Human Computer Interaction:
Issues and Challenges

Qiyang Chen
Montclair State University, USA

IDEA GROUP PUBLISHING
Hershey • London • Melbourne • Singapore

Acquisition Editor: Mehdi Khosrowpour
Managing Editor: Jan Travers
Development Editor: Michele Rossi
Copy Editor: Brenda Zboray Klinger
Typesetter: Tamara Gillis
Cover Design: Deb Andree
Printed at: Sheridan Books

Published in the United States of America by
Idea Group Publishing
1331 E. Chocolate Avenue
Hershey PA 17033-1117
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@idea-group.com
Web site: http://www.idea-group.com

and in the United Kingdom by
Idea Group Publishing
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 3313
Web site: http://www.eurospan.co.uk

Library of Congress Cataloging-in-Publication Data

Human computer interaction : issues and challenges / Qiyang Chen [ed.].
p. cm.
Includes bibliographical references and index.
ISBN 1-878289-91-8 (paper)
1. Human-computer interaction. I. Chen, Qiyang, date.
QA76.9.H85 H858 2001
004'.01'9--dc21 00-054108

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter VIII

Knowledge Engineering in Adaptive Interface and User Modeling

Qiyang Chen, Montclair State University, USA
A. F. Norcio, University of Maryland, Baltimore County, USA

ABSTRACT

This chapter presents the issues of user modeling and its role in adaptive human-computer interface (HCI). Particularly, it focuses on knowledge acquisition and representation in user modeling. Several related problems in the traditional user modeling systems are also identified and discussed.

ADAPTIVE INTERFACE AND USER MODELING

The concept of an adaptive human-computer interface (HCI) has been recognized as a very promising and challenging area for both research and applications (Norcio and Stanley, 1989). The objective of an adaptive interface is to adapt system responses to the user effectively in a complex computer-based task. In the context of human computer interaction, the relationship between a human and a computer involves many factors such as the computing environment, the nature of the tasks to be performed, as well as various characteristics of the users. The effectiveness of a human-computer interface system is influenced greatly by its ability to adapt to these factors.

As the functionality of computer systems becomes more complex and users' tasks vary, users must adjust their behavior and problem solving strategies to the systems. This situation is often made worse because users lack the knowledge and experience to be effective. A well-designed interface can provide much more helpful information in a more appropriate manner during the interaction, especially for those users who have limited experiences. Therefore, an adaptive interface is not only beneficial for users but also for system resources management since the tasks and the related resources are better allocated between user and the computer.

Generally, an adaptive interface should be able to offer an effective interaction and allocate tasks dynamically between the user and the computer system. It is not unusual that users are frequently confronted with an overwhelming amount of information in interacting with computers. Users must decide what information to request and use. In these situations, it has been found that users do not usually perform optimally (Card, Moran and Newell, 1983). In addition, users may not have the necessary information or expertise to adjust their

behavior. The adaptability of an interface is helpful for the users with different backgrounds, because the interface system can individualize its responses. An adaptive interface can increase user proficiency with a new system and allow novices and experts to use the system with equal ease.

To adapt to a user effectively and correctly, an interface system must be able to characterize and distinguish individual users. User modeling, a process of establishing a collection of the system's beliefs about various users' characteristics, has become an important component in adaptive interface systems. An interface equipped with user modeling component is able to tailor its responses to individual users. Generally, an adaptive interface has the following advantages:

- Economy of the interaction: The dialogue may be short, more precise, better focused and understandable. Since system responses can rely on default knowledge already stored in the user model, the system *knows* what the next optimal response should be to help the user perform the task.
- User acceptability: The dialog is individually tailored. Thus, it becomes more acceptable to the user. In addition to providing a clearer dialogue, a user model can provide the basis of explanations of the solution to users. It can also detect the user's misconceptions in the task performance; and therefore, it can provide justifications for the adaptation.
- Effectiveness and efficiency of use: The access to the target system and its use may become more effective and efficient, in terms of both quality and cost of the performance.

Research in the field of user modeling dates to the early seventies (Self, 1974; Bruce, 1975; Allen and Perrault, 1978; Cohen, 1978; Rich, 1979). It is widely recognized that computer systems should adapt to users in an intelligent and cooperative manner. It is also evident that computer systems can acquire these capabilities on a large scale only if they have the knowledge about users and the tasks that the users are performing. Constructing, maintaining, and utilizing user models has become an active research area. Since the mid-eighties, user modeling research has focused on four different but interrelated domains: human-computer interface (Murray, 1987; Dede, 1986; Botman *et al.,* 1987), natural language dialog systems (Wahlster and Kobsa, 1989), intelligent tutoring systems (Self, 1974; Kass, 1989, Selker, 1994), and information retrieval (Daniel, 1986; Allen, 1990, Kramer *et al*, 2000). From the viewpoint of system functionality, user modeling can be considered as a component of the interface for optimizing human-computer interactions.

Some studies of user interface management systems (UIMS) partially address adaptation of information displays based on features of the system's operation (Eberts and Eberts, 1989). With the UIMS approach, the interface becomes an important and separate system component to which software engineering techniques are applied. The UIMS approach facilitates interface design by providing a set of tools. A UIMS provides a definition language for representing the dialogue required and a generator that automatically produces the necessary code from a source definition in this language. A UIMS typically includes screen generation tools, a graphics package, and editors for help messages, error messages, prompts, forms, icons, and so on. The typical runtime functions are management of multiple windows and conversion of task output to user representation. However, the characteristics of users and tasks to which the system needs to adapt are not emphasized. The UIMS approach provides a flexible and reliable interface design strategy but fails to make the interface adapt dynamically because the underlying cognitive models

and other relative characteristics of users are not considered (Norcio and Stanley, 1989). It is acknowledged that in many applications such as tutorial systems, information retrieval systems, or security control systems, personalized interfaces are crucial to the success of the system (Kramer *et al*, 2000).

In an adaptive interface, user modeling is a primary mechanism that allows the system to tailor its responses to individual users during the interaction. A wide variety of user models have been proposed. In order to adapt to users, a user modeling system must address the following issues:

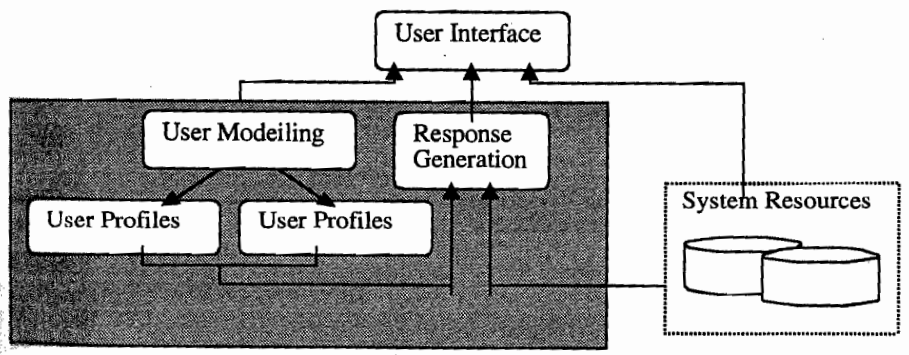
- The existence of information or knowledge about the user, the task, and the context of interaction;
- The approaches of eliciting the above information or knowledge;
- The strategies of utilizing this information or knowledge.

