribution to represent attribute preferences, and independent probability distributions to represent preferences for each attribute’s set of values. For attribute preferences, domain knowledge is used to initialize the weights; for example price is usually more important than parking. In the absence of such information, the system could begin with a uniform distribution, which is used for value preferences.

The system represents item preferences as a ratio of the number of times an item was accepted to the number of times it was presented; this is initialized by assuming that all items have been presented and then accepted a large percentage (90%) of the time. While this may cause updates (see below) to have a small effect, it has the effect of not quickly discounting alternatives early in the learning process. This in turn encourages the user to explore alternatives, allowing the system to learn more about additional items. In sum, item preferences represent the probability of the user accepting a particular item after it is presented, rather than a probability distribution over all items.

Once in place, the user preferences affect the behavior of the system; specifically it is exploited by the Retrieval Engine as described next.

3.2 The Retrieval Engine

The Retrieval Engine interacts with a database to retrieve the items, if any, that satisfy the currently agreed upon constraints. It also interacts with the user model to determine how similar those items are to the user’s preferences and to determine the best ordering of the attributes for constraining or relaxing, as appropriate. Both functions support the goal of quickly narrowing down the search for a satisfactory item.

Similar to the way a human advisor bases assumptions regarding the inquirer on their previous interactions, our system uses its cumulative experience, reflected in the user model, as a basis for its computation. The Retrieval Engine represents the user’s preferences and requirements in the query, a partial, probabilistic item specification determined by both the conversation and the user model. The query is initialized from the user model, and thus contains preference-based probabilities for the attributes and values the user has not yet explicitly specified. In the course of the conversation, the query is updated to reflect the values the user specifies. For each attribute, the probabilities for all values agreed upon in the conversation are normalized while the probabilities for all other values are set to zero. For example, if the user says “Chinese or Italian,” the value probabilities for Chinese and Italian are each set to 0.5, and all other cuisine probabilities are set to zero.

Unlike a typical case-based reasoning similarity computation, which retrieves items beyond those that match a query exactly, the computation used by the system restricts the retrieved items to those most desirable to the user. The system filters the items to be included in the current case-base according to the characteristics explicitly specified by the user and sorts the remaining items according to their similarity to items the user liked in the past.

Thus, the Engine first queries the database to retrieve all items matching the current constraints exactly.⁵ Then, the probability distributions in the query are used to compute how likely the user is to choose an item. The similarity between the current query, Q , and an item, I , is calculated as:

$$Sim(Q, I) = R_I \times \sum_{j=1}^n w_j \times P(V_j) ,$$

where R_I is the user’s item preference ratio for item I , n is the number of attributes, w_j is the weight of attribute j in Q , V_j is the value of attribute j in I , and $P(V_j)$ is the value preference (probability in Q) for this value. Similarity in this formula is based on the user model and thereby on the probability of the user accepting the value present in the item for each unconstrained attribute. Then, the system compares the similarity of each item to a constant similarity threshold, and only retains those items exceeding the threshold. Thus, when we later discuss the number of items matching the constraints, it refers to the items remaining after this similarity filtering.

The other function of the Retrieval Engine is to rank attributes in under- and over-constrained situations. For both situations, one option is to order attributes randomly, which is a model used in some simple dialogue systems. Another option is to use a conditional entropy measure to select attributes (see Section 6.1). A third is to rank attributes

5. For attributes where the user has selected more than one value, we assume that any supplied value would be acceptable.

in order by their desirability to the user, as reflected in the user model, and we take this option.

We actually take the option of utilizing the user model for attribute ranking one step further, by using the weights in the query; we will see later that the query’s attribute weights are also influenced by the conversation. In an over-constrained situation, the ranking order is from highest to lowest, and in an under-constrained situation, it is the reverse.⁶ Using the attribute weights rather than the conditional entropy allows us to avoid pitfalls arising from the continuously changing value distribution in the data (new restaurants, restaurants going out of business, etc.). Further, not every question that has a high information gain is of high relevance for a user selecting a destination (e.g., “parking options” may have a very high information gain but should only be asked after the user has decided on cuisine and location.)

In summary, the user model influences item retrieval, item ranking, and attribute ranking, which in turn influence the system’s utterances during the conversation.

3.3 Conversing with the User

As it converses with the user, the Dialogue Manager uses the results of the Retrieval Engine’s functions. The system uses a frame containing a simple list of constraints to support the interactive constraint-satisfaction search (see Jurafsky et al. (1994) or Dowding et al. (1993) for a similar formulation). As is usual in this type of system, the user can respond to a system request to fill a constraint by ignoring that attribute and specifying the value for different one(s) instead (Ward & Issar, 1996; Goddeau et al., 1996). The speech acts supported are listed in Table 2.

There are two main phases of the dialogue, the interactive constraint-satisfaction portion, and the item presentation portion. The constraint-satisfaction portion is further divided into over- and under-constrained situations. The dialogue state (Table 3) determines the system’s utterance and the range of responses expected at each point. The dialogue state’s variables are updated throughout the conversation.

In more detail, the system’s speech act (or *move*) during interactive constraint-satisfaction is determined by the `Num-Items` dialogue state variable. The most common situation is when many items (more than some small threshold, here three) match the current constraints. In this situation, the system makes an `ATTEMPT-CONSTRAIN` move, in which it asks the user to fill in the value for an attribute. This move, if responded to appropriately by the user, would reduce the number of items considered to be satisfactory to the user. The attribute to `Constrain` is the one ranked highest by the Retrieval Engine that has not already been `Constrained` or `Rejected`. In our first sample conversation, repeated in Figure 2, utterances 2, and 6 illustrate `ATTEMPT-CONSTRAIN`’s by the system.

One user response to an `ATTEMPT-CONSTRAIN` is a `PROVIDE-CONSTRAIN`, in which he provides a value for the specified attribute or for additional attributes, as in utterances 5 and 7. A second possible response is a `REJECT-CONSTRAIN`, in which the user indicates

6. Weighting factors for similarity computation and question ordering are not necessarily treated identically in all CBR applications (e.g., diagnosis). However, for our application area, the assumption holds that the importance of an attribute is identical to its impact on similarity computation.

Table 2: Speech acts supported in the Adaptive Place Advisor

System Speech Acts	
ATTEMPT-CONSTRAIN	Asks a question to obtain a value for an attribute.
SUGGEST-RELAX	Asks a question to remove a value for an attribute.
RECOMMEND-ITEM	Recommends an item that satisfies the constraints.
QUIT-START-MOD	States that no matching items remain and asks whether to modify the search, start over, or quit.
PROVIDE-VALUES	Lists a small set of values for an attribute.
CLARIFY	Asks a clarifying question.
User Speech Acts	
PROVIDE-CONSTRAIN	Provide a value for an attribute.
REJECT-CONSTRAIN	Rejects the proposed attribute.
ACCEPT-RELAX	Accepts the proposed removal of an attribute value.
REJECT-RELAX	Rejects the proposed removal of an attribute value.
PROVIDE-RELAX	Provides an attribute value for removal.
ACCEPT-ITEM	Accepts the proposed item.
REJECT-ITEM	Rejects the proposed item.
START-OVER	Indicates a desire to reinitialize the constraints and begin again.
QUIT	Indicates a desire to stop the conversation.
QUERY-VALUES	Asks for information about possible values of an attribute.

