FUZZY DATABASE SYSTEMS

Motohide Umano*
Masaharu Mizumoto**
Kokichi Tanaka***

* Department of Applied Mathematics
Faculty of Science
Okayama University of Science
Ridai-cho, Okayama 700
Japan

** Department of Management Engineering
Faculty of Engineering
Osaka Electro-Communication University
Neyagawa, Osaka 572
Japan

*** Department of Information
and Computer Sciences
Faculty of Engineering Science
Osaka University
Toyonaka, Osaka 560
Japan

ABSTRACT

An extended relational model for fuzzy databases and its manipulation language are described. This model is an extended version of Codd's relational model of data and allows fuzzy sets and NULL as attribute values for representing ambiguous data. The interpretation for such ambiguous data is based on the concept of possibility distribution proposed recently by L.A.Zadeh.

The data manipulation language provides QUERY, INSERT, DELETE, DEFR (DEfine Fuzzy Relations) and DEFP (DEfine Fuzzy Predicates) statements and can be embedded in FSTDSL/FORTRAN. Several examples using QUERY statements for a fuzzy database are illustrated.

This manipulation language is implemented in FSTDSL/FORTRAN and it is currently running on a FACOM 230-45S computer.

1. INTRODUCTION

Database systems have been vigorously studied since Codd (1970) proposed the relational model of data in 1970. Such database systems can only deal with well-defined and unambiguous data. In the real world, however, there exist uncertain or ambiguous data and information which cannot be defined in certain and well-defined form by any means. Since in everyday life we often make decisions based on such fuzzy data, the formulation and construction of a database which can represent and manipulate fuzzy data will increase the application areas of database systems and improve the interface for the smooth communication between men and machines. We will refer to such a database as a fuzzy database [Kunii (1976)].

Based on the theory of possibility distribution proposed by Zadeh (1978a, 1978b), we formulate an extended relational model as a data model of fuzzy database which is an extended version of Codd's relational model, and design and implement a data manipulation language for such a fuzzy database [Fukami, Umano, Mizumoto and Tanaka (1979)]. This language is implemented in FSTDSL/FORTRAN [Umano, Mizumoto and Tanaka (1978)] and currently running on a FACOM 230-45S computer.

2. POSSIBILITY DISTRIBUTION

To understand the concept of possibility distribution [Zadeh (1978a, 1978b)], we shall consider initially a simple non-fuzzy proposition such as

$$P_1: \text{Tom is 20 or 21 years old.} \tag{2.1}$$

The information we can obtain from this proposition is (a) it is possible that Tom's age is 20 or 21 years old, and (b) it is not possible that Tom's age is other than 20 or 21 years old. If we denote the possibility by two values {0, 1}, with 1 and 0 representing the situation in which there is possibility and no possibility, respectively, then the proposition P_1 induces a possibility distribution $\Pi_{AGE(Tom)}$ such as

$$\text{Poss}\{\text{AGE(Tom)}=u\} = \begin{cases} 1 & \text{for } u \in \{20,21\} \\ 0 & \text{for } u \notin \{20,21\} \end{cases} \qquad (2.2)$$

e AGE(Tom) denotes the AGE attribute of Tom and Poss{AGE(Tom)=u} possibility that AGE(Tom) may assume the value u in the numerical erse of discourse of age in the absence of any information rding AGE(Tom) other than P_1. And corresponding to the ibility distribution, a function $\pi_{\text{AGE(Tom)}}$ is defined as follows:

$$\pi_{\text{AGE(Tom)}}(u) = \begin{cases} 1 & \text{for } u \in \{20,21\} \\ 0 & \text{for } u \notin \{20,21\} \end{cases} \qquad (2.3)$$

h is called a <u>possibility distribution function</u>. Next, let us ider a fuzzy proposition:

$$P_2\text{: Tom is young.} \qquad (2.4)$$

this case, we cannot express a possibility by only two values }. So we extend permitted values of possibility to the unit rval [0,1], and we may have the possibility distribution (Tom), say, as shown in Fig.1.