Figure 1 shows the framework of an interface system equipped with a user modeling component. The user modeling subsystem monitors the interaction and dynamically produces profiles for both the user and the tasks. According to these profiles, the system tailors personalized information from various system resources to provide to the user.

Without considering the application domain, the possible roles of user models in interactive systems can be illustrated as follows:

- User models can be used to analyze input and generate output. User models are helpful for determining what and how to prompt a user and interpreting a user's response without ambiguity. In a natural language system, it is often necessary to determine what a user really means. Uncertainties involved in a user's language need to be minimized. In addition, the system needs to determine a user's intended meaning, even though it is not explicitly stated. User models may provide prerequisite information that may not be directly stated in a user input, construct referring expressions, and make lexical choice so that the system is able to decide what and how to respond to a user input.
- User models assist in recognizing a user's goals and plans so that the system can better understand the user's information-seeking behavior. A user's goals and plans are related. A user's goal is what the user wants to achieve after interacting with a computer system, while the plan is considered as a sequence of actions that results in the realization of a goal. Furthermore, each action in a plan may have its own subgoals. One approach to recognize a user's goal is to observe and analyze the user's actions.
- User models help to evaluate when to provide information, as well as recognize and

Figure 1. A Framework of Adaptive Interface System



correct misconceptions, so that the system can provide more meaningful help and advice. Knowing what the user believes to be true is especially useful for many kinds of systems. Knowledge of the concepts that a user understands or misunderstands allows the system to produce appropriate responses. This feature is particularly necessary in instructional or tutorial systems.

DESIGN USER MODEL

The term "user model" is used in many different contexts to describe task-oriented information or knowledge that is used to support human-computer interaction. There are a variety of views of user model, and none is universally accepted. Thus, the best way for understanding user modeling mechanisms is to explore their features in terms of their organization and utilization.

Knowledge Sources of a User Model

The contents of a user model can directly affect the user-computer interaction. Given a task, different user models usually cause different system responses. Some or all of the following aspects might be incorporated into a user model:

- User's knowledge about the system and proficiency regarding the system operations.
- User's goals that underlie the tasks being performed; the plans that a user follows to achieve the goal; and the beliefs held by a user regarding a particular domain.
- User's preferences and cognitive characteristics.
- Characteristics about the tasks.

The contents of the user model vary greatly from application to application. For example, a tutorial system may need to capture all concepts known (or unknown) to a student (Paris, 1989; Nwana, 1991). A database query system may need to model data items with which a user is concerned (Chen and Norcio, 1997; Heger *et al.*, 1991). An expert system may need to model a user's domain goals (Chin, 1989). Rich (1979, 1983) classifies the information incorporated in a user model through three domain-independent dimensions:

- *The long-term/short-term dimension:* Long-term models describe relatively static characteristics of users, while short-term models describe specific topics and goals in the current discourse. The dimension characterizes the persistence of a user model established during an interaction.
- *Canonical / individual dimension:* A canonical model only describes the characteristics of a "typical" system without considering the individual differences among users, while an individual model concerns those characteristics that are highly salient and relevant to the task being performed by a specific user.
- *Explicit/implicit dimension:* This dimension concerns how to elicit and represent the information from the previous two dimensions. The model is specified directly by a user or established through observing user behavior in the interaction. A mechanism for inferring the individual models from observed events must be provided.

Given this context, similar classifications, such as given vs. inferred, or static vs. dynamic, are also proposed (Daniel, 1986; Brajnik *et al.*, 1990; Clowes *et al.*, 1985; Allen, 1990). The idea of this classification is to provide a generic consideration on how the user model is established and utilized. Obviously, this classification does not provide an insight into the dynamics of a user model, which includes a user's progressive or evolutionary capability, a user's cognitive style, as well as a user's mental workload in task performance. Allen (1990) also suggests that this classification is not clear about relationships or

transitions between these dimensions.

These dimensional settings are consistent with each other. Some also imply others. Essentially, they attempt to characterize knowledge about a user in four aspects (Chen and Norcio, 1997):

- Information persistency or temporal extent
- Methods of knowledge elicitation and representation
- Degree of specification
- Relationships between these three aspects

The knowledge for constructing a user model may encompass three areas:

1) *User's domain knowledge and task related characteristics.* The user model should be able to characterize and to distinguish between individuals. The information about a user's domain knowledge, capacities, and cognitive preferences should be obtained for each user in order to adapt to the task being performed. Also, the predicted user's beliefs, goals and plans are the components of a user model.
2) *Task information.* While the term "user model " emphasizes the information about the person's long term and short-term characteristics, it is obvious that a great deal of task information may be encoded in the model. It is suggested that dynamic information about a user's current state in task performance is more important than the user's long-term characteristics for generating an appropriate system response. The GOMS model is one of the common approaches for predicting the task performance (Card *et al.*, 1983).
3) *Knowledge of the interaction.* A user model should be able to keep track of the current human computer dialogue. It requires knowledge about how the interaction is constructed and what information may be implied from it. A transition diagram of a dialogue can be considered as a representation of such knowledge in a user model.

In order to incorporate knowledge about users and their tasks in a structured way, some domain independent features must be implemented into a user modeling system. Finin (1989) suggests several important features that can be synthesized:

- *Separate Knowledge Base*: Information about a user is organized in a separate module rather then distributed throughout the system. This knowledge base distinguishes the system's knowledge about its user from the other target resources such as application databases, programs, etc.
- *Explicit and Declarative Representation*: Knowledge in a user model is encoded in a declarative rather than procedural manner. A user's profile is represented explicitly.
- *Support for Abstraction*: The modeling system provides a function of generalization so that it can address a class of users as well as individuals.
- *Multiple Use*: Since the user model is represented as a separate module, it can be utilized in several ways such as classifying a user, monitoring a user's performance, and tailoring the system response.

Knowledge Acquisition and Representation in User Model

Norcio and Stanley (1989) summarize three common approaches for establishing a user model. They are:

- Classify users from novices to experts according to their operational proficiency and domain expertise,
- Compare a user's knowledge against an expert's knowledge, and
- Characterize users by a set of stereotypical traits.

The ways of constructing a user model can be classified as explicit elicitation and implicit elicitation:

1) *Explicit information elicitation.* The information needed for a user model is learned directly from the user's input. The system may prompt the user for characteristics regarding a certain task. The user's response to these questions can be formed as an initial profile. An example is the Grundy system that is used to model a user's social status such as sex, age, and occupation (Rich, 1979; Hoeppner *et al.*, 1986).
2) *Implicit information elicitation.* Through monitoring the human-computer interaction, information about a user can be obtained indirectly. The system deducts the user model contents according to observations on various factors. These factors are usually domain dependent. However, they could be domain-independently classified as *behavior data* and *conceptual objects*.

Behavior data are statistics that focus on the user's performance during interactive sessions, such as the types of commands used, the error rate, the time span of a dialogue, the number of help requests, the usage of optimal command sequence, etc. Also, these data can be considered as explicitly obtained since they are primarily based on objective observation. However, the explanation of the implication from these data is crucial in user modeling. The most common usage of such statistics is to classify the user's operational proficiency toward the target system.