Table 3: Dialogue State

<i>Variable</i>	<i>Description</i>
Constrained	Attributes whose values have been specified
Rejected	Attributes whose value the user has declined to provide
Fixed	Constrained attributes that the user has indicated should not be relaxed
Constrain	The next attribute to constrain, if any
Relax	The next attribute to relax, if any
Query	Probability model of desired item constraints
Num-Items	Number of database items matching the query
Ranked-Items	The matching items ranked in similarity order
Rejected-Items	Items that the user has rejected
User-Move	The user's most recently uttered speech act

disinterest or dislike in an attribute, as in the first part of utterance 7. As illustrated by some of these examples, the user can combine more than one move in a single utterance.

A second situation, an *over-constrained* query, is that in which no items satisfy the agreed upon constraints and are similar enough to the user's preferences, and thus the Retrieval Engine returns an empty set ($\text{Num-Items} = 0$). In this case, the system performs a SUGGEST-RELAX move informing the user of the situation and asking if he would like to relax a given constraint. The attribute to **Relax** is chosen from the Retrieval Engine's highest

1. Inquirer: Where do you think I should eat tonight?
2. Advisor: What type of food would you like?
3. Inquirer: What types are there?
4. Advisor: You can say things like Chinese, Indian, and Mediterranean.
5. Inquirer: Oh, maybe a cheap Indian place.
6. Advisor: What kind of parking would you like?
7. Inquirer: I don't care, as long as it's in Palo Alto.
8. Advisor: I'm sorry, I don't know of any restaurants like that, would you like to search for any price?
9. Inquirer: No, I think I'd like Chinese instead.
10. Advisor: I know three inexpensive Chinese restaurants in Palo Alto. One is Mandarin Gourmet on 420 Ramona. How does that sound?⁷
11. Inquirer: No, what else do you have?
12. Advisor: How about Jing-Jing Szechwan Hunan Gourmet on 443 Emerson Street?
13. Inquirer: Sure, that sounds fine.

Figure 2: Sample Conversation

ranked attribute that has not already been **Fixed**. This is illustrated in utterance 8 of the conversation in Figure 2. As in utterance 9 of that conversation, the user can respond by rejecting (**REJECT-RELAX**) the system's suggestion, or he can accept it (**ACCEPT-RELAX**). In the former case, the attribute is **Fixed** so that the system will not try to relax it again. In combination with either of these speech acts, the user can specify other attributes to relax in addition to or instead of the system suggested attribute (**PROVIDE-RELAX**).

When only a few items satisfy the constraints, the system ends the interactive search and begins to suggest items to the user (**RECOMMEND-ITEM**) in order by similarity, as in utterances 10 and 12 above. The user can either accept or reject an item. If the user accepts an item (**ACCEPT-ITEM**), the system ends the conversation, having reached the goal state. If the user rejects an item (**REJECT-ITEM**), the system presents an alternative, if any remain.

There are three special situations not covered by the above. The first is when the query is over-constrained, but the user has **Fixed** all attributes that could be relaxed. The second is when the user has rejected all items that match the constraints. In these two situations, the system informs the user of the situation, asks him whether he would like to quit, start over, or modify the search (**QUIT-START-MOD**), and reacts accordingly. The third special situation is when **Num-Items** exceeds the presentation threshold, but all attributes have been **Constrained** or **Rejected**. In that case, the Place Advisor begins to present items to the user.

To support the spoken natural language input and output, we use a speech recognition package from Nuance Communications, Inc.⁸ This allows us to write a different recognition grammar for each of the main situations described above, and to use human-recorded prompts (rather than text to speech). The string of words recognized by the system is parsed by recognition grammars that we wrote, which were used for all users and not adapted. Future work could include personalized recognition grammars as well as personalized information preferences. The grammars use semantic tags to fill in each slot: besides slots for

8. www.nuance.com

Table 4: The effects of speech acts on the query

<i>Speech Act</i>	<i>Effect on Query</i>
PROVIDE-CONSTRAIN	Normalize probabilities of all provided values so they add to one. Set probability of other values for constrained attributes to zero. If the attribute has been rejected previously, reset its attribute probability from user model.
REJECT-CONSTRAIN	Drop attribute by setting its probability to zero.
ACCEPT-RELAX	Reset value probabilities for the attribute from user model.
REJECT-RELAX	No effect; Dialogue Manager selects next attribute.
PROVIDE-RELAX	Reset value probabilities for the attribute from user model.
ACCEPT-ITEM	None.
REJECT-ITEM	None.
START-OVER	Initialize from user model.

each attribute, we defined slots for rejection or acceptance of the system’s suggestions. In more complex domains, more sophisticated parsing methods may be required, but this simple scheme allows the user a reasonably diverse set of utterances. The Nuance modules also generate a response to user requests for help (QUERY-VALUES) with a PROVIDE-VALUES speech act, and enters clarification dialogues when the confidence in a recognized utterance is below a given threshold (CLARIFY). These are currently simple interactions where the system provides examples of answers to the most recently uttered prompt, or asks the user to repeat themselves. Finally, for the item presentation portion of the dialogue only, we display the restaurant information (name, address and phone number) on the screen. We chose this presentation modality due to our reluctance to use text-to-speech generation and the large number of prompts we would have had to record to produce spoken language for each restaurant.

Each system-user interaction affects subsequent rounds of database retrieval and similarity calculation via updates to the query. Table 4 shows the effects of relevant speech acts on the query, which is in turn used in the similarity calculation as described in Section 3.2.

3.4 Updating the User Model

Our main goal and contribution is to add personalization to the above conversational recommendation model. The user model (Section 3.1) represents that personalization, but the Adaptive Place Advisor must update it appropriately. While some adaptive recommendation systems (Pazzani et al., 1996; Lang, 1995; Linden, Hanks, & Lesh, 1997; Smyth & Cotter, 1999) require the user to provide direct feedback to generate the user model, our basic approach is to unobtrusively derive the user preferences. Thus, the system does not introduce unnecessary interactions, but learns from the interactions needed to support item recommendation. We describe here how the system gathers item, attribute, and value preferences. For the latter two, feature and value weights are modified (Fiechter & Rogers,

2000; Zhang & Yang, 1998; Bonzano, Cunningham, & Smyth, 1997; Wettschereck & Aha, 1995), and for items, we increase the counts in the ratio of accepted to presented items.

When determining the points in the dialogue at which to update the user model, we considered several factors. We wanted to allow the system to acquire information quickly, but not make erroneous assumptions. We thought that users might explore the search space the most while constraining attributes, so we decided not to update value preferences after each user-specified constraint. However, if we instead chose to only update the model after an item suggestion, the learning process might be too slow. The choices described below are, we feel, a good tradeoff between the extremes.

The three circumstances that we chose for user model update were after the user’s ACCEPT-ITEM, REJECT-ITEM, and ACCEPT-RELAX speech acts. First, we assume that when a user accepts an item, he is indicating: (1) a preference for the item itself, (2) preferences for the attributes he constrained to find this item, and (3) preferences for the values he provided for those attributes. Thus, when a user accepts an item presented by the system, the probabilities for the appropriate item, attributes, and values are increased. For the item preference, the system simply adds one to the presentation and acceptance counts. For attribute and value preferences the probability of the appropriate weight is increased by a small amount proportionate to its current weight, and all weights are renormalized. Thus attribute and value preferences do not truly reflect the probability that a user will choose the attribute or value, but is a biased measure that avoids zero counts for values never chosen by the user.