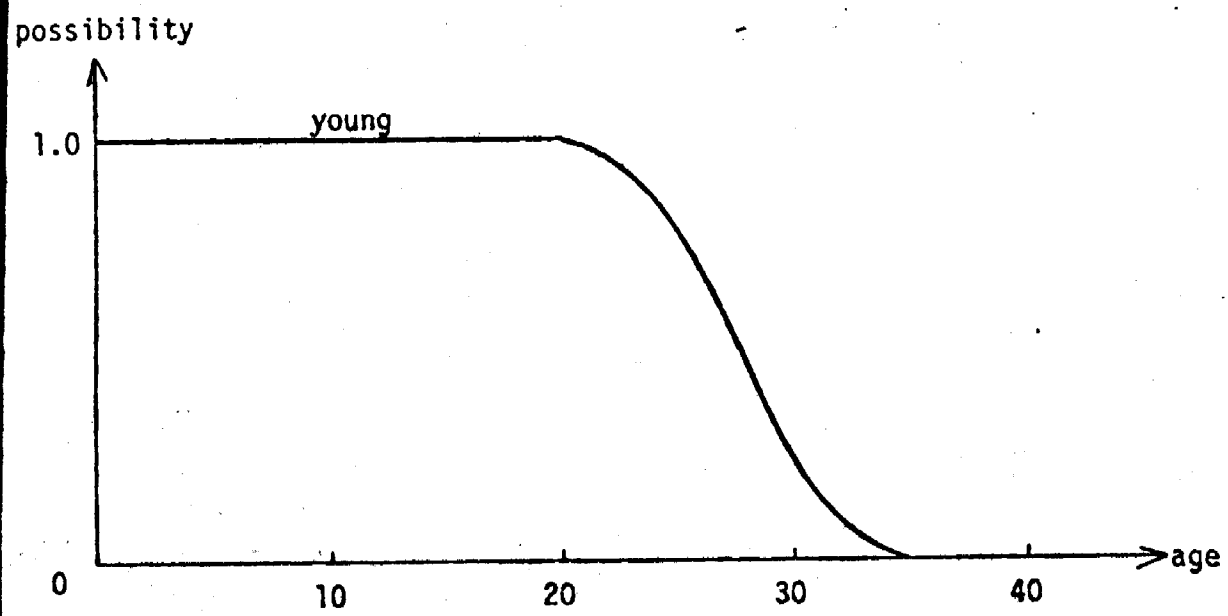


Fig.1. Possibility distribution function $\pi_{\text{AGE(Tom)}}(u)$ obtained from the proposition (2).

P_2

More generally, the possibility distribution function $\pi_{A(x)}$ associated with an attribute A of an object x is defined as:

$$\pi_{A(x)} : U \longrightarrow [0,1], \tag{2.5}$$

where U is a universe of discourse of A(x). $\pi_{A(x)}(u)$ represents the possibility that A(x) assumes the value u in U. For the convenient notation of possibility distribution $\Pi_{A(x)}$ associated with an attribute A of an object x, we use

$$\Pi_{A(x)} = \{\pi_{A(x)}(u_1)/u_1, \pi_{A(x)}(u_2)/u_2, \ldots, \ldots, \pi_{A(x)}(u_n)/u_n\}p, \tag{2.6}$$

where $\pi_{A(x)}$ is an associated possibility distribution function and u_i, i=1,2,...,n, represent the elements of U.

If a possibility distribution function $\pi_{A(x)}$ only takes the values of 0 and 1, it is called <u>non-fuzzy</u>. Otherwise it is called <u>fuzzy</u>.

In this juncture, we can define a certain, uncertain and ambiguous information or data on the attribute A of an object x. The information regarding A(x) is:

(i) <u>certain</u> if we have the only one value u_i in the universe of discourse U such that

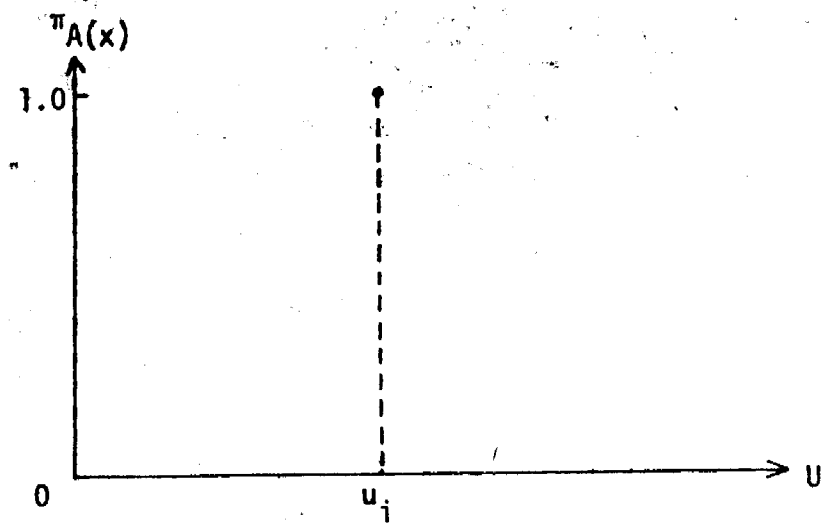
$$\pi_{A(x)}(u) = \begin{cases} 1 & \text{for } u = u_i \\ 0 & \text{for } u \neq u_i \end{cases} \tag{2.7}$$

(ii) <u>uncertain</u> if we have in U a non-fuzzy set S which contains more than one elments such that

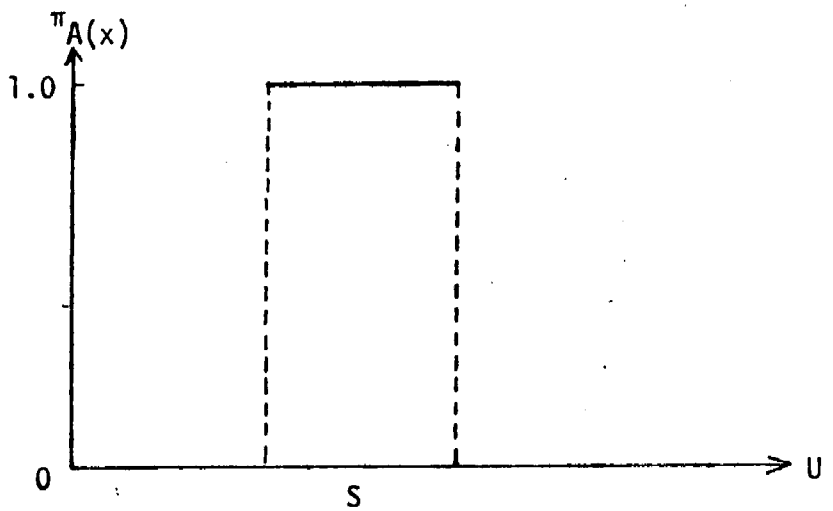
$$\pi_{A(x)}(u) = \begin{cases} 1 & \text{for } u \in S \\ 0 & \text{for } u \notin S \end{cases} \tag{2.8}$$

(iii) <u>ambiguous</u> if $\pi_{A(x)}$ is fuzzy.