A user model functions only as the system's knowledge base about the user. Its establishment should not become the user's obligation and workload. Therefore, the balance between explicit elicitation and implicit elicitation is important for effectiveness of the interface system. The modeling process should be *transparent* to the user. Actually, explicit elicitation may not guarantee the validity of obtained information. For instance, a user's claimed programming skill as "excellent" may not fit the system's criteria. However, this transparency may raise some social or psychological issues. For example, a user may be reluctant to be modeled and may make his/her behavior abnormal for various reasons (Murray, 1987).

In the user modeling process consistency is a major concern in this process. The acquired knowledge should be incorporated into the existing user model without causing any contradiction. Usually default reasoning and evidential reasoning are utilized.

Default reasoning allows the modeling process to maintain hypothetical knowledge about a user in the absence of evidence to the contrary. In order to make a large number of inferences with a small number of observations, extensive default assumptions must be established, even though they may not be true during the interaction. Therefore, techniques for handling incomplete and inconsistent information must be involved in the modeling process.

If the model construction is driven by evidential reasoning, the numerical values of the beliefs about the facts and rules must be carefully defined. Further, modification of such belief values is necessary until adequate modeling performance is obtained. However, even if the system can identify the correlation between the observations and their implications, it is still difficult to make a probabilistic assignment especially for some intermediate concepts or conclusions that support inferences and explanations. Even with a well-understood situation, it is difficult to maintain probabilistic formalisms (*e.g.*, independence of probabilities) over the whole production system (Zadah, 1989). Many sources of uncertainties can lead reasoning systems to draw inconsistent conclusions (Bonissone *et al.*, 1985). In addition, off-line tuning of belief values is both time- consuming and inexact. It is often based on inadequate tests and a subjective judgment of how the interaction is

progressing, and local improvement obtained by tuning one capability of the modeling process might be detrimental to other capabilities (Bonissone *et al.*, 1985).

ASSUMPTIONS AND STEREOTYPES

It is a common practice to use a set of predefined assumptions to initialize the system's beliefs about its users and the tasks they are performing. It is called the stereotype approach (Rich, 1979; Kobsa, 1990). Stereotypical knowledge is organized into a generalization hierarchy in which the stereotypes inherit knowledge from their ancestors. A stereotypes hierarchy is shown in Figure 2, in which each node in the hierarchy represents a stereotype regarding a certain task. Each node can also be associated with a set of attributes or assumptions that are used to justify or describe the corresponding stereotype. One stereotype subsumes another if it can be considered to be more general. For example, a hierarchical user model S_h is a substructure of the generalization hierarchy S that can be represented as follows:

$S_h \subseteq S = \{ S_i,\ S_i$ is a stereotype in the hierarchy$\}$
$S_i = \{ a_j^{(i)}, a_j^{(i)}$ is an assumption in $S_i \}$

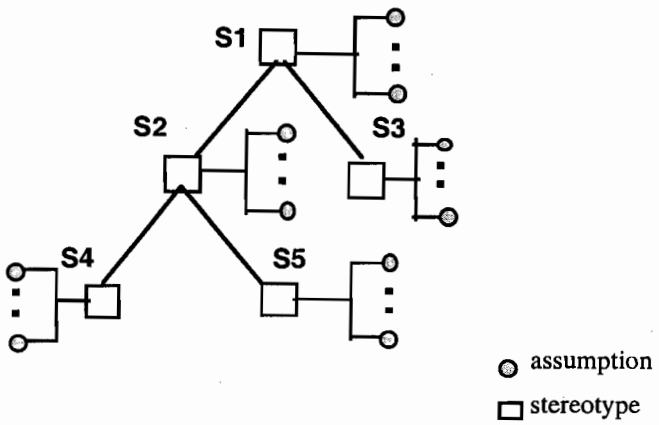
Usually, the modeling process proceeds from the root of the hierarchy. If one or more assumptions are verified by certain facts or observations during an interaction, the stereotype that contains these assumptions is activated and then added into the current user model. As the interaction progresses, more and more stereotypes are activated and the user model becomes more informative. For example, in Figure 2 if *S1* is defined as a computer user, then *S2* can be defined as microcomputer user and *S3* as mainframe user. Furthermore, *S4* may be specified as an IBM PC user and *S5* as Macintosh user respectively. If a user is modeled as a PC user, then the user's profile consists of all assumptions contained in *S1*, *S2* and *S4*.

Initial knowledge about a user comes from the system assumptions. To infer a user's characteristics, user modeling system must establish extensive assumptions. Each assumption is predefined with conditions that need to be satisfied before that assumption can be applied to the user. If a user fulfills the conditions, the assumption can be ascribed to the user. This ascription is normally contingent on an absence of conflict with other information previously known to the system.

Applications of Stereotypes

The earliest application of the stereotype approach is Rich's GRUNDY system that is used to recommend books for users (Rich, 1979). Stereotypes are used to model a user's social status such as occupation, reading preferences, sex, age, etc. At the beginning of each session, a user is

Figure 2. Hierarchical Stereotypes



asked for a self-description regarding these social traits. Stereotypes are activated by the user's input in terms of matching the user's responses with the assumptions attributed to these stereotypes. This approach may work well for modeling a user's long term, static characteristics, but it does not fit the situation in which a user does not know how to describe the task-related characteristics. For example, in a tutorial system, the user (*i.e.*, a student) may not be able to respond correctly to the request about the domain knowledge level (Wenger, 1987). Even if the user does respond, the user's judgment may not fit the system's criteria. In addition, the traditional stereotype system does not provide any insight about the task being performed (Clowes *et al.*, 1985).

KNOME, a user modeling system developed by Chin (1989) for tutoring UNIX, avoids eliciting information from a user directly. Instead, it infers the user's knowledge levels by examining the evidence that the user knows or does not know some key concepts. It classifies four groups of users with different levels of expertise and associates each group with stereotypical expectations about the user's familiarity with UNIX commands of a certain difficulty level. However, such stereotypical knowledge does change as the user's proficiency progresses. This limits the adaptability of a user model (Totterdell *et al.*, 1990).

Stereotyping is also used in some user modeling shells that provide tool kits for constructing user models for various applications. Finin's GUMS system addresses the problems of revising stereotypical knowledge about the user's beliefs (Finin, 1989). It does not derive the assumptions itself, but provides a set of services for maintenance. It accepts and stores new facts about the user. The knowledge about the user is organized in a stereotype tree where each node represents a class of users.

The user model is revised when the application system observes new facts that conflict with the active stereotypes. It deduces the negated fact from the current assumptions, informs the application system about recognized inconsistencies and answers queries concerning its current assumptions about the user's level of expertise. The revision is done by replacing the active stereotype with its closest ancestor that does not conflict with the observed facts. The stereotypes in a parallel path are considered as possible replacements. As some authors pointed out (Huang, 1991; Kobsa, 1990), since the hierarchy is constrained to a tree, and only a single stereotype may apply to a user at a time, this revision approach is inappropriate because it only decreases the stereotypical knowledge. If the initial stereotype is chosen inappropriately, additional information about the user increases the likelihood that fewer stereotypes can be attributed to the user. Therefore, better fitting stereotypes can never be found. In addition, the GUMS exploits the closed world assumption as the basis of inference; whereby, any hypothesis that is unknown is assumed to be false. This might oversimplify the real world situation.

Kobsa (1990) presents a more comprehensive shell system, BGP-MS, to support the definition of the stereotype architecture in a user model for a particular application domain. A stereotype management utility is presented to monitor the current assumptions about the user and to automatically activate and retract stereotypes according to certain conditions specified by the user model designer. It attempts to maintain a sufficient number of activated and consistent stereotypes in run time. However, it seems to leave too much obligation to the shell users (i.e., user model designers).

