Cooperative Interactions in Fuzzy Decision Support Systems

Koichi YAMADA
Advanced Technology Center, Yamatake-Honeywell Co., Ltd.
4-28-1 Nishirokugo Ohta-ku, Tokyo 144, JAPAN
E-mail: yamada@atc.yamatake.co.jp

INTRODUCTION

Human-computer interaction (HCI), or the human interface, is one of the most vigorous research topics in computer science and the related areas. However, there are many conflicting images of HCI research, and sometimes the field seems to be chaotic. Whatever the viewpoint, the final goal is a common one: realization of easy, intelligent and friendly systems -- systems with cooperative interactions.

There are two major streams of research that deal directly with cooperative interactions. One is cooperative answering in information retrieval[1]. However, most research in the field is dependent on a priori domain knowledge, or on heuristics in the domain. The other is an approach that deals with the user's intentions in more general dialogues[2][3]. They are, however, too precise for use in practical systems.

In this paper, we introduce some alternative approaches employing fuzzy logic which we have investigated at the Laboratory for International Fuzzy Engineering Research (LIFE). First, we demonstrate an approach for cooperative interactions in information retrieval systems which requires no a priori knowledge or heuristics dependent on the domain. Then, we discuss another approach that utilizes the user's preferences.

COOPERATIVE ANSWERS TO QUERIES [4]

Information retrieval (IR) is one of the most basic decision-making tasks. However, the task is not so simple for most decision makers, who often have to access unfamiliar databases using unfamiliar IR systems. In such cases, queries frequently fail (retrieve no data) or retrieve too many data to examine all of them. Then, they have to construct a new query to retrieve a manageable number of data. But there is still no guarantee that this next trial will be successful.

Cooperative answering is one technique to support such unhappy users. Cooperative answers are those that give the user some helpful hints for construction of the next query. These are given when 1) the query fails, or when 2) too many data are retrieved. Unlike approaches proposed so far, our approach generates cooperative answers without domain dependent heuristics and knowledge given a priori. Instead, it creates knowledge about the distribution of data in the database for itself, and utilizes it. The created knowledge is called a macro database, because it describes the contents of the database globally.
**Generation of the macro database**

The macro database is a set of linguistic expressions each of which describes a fuzzy cluster of data in the data space. It is generated by the following algorithm:

1. Express nominal attributes by numerical ones; e.g. in the case of an apartment database, the attribute "floor type" (see Fig. 2) is expressed by its average floor space, and "the nearest station" by a pair of map coordinates.

2. Define linguistic labels by membership functions on these numerical attributes (See Fig. 1). Each linguistic label expresses a certain concept on the attribute. These membership functions must cover the entire universe of discourse of the attributes.

3. Apply a fuzzy clustering method called Fuzzy C-means to data in the database, and create fuzzy clusters from the data. In the first trial, set the number of clusters to two.

4. Try to express the generated fuzzy clusters by the linguistic labels defined before. If the fuzzy clusters are expressed well by the linguistic labels -- that is, their projections to attribute axes are almost included in membership functions defined on the attributes, go to the next step. Otherwise, increase the number of cluster by one, and go to (3).

5. Express all fuzzy clusters by the combinations of linguistic labels. Then, each cluster is a factor of the macro database.

![Diagram](image-url)

**Fig.1** Fuzzy clusters and their linguistic expressions

**Cooperative Answering when a query fails**

The user's query is given in a conjunction of fuzzy conditions expressed in fuzzy sets on attributes. If the matching degree of data to the query is greater than a certain value, that data is retrieved from the database. If there is no such data, the query fails.

When a query fails, the system gives the user the following information in order; 1) the nearest cluster to the query, as a set of data that the user might accept, 2) the nearest data to the query as an alternative, then 3) compromises that the user must accept if not satisfied with the alternative (Fig.2). The linguistic expressions of the macro database described before are used to provide the information.
Cooperative answering when too much data are retrieved

Queries sometimes return too many data from the database for the user to examine all of the data precisely. In that case, the user might want to refine the retrieved data set by giving additional conditions. However, this is not so easy, because there is no extra information to shrink it to a proper size. So, the system gives the user the linguistic expression of the retrieved data set and the number of data, then, after checking the distribution of the retrieved data set for each attribute, it urges the user to provide an extra condition for the attributes for which the distribution is the largest.

DECISION SUPPORT WITH A USER MODEL

In consulting systems such as apartment-hunting[5] and book-selection[6] advisors, the user often does not have any clear prior image about his/her decision. What the system must propose in these applications is "a favorable decision" rather than "the best one". A user model expressing the user's preferences is effective for this purpose. Preferences, for example, can be used as additional conditions for queries in data retrieval systems, when too many data are retrieved, and can also be used as information for derivation of fuzzy goals for interactive fuzzy programming[7], which is a fuzzy version of multi-objective programming.

User modeling in decision support systems

In general, there are two ways to build a user model. One is explicit modeling, which requires the user to inform the system of his/her preferences. The other is implicit modeling, where the system guesses the user's preferences during interactions. For the aim of cooperative interactions, implicit modeling is preferable.
Stereotyping[6] is one of the major approaches for implicit modeling. A stereotype is knowledge that represents typical traits of a typical user. It also has its activating conditions, which are usually based on the basic profile of the user, such as occupation, gender, age, etc. Using stereotypes as knowledge, the system can guess many aspects of the user's preferences in the first stage of interactions.