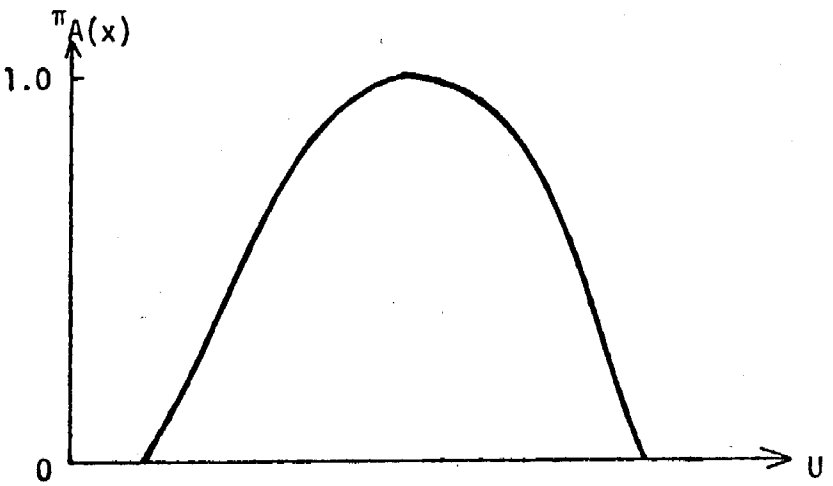
These are shown in Fig.2. Such terms as certain, uncertain and ambiguous information and data are used according to the above definition.



(a) Certain information



(b) Uncertain information



(c) Ambiguous information

Fig.2. Possibility distributions for certain, uncertain and ambiguous information.

We shall have two special possibility distributions. One is a possibility distribution whose $\pi_{A(x)}$ is identical to unity, i.e.,

$$\pi_{A(x)}(u) = 1 \qquad \text{for all } u \text{ in } U \tag{2.9}$$

which is called unknown since there is possibility that A(x) could be any value in U and we cannot obtain no information about A(x) from the possibility distribution $\pi_{A(x)}$. The other is a possibility distribution whose $\pi_{A(x)}$ is identical to 0, i.e.,

$$\pi_{A(x)}(u) = 0 \qquad \text{for all } u \text{ in } U \tag{2.10}$$

which is referred to as undefined because there is no possibility that the value of A(x) could exist in the universe of discourse U.

3. FUZZY DATABASE

We shall define an extended relational model for a fuzzy database, which is an extension of Codd's relational model of data [Codd (1970)].

A fuzzy database D_f is defined as a set of extended relations R_i, i=1,2,...,n, i.e.,

$$D_f = \{R_1, R_2, \ldots, R_n\} \tag{3.1}$$

in which an extended relation R_i is defined as a subset of the Cartesian product of a collection of possibility distributions, i.e.,

$$R_i \subseteq (P(U_{i1}) \cup \{\text{NULL}\}) \times (P(U_{i2}) \cup \{\text{NULL}\}) \times \ldots \times (P(U_{im}) \cup \{\text{NULL}\}) \tag{3.2}$$

where the symbols $\times$, $\cup$ and $\subseteq$ denote the Cartesian product, the union and the subset, respectively, in ordinary set theory, $P(U_{ij})$, j=1,2,...,m, are collections of all possibility distributions on a universe of discourse U_{ij} and NULL is a special value for representing the sutuation that we do not know even whether the attribute value is defined or not. The U_{ij} and $P(U_{ij}) \cup \{\text{NULL}\}$ are called a basic set and a domain, respectively.

[Example 1] Let us consider an extended relation PERSON whose attributes are NAME, AGE and CHILD_NAME. Let the basic set U_1 of attributes NAME and CHILD_NAME be a set of individual's names, e.g., practically a set of character strings, and U_2 of AGE be a set of numerical ages, e.g., the interval [0,150] of integer. Then, the extended relation PERSON is defined as

$$\text{PERSON} \subseteq (P(U_1) \cup \{\text{NULL}\}) \times (P(U_2) \cup \{\text{NULL}\}) \times (P(U_1) \cup \{\text{NULL}\}) \quad (3.3)$$

and one of its occurrences is, for example, shown in Fig.3. Note that attribute values which are not enclosed by { }p mean names of possibility distributions which have been defined other than PERSON.

This extended relation PERSON represents the following meaning. We know that Tom is 23 years old and has a child named Ted. Susan is 35 years old and has two children whose names are John and Mike, whereas Richard's age is 40 and he has one child whose name may be Judy or Anna. Note that Richard has only one child rather than two. If he had two children, we would consume two rows for Richard like Susan. Raymond's age is a possibility distribution young and the value unknown in the attribute CHILD_NAME means that we know he has a child but we don't know his child's name. For Victor, we do not know his age and the value undefined in the attribute CHILD_NAME

PERSON

NAME	AGE	CHILD_NAME
{Tom}p	{23}p	{Ted}p
{Susan}p	{35}p	{John}p
{Susan}p	{35}p	{Mike}p
{Richard}p	{40}p	{Judy, Anna}p
{Raymond}p	young	unknown
{Victor}p	unknown	undefined
{Smith}p	{50,51}p	NULL

Fig.3. Extended relation PERSON.

represents that he has no children. Finally, Smith may be 50 or 51 years old and the NULL value in the attribute CHILD_NAME means that we do not know even whether he has any children or not.

Since the extended relational model defined here treats uncertain and ambiguous information, it is impossible to deal with it in traditional two-valued and even multi-valued logic systems. So we shall discuss a logic system which can deal with uncertain and ambiguous information represented by a possibility distribution[1].