Forming Assumptions

In a stereotype hierarchy, all assumptions are directly or indirectly activated by the user's input. There are various techniques for transforming a user's input into. Wahlster and

Kobsa (1989) summarized two types of approaches for forming assumptions:

1) *Forming the assumptions based on the user's input.* Sometimes a user's input can be directly incorporated into a user model as long as this input directly states the information about what is required and what the user believes. For example, the system can pose questions to the users that have been devised to elicit responses that are most relevant to the preconditions of the assumptions. If the assumption cannot be made directly from a user's input, a content-dependent inference must be conducted. For example, if a user's input is a question, it usually implies several alternative assumptions. The syntactic and linguistic analysis on how this question is prompted can also underlie a set of assumptions.
2) *Forming the assumptions based on the dialogue.* The interaction between the user and the computer system also contributes new entries to a user model. All traits involved in a dialogue, such as system's navigation plan, system response, user's utterances, and feedback actions can be these entries. For example, an answered question is assumed to be understood by a user. Therefore, the user should not ask the same question in a later dialogue.

In the context of modeling a user's domain knowledge or conceptual knowledge, some "common sense" rules can be used to form assumptions directly from the user's direct input.

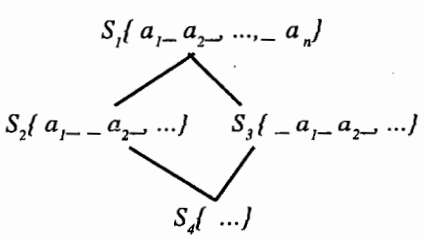
- *Transitive rule.* If a user knows that the concept C implies B, and that B implies A, then the user knows that C implies A.
- *Inheritance rule.* If a user knows that concept A has attribute description X, and B is a subtype of A, then B also has attribute description X.
- *Compass rule.* If a user knows that concept A has attribute descriptions $X_1,..., X_n$, and B has $Y_1,..., Y_m$, and that each of the X_i is a special case of some Y_j, then the user knows B implies A.

The following rules can also be useful for indirectly adding the information into the user model.

- If a user mentions a concept in a dialogue, then the user either knows the concept or does not know the concept, depending on whether or not the user requests an explanation for this concept.
- If a user mentions an attribute description of a concept, then the user believes that the concept has such an attribute.
- If a user uses new terminology and the user does not ask for clarification, then the user understands the terminology.

Very often more than one stereotype can be ascribed to a user, and a given stereotype may be a subtype of more than one other stereotype (i.e., a node has more than one parent node in the hierarchy). This feature complicates the inheritance of information. The inherited information might conflict with each other. This is shown in Figure 3, where each stereotype has some inherited information (i.e., assumptions or attributes applied to a stereotype), except S_1. The S_2 and S_3 are the subtypes of S_1. S_4 is the subtype of both S_2 and S_3. The problem in this case is that, according to the feature of generalization, S_4 may inherit conflicting assumptions (i.e., a_1 and $\ddot{y}a_1$), which is not allowed in user modeling.

Figure 3 Inheritance conflict



Conflict Resolution

Conflicts between the conclusions/predictions based on different stereotypes are difficult to solve. Rich (1979) presents several domain-dependent and domain independent resolution strategies. Domain-independent strategies suggest the selection of the following predictions instead of the conflicting stereotype predictions:

- The average value, if the predictions in a conflict are numerically valued.
- The least upper bound, namely the closest shared ancestor of two stereotypes in the generalization hierarchy.
- The prediction based on the stereotype that has been active longer since it should be better established.
- The prediction based on the more recently activated stereotype since it is more current.
- The prediction that has never been involved in any conflict.

In addition, as Ballim and Wilks (1991) suggest, an extension to inheritance hierarchy is necessary to allow for exceptions. However, the introduction of exceptions brings further problems and complexities to the inheritance reasoning. One such problem is that of ambiguity. Also, the solutions are often computationally expensive because of the time-consuming search that is required.

Ability of Learning

The learning mechanism in a user modeling system allows the user model to adapt dynamically to the user's current performance or status. This mechanism is crucial for the usability of a user modeling system. Usually the learning mechanism is enforced by revising the contents of an individual user model dynamically.

The requirement for learning stems from two realistic situations: first, it is difficult to predefine an accurate knowledge set about the system's user community; and secondly, the characteristics (*e.g.,* familiarity with the system, beliefs, goals, and preferences, etc.) about an individual user or a class of users may change over time. As the interaction progresses, the system's belief values or rules must be added, deleted or modified to reflect real time changes. These revisions may in turn cause further data generalization and specification. However, this kind of adaptation is passive because it lacks the ability of deductive inference and prediction.

Some user modeling systems utilize techniques for handling uncertainties as a learning mechanism in which the model is refined by modifying the belief values until adequate modeling performance is obtained (Rich, 1979; Brajnik *et al.,* 1990; Hoeppner *et al.*, 1986; Howard *et al.,* 1987; Simon, 1988). Off-line tuning is both time-consuming and inexact. It is often based on inadequate tests and a subjective judgment of how the interaction is progressing. The local improvement obtained by tuning one capability of the modeling process might be detrimental to other capabilities. In addition, if such a revision is not based on a view of the behavior pattern, the modeling process may face the dilemma of frequent conflict-resolution (Huang, *et al.*, 1991). In addition, when a conflict is detected, it is necessary to determine a minimal set of assumptions that is associated with this conflict. This is usually an NP-complete process (Barr and Feigenbaum, 1982).

Default Reasoning and Truth Maintenance

A user modeling system can draw a number of plausible conclusions about a user from a small base of definite knowledge. Thus, default reasoning becomes a major process in

most user modeling systems. The stereotyping approach is one form of default reasoning. In a default reasoning system if the underlying knowledge never changes, or if a reasoning system never reexams any conclusion, it is called a monotonic system. A non-monotonic reasoning system differs from the monotonic reasoning in that it is often necessary, when a conclusion is deleted from the current knowledge base, to review other conclusions whose proofs depend on the deleted one.

In a default reasoning system, assumptions may need to be either eliminated or to have new proofs that are valid based upon the current knowledge base. Deleting a single assumption may have a significant effect on an entire knowledge base because all assumptions must be deleted to maintain the consistency. In turn, these deletions can cause further deletions in the knowledge base. Thus, a change is propagated throughout the knowledge base. It is very important that a non-monotonic reasoning process not spend all its time propagating changes. Otherwise, it is not a stable system, and the credibility of any conclusion it draws is doubtful.

Default reasoning and non-monotonic logic have been originally studied by McCarthy, McDermott and Doyle, as well as Reiter. McDemott and McCarthy introduced a formal discussion about non-monotonic logic (McDermott, 1980; McCarthy, 1980). Reiter presented a survey of the areas of AI in which such reasoning is involved (Reiter, 1980), and introduced a mathematical treatment of this subject (Reiter, 1980). Reiter (1980) presents the following reasons for the necessity of a non-monotonic reasoning system:

- The presence of incomplete knowledge requires default reasoning.
- A changing external environment must be adapted by the knowledge base.
- Generating a complete solution to a problem may require temporary assumptions about potential solutions.