On the other hand, when a user rejects an item presented by the system, we only assume that he has a dislike for the particular item. We do not assume anything about the particular characteristics of that item, since the user has specified some of those characteristics. Therefore, for rejected items the system simply adds one to the presentation count.

The third situation in which the system updates the user model is when the query has become over-constrained, the system presents an attribute for relaxation, and the user accepts that relaxation. In this situation, we assume that had there been a matching item, the user would have been satisfied with it, since the characteristics specified in the conversation so far were satisfactory. Therefore, before the relaxation occurs, the attribute preferences for the constrained attributes and the value preferences for user-specified values are increased, in a manner similar to an ACCEPT-ITEM situation. This enables the Adaptive Place Advisor to more quickly make inferences about the user’s preferences.

4. System Evaluation

As stated earlier, we believe that user modeling increases the effectiveness and efficiency of conversations with the system over time. To test this hypothesis, we carried out an experiment with a version of the Adaptive Place Advisor that recommends restaurants in the San Francisco Bay Area. The system describes items in terms of seven attributes: cuisine, rating, price, location, reservations, parking options, and payment options. Most attributes have few values, but some, such as cuisine and location, have dozens. There are approximately 1900 items in the database.

We asked several users, all from the Bay Area, to interact with the system to help them decide where to go out to eat. The users were given no external guidance or instructions

on which types of restaurants to select, other than to look for and choose those that they might actually patronize. An experimenter was present during all these interactions, which were filmed, but his help was not needed except on rare occasions when a subject repeatedly tried words that were not included in the speech recognition grammar.

4.1 Experimental Variables

To test our hypothesis about the benefits of personalization in the Adaptive Place Advisor, we controlled two independent variables: the presence of user modeling and the number of times a user interacted with the system. We predicted the system’s interactions would improve over time, as it gained experience with each user, so we observed its behavior at successive points along this learning curve. In particular, each subject interacted with the system in 15 successive sessions. We tried to separate each subject’s sessions by several hours, but this was not always possible. However, in general the subjects did use the system to actually help them decide where to eat either that same day or in the near future; we did not provide constraints other than telling them that the system only knew about restaurants in the Bay Area. Because we anticipated that users might also improve their interactions with the Place Advisor over time, we divided subjects into an experimental or modeling group and a control group. The 13 subjects in the modeling group interacted with a version of the system that updated its user model after each conversation. The 11 subjects in the control group interacted with a version that did not update the model, but which selected attributes and items from the default distribution described in Section 3.1. Naturally, the users were unaware of their assigned group.

To determine each version’s efficiency at recommending items, we measured several conversational variables. One involved the average number of *interactions* needed to find a restaurant accepted by the user. We defined an interaction as a cycle that started with the system providing a prompt and ending with the system’s recognition of the user’s utterance in response, even if that response did not answer the question posed by the prompt. We also measured the *time* taken for each conversation. This began when a “start transaction” button was pushed and ended when the system printed “Done” (after the user accepted an item or quit).

We also collected two statistics that should not depend on whether or not user modeling is in effect. First is the number of *system rejections*, that is, the number of times either no recognition result was obtained or the system’s confidence was too low. In either case the system asks the user to repeat himself. Since they are a measure for recognition quality and not the effects of personalization, we omitted these from the count of interactions. A second, more serious problem is a speech *misrecognition* error in which the system assigns an utterance a different meaning than the user intended.

Effectiveness, and thus the subjective quality of the results, was somewhat more difficult to quantify. We wanted to know each user’s degree of satisfaction with the system’s behavior. One such indication is the *rejection rate*: the proportion of attributes about which the system asked but the subject did not care (REJECT-CONSTRAIN’s). A second measure is the *hit rate*: the percentage of conversations in which the first item presented is acceptable to the user. Finally, we also administered a questionnaire to users after the study to get more subjective evaluations.

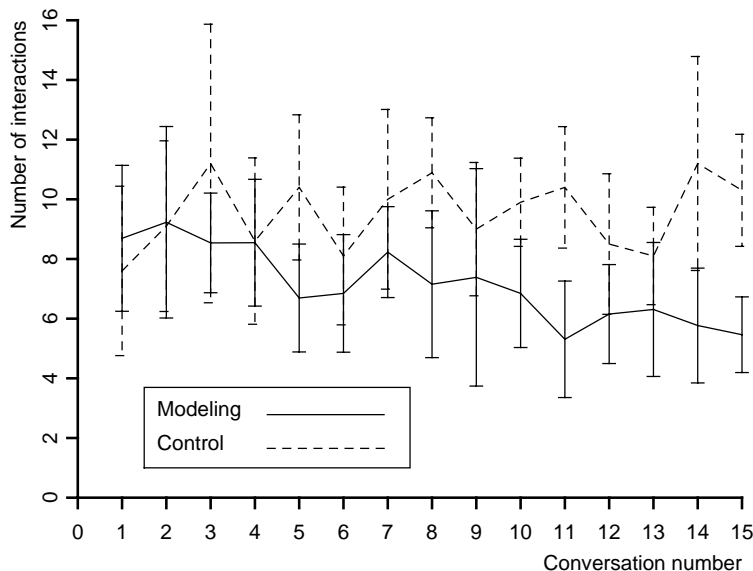


Figure 3: Average number of interactions per conversation

4.2 Experimental Results

The results of this experiment generally supported our hypothesis with respect to efficiency. We provide figures that show average values over all users in a particular group, with error bars to show the 95% confidence intervals. The x-axis always shows the progression of user’s interactions with the system over time: each point is for the n th completed (either found an acceptable restaurant or user quit) conversation.

Figure 3 shows that, for the modeling group, the average number of interactions required to find an acceptable restaurant decreased from 8.7 to 5.5, whereas for the control group this quantity actually increased from 7.6 to 10.3. We used linear regression to characterize the trend for each group and compared the resulting lines. The slope for the modeling line differed significantly ($p = 0.017$) from that for the control line, with the former lower than the latter, as expected.

The difference in interaction times (Figure 4) was even more dramatic. For the modeling group, this quantity started at 181 seconds and ended at 96 seconds, whereas for the control group, it started at 132 seconds and ended at 152 seconds. We again used linear regression to characterize the trends for each group over time and again found a significant difference ($p = 0.011$) for the two curves, with the slope for the modeling subjects being lower than that for the control subjects. We should also note that these interaction times include some time for system initialization (which could be up to 10% of the total dialogue time). If we had instead used as the start time the first system utterance of each dialogue, the difference between the two conditions would be even clearer.

The speech recognizer rejected 28 percent of the interactions in our study. Rejections slow down the conversation but do not introduce errors. The misrecognition rate was much lower – it occurred in only seven percent of the interactions in our experiment. We feel both these rates are acceptable, but expanding the number of supported utterances could reduce

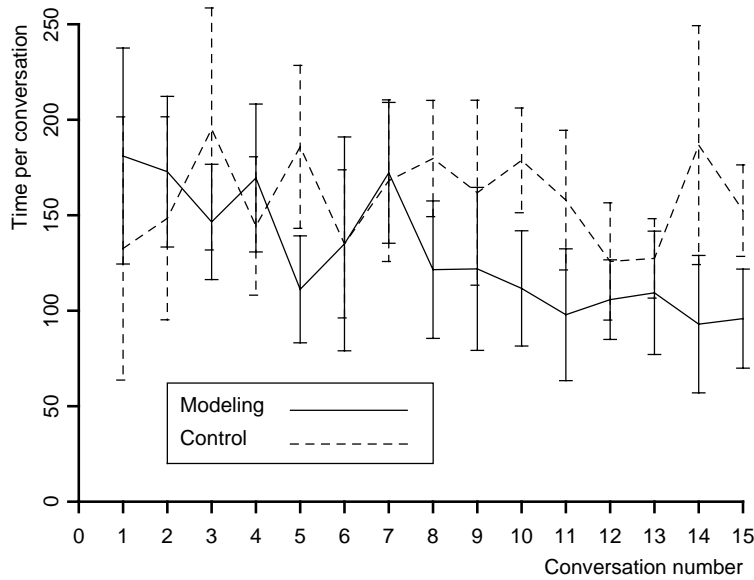


Figure 4: Average time per conversation

the first number further, while potentially increasing the second. In the most common recognition error, the Adaptive Place Advisor inserted extra constraints that the user did not intend.