Assume that $\Pi_{A(x)}$ and $\Pi_{A(y)}$ are possibility distributions regarding an attribute A of objects x and y, respectively. Let us consider a proposition Q as follows:

$$Q: A(x) = A(y) \tag{3.4}$$

in the two cases where

(a) $\Pi_{A(x)} = \{1/u_i\}p$ and $\Pi_{A(y)} = \{1/u_j\}p$;

(b) $\Pi_{A(x)} = \{0.5/u_1, 1/u_2\}p$ and $\Pi_{A(y)} = \{1/u_2, 0.6/u_3\}p$.

For the case (a), if $u_i=u_j$, then the truth value of the proposition Q is 1, and otherwise 0. For the case (b), the problem is rather complicated. Since the true value of A(x) may be u_1 or u_2 and that of A(y) may be u_2 or u_3, it is possible that we have not only $A(x)=A(y)=u_2$ but also $A(x)\neq A(y)$.

We can not learn the fact more than the above from this ambiguous information. We introduce the parameter other than the truth value for solving this situation. We represent the truth value by an ordered pair <c,t> which is called a p-truth value[2]. In the p-truth value <c,t>, t is the same as the truth value in the sense of multi-valued logic and takes a value in the unit interval [0,1], in the other hand, c is a factor for reflecting the certainty of t and takes a value of either P or T, which means that it is possible and certain, respectively. Thus by the p-truth value <T,t>, t in [0,1], we mean that the truth value for a proposition is certainly t, and by <P,t> we mean that the maximum of the truth value is t under the

---

1 Another logic system to deal with it will appear in the subsequent paper.

2 The term p-truth value means a pair truth value or possibility truth value.

uncertain or ambiguous information. It should be noted that c-part involves possibility distributions, whereas t-part does predicates in the propositions. It is obvious that the p-truth values <T,0> and <P,0> have the same meaning. Moreover, we assume the ordered relation T>P between T and P.

In the above case (a), if $u_i=u_j$, then we have p-truth value <T,1> for the proposition Q, and if $u_i \neq u_j$, we have <T,0>. In the case (b), the p-truth value is <P,1>.

More generally, we have the following p-truth values for the proposition Q.

(i) If $\Pi_{A(x)} \cap \Pi_{A(y)} = \phi$, the p-truth value is <T,0>.
(ii) If $\Pi_{A(x)}=\{u_i\}p$ and $\Pi_{A(y)}=\{u_i\}p$, it is <T,1>. (3.5)
(iii) Otherwise, it is <P,1>.

Note that the proposition Q contains the non-fuzzy predicate equal(x,y) but not a fuzzy one, so the t-part in the above included only 0 and 1. When we use fuzzy predicates such as young(x), about_20_years_old(x) and approximately_equal(x,y) which return a value in the unit interval [0,1], the value other than 0 and 1 appears in the t-part. The t-part has no relationship to values of possibility in the possibility distribution but involves a fuzzy proposition. The processing for propositions which contains fuzzy predicates is as follows. Let $t_1, t_2, \ldots, t_n$ be the evaluated truth values of the predicate for all combinations of elements u_x and u_y in the possibility distributions $\Pi_{A(x)}$ and $\Pi_{A(y)}$.

(i) If $t_1=t_2=\ldots=t_n$, then the p-truth value is <T,t>.
(ii) Otherwise, it is <P,t>, where $t=\max(t_1,t_2,\ldots,t_n)$. (3.6)

Note that the definition (3.5) is a special case of this definition.

In what follows, we define logical operations; the conjunction $\wedge$, the disjunction $\vee$ and the negation $\sim$ for p-truth values.

(1) Conjunction.

$$<c_1,t_1> \wedge <c_2,t_2> = <\min(c_1,c_2), \min(t_1,t_2)> \quad (3.7)$$

where min(T,T)=T, min(T,P)=P and min(P,P)=P.

(2) Disjunction.

1. For $c_1=c_2$, we have

$$\langle c_1, t_1 \rangle \vee \langle c_2, t_2 \rangle = \langle c_1, \max(t_1, t_2) \rangle. \quad (3.8)$$

2. For $c_1 \neq c_2$, let $c_1 = T$ and $c_2 = P$, and we have
   (i) $t_1 \geq t_2$, or $t_1 \leq t_2$ and $t_1 \geq 0.5$:

$$\langle T, t_1 \rangle \vee \langle P, t_2 \rangle = \langle T, t_2 \rangle, \quad (3.9)$$

   (ii) otherwise:

$$\langle T, t_1 \rangle \vee \langle P, t_2 \rangle = \langle P, t_2 \rangle. \quad (3.10)$$

(3) Negation.

1. $\sim\langle T, t \rangle = \langle T, 1-t \rangle$ (3.11)

2. $\sim\langle P, 1 \rangle = \langle P, 1 \rangle$ (3.12)

Note that we have given no definition of $\sim\langle P, t \rangle$, $t \neq 1$, since we do not use it in the current version of our fuzzy database system.

This logic system has not been developed enough yet. We shall discuss several topics such as its properties and algebraic structure in the subsequent paper.

4. DATA MANIPULATION LANGUAGE

In this section, we shall describe a data manipulation language for the fuzzy database defined in the previous section. Its syntax is similar to DEDUCE language developed by Chang (1976).

Our language provides QUERY, INSERT, DELETE, DEFR (DEfine Fuzzy Relation) and DEFP (DEfine Fuzzy Predicate) statements and can be embedded into FSTDSL/FORTRAN [Umano, Mizumoto and Tanaka (1978)].