To able to propagate changes in the knowledge base and to check the validity of the proofs, the default reasoning needs to keep track of all results in the process. Thus, a non-monotonic system requires more storage space and processing time compared to the monotonic system.

The Truth Maintenance System (TMS) by Doyle is an implemented system that supports non-monotonic reasoning (Doyle, 1979). It serves as a truth maintenance subsystem affiliated with a reasoning system. It does not generate new inferences, but maintains consistency among the conclusions generated by the reasoning system. When an inconsistency is detected, it evokes its own reasoning mechanism, called dependency backtracking, to resolve the inconsistency or conflict by updating a minimal set of beliefs. It has the following three essential functions (Boy, 1991):

- It stores all the inferences made by the problem solver. As a result, inferences that have already been made do not need to be repeated and the contradictions, once discovered, will be avoided in the future.
- It enables the problem solver to make non-monotonic inferences. The presence of non-monotonic justification forces the TMS to use a procedure that satisfies constraints in order to determine which facts should be believed.
- It ensures that the knowledge base does not contain contradictions. Contradictions are removed by identifying the missing justifications and adding corresponding justifications. The procedure of identifying and adding justifications is called dependency-directed backtracking.

SOME LIMITATIONS OF CONVENTIONAL STEREOTYPE APPROACHES

Although the stereotype approach provides a simple way to initialize the modeling process and is successful in some applications, this approach limits the representational power of a user model in the following three aspects:

1) Reasoning under Uncertainties. Since reasoning is conducted with extensive default assumptions that may conflict with the new evidence obtained as the interaction progresses, the revision of stereotypical knowledge is necessary to handle the inconsistencies. This in turn can cause dependency-directed backtracking that is a non-monotonic process (de Kleer, 1986). This process is called truth maintenance and has been a fertile field for AI research. Since the conventional truth maintenance approach examines one piece of evidence at a time, it is often ineffective or even impossible to detect noisy or inconsistent inputs that should be ignored. This phenomenon, for instance, may be observed in a tutoring system where the student's input often carries some inconsistencies (*e.g.*, typing mistake, grammar mistake, or the way the student raises the question) that may have nothing to do with understanding the tutorial concepts and, therefore, should be ignored in the modeling process.

In addition, the stereotyping approach must be able to examine the continuity and overall user behavior pattern in task performance. Otherwise, the modeling process may not reflect the real world situation, and it is possible that current efforts of maintaining consistency may bring further conflicts in the subsequent interaction. Thus, model construction may fall into a dilemma where a non-monotonic process of conflict-resolution is frequently involved, and eventually no modeling decisions can be made after a period of interaction (Chen and Norcio, 1997).

2) Individualization. A generalization hierarchy also provides a simple way to classify a user into certain stereotypes. The user model must capture the individual differences in order to tailor the system response to fit a particular user. Therefore, very often this classification must be as subtle as possible. But the predefined hierarchical structure is sometimes too inexact to characterize a particular user. This limits the degree of individualization in the sense that all assumptions or attributes are confined within a stereotype. They must apply to a certain stereotype at all times. However, it is very possible that a user fits one stereotype, but may not fit some attributes defined in that stereotype. The generalization hierarchy lacks the ability to justify such a situation. In addition, it is obvious that the representation power depends on the number of stereotypes rather than the number of assumptions.

3) Feature identification. Since the stereotype hierarchy is logically organized, it may not be appropriate for dynamically extracting a user's short-term characteristics that change over time and temporally exhibit great varieties regarding the current task. For instance, it is very realistic that a user might demonstrate both expert and novice traits for the same task. In that case, the hierarchical classification of characteristics via few dimensions might not be able to provide consistent information for the system to adapt the current interaction. It may fail to recognize the *irrational* data about a user's behavior. Whether such data need to be taken into account should depend on the examination of overall behavior patterns.

Since the predefined attributes or assumptions are confined within each stereotype and can be only inherited by the descendant stereotypes, there is no effective way to update those attributes that are no longer significant in the context of task performance. In addition, most stereotyping approaches solve conflicts by simply replacing the active stereotypes with their

ancestors in the hierarchy (Rich, 1979; Huang *et al.*, 1991). Thus, some assumptions that are still consistent to the current situation are lost. It is possible that a user may fail to fit any set of stereotypes, so that the hierarchical modeling process fails to associate any system decision to that user. In such a situation, some assumptions distributed among the stereotypes might be still useful for characterizing that particular user. Hierarchical approaches lack the ability to identify and reorganize the useful assumptions distributed among the stereotypes.

ARTIFICIAL NEURAL NETWORKS IN ADAPTIVE HUMAN-COMPUTER INTERFACE

It has been recognized that the ability of an Artificial Neural Network (ANN) to learn and recognize patterns in data can be utilized in the area of human-computer interaction (HCI). Unlike classical AI techniques, neural networks have the ability to adapt and to generalize. Many different types of HCI activities can benefit from these characteristics of neural nets (Maren, 1990; Stacey *et al.*, 1992; Chen and Norcio, 1997). The primary concern for utilizing ANN techniques can be summarized as follows:

1) Learning techniques. ANN models provide a family of numerical learning techniques. They are particularly useful in the applications of data-intensive pattern recognition. They can extract the rules from the examples. The feature of self-organization allows dynamic modeling of the run-time environment. ANN models have also shown the potential for the development of knowledge-based system in which the knowledge acquisition is often a bottleneck (Hayes-Roth, 1991). In human computer interaction, dynamic learning is particularly important for modeling user performance. The historical log of interaction can provide training samples for the classification of a user's behavior (Stacey *et al.* 1992).

2) Pattern completion and noise tolerance. The ANN technique is also useful for processing incomplete information. Successful matches between the input and output pattern can be accomplished with incomplete or imperfect patterns, even when there is noisy or ambiguous input. The performance is said to degrade gracefully. Similarly, when presented with unfamiliar data that are within the range of its exemplars, the network will generally produce an output that is a reasonable interpolation between the exemplar outputs. This feature also facilitates uncertainty management in the sense that it examines the current knowledge base in pattern-formatted view so that incomplete and imprecise information can be matched to the closest output pattern (Pao, 1989).

3) Efficient *system development and maintenance.* For some applications, ANN techniques provide an efficient development alternative. Building and adjusting an ANN model are relatively easier than building a large number of production rules (Fu, 1990). In the applications of data-intensive pattern recognition and classification, it is especially difficult to define a set of rules that covers all evidence-conclusion situations in pattern recognition. In addition, the parallel processing model makes the computation cost-effective. It avoids the overhead of required maintenance in conventional rule-based systems (Carpenter, 1989; Pao, 1989).