However, stereotyping has inherent shortcomings -- the initially obtained user model usually includes some wrong guesses, so, the system always has to monitor the interactions, infer the user's preferences from clues in interactions and correct the wrong guesses.

The user's preferences, however, are not easy to infer from the interactions, because the user talks about different levels of preferences. Sometimes he/she may say directly: "I prefer a cheaper apartment". In other cases, the user may mention a higher level of preference such as "I want safety", which lets the system deduce that he/she wants an apartment with a caretaker and/or a remote lock system. Furthermore, when the user talks about concrete requirements such as the desire for an apartment with tiled walls, it is possible to hypothesize that he/she prefers a "good-looking" apartment. From the hypothesis, you can also assume that the user prefers a new apartment.

Though the system which we are implementing at LIFE employs only the direct and the deductive approaches, we discuss the third approach of hypothesizing, which involves the inferencing process called abduction.

**Fuzzy abduction**

Abduction is the inferencing process by which to derive a set of hypotheses that explains a given set of events with a set of rules. A derived set of hypotheses is called an explanation. As a simple example of abduction, suppose that you love to go to the races, and that a friend says you must be a gambler. In this case, probably, his inference is NOT a deduction, because he does not seem to use a rule saying "you are a gambler, because you like to go to the races". A more probable rule is "You like to go to the races, because you are a gambler". So, what he did was finding a hypothesis that explains the fact that you like to go to the races. That is an abduction.

Fuzzy abduction[8] is defined as follows; When a set of rules R, where each rule 
"Rij: P_i -> Q_j" has a truth value r_{ij} in [0,1], and a fuzzy set \( Q \) on \( Q \) (the set of \( Q \) s) are given, fuzzy abduction is the procedure by which to obtain a fuzzy set (fuzzy explanation) \( P \) on \( P \) (the set of \( P \) s) that derives \( Q \) with \( R \). Here, "\( P \) derives \( Q \)" means that the following equation holds:

\[
q_{ij} = \max_{i, p_i+r_{ij} \geq 1} (p_i + r_{ij} - 1), \quad (1)
\]
where \( p_i \) and \( q_j \) are truth values of \( P_i \) and \( Q_j \), and are also membership values of \( P_i \) in \( P \) and \( Q_j \) in \( Q \), respectively.

Theoretically, there is no guarantee that fuzzy explanations always exist. However, if any fuzzy explanations do exist, there are, in general, only the largest and multiple minimal fuzzy explanations. The largest fuzzy explanation, \( P_{\text{max}} \) is given by the following equations:

\[
P_{\text{max}} = \hat{\phi} \left( \frac{p_{ij}}{P_i} \right),
\]

\[
p_{ij} = \begin{cases} 
q_j - r_{ij} + 1 & \text{if } r_{ij} \geq q_j, \\
1.0 & \text{otherwise},
\end{cases}
\]

where \( \hat{\phi} \) is an operator defined as:

\[
a/A \hat{b}/B = \begin{cases} 
\frac{a}{A} + \frac{b}{B} & \text{if } A \neq B \\
\min(a, b) / A & \text{if } A = B
\end{cases}
\]

Then, \( P_{k_{\text{min}}} \) \((k=1, \ldots, N_{\text{min}})\), given in the next equation, is a minimal fuzzy explanation, if it is a subset of \( P_{\text{max}} \) and does not include other \( P_{k'_{\text{min}}} \) \((k' \neq k)\).

\[
P_{k_{\text{min}}} = \sum_{j, q_j \neq 0} \left( \Delta \frac{(q_j - r_{ij} + 1 / P_i)}{i \in \phi(j)} \right),
\]

where \( \phi(j) \) means a set of "i"s such that \( r_{ij} - q_j \geq 0 \) for the given \( j \), and \( \Delta j \) is an operator which chooses a term from amongst those with different "i"s.

Reasoning preferences by fuzzy abduction

Suppose that there is a real estate agent who has some knowledge about the tendencies of apartment hunters. For example, customers who want good-looking apartments always require new ones with tiled walls, and tend to want remote lock systems as shown in R1, R2 and R3 in Fig.3 (a). In these rules, preferences in the antecedents are customers' basic inclinations, or higher-level preferences. Those in the consequents are their concrete requirements, or lower-level preferences.

Now, suppose again that a customer has come to the agency, and that the agent understands him/her to want an apartment shown in Fig.3 (b), after some discussion with him. Then, the agent can apply fuzzy abduction to infer the customer's higher-level preferences. Since the best explanation of the given facts is usually given in minimal explanations[8], we use eq. (5) to obtain a solution. In this case, only a minimal fuzzy explanation is obtained, as shown in Fig.3 (c).
CONCLUSIONS

A few approaches related to cooperative interactions in fuzzy decision support systems are proposed. In these approaches, fuzzy logic plays important roles in dealing with the fuzziness which is essential in macro expression of a database, and in users' preferences.

![Fig. 3 Inferencing higher-level of preferences by fuzzy abduction](image)

REFERENCES