The QUERY statement retrieves data from a fuzzy database. The INSERT statement inserts and the DELETE statement deletes several tuples into and from the specified extended relation, respectively. The DEFR statement declares extended relations together with its attributes and their types before the INSERT statements insert tuples into them. The DEFP defines fuzzy predicate in order to use in a QUERY statement.

REPRESENTATION OF EXTENDED RELATIONS

We shall have a representation of extended relations in t language. To manipulate possibility distributions, we represent t as fuzzy sets, that is, a possibility distribution $\Pi_{A(x)}$ w possibility distribution function $\pi_{A(x)}$ is represented by a fuzzy F whose membership function μ_F is identical to $\pi_{A(x)}$, i.e.,

$$\mu_F(u) = \pi_{A(x)}(u) \qquad \text{for all } u \text{ in } U. \qquad (4$$

We call it a representation of a possibility distribution $\Pi_{A(x)}$ b fuzzy set F. It is obvious that the mapping from a collection of possibility distributions on U to a collection of all fuzzy sets o is one-to-one correspondence. In this language, we treat possibil distributions by their representations by corresponding fuzzy sets.

Thus the representation in this language is slightly differe from that of Fig.3 in Example 1. The different points are follows[3].

(1) The possibility distribution which has only one element wi the possibility 1 is expressed only by the element without t symbols { }p. Thus {Ted}p and {20}p are expressed only by Ted and 2 respectively.

(2) The possibility distribution which has more than one elemer must have a name of fuzzy set which starts with the dollar mark $ differenciate the case (1). The definition of fuzzy set is written FSTDSL notation. Note that since the unknown and undefined possibility distributions, we express them as $UNKNOWN a $UNDEFINED. And the NULL is expressed as $NULL because of i speciality although it is not possibility distribution.

[Example 2] The extended relation PERSON in Fig.3 is translated ir Fig.4 using the above convention. Note that $JA and $A50 have to defined as

$$\$JA = \{Judy, Anna\}p, \qquad (4.$$
$$\$A50 = \{50, 51\}p \qquad (4.$$

---

3 In the implementation of the language, we really represent a possibil distribution which has only one element by the element, and the other possibil distribution by fuzzy sets whose names start with the dollar mark $ except Unkn and UNDEFINED.

PERSON

NAME	AGE	CHILD_NAME
TOM	23	TED
SUSAN	35	JOHN
SUSAN	35	MIKE
RICHARD	40	$JA
RAYMOND	$YOUNG	$UNKNOWN
VICTOR	$UNKNOWN	$UNDEFINED
SMITH	$A50	$NULL

Fig.4. Extended relation PERSON in the language.

which can be written in FSTDSL as:

```
$JA  := FSET(JUDY, ANNA);                    (4.4)
$A50 := FSET(50,51);                         (4.5)
```

where the symbol := means the assignment operator and FSET is the fuzzy-set construction operator in FSTDSL.

QUERY STATEMENTS

The QUERY statement is a main feature in this language. The QUERY statement has a general form as:

```
QUERY relation_name (target_list):
        conditional_part                     (4.6)
QEND
```

By (4.6) we can obtain a set of tuples, whose attributes are seen in the target_list, which satisfy the conditional_part.

The conditional_part may consist of conditional expressions or again other QUERY statements. The evaluation is based on the logic of p-truth value. The conditional expression and its evaluation are defined as follows.

1. Factors.
   (i) A constant and a variable are factors.

(ii) If $f_1, f_2, \ldots, f_n$ are factors and F is a function name, then $F(f_1, f_2, \ldots, f_n)$ is a factor.

A constant is classified into a character string, a number and a relation name. We adopt the inverse qoute notation. So a character string for a constant is not enclosed by the quotation marks ' '. A relation name had to be defined by a DEFR or QUERY statement and only occurs in functions.

The inverse quote notation makes a character string which begins with ? or * to be a variable. Variables with ? and * are called variables in ?-mode and *-mode, respectively. A variable in *-mode may appear in functions and relational terms and means its value, which may generally be a fuzzy set representing a possibility distribution. In the other hand, a variable in ?-mode occurs only in a relational term. It will be described in the explanation of a relational term.

For a function, COUNTS, SUM and AVG are currently available and compute the number of elements and the sum and the average of the specified attribute in a relation, respectively.

2. Terms.

(i) If $f_1, f_2, \ldots, f_n$ are factors and P is a built-in predicate symbol or a fuzzy predicate symbol, then $P(f_1, f_2, \ldots, f_n)$ is a predicate term.

(ii) If $f_1, f_2, \ldots, f_n$ are numbers, character strings or variables and R is an relation name and $a_1, a_2, \ldots, a_n$ are a subset of its attribute names, then $R(a_1=f_1, a_2=f_2, \ldots, a_n=f_n)$ is a relational term.

The built-in predicates SETEQ, DISJOINT, CONTAINS, EQ, GE, GT, FEQ and FCONT are available now. The SETEQ, DISJOINT and CONTAIN take two parameters of sets S_1 and S_2 and return the p-truth value <T,1> or <T,0> corresponding to $S_1=S_2$, $S_1 \cap S_2=\phi$ and $S_1 \supseteq S_2$, respectively. The EQ, GE and GT return the p-truth value <T,1>, <P,1> or <T,0> for two constants or variables in *-mode. The FEQ and FCONT compare two fuzzy sets F_1 and F_2 of constants and variables in *-mode and return the p-truth value <T,t>, where t is in the unit interval [0,1], corresponding to $F_1 = F_2$ and $F_1 \supseteq F_2$, respectively.