ANN and Rule-Based Systems: Comparison and Integration

The ANN approaches and rule-based systems are different in several aspects. The knowledge of ANN is implemented by its connections and associated weights; whereas, the

Table 1. A comparison between the ANN-based system and the rule-based system

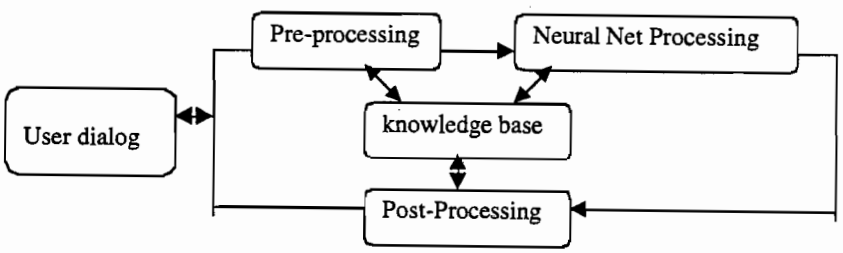
	ANN-based system	***rule-based system***
knowledge representation	Connections and weights	production rules
computing unit	Numbers	symbols, belief values
information level	Signal level	knowledge level
adaptation	Good	poor
development cost	Low	high
explanation	None	good
pattern recognition ability	Good	poor
processing mode	Parallel	sequential
inference mode	Summation and thresholding propagation of activation	propagation of belief values, unification and resolution

knowledge of rule-based system lies in rules. The ANN processes information by numerically propagating and combining activations through the networks, but a rule-based system processes information through symbol matching and generation. Table 1 shows the primary differences between the two methods.

Since the interface systems must provide the dialogue in the symbolic form, any result from neural nets must be transformed, which needs the rule based processing. On other side, the ANN can facilitate the conventional rule-based systems in the aspects of adaptation, pattern recognition, and classification. It has been suggested that combining these two systems can enhance both of them (Caudill, 1990). Figure 4 illustrates an interface framework that integrates the two systems.

The neural network module functions as a component that analyzes the user's input and provides the intermediate results for generating system response and explanation. Both the pre-processing module and the post-processing module are the rule-based subsystems. The pre-processing component is used to transform the user's input into numerical stimuli for further neural-net processing. The output from the neural network activates system operations and needs to be symbolically interpreted. This is achieved by another rule-based module, the post-processing component. The system's knowledge base is a working memory that contains all supporting information, such as rules of transformation, meta-knowledge, and user or task profiles, usually in symbolic form. For example, if this

Figure 4. An integration of neural networks and rule-based system



framework is implemented in an information retrieval environment, then the preprocessing component translates a user's query (e.g., subjects) into network input, and the network will propagate the activation. The output is a vector that corresponds to some subjects or documents. The post-processing component then searches the database according to the analysis on this vector. It also generates explanations if the system response to a user's query needs to be justified.

There are several strategies for combining neural networks with expert systems or other systems. The strategies for the integration vary from stand-alone systems to a fully integrated neural net and expert system (Caudill, 1990; Bailey and Thompson, 1990; Hillman, 1990; Bailey and Medsker, 1991; Boy, 1991). Bailey and Medsker (1991) classify these strategies in terms of structural topology: stand alone structure; transformation structure where two systems are linked one after the other, and the successor uses the output of the predecessor; loose coupling structure where the systems have dependencies such as feedback from one system to the other or both systems feed into a third; tight coupling structure; and full integration structure.

Applications of ANN in HCI

Artificial Neural Network (ANN) techniques that specialize in human computer interaction have been proposed and developed by a number of investigators. The earlier application of ANN in HCI is NETtalk, a system developed by Sejnowski and Rosenberg (1987), which uses three-layer back-propagation ANN to learn to "read aloud." The goal was to have an ANN convert English text to speech by learning through exposure to examples as opposed to entering phonological rules and handling exceptions with a look-up table. A back-propagation net was trained to transform an input vector that encodes a sequence of letters into an output vector that encodes the phonemic features of the sequence and which can be used to drive a speech synthesizer.

Maren (1990) summarizes three areas where ANN can be utilized to improve a system's ability to adapt in HCI: information control, diagnosis, and dialog monitoring. She suggests that neural networks can enhance the learning capability of adaptive interfaces in terms of sensing the extensive data about the interaction through various channels.

ANN in user modeling is also proposed for identifying user's domain knowledge (Chen and Norcio, 1997). The data about a user's behavior can be formed as stimuli for classification conducted by neural networks. The clues from the task performance can also be matched to a complete task path for system's navigation. Stacey *et al.* (1992) present an ANN paradigm for pattern analysis of the trace data generated by human-computer interaction. It is suggested that since a user interacting with a system follows patterns within a certain context, ANN techniques can be used to observe and predict the behavior pattern. Eberts (1991) proposed to use a back-propagation network to elicit knowledge about a user's proficiency of the UNIX command system for the design of help subsystems. Finley and Beale (1992) suggest modeling individual differences and habits of users as a means of providing support for common activities. They use a minimalist approach to the problem, and view user modeling as the composition of a number of stereotypical classifications. They propose the use of example-based classifiers as user modeling components.

ANN can provide an alternative intelligent interface for an information retrieval system. Heger and Koen (1991) developed an associative memory for database retrieval in which the causal relationships among key terms in a database are established and weighted. Given a user's initial query, the interface can activate all related key terms and induce the

new queries. McGrew (1992) proposes the use of a neural network in the rapid validation and prototyping of task analysis in interface design. By mapping the task analysis onto a network, one can simulate the result of the analysis and quickly identify any errors. This is particularly useful for usability studies. Eichmann and Srinivas (1992) propose the use of an unsupervised network based on Adaptive Resonance Theory (ART) to perform retrieval software modules from a repository. It is done by partial matching and exploiting the generalization properties of the network. Habibi and Eberts (1992) also use neural networks to develop a filter to select relevant articles based on keywords of interests. Oakes and Taylor (1992) present a useful comparison of a number of retrieval techniques, ranging from string matching through the neural network methods. Their application is to classify terms in a medical thesaurus in which the terms may be related literally or conceptually.

There are also many studies of using neural network technology to design video/audio equipment for HCI. For instance, Bryn and Eichmann (1997) present a design and experiment of neural network approach to tracking eye position. The results show that accurate tracking of a human's eye position is possible by neural network based image processing built in the goggle-mounted miniature charge-coupled device.

All these studies demonstrate benefits of various useful properties of neural networks, which open up new research and applications in the field of HCI.

An Alternative Approach to User Modeling

Chen and Norcio (1997) proposed and tested an alternative to conventional stereotyping approaches. A set of associative networks is utilized as the knowledge base for modeling users' domain knowledge. In contrast to the conventional rule-based user modeling approaches, this study uses artificial neural network (ANN) techniques to store, maintain, and infer various task-related attributes about users. The proposed approach is referred to as associative user modeling. It is suggested that this approach provides an effective and simple way of user modeling in terms of the inherent features of ANN techniques.

In associative user modeling, all characteristics or assumptions about a user are referred to as concepts. These concepts are represented by the network nodes. There are two perspectives for defining the connections among the nodes (*i.e.*, concepts):

- The connections are viewed as the causal relationships among the concepts, which are conditionally weighted.
- The connections are obtained by supervised or unsupervised network training.

In associative user modeling, there is no need to group assumptions *a priori* for predefined user profiles. User modeling is a process that extracts the concepts and their weighted connections from associative networks to form a unique profile that fits a particular user or task. This process is conducted by propagating the activation level throughout the network. Since the propagation process allows any number of concepts to be activated, more combinations of concepts can be utilized to characterize individual users. This underlies a more flexible way of constructing of a user model.

Associative user modeling provides a pattern-formatted view of a user's characteristics. It examines the information about users through pattern recognition. The default reasoning is viewed as a process of pattern completion. The produced user models can be applied to predicting users' performance and evaluating users' domain expertise. The system equipped with such models can optimize its performance by making the dialogue strategy adapt to users' skills and tasks.