The results for effectiveness were more ambiguous. Figure 5 plots the rejection rate as a function of the number of sessions. A decrease in rejection rate over time would mean that, as the system gains experience with the user, it asks about fewer features irrelevant to that user. However, for this dependent variable we found no significant difference ($p = 0.515$) between the regression slopes for the two conditions and, indeed, the rejection rate for neither group appears to decrease with experience. These negative results may be due to the rarity of rejection speech acts in the experiment. Six people never rejected a constraint and, on average, each person used only 0.53 REJECT-CONSTRAIN speech acts per conversation (standard deviation = 0.61).

Figure 6 shows the results for hit rate, which indicate that suggestion accuracy stayed stable over time for the modeling group but decreased for the control group. One explanation for the latter, which we did not expect, is that control users became less satisfied with the Place Advisor’s suggestions over time and thus carried out more exploration at item presentation time. However, we are more concerned here with the difference between the two groups. Unfortunately, the slopes for the two regression lines were not significantly different ($p = 0.1354$) in this case.

We also analyzed the questionnaire presented to subjects after the experiment. The first six questions (see Appendix A) had check boxes to which we assigned numerical values, none of which revealed a significant difference between the two groups. The second part of the questionnaire contained more open-ended questions about the user’s experience with the Adaptive Place Advisor. In general, most subjects in both groups liked the system and said they would use it fairly often if given the opportunity.

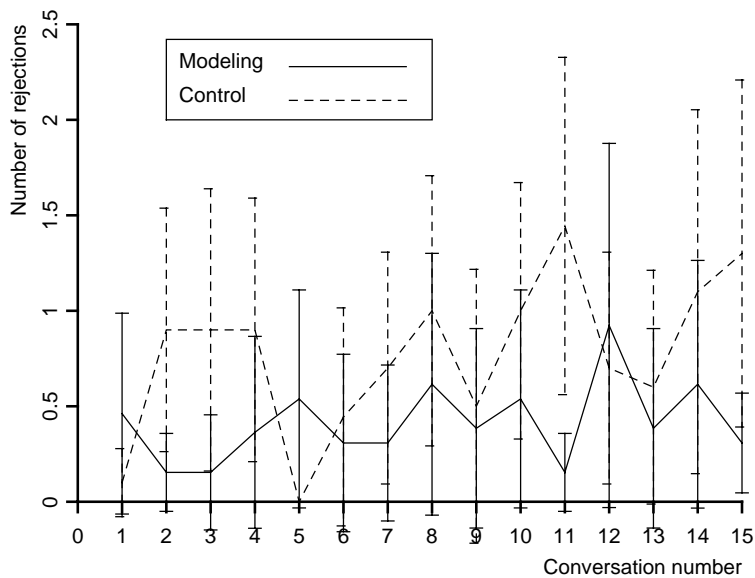


Figure 5: Rejection rate for modeling and control groups

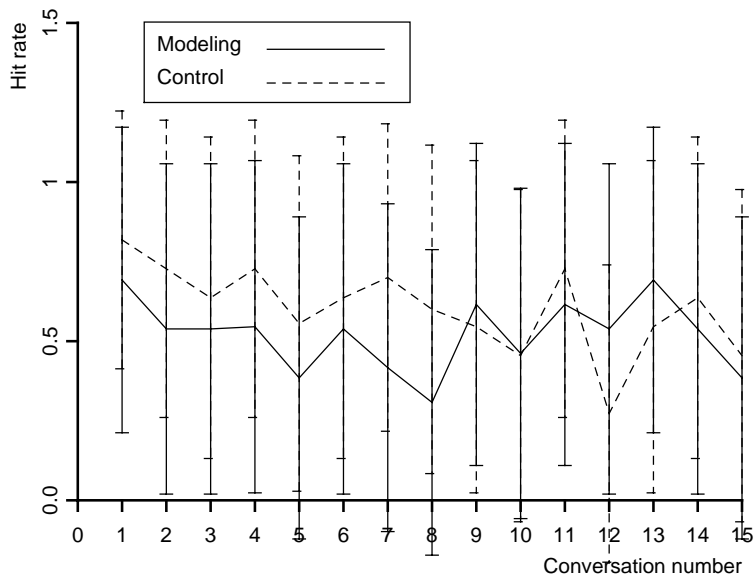


Figure 6: Hit rate for modeling and control groups

4.3 Discussion

In summary, our experiment showed that the Adaptive Place Advisor improved the efficiency of conversations with subjects as it gained experience with them over time, and that this improvement was due to the system’s update of user models rather than subjects learning how to interact with the system. This conclusion is due to the statistical significance results between the user modeling and control group, for both number of interactions and time per conversation. This significance is in effect even in the face of large error bars and a small sample size. This in turn implies that the differences are large and the system could make a substantial difference to users.

The results for effectiveness were more ambiguous, with trends in the right direction but no significant differences between the modeling and control groups. Subjects in both conditions generally liked the system, but again we found no significant differences along this dimension. Here, a larger study may be needed to determine whether differences exist between the two groups of users.

Further user studies are warranted to investigate the source of the differences between the two groups. One plausible explanation is that the largest influence on the user model is due to item accepts and rejects. The user model in turn influences the order of questions asked, which in turn influences the number of items matching at each point in the conversation. This in turn determines how soon items are presented in a conversation. To test this, we need to conduct a larger experiment so we can obtain a better measure of the influence of question order on how soon item presentation is begun. A preliminary test of this in the current study is the average number of interactions per user, before the first item is presented after an item search begins. This value decreased on average for the user modeling group (from 4.7 to 3.9) and increased for the control group (from 4.5 to 5.8). This is a reasonably large difference but the difference in slope for the two regression lines is not statistically significant ($p=0.165$).

5. Related Research

Previous research on related topics can be roughly broken up into three areas, the first focusing on personalized recommendation systems, the second on conversational interfaces, and the third on adaptive dialogue systems. We restrict our discussion here to the most strongly related work.

5.1 Personalized Recommendation Systems

Although research in personalized recommendation systems has become widespread only in recent years, the basic idea can be traced back to Rich (1979), discussed below with other work on conversational interfaces. Langley (1999) gives a more thorough review of recent research on the topic of adaptive interfaces and personalization.

Several other adaptive interfaces attempt to collect user information unobtrusively. An interesting example is the CASPER project (Rafter et al., 2000), an online recruitment service. The project investigates methods to translate raw click and read-time data into accurate relevancy information, given that the raw data is inherently noisy. This work shows that such raw data can be automatically translated into useful relevancy information.