A fuzzy predicate is defined using the DEFP statement by a user and returns the p-truth value <T,t> or <P,t>, where t is in [0,1]. Currently, only unary fuzzy predicates are provided.

As for a relational term, let the values of components in a tuple in a relation R corresponding to the attributes a_1, a_2, ..., a_n to be v_1, v_2, ..., v_n, respectively, and the evaluated values of the factors f_1, f_2, ..., f_n to be w_1, w_2, ..., w_n, respectively. Then the p-truth value of a relational term as:

$$R(a_1=f_1,\ a_2=f_2,\ \ldots,\ a_n=f_n) \tag{4.7}$$

is defined as

$$(v_1=w_1) \wedge (v_2=w_2) \wedge \ldots \wedge (v_n=w_n) \tag{4.8}$$

where $v_i=w_i$, i=1,2,...,n, can be evaluated by (3.5) and the operation $\wedge$ by (3.7). If f_j is a variable in ?-mode, the value v_j is assigned to the variable and the p-truth value always becomes <T,1>. The above process continues until all tuples in the relation are exhausted.

3. Literals.

If t is a term, t and NOT(t) are literals.

NOT means the negation of t and the p-truth value of NOT(t) is defined by (3.11) and (3.12).

4. Clauses.

(i) A literal is a clause.

(ii) If ℓ_1, ℓ_2, ..., ℓ_n are literals which contain only predicate terms, then OR(ℓ_1, ℓ_2, ..., ℓ_n) is a clause.

OR corresponds to the disjunction and the p-truth value is defined by (3.8)-(3.10).

5. Conditional expressions.

If c_1, c_2, ..., c_n are clauses, $c_1;c_2;\ldots;c_n$ is a conditional expression, which means the conjunction $c_1 \wedge c_2 \wedge \ldots \wedge c_n$. The conjunction is evaluated by (3.7).

[Example 3] We have an extended relation shown in Fig.4, where $YOUNG is defined as in Fig.1, and $JA and $A50 are of (4.4) and (4.5), respectively. Let us consider the following QUERY statement:

```
QUERY A (NAME = X):
  PERSON (AGE = ?Y, NAME = ?X);
  GT(*Y, 25)
QEND
```

We have the result as:

```
A@1=FSET(1/SUSAN, 1/RICHARD, 1/SMITH);
A@2=FSET(1/RAYMOND, 1/VICTOR);
```

where A@1 contains the elements whose p-truth values are $\langle T,t\rangle$, $t \geq \theta$ and A@2 consists of the elements which have the p-truth value $\langle P,t\rangle$, $t \geq \theta$. In other words, A@1 certainly satisfies the conditional expression, whereas A@2 possibly satisfies it. θ is a threshold value, which can be specified by a user. If a user does not specify it, the default value 0.5 is assumed. Note that although the information on Smith's age is uncertain, his age is certainly greater than 25. It is often the case that we can obtain certain answers for ambiguous queries even based on ambiguous information or data.

INSERT AND DELETE STATEMENTS

The INSERT and DELETE statements in this language are so restricted. They can only insert and delete several tuples into and from the specified relation, respectively.

The general forms of INSERT and DELETE statements are

```
INSERT relation_name tuple_list IEND          (4.9)
DELETE relation_name tuple_list DEND          (4.10)
```

where the relation_name has to be defined by the DEFR statement and the tuple_list is a finite times of tuples separated by the comma ,. For example, we can obtain the extended relation PERSON in Fig.4 by

```
INSERT PERSON <TOM,23,TED>, <SUSAN,35,JOHN>,
              <SUSAN,35,MIKE>,
              <RICHARD,40,$JA>,
              <RAYMOND,$YOUNG,$UNKNOWN>,
              <VICTOR,$UNKNOWN,$UNDEFINED>,
              <SMITH,$A50,$NULL>     IEND
```

where $YOUNG, $JA and $A50 must be defined before.

Examples of the DELETE statement is omitted because of the same syntax as the INSERT statement.

DEFR STATEMENT

The DEFR (DEfine Fuzzy Relation) statement is used to declare extended relation with attribute names and their types.

The general form is as follows:

$$\text{DEFR relation\_name } \langle a\_name_1:type_1,\ a\_name_2:type_2,\ \cdots,\ a\_name_n:type_n \rangle \text{ DEFEND} \qquad (4.11)$$

where the a_name_i, $i=1,2,\ldots,n$, mean attribute names and $type_i$, $i=1,2,\ldots,n$, are either CHAR (character strings), INTEGER or REAL.

The PERSON in Fig.4 is defined by

```
DEFR PERSON <NAME:CHAR, AGE:INTEGER,
             CHILD_NAME:CHAR>  DEFEND
```

DEFP STATEMENT

The DEFP (DEfine Fuzzy Predicate) is used to define a fuzzy predicate. The definition is similar to a fuzzy set definition. Really, a fuzzy predicate is represented by a fuzzy set in the system. Unary fuzzy predicates can only be defined.

The general form is as follows:

$$\text{DEFP f\_predicate\_name} = (t_1/u_1,\ t_2/u_2,\ \ldots,\ t_n/u_n) \text{ PEND} \qquad (4.12)$$

where t_i, $i=1,2,\ldots,n$, are numbers in the interval $[0,1]$, and u_i, $i=1,2,\ldots$, are elements in the universe of discourse.