1) The ability of default reasoning. ANN's abilities of pattern completion and noise tolerance allow production of default assumptions based on incomplete, imprecise

and inconsistent input. This means that based on a small number of inputs the system can produce a large number of outputs. Also, the inconsistent information about the user can be reconciled through pattern recognition. This approach avoids the complexity of truth maintenance in default reasoning that is required in conventional stereotype approaches.

2) *The ability of learning.* ANN models can learn from the exemplars (*i.e.*, training data). Using supervised or unsupervised training procedures, the ANN models can generalize the hypothetical information about user's characteristics such as the beliefs about user's domain knowledge and user classifications. The learning ability of ANN models can enhance the adaptability of user modeling that requires dynamic revision of its knowledge base.
3) *The system implementation and maintenance.* It is widely recognized that the implementation and maintenance of neural network models are much simpler than that of rule-based systems. Conventional rule-based systems require extensive work on knowledge elicitation to set up the appropriate rule bases, which tends to be a time-consuming and error-prone process. In contrast, the learning ability of a neural network eases the knowledge elicitation process. The consistency can be easily maintained in neural network systems due to their ability to handle inconsistent or incomplete information. In addition, the useful features of conventional stereotyping approaches, such as inheritance and fast classification, can be easily implemented by ANN-based pattern association process.

SUMMARY

This chapter presents the importance of user modeling techniques for improving the adaptability of human-computer interfaces. User models have become important components of adaptive human-computer interfaces. A system will be able to exhibit flexible user-oriented behavior if it has access to a model that consists of assumptions about the user's characteristics in performing certain tasks. These characteristics are usually task-related, such as the user's plans, goals, background knowledge, and cognitive preferences. There are different taxonomies about these characteristics depending on the time period while they hold, the way by which they are elicited and represented, as well as the degree to which they are specified.

It has been a common practice to use a set of stereotypes to initialize a system's beliefs about its users and the tasks being performed. The predefined stereotypical knowledge is organized into a generalization hierarchy. The modeling process proceeds with a stereotype assignment using default reasoning. This chapter discussed the issues of knowledge elicitation and representations involved in the modeling process. The limitations of the stereotyping approach in generalization, individualization, and default reasoning were presented. The uncertainties involved in the user modeling were identified. These uncertainties primarily originate from the incomplete and inconsistent information about users. Most user modeling systems use the *ad hoc* approach to handle the uncertain information, which may fail to generalize, and weakens the theoretical foundation of uncertainty management considerably.

In addtion, the study of user cognitive characteristics (*e.g.*, performance models or mental models) is also very important to user modeling. The performance models emphasize the decomposition of tasks in a generic way, and mental models are concerned about individual differences. These models may supplement each other to facilitate the understanding of human-computer interactions.

The ANN techniques have been used in the field of human computer interface design. The ANN techniques demonstrate strong ability of learning, pattern completion, and handling inconsistent information, which is required in the design of adaptive interface systems. They can also be used in user modeling. In addition, the ANN-based modeling systems are easier to develop and maintain compared with the traditional rule based systems, especially for data-intensive pattern recognition and classification.

REFERENCES

Allen, J. F. and Perrault, C. R., "Participating in Dialogue Understanding via Plan Deduction," *Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence,* 1978.

Allen, R. B. " User Models: Theory, Methods, and Practice," *International Journal of Man-Machine Studies,* Vol.32, 1990, 511-543.

Bailey, D. L. and Medsker, L. R., "Integrating Expert Systems and Neural Networks: Tutorial mp3." in *the Ninth National Conference on Artificial Intelligence (*AAAI-91), July 1991.

Bailey, D. L. and Thompson, D. M. "Developing Neural-Network Application," *AI Expert*, November 1990.

Ballim, A. and Wilks, Y. "Beliefs, Stereotypes and Dynamic Agent Modeling," *User Modeling and User-Adapted Interaction,* Vol. 1, No.1, 1991, 33-56.

Barr, A. and Feigenbaum, E. A., *The Handbook of Artificial Intelligence* 2. Los Altos, Kaufmann, 1982.

Bonissone, P. and Tong, R. M. "Editorial: Reasoning with Uncertainty in Expert Systems," *International Journal of Man-Machine Studies*, 1985, Vol. 30, 69-111.

Boy, G. A. *Intelligent Assistant Systems*, Academic Press, 1991.

Brajnik, G., Guida, G., and Tasso, C. "User Modeling in Expert Man-Machine Interfaces: A Case Study in Intelligent Information Retrieval," *IEEE Transactions on Systems, Man, and Cybernetics*, 1990, Vol. 20, No. 1, 166-185.

Bruce, B. C., Belief System and Language Understanding, *Report No. 2973*, Bolt, Beranek Newman (Ed.) Cambridge, MA, 1975.

Bryn, W. and D. Eichmann, "A Neural Network Approach to Tracking Eye Position," *International Journal of Human-Computer Interaction*, 1997, Vol. 9, No. 1, 59-79.

Buchanan, B. and Smith, R. G., Fundamentals of Expert Systems, *Ann. Rev., Computer Science*, 1988, Vol. 3, 23-58.

Card, S. K., Moran, T. P. and Newell, A., *The Psychology of Human-Computer Interaction,* Lawrence Erlbaum Associates, 1983.

Carpenter, G. A. and Grossberg, S., "A Massively Parallel Architecture for Self-Organizing Neural Pattern Recognition Machine," Computer *Vision, Graphics and Image Processing*, Vol. 37,1987, 54-115.

Carpenter, G. A. " Neural Network Models for Pattern Recognition and Associative Memory," Neural Networks, Vol. 2, 1989, 243-257.

Caudill, M. "Using Neural Nets: Fuzzy Cognitive Maps," *AI Expert*, 1990, 49-53.

Chen, Q. and A. F. Norcio, "Modeling a User's Domain Knowledge with Neural Networks", *International Journal of Human-Computer Interaction*, Vol. 9, 1997, No. 1, 25-40.

Chin, D. N., "Knome: Modeling What the User Knows in UC," *User Models in Dialogue Systems,* A. Kobsa and W. Wahlster (Ed), Springer-Verlag, Berlin, 1989.

Clowes, I., Cole, I., Arshad, F., Hopkins, C., and Hockley, A., "User Modeling Techniques

for Interactive Systems," People *and Computers: Designing the Interface*, Ed. by Johnson P. and Cook, S., Cambridge University Press, 1985, 35-45.

Cohen, P. R. "On Knowing What to Say: Planning Speech Acts," Ph. D. Thesis, Computer Science Department, University of Toronto, 1978.

Daniel, P. J. "Cognitive Models in Information Retrieval - An Evaluative Review," *Journal of Documentation,* Vol. 42, No. 4, 1986, 272-304.

DARPA *Neural Network Study*, AFCEA International Press, 1988.

Dede, C. "A Review and Synthesis of Recent Research in Intelligent Computer-Assisted Instruction," International *Journal of Man-Machine Studies*, Vol. 24, 1986, 329-353.

de Kleer, J. "An Assumption-Based TMS," *Artificial Intelligence*, Vol. 28 No. 2, 1989, 127-162.

Doyle, J. "A Truth Maintenance System," *AI,* Vol. 12, 1979, 231-272.

Eberts R. "Knowledge Acquisition Using Neural Networks for Intelligent Interface Design," *Proceedings of 1991 IEEE International Conference on Systems, Man, and Cybernetics*, 1331-1335.