Another example is the Adaptive Route Advisor (Rogers, Fiechter, & Langley, 1999), which recommends driving routes to a specified destination. The system collects preferences on attributes such as number of turns and driving time through the user's selections and modifications of the system's proposed routes. Finally, Goecks and Shavlik (2000) describe a technique for learning web preferences by observing a user's browsing behavior.

The constraint-based interaction style used by the Adaptive Place Advisor to search for items is not the only option. An alternative is taken by the candidate/critique, or tweaking approach. Tweaking systems, such as the Find Me suite (Burke, 1999) typically require the user to fill in values for a few predetermined attributes. They then present an item, at which point the user has the opportunity to change some search parameters to try to find a more desirable item. Eaton, Freuder, and Wallace (1997) take a similar approach in their Matchmaking system. In addition, they exploit constraint satisfaction to manage the search. Neither the Find Me systems nor the Matchmaking system, however, learn user models. A related system that does include a learning component is that of Shearin and Lieberman (2001), which learns attribute preferences unobtrusively. While tweaking is a valid method, it is not appropriate, we feel, as the primary method in an environment in which speech is the only mode of interaction, since presenting the user with his options would be somewhat cumbersome.

5.2 Conversational Interfaces

There is considerable ongoing work in the area of conversational systems, as evidenced in the general surveys by Dybkjær et al. (2000) and Maier et al. (1996). Zukerman and Litman (2001) give a more thorough overview of user modeling in dialogue systems. Rich (1979) reported one of the earliest (typewritten) conversational interfaces, which focused on book recommendation. At the beginning of an interaction the system asked several questions in order to place the user in a stereotype group, thereby initializing the user model. As each conversation progressed, this model was adjusted, with the system using ratings to represent uncertainty. However, the language understanding capabilities of the system were limited, mostly allowing only yes/no user answers. More recently, work within the dialogue community (Kobsa & Wahlster, 1998) models user's beliefs and intentions to aid in dialogue management and understanding, though typically these models are maintained only over the course of a single conversation.

As noted in Section 2.3, an important distinction is whether only one conversational participant keeps the initiative, or whether the initiative can switch between participants. An ambitious mixed-initiative system for planning tasks are the TRAINS (Allen et al., 1995) and more recent TRIPS (Allen et al., 2001) systems. Like the Place Advisor, the program interacts with the user to progressively construct a solution, though the knowledge structures are partial plans rather than constraints, and the search involves operators for plan modification rather than for database contraction and expansion. TRAINS and TRIPS lack any mechanism for user modeling, but the underlying system is considerably more mature and has been evaluated extensively.

Smith and Hipp (1994) describe another related mixed-initiative system with limited user modeling, in this case a conversational interface for circuit diagnosis. Their system aims to construct not a plan or a set of constraints, but rather a proof tree. The central

speech act, which requests knowledge from the user that would aid the proof process, is invoked when the program detects a ‘missing axiom’ that it needs for its reasoning. This heuristic plays the same role in their system as does the Place Advisor’s heuristic for selecting attributes to constrain during item selection. The interface only infers user knowledge during the course of one conversation, not over the long term, as in our approach.

With respect to dialogue management, several previous systems have used a method similar to our frame-based search. In particular, Dowding et al. (1993) and Seneff et al. (1998) developed conversational interfaces that give advice about air travel. Like the Place Advisor, their systems ask the user questions in order to reduce candidates, treating the choice of selecting airline flights as the interactive construction of database queries. However, the order of the questions is typically fixed in advance, despite the clear differences among individuals in this domain. Also, these systems usually require that all constraints be specified before item presentation begins.

An alternative to selecting questions to ask during information elicitation is presented in Raskutti and Zukerman (1997). Their overall system necessitates that the system recognize plans the user is attempting to carry out. Then the system must decide how to best complete those plans. It is in the situation when insufficient information is available to form a plan that their system enters an information seeking subdialogue similar to the constraint-satisfaction portion of our dialogues. Their system can decide which question to ask based on domain knowledge or based on the potential informativeness of the question.

Another approach to dialogue management is “conversational case-based reasoning” (Aha et al., 2001), which relies on interactions with the user to retrieve cases (items) from memory that will recommend actions to correct some problem. The speech acts and basic flow of control have much in common with the Adaptive Place Advisor, in that answering questions increasingly constrains available answers. One significant difference is that in their approach the system generates several questions or items, respectively, at a time, and the user selects which question to answer next or which item is closest to his or her needs, respectively.

5.3 Adaptation in Dialogue Systems

Finally, another body of recent work describes the use of machine learning or other forms of adaptation to improve dialogue systems.⁹ Researchers in this area have goals such as learning user preferences, improving task completion, and adapting dialogue strategy to an individual during a conversation.

The closest work is that also pursuing our goal of learning user preferences. Carberry, Chu-Carroll, and Elzer (1999) report one such example for consultation dialogues, but take a different approach. Their system acquires value preferences by analyzing both user’s explicit statements of preferences and their acceptance or rejection of the system’s proposals. Their system uses discrete preference values instead of our more fine-grained probability model. Also, their system does not use preferences during item search, but only at item presentation time to help evaluate whether better alternatives exist. Finally, their evaluation is based on subject’s judgements of the quality of the system’s hypotheses and recommendations, not

9. The work on adaptation of speech recognition grammars (e.g., Stolcke et al. (2000)), while related, addresses a different problem and uses different learning techniques, so we do not discuss it here.

on characteristics of actual user interactions. We could, however, incorporate some of their item search ideas to allow partial matches between user-specified constraints and actual items.

Another system that focuses on user preferences is an interactive travel assistant (Linden et al., 1997), that carries out conversations through a graphical interface. The system asks questions in an effort to narrow down the available candidates, using similar speech acts to ours, and also aims to satisfy the user with as few interactions as possible. To attempt to keep the number of interactions small, they rely on a candidate/critique approach. From user’s responses, the system infers a model represented as weights on attributes such as price and travel time. Unlike the Adaptive Place Advisor, it does not carry these profiles over to future conversations, but one can envision a version that stores longer-term models.

Several authors use reinforcement learning techniques to improve the probability of or process of task completion in a conversation. Singh et al. (2002) use reinforcement learning to determine the system’s level of initiative and amount of confirmation of user utterances. Their goal is to optimize, over all users, the percentage of dialogues for which a given task is successfully completed. This system applies the learned information across all users, rather than personalizing the information. Also, Levin, Pieraccini, and Eckert (1997) use reinforcement learning to determine which question to ask at each point during an information seeking search, but do not demonstrate the utility of their approach.

Finally, a number of systems adapt their dialogue management strategy over the course of a conversation based on user responses or other dialogue characteristics. For example, Litman and Pan (2002) use a set of learned rules to predict whether a user is having difficulty achieving their task, and adapt the level of system initiative and confirmation accordingly. Maloor and Chai (2000) present a help-desk application that first classifies the user as a novice, moderate, or expert based on responses to prompts. It then adapts the complexity of system utterances, the jargon, and the complexity of the path taken to achieve goals. Horvitz and Paek (2001) apply user modeling to dialogue systems, using evidence from the current context and conversation to update a Bayesian network that helps refine a spoken language recognition hypothesis and adapt the level of initiative accordingly. Chu-Carroll (2000) adapts both language generation and initiative strategies for an individual user within a single dialogue. Also, Jameson et al. (1994) use Bayesian networks in a system that can take the role of either the buyer or seller in a transaction, and that changes its inquiry or sales strategy based on beliefs inferred from the other participant’s utterances.