For example, a fuzzy predicate as

$$\text{about25}(u) = \begin{cases} 1 & \text{for } u = 25 \\ 0.8 & \text{for } u \in \{24,26\} \\ 0.5 & \text{for } u \in \{23,27\} \\ 0 & \text{otherwise} \end{cases} \qquad (4.13)$$

is defined by

```
DEFP ABOUT25 = (0.5/23, 0.8/24, 1/25,
                0.8/26, 0.5/27)  PEND
```

We have briefly described each statement provided in our language. In the next section, we shall have several examples of the QUERY statement to show some facilities of our language.

5. EXAMPLES OF QUERY STATEMENT

This section describes some facilities of our manipulation language by writing several queries in it.

First of all, we have three extended relations as:

```
CANDIDATE (NAME, SEX, AGE, SCAREER)
PROF (NAME, PROFES, INCOME)
BODY (HEIGHT, WEIGHT, NAME)
```

which have been defined by the DEFR statements and insertions of several tuples generate the occurrences shown in Figs.5(a)-(c). A basic set of the attribute SEX is {MALE, FEMALE}. The SCAREER means a school career and its basic set is {U, H}, where U denotes a university and H a high school. Basic sets of the attributes NAME, PROFES are sets of names of individuals and professions, respectively. The AGE, INCOME, HEIGHT and WEIGHT take a set of integers as a basic set. The fuzzy sets $An represent "about n", which are defined as in AGE, HEIGHT and WE_GHT

```
$An = FSET(0.5/n-1, 1/n, 0.5/n+1);          (5.1)
```

and in INCOME

```
$An = FSET(0.5/n-5, 1/n, 0.5/n+5);          (5.2)
```

1. Retrieve the name of a person who graduates from a university.

```
QUERY A (NAME = X):
  CANDIDATE (NAME=?X, SCAREER=U);
QEND
```

CANDIDATE

NAME	SEX	AGE	SCAREER
SMITH	MALE	30	U
JOHN	MALE	$A28	H
RICHARD	MALE	$A25	$UNKNOWN
ANNA	FEMALE	22	H
MARY	FEMALE	$A25	U
LUCY	FEMALE	$A20	H
SUSAN	FEMALE	23	U

(a) *Extended relation CANDIDATE*

PROF

NAME	PROFES	INCOME
SMITH	PROGRAMMER	300
JOHN	ENGINEER	$A200
RICHARD	$UNDEFINED	$UNDEFINED
ANNA	CLERK	$A150
MARY	TEACHER	$A300
LUCY	$UNDEFINED	$UNDEFINED
SUSAN	PROGRAMMER	$A200

(b) *Extended relation PROF*

BODY

HEIGHT	WEIGHT	NAME
180	$A60	SMITH
$A175	80	JOHN
165	$A60	RICHARD
$A170	$A55	ANNA
165	55	MARY
170	60	LUCY
$A175	$A55	SUSAN

(c) *Extended relation BODY*

Fig.5. The extended relations CANDIDATE, PROF and BODY.

```
Result: A@1=FSET(1/SMITH, 1/MARY, 1/SUSAN);
        A@2=FSET(1/RICHARD);
```

This is one of the simplest queries in our language. In relational algebra [Codd (1972)], it restricts the relation CANDIDATE to the value U in the attribute SCAREER and projects them to the attribute NAME. As for the result, SMITH, MARY and SUSAN satisfy the condition in the p-truth value <T,1>, whereas RICHARD's SCAREER is unknown and he may graduate from a university, so he is in A@2, which means that the p-truth value is <P,1>.

2. (i) Retrieve the name and age of a person who is 25 years old.
(ii) Retrieve the name and age of a person who is about 25 years old.

```
(i) QUERY A0 (NAME=U, AGE=V):
      CANDIDATE (NAME=?U, AGE=?V);
      EQ(*V, 25);
    QEND

Result: A0@1=EMPTY;
        A0@2=FSET(1/<RICHARD,$A25>, 1/<MARY,$A25>);

(ii) QUERY A1 (NAME=U, AGE=V):
       CANDIDATE (NAME=?U, AGE=?V);
       FEQ(*V, @A25);
     QEND

Result: A1@1=FSET(0.7/<RICHARD,$A25>, 0.7/<MARY,$A25>);
        A1@2=EMPTY;
```

The symbol @ followed by A25 in the third line in (ii) means that the name A25 is a fuzzy set name. This fuzzy set is defined as

```
A25 := FSET(0.4/23, 0.8/24, 1/25, 0.7/26);
```

Note that this fuzzy set is different from the fuzzy set $A25 representing a possibility distribution in the extended relation CANDIDATE. Really, the fuzzy set $A25 is defined as

```
$A25 := FSET(0.5/24, 1/25, 0.5/26);
```

In (i), the predicate EQ compares two parameters by elements. In the

other hand, the predicate FEQ compares by fuzzy sets. As is shown in results, MARY and RICHARD whose AGE attribute value is $A25 are in A0@2 in (i) but in A1@1 in (ii). The grade value 0.7 in A1@1 is the compatibility of A25 with $A25.

3. Retrieve the name of person who is young.

```
QUERY B (NAME = X):
  CANDIDATE (NAME=?X, AGE=?Y);
  YOUNG(*Y);
QEND
```

Result: B@1=FSET(1/ANNA, 1/LUCY, 0.8/SUSAN);
B@2=FSET(0.6/RICHARD, 0.6/MARY);

A fuzzy predicate YOUNG has been defined by the DEFP statement:

```
DEFP YOUNG = (1/19, 1/20, 1/21, 1/22,
                    0.8/23, 0.6/24, 0.3/25)  PEND
```

Using the predicate YOUNG, the truth value that ANNA and SUSAN is YOUNG is obtained as 1 and 0.8, respectively. For LUCY, since her age is

$$\$A20 = \{0.5/19,\ 1/20,\ 0.5/21\}p \tag{5.3}$$

and the truth value of YOUNG for 19, 20 and 21 is 1, LUCY is in B@1. This is the representative case where ambiguous data and ambiguous query make a certain answer. For RICHARD and MARY, the grade value 0.6 is obtained from the maximum value of 0.6 and 0.3 which are truth values of YOUNG for their possible ages 24 and 25, respectively.