Eberts, R. E. and Eberts, C. G. "Four Approaches to Human-Computer Interaction," Intelligent Interfaces: Theory, Research and Design, P. A. Hancock and M. H. Chignell (ed.), Elsevier Science, NY, 1989.

Eichmann, D. and Srinivas, K. "Neural Network-Based Retrieval from Software Reuse Repositories," *Neural Networks and Pattern Recognition in Human-Computer Interaction*, Beale, R. and Finlay, J. (Ed.), Ellis Horwood Publisher, 1992, 216-228.

Finin, T. W., "GUMS: A general user modeling shell," in *User Models in Dialogue Systems*, A. Kobsa and W. Wahlster (Ed.), Springer-Verlag, Berlin, 1989.

Finlay, J. and Beale, R. "Pattern Recognition and Classification in Dynamic and Static User Modeling," *Neural Networks and Pattern Recognition in Human-Computer Interaction.* R. Beal and J. Finlay (Eds.), Ellis Horwood Pub. 1992, 65-89.

Fu, L., "Integration of neural heuristics into knowledge-based inference," *Connection Science*, 1990, Vol. 1, No. 3.

Habibi, S. and Eberts, R. "Using Neural Networks to Route Messages," *Neural Networks and Pattern Recognition in Human-Computer Interaction*, Beale, R. and Finlay, J. (Ed.), Ellis Horwood Pub., 1992, 229-239.

Hayes-Roth, F, "Making Intelligent Systems Adaptive," in *Architecture for Intelligence*, K. Vanlehn (ed.), Lawrence Erbaum Associates, Hillsdale, NJ, pp. 301-321, 1991.

Heger, A. S. and Koen, B. V. "KNOWBOT: An Adaptive Data Base Interface," *Nuclear Science and Engineering*, Vol. 107, pp. 142-157, 1991.

Hillman, D. V., "Integrating Neural Networks and Expert Systems," *AI Expert*, June 1990, 54-59.

Huang, X., Mccalla, G. I., Greer, J. E. and Neufeld, E., "Revising deductive knowledge and stereotype knowledge in a student model," *User Modeling and User-Adapted Interaction,* 1991, Vol. 1, No. 1.

Hoeppner, W., Morik, K. and Marburgber, H., "Talking It Over: The Natural Language Dialog System HAM_ANS," *Cooperative Interfaces to Information Systems*, L. Bolc and M. Jarke (Ed.), Berlin, Springer-Verlag, 1986, 189-258.

Howard. S. and Murray, D., "A Taxonomy of Evaluation Techniques for HCI," *Proceedings of Human Computer Interaction (INTERACT'87)*, 1987, 453-459.

Kass, R. "Student Modeling in Intelligent Tutoring Systems: Implications for User Modeling," in *User Models in Dialogue Systems,* A. Kobsa and W. Wahlster, (Ed.), Springer-

Verlag, Berlin, 1989.
Kobsa, A. "Modeling the User's Conceptual Knowledge in BGP-MS, A User Modeling Shell System," *Computational Intelligence,* Vol. 6, No. 4, 1990.
Kramer, J., Noronha, S. and Vergo, J. "A User-Centered Design Approach to Personalization," *Communications of ACM*, Vol 43, No. 8, 2000.
Maren, A. J., "Neural Network for Enhanced Human-Computer Interaction," *IEEE Control Systems*, August, 1991
McCarthy, J., "Circumscription - A Form of Non-Monotonic Reasoning," *AI*, Vol. 13, 1980, pp. 27-39.
McDermott, D. and Doyle, J., "Non-monotonic Logic," *AI* Vol. 13, 1980, 41-72.
MaGrew, J., "Task Analysis, Neural Nets, and Very Rapid Prototyping," *Neural Networks and Pattern Recognition in Human-Computer Interaction*, Beale, R. and Finlay, J. (Ed.), Ellis Horwood Pub., 1992.
Minskey, M. L, "A Framework of Representing Knowledge," *The Psychology of Computer Vision*, P. H. Winston (ed.) NY: McGraw-Hill, 1975.
Murray, D., "A Survey of User Cognitive Modeling," *Report DITC 92/87,* National Physics Laboratory, Teddington, England, 1987.
Norcio, A. F. and Stanley, J., "Adaptive Human-computer Interfaces, A Literature Survey and Perspective," *IEEE Transactions on System, Man and Cybernetics*, Vol. 19, No. 2, 1989, 399-408.
Nwana, H. S., "User Modeling and User Adapted Interaction in an Intelligent Tutoring System," *User Modeling and User-Adapted Interaction,* Vol. 1, No. 1, 1991, 1-32.
Oakes, M.P. and Taylor, M. J. "Clustering of Thesaurus Terms Using Adaptive Resonance Theory, Fuzzy Cognitive Maps and Approximate String-Matching Techniques," *Neural Networks and Pattern Recognition in Human-Computer Interaction*, Beale, R. and Finlay, J. (Ed.), Ellis Horwood Pub.1992, 243-263.
Pao, Y., *Adaptive Pattern Recognition and Neural Networks*, Addision-Wesley Publishing Co., 1989.
Paris, C. L. "The Use of Explicit User Models in a Generation System for Tailoring Answers to the User's Level of Expertise," in *User Models in Dialogue Systems,* A. Kobsa and W. Wahlster, (Ed.), Springer-Verlag, Berlin, 1989, 200-232.
Reiter, R. "A Logic for Default Reasoning," *Artificial Intelligence*, Vol. 13, 1980, 81-132.
Rich, E., "User Modeling via Stereotypes," *Cognitive Sciences,* Vol. 3, 1979, 329-354.
Schank, R. *Scripts, Plans, Goals and Understanding.* Hillsdale, NJ: Lawrence Erlbaum, 1977.
Sejnowski, T. J. and Rosenberg, C. R. "Parallel Networks that Learn to Pronounce English Text," *Complex Systems,* Vol. 1, 1987, 145-168.
Self, J. A., "Student Models in Computer-Aided Instruction," *International Journal of Man-Machine Studies,* Vol. 6, 1974, 261-276.
Selker, T. "Coach: A Teaching Agent that Learns," *Communication of ACM*, Vol. 37, No.7, 1994, 92- 99.
Simon, T. "Analyzing the Scope of Cognitive Models in Human-Computer Interaction: A Trade-Off Approaches," *People and Computer IV*, D. M. Jones and R. Winder Ed., Cambridge Univ. Press 1988, 79-93.
Stacey, D. Calvert, D. and Carey T., "Artificial Neural Networks for Analyzing User Interactions," *Neural Networks and Pattern Recognition in Human-Computer Interaction*, Beale, R. and Finlay, J. (Ed.), Ellis Horwood Pub., 1992.

Wahlster, W. and Kobsa, A. "User Models in Dialogue Systems," in *User Models in Dialogue Systems,* A. Kobsa and W. Wahlster, (Ed.) , Springer-Verlag, Berlin, 1989.

Wenger, E. *Artificial Intelligence and Tutoring Systems*, Morgan Kaufmann Publishers, Los Altos, CA. 1987.

Zadeh, L. A. "Knowledge Representation in Fuzzy Logic," *IEEE Transactions on Knowledge and Data Engineering,* Vol. 1, No. 1, 1989, 89-100.