6. Directions for Future Work

Our results to date with the Adaptive Place Advisor are promising but much remains to be done. In this section, we discuss ways to make the search model more flexible, expand the conversational model, and enrich the user model and learning technique. We also consider more extensive evaluations of the system.

6.1 Search Model

With respect to the search mechanism, we first plan to investigate alternative techniques for using item similarity values to determine which to include, for example by cutting off items at the point at which similarity drops off most steeply, instead of our current use of

a threshold. We also note that work such as that of Cohen, Schapire, and Singer (1999) on learning to rank instances could apply nicely to this work, augmenting our current item ranking scheme. Additionally, we plan to allow the system to generate alternative items or values in an over-constrained situation (Qu & Beale, 1999), for example by using the user model to estimate the strength of a stated constraint, or by merging our preference-based similarity metric with a more traditional domain-specific similarity metric (Pieraccini et al., 1997). We also plan to evaluate the effect of making even stronger assumptions about user preferences. For example, if the system is certain enough about value preferences, a question about the associated attribute may not have to even be asked.

A final improvement of the search mechanism concerns the techniques for ranking attributes for constraining and relaxing. For the former, we have implemented but not yet evaluated a conditional entropy measure. The attribute to constrain is selected by determining the attribute with the lowest conditional entropy among the unconstrained attributes. This scheme would not be useful for ranking attributes to relax. Therefore, the system simply determines the size of the case base that would result if each attribute were relaxed, ranks these case bases from smallest to largest, and orders the attributes accordingly, excluding those attributes that if relaxed would still result in an empty case base.

We also plan to investigate the combination of the user model with information gain, as well as with alternative ranking techniques such as that used by Abella, Brown, and Buntschuh (1996). Another option is to add personalization to or otherwise adapt the variable selection techniques used by constraint-satisfaction solvers, especially as we move to more complex dialogues with more complex constraints.

6.2 Conversational Model

Our plans for the conversational model are less immediate. First, we plan to increase the number of speech acts available to the user. For example, we will add confirmation and better clarification dialogues, thus allowing other types of adaptation strategies, as in Singh et al. (2002). Moreover, the presence of a user model should aid speech recognition by providing probability distributions over the user's answers. In a longer term investigation, we plan to extend our adaptation techniques as needed to handle more complex travel planning dialogues (Seneff, Lau, & Polifroni, 1999; Walker & Hirschman, 2000). These dialogues will require more complex user models, as discussed below, and the possible addition of preferences regarding language and dialogue style, such as initiative, system verbosity, and vocabulary. These will in turn need to be appropriately acquired and utilized by the system. In general, the insights gained here into utilizing and acquiring user preferences at different junctures of the dialogue and search process should prove useful in supporting personalization in other tasks.

6.3 User Model

To improve the user model, we first plan to add more types of preferences. First, preferences for certain constraint combinations represent interactions among item characteristics (e.g., accepts Mexican restaurants only if they are cheap, or cares about parking only if going to San Francisco). Second, diversity preferences more fully capture the user's desire for vari-

ability in particular items or in attribute values. We plan to incorporate both combination and diversity preferences into the next version of our system.

Combination preferences can be used to predict either values or acceptable attributes, based on previously provided constraints. The first can be modeled by learning association rules (Agrawal, Imielinski, & Swami, 1993) or extending to a Bayesian network, which we would then use to influence the query, in turn influencing the similarity calculation, case base, and information gain calculation. For combination preferences about acceptable attributes, we can learn conditional probabilities based on past interactions, and use this to influence the ranking of attributes to constrain or relax.

While “drifting” preferences are not likely to cause problems in item selection applications, our model could be extended for user variability. One way to do this is to capture the user’s desired time interval between the suggestion of a particular item or value. This can be calculated by determining the mean time between explicit user selection or rejection of a value (value diversity preferences) or item (item diversity preferences). We will incorporate these into the similarity calculation from Section 3.2 by extending R_I and $P(V_j)$ in that equation to incorporate time effects. We define $R_D(I)$ and $P_D(V_j)$ as:

$$R_D(I) = R_I \times \frac{1}{1 + e^{-k_I(t-t_I-t_{ID})}}$$

$$P_D(V_j) = P(V_j) \times \frac{1}{1 + e^{-k_V(t-t_V-t_{VD})}} ,$$

where t is the current time, t_I and t_V are the time when the item or value was last selected, and t_{ID} and t_{VD} are the time differences the user wants to have between having the item or value suggested again. R_D and P_D are in form of a sigmoid function where k_I and k_V determine the slope of the curve. One empirical question is whether users also have attribute diversity preferences. We hypothesize that diversity preferences change on a value-by-value basis, and that this implicitly overrides attribute diversity. For example, a user may have strong preferences about the frequency with which expensive restaurants are suggested, but may not care about how often the price range of the suggested restaurants varies in general. We plan to investigate this hypothesis.

There are other improvements we might add to our user modeling technique. For example, the system may learn more quickly if it updates the user model from speech acts other than the current three of ACCEPT-ITEM, REJECT-ITEM, and ACCEPT-RELAX. Also, using collaborative user models (Konstan et al., 1997; Billsus & Pazzani, 1998; Smyth & Cotter, 1999) to initialize individual models could speed up the learning process. A more explicit combination of collaborative and individual user models is also a viable direction (Jameson & Wittig, 2001; Melville, Mooney, & Nagarajan, 2002).

6.4 Evaluation

Finally, we are planning to carry out a larger user study. In the current study, we did not control for the difficulty of finding a particular item, in terms of the number of constraints required to reduce the matching items enough to begin presentation, so we must verify that differences were not due to task difficulty differences. To support this expanded evaluation we have implemented a version of the system that recommends movies, thus allowing a

broader user base for our studies. This should help us measure user satisfaction more easily, as it has been noted (Walker et al. (1998)) that efficiency is not the only important consideration, and that users might tend to prefer more predictable interfaces.

7. Conclusions

Overall, we have made significant inroads into methods for unobtrusively acquiring an individual, long term user model during recommendation conversations. Our long-term goal is to develop even more powerful methods, capable of adapting to the needs, goals, and preferences of a user over multiple conversations. In this paper, we described an intelligent adaptive conversational assistant designed to help people select an item. We expand on previous work on adaptive recommendation systems that were not conversational, and on dialogue systems that were not user adaptive. While we have leveraged off the feedback between conversation and recommendation, such feedback is likely to be present in other tasks such as planning or scheduling.

The two key problems addressed by our research are the design of adaptive recommendation systems when conversations are the interaction mode, and the addition of personalization to dialogue systems, starting here with dialogues for recommendation. Thus, unlike many recommendation systems that accept keywords and produce a ranked list, this one carries out a conversation with the user to progressively narrow his options. In solving these problems, we designed a novel approach to the acquisition, use, and representation of user models. Unlike many other adaptive interfaces, our system constructs and utilizes user models at levels of detail beyond that of complete items to help better support personalization in conversations. We have started with a relatively simple model of dialogue in order to focus on the issues involved in personalization. We also described experimental results showing the promise of our technique, demonstrating a reduction in both the number of interactions and conversation time for users interacting with our system when compared to a control group.