4. Retrieve the name of a female whose profession is a programmer.

```
QUERY C (NAME = X):
  CANDIDATE (NAME=?X, SEX=FEMALE);
  PROF (NAME=*X, PROFES=PROGRAMMER);
QEND
```

Result: C@1=FSET(1/SUSAN);
C@2=EMPTY;

This query means that the relational algebra joins naturally the

relations CANDIDATE and PROF on the attribute NAME, restricts to FEMALE in the attribute SEX and PROGRAMMER in the attribute PROFES and projects to the attribute NAME.

5. Retrieve the name and weight of a female who graduates from only a high school and whose weight is greater than or equal to the average of all persons.

```
QUERY E (NAME=U, WEIGHT=V):
  QUERY W (X, Y):
    BODY (NAME=?X, WEIGHT=?Y);
  QEND
  CANDIDATE (NAME=?U, SEX=FEMALE, SCAREER=H);
  BODY (NAME=*U, WEIGHT=?V);
  GE(*V, AVG(W@1,2));
QEND
```

```
Result: E@1=EMPTY;
        E@2=FSET(1/<LUCY,60>);
```

AVG is a function which computes the average. Its first parameter W@1 is a set obtained by the QUERY statement of the 2nd line to the 4th line. If a relation does not need its attribute names, e.g., it never appears as a relational term, we can omit the attribute names like in this query.

6. Retrieve the name of a female who is 23 years old and graduates from a university.

```
QUERY EEX (NAME=X, AGE=W):
  QUERY FEMALE (N=XX, A=V, SC=Y):
    CANDIDATE (NAME=?XX, SEX=FEMALE,
                  AGE=?V, SCAREER=?Y);
  QEND
  FEMALE@1 (N=?X, A=?W, SC=U);
QEND
QUERY EEY (NAME = X):
  EEX (NAME=?X, AGE=23);
QEND
```

```
Result: EEY@1=FSET(1/SUSAN);
        EEY@2=EMPTY;
```

This query can be written in more simple QUERY statement. But this shows that the relation obtained by QUERY statement with attribute names can be used as if it is in the database, that is, it can be used in another QUERY statement without the DEFR statement.

We have described some facilities of our language by illustrating a variety of queries.

6. CONCLUSIONS

We have defined an extended relational model as a model for representing and manipulating uncertain or ambiguous data and described a data manipulation language for such a fuzzy database. This language is implemented in FSTDSL/FORTRAN and currently running on a FACOM 230-45S computer. The preprocessor for a program embedded into the host language FSTDSL/FORTRAN has been implemented in PL/I, although we omitted its description.

The extended relational model in this paper facilitates the representation of uncertainty and ambiguity contained in data itself, but cannot easily represent uncertainty and ambiguity in the relationship between data. The data model which overcomes this problem is defined in connection with a fuzzy version of relational algebra by Umano, Fukami, Mizumoto and Tanaka (1979).

By the introductions of higher type possibility distributions whose values of possibility may be again a possibility distribution and higher level possibility distributions whose elements may be possibility distributions, fuzzy databases can represent a hierarchical fuzzy data and they can be used as a good tool for the representation of complex fuzzy data and knowledge.

Fuzzy database systems will find a number of applications in such fields as natural language processing, question-answering and artifitial intelligence where fuzzy data play an important role in nature.

REFERENCES

Chang, C.L. (1976). DEDUCE — A Deductive Query Language for Relational Data Bases. In Chen, C.H. (ed.) Pattern Recognition and

Artificial Intelligence, Academic Press, New York, USA, pp.108-134.

Codd, E.F. (1970). A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, Vol.13, pp.377-387.

Codd, E.F. (1972). Relational Completeness of Data Base Sublanguages. In Rustin, R. (ed.) Data Base Systems, Courant Computer Science Symposium 6, Prentice-Hall, Englewood Cliffs, USA, pp.65-98.

Fukami, S., Umano, M., Mizumoto, M. and Tanaka, K. (1979). Fuzzy Database Retrieval and Manipulation Language. Technical Reports of IECE of Japan, Vol.78, No.233 (on Automata and Languages), pp.65-72, AL78-85, January, 1979 (in Japanese).

Kunii, T.L. (1976). DATAPLAN: An Interface Generator for Database Semantics. Information Sciences, Vol.10, pp.279-298.

Umano, M., Mizumoto, M. and Tanaka, K. (1978). FSTDS System: A Fuzzy-Set Manipulation System. Information Sciences, Vol.14, pp.115-159.

Umano, M., Fukami, S., Mizumoto, M. and Tanaka, K. (1979). On Fuzzy Relational Algebra. 20th Natioanl Convention Records of IPS of Japan, pp.693-694, No.1D-6, July, 1979 (in Japanese).

Zadeh, L.A. (1978a). Fuzzy Sets as a Basis for a Theory of Possibility. Fuzzy Sets and Systems, Vol.1, pp.3-28.

Zadeh, L.A. (1978b). PRUF —— A Meaning Representation Language for Natural Languages. International Journal of Man-Machine Studies, Vol.10, pp.395-460.