Of course, there are still several open questions and opportunities for improvement. The user model, conversational model, and search models are functional but we plan to improve them further. We are also extending our conversational approach to items other than destinations, such as books and movies, and we plan to link the system to other assistants like the Adaptive Route Advisor (Rogers et al., 1999). Our goal for such additions is to provide new functionality that will make the Adaptive Place Advisor more attractive to users, but also to test the generality of our framework for adaptive recommendation. In turn, these should bring us closer to truly flexible computational aides that carry out natural dialogues with humans.

Acknowledgements

This research was carried out while the first author was at the Center for the Study of Language and Information, Stanford University, and the other authors were at the DaimlerChrysler Research and Technology Center in Palo Alto, California. We thank Renée Elio, Afsaneh Haddadi, and Jeff Shrager for the initial conception and design of the Adaptive Place Advisor, Cynthia Kuo and Zhao-Ping Tang for help with the implementation effort,

and Stanley Peters for enlightening discussions about the design of conversational interfaces. Robert Mertens and Dana Dahlstrom were crucial in carrying out the user studies.

Appendix A. Questionnaire

1. What did you think about the interaction with the system, did it

0	1	2	3	4	5	6	7	8
talk				right				not
too				amount of				enough
much				talking				talking

2. How easy was it to find a restaurant you liked?

0	1	2	4
very			not
easy			easy

3. Did the system deliver restaurants you liked?

0	1	2	4
yes			no

4. Please rate the interaction with the system on a scale between standard human-computer-interaction and a person to person conversation via telephone.

0	1	2	3	4	5	6
human-						phone
computer						conversation
interaction						

5. Do you think the APA is a useful system?

0	1	2	3	4
yes				no

6. Do you think the conversation was significantly more distracting than a similar conversation with a real person?

0	1	2	3	4
no				yes

References

- Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, 7(1), 39–59.
- Abella, A., Brown, M. K., & Buntschuh, B. (1996). Development principles for dialog-based interfaces. In *Proceedings ECAI-96 Spoken Dialog Processing Workshop*, pp. 1–7 Budapest, Hungary.
- Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. In Buneman, P., & Jajodia, S. (Eds.), *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207–216 Washington, D.C.
- Aha, D., & Breslow, L. (1997). Refining conversational case libraries. In *Second International Conference on Case-Based Reasoning*, pp. 267–278 Providence, RI. Springer Verlag.
- Aha, D., Breslow, L., & noz Avila, H. M. (2001). Conversational case-based reasoning. *Applied Intelligence*, 14, 9–32.
- Allen, J. (1999). Mixed-initiative interaction. *IEEE Intelligent Systems*, September/October, 14–16.
- Allen, J., Byron, D., Dzikovska, M., Ferguson, G., Galescu, L., & Stent, A. (2001). Towards conversational human-computer interaction. *AI Magazine*, 22(4), 27–37.
- Allen, J., Schubert, L., Ferguson, G., Heeman, P., Hwang, C. H., Kato, T., Light, M., Martin, N., Miller, B., Poesio, M., & Traum, D. (1995). The TRAINS project: A case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI*, 7, 7–48.
- Allen, J. F. (1995). *Natural Language Understanding (2nd Ed.)*. Benjamin/Cummings, Menlo Park, CA.
- Ardissono, L., & Goy, A. (2000). Tailoring the interaction with users in web stores. *User Modeling and User-Adapted Interaction*, 10(4), 251–303.
- Billsus, D., & Pazzani, M. (1998). Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 46–54 Madison, WI. Morgan Kaufman.
- Bobrow, D., Kaplan, R., Kay, M., Norman, D., Thompson, H., & Winograd, T. (1977). Gus, a frame driven dialog system. *Artificial Intelligence*, 8, 155–173.
- Bonzano, A., Cunningham, P., & Smyth, B. (1997). Using introspective learning to improve retrieval in CBR: A case study in air traffic control. In *Second International Conference on Case-Based Reasoning*, pp. 413–424. Springer Verlag.

- Brusilovsky, P., & Maybury, M. (2002). Introduction to special section on the adaptive web. *Communications of the ACM*, 45(5), 30–33.
- Burke, R. (1999). The wasabi personal shopper: A case-based recommender system. In *Proceedings of the 16th National Conference on Artificial Intelligence*, pp. 844–849.
- Burke, R., Hammond, K., & Young, B. (1996). Knowledge-based navigation of complex information spaces. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pp. 462–468.
- Carberry, S. (1990). *Plan recognition in natural language dialogue*. MIT Press, Cambridge, MA.
- Carberry, S., Chu-Carroll, J., & Elzer, S. (1999). Constructing and utilizing a model of user preferences in collaborative consultation dialogues. *Computational Intelligence Journal*, 15(3), 185–217.
- Chin, D. (1989). KNOOME: Modeling what the user knows in UC. In Kobsa, A., & Wahlster, W. (Eds.), *User models in dialog systems*, pp. 74–107. Springer Verlag, Berlin.
- Chu-Carroll, J. (2000). MIMIC: an adaptive mixed initiative spoken dialogue system for information queries. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pp. 97–104 Seattle, WA.
- Cohen, P. R., & Perrault, C. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, 3, 177–212.
- Cohen, W., Schapire, R., & Singer, Y. (1999). Learning to order things. *Journal of Artificial Intelligence Research*, 10, 243–270.
- Cotter, P., & Smyth, B. (2000). PTV: Intelligent personalized TV guides. In *Proceedings of the 12th Innovative Applications of Artificial Intelligence Conference*.
- Cucchiara, R., Lamma, E., Mello, P., & Milano, M. (1997). Interactive constraint satisfaction. Tech. rep. DEIS-LIA-97-00, University of Bologna.
- Dowding, J., Gawron, J., Appelt, D., Bear, J., Cherny, L., Moore, R., & Moran, D. (1993). Gemini: A natural language system for spoken-language understanding. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 54–61.
- Dybkjær, L., Hasida, K., & Traum, D. (Eds.). (2000). *1st SIGdial Workshop on Discourse and Dialogue*, Hong Kong. Association for Computational Linguistics.
- Eaton, P., Freuder, E., & Wallace, R. (1997). Constraint-based agents: Assistance, cooperation, compromise. In *CP97 Workshop on Constraint Reasoning on the Internet* Schloss Hagenberg, Austria.
- Elio, R., & Haddadi, A. (1998). Dialog management for an adaptive database assistant. Tech. rep. 98-3, Daimler-Benz research and Technology Center, Palo Alto, CA.

- Elio, R., & Haddadi, A. (1999). On abstract task models and conversation policies. In *Proceedings of the Agents'99 Workshop on Specifying and Implementing Conversation Policies* Seattle, WA.
- Ferrario, M., Waters, K., & Smyth, B. (2000). Collaborative maintenance in ULYSSES. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems* Trento, Italy.
- Fiechter, C., & Rogers, S. (2000). Learning subjective functions with large margins. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 287–294 Stanford University, CA. Morgan Kaufmann Publishers.
- Goddeau, D., Meng, H., Polifroni, J., Seneff, S., & Busayapongchai, S. (1996). A form-based dialogue manager for spoken language applications. In *Proceedings of the Fourth International Conference on Spoken Language Processing*, Vol. 2, pp. 701–704 Philadelphia, PA.
- Goecks, J., & Shavlik, J. (2000). Learning users' interests by unobtrusively observing their normal behavior. In *Proceedings of the 2000 International Conference on Intelligent User Interfaces*, pp. 129–132 New Orleans, LA.
- Göker, M., & Roth-Berghofer, T. (1999). The development and utilization of the case-based help-desk support system HOMER. *Engineering Applications of Artificial Intelligence*, 12, 665–680.
- Haller, S., & McRoy, S. (1998). Special issue: Computational models of mixed-initiative interaction. *User Modeling and User-Adapted Interaction*, 8.
- Horvitz, E., & Paek, T. (2001). Harnessing models of users' goals to mediate clarification dialog in spoken language systems. In *Proceedings of the Eighth International Conference on User Modeling*.
- Jameson, A., Kipper, B., Ndiaye, A., Schäfer, R., Simons, J., Weis, T., & Zimmermann, D. (1994). Cooperating to be noncooperative: The dialog system PRACMA. In *KI-94: Advances in artificial intelligence*, pp. 106–117.
- Jameson, A., & Wittig, F. (2001). Leveraging data about users in general in the learning of individual user models. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pp. 1185–1192.
- Jurafsky, D., & Martin, J. (2000). *Speech and Language Processing*. Prentice Hall.
- Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., Fosler, E., & Morgan, N. (1994). The berkeley restaurant project. In *International Conference on Spoken Language Processing*, pp. 2139–2142 Yokohama, Japan.
- Kass, R. (1990). Building a user model implicitly from a cooperative advisory dialog. In *2nd International Workshop on User Modeling*.

- Kay, J., & Thomas, R. C. (2000). Personal usability based upon a scrutable, dynamic, individual user model. In *Australasian Computer Human Interfaces Conference*, pp. 292–298.
- Kobsa, A., & Wahlster, W. (1998). Special issue on user modeling. *Computational Linguistics*, 14.
- Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., & Riedl, J. (1997). Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3), 77–87.
- Kumar, V. (1992). Algorithms for constraints satisfaction problems: A survey. *The AI Magazine*, 13(1), 32–44.
- Lang, K. (1995). NewsWeeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 331–339 San Francisco, CA. Morgan Kaufman.
- Langley, P. (1999). User modeling in adaptive interfaces. In *Proceedings of the Seventh International Conference on User Modeling*, pp. 357–370 Banff, Alberta. Springer.
- Levin, E., Pieraccini, R., & Eckert, W. (1997). Learning dialogue strategies within the Markov decision process framework. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 72–79.
- Linden, G., Hanks, S., & Lesh, N. (1997). Interactive assessment of user preference models: the automated travel assistant. In *Proceedings of the Sixth International Conference on User Modeling*, pp. 67–78 Chia Laguna, Sardinia. Springer.
- Litman, D., & Pan, S. (2002). Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction*, 12(2/3), 111–137.
- Maier, E., Mast, M., & Luperfoy, S. (Eds.). (1996). *Dialogue processing in spoken language systems: ECAI'96 workshop*, Budapest, Hungary. Springer Verlag.
- Maloor, P., & Chai, J. (2000). Dynamic user level and utility measurement for adaptive dialog in a help-desk system. In *1st SIGdial Workshop on Discourse and Dialogue*.
- Melville, P., Mooney, R., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pp. 187–192 Edmonton, Canada.
- Pazzani, M., Muramatsu, J., & Billsus, D. (1996). Syskill & Webert: Identifying interesting web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pp. 54–61 Portland, OR.
- Pieraccini, R., Levin, E., & Eckert, W. (1997). AMICA: The AT&T mixed initiative conversational architecture. In *Proceedings European Conference on Speech Communication and Technology*, pp. 1875–1878 Rhodes, Greece.

- Qu, Y., & Beale, S. (1999). A constraint-based model for cooperative response generation in information dialogues. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pp. 148–155 Orlando, FL.
- Rafter, R., Bradley, K., & Smyth, B. (2000). Personalized retrieval for online recruitment services. In *Proceedings of the 22nd Annual Colloquium on Information Retrieval*.
- Raskutti, B., & Zukerman, I. (1997). Generating queries and replies during information-seeking interactions. *International Journal of Human Computer Studies*, 47(6), 689–734.
- Resnick, P., & Varian, H. (1997). Recommender systems. *Communications of the ACM*, 40(3).
- Rich, E. (1979). User modeling via stereotypes. *Cognitive Science*, 3, 329–354.
- Rogers, S., Fiechter, C., & Langley, P. (1999). An adaptive interactive agent for route advice. In *Proceedings of the Third International Conference on Autonomous Agents*, pp. 198–205 Seattle, WA.
- Sadek, M., Bretier, P., & Panaget, F. (1997). ARTIMIS: natural dialogue meets rational agency. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* Nagoya, Japan.
- Segal, R., & Kephart, J. (1999). Mailcat: An intelligent assistant for organizing e-mail. In *Proceedings of the Third International Conference on Autonomous Agents*, pp. 276–282 Seattle, WA. ACM Press.
- Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., & Zue, V. (1996). Galaxy-II: A reference architecture for conversational system development. In *International Conference on Spoken Language Processing*, pp. 931–934 Sydney, Australia.
- Seneff, S., Lau, R., & Polifroni, J. (1999). Organization, communication, and control in the Galaxy-II conversational system. In *Proceedings of Eurospeech 1999*.
- Shardanand, U., & Maes, P. (1995). Social information filtering: algorithms for automating ‘word of mouth’. In *Proceedings of the Conference on Human Factors in Computing Systems*, pp. 210–217. ACM Press.
- Shearin, S., & Lieberman, H. (2001). Intelligent profiling by example. In *Proceedings of the International Conference on Intelligent User Interfaces*, pp. 145–152 Santa Fe, NM.
- Singh, S., Litman, D., Kearns, M., & Walker, M. (2002). Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research*, 16, 105–133.
- Smith, R., & Hipp, D. (1994). *Spoken natural language dialog systems: a practical approach*. Oxford University Press, New York, NY.

- Smyth, B., & Cotter, P. (1999). Surfing the digital wave, generating personalized TV listings using collaborative, case-based recommendation. In *Proceedings of the Third International Conference on Case-Based Reasoning*, pp. 561–571.
- Stent, A., Dowding, J., Gawron, J., Bratt, E., & Moore, R. (1999). The CommandTalk spoken dialogue system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics* College Park, MD.
- Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C. V., & Meteer, M. (2000). Dialog act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3), 339–373.
- Walker, M., Fromer, J., Fabbriozio, G., Mestel, C., & Hindle, D. (1998). What can I say?: Evaluating a spoken language interface to email. In *Proceedings of ACM CHI 98 Conference on Human Factors in Computing Systems*, pp. 582–589.
- Walker, M., & Hirschman, L. (2000). Evaluation for DARPA communicator spoken dialogue systems. In *Proceedings Second International Conference on Language Resources and Evaluation*.
- Ward, W., & Issar, S. (1996). Recent improvements in the CMU spoken language understanding system. In *Proceedings ARPA Human Language Technology Workshop*, pp. 213–216.
- Wettschereck, D., & Aha, D. (1995). Weighting features. In *Proceedings of the First International Conference on Case-Based Reasoning*, pp. 347–358. Springer Verlag.
- Winograd, T., & Flores, F. (1986). *Understanding computers and cognition: a new foundation for design*. Ablex Publishing, Northwood, NJ.
- Zhang, Z., & Yang, Q. (1998). Towards lifetime maintenance of case base indexes for continual case based reasoning. In *Proceedings of the 1998 International Conference on AI Methodologies, Systems and Applications*, pp. 489–500 Bulgaria.
- Zukerman, I., & Litman, D. (2001). Natural language processing and user modeling: Synergies and limitations. *User Modeling and User-Adapted Interaction*, 11, 129–158.