Fuzzy Relational DB and its Application to Expert System

Yang Yun
Dept. of Computer, Beijing Inst. of Aeronautics & Astronautics, PRC

In this extended abstract, fuzzy data representation and query are discribed. An approach of fuzzy DB to support expert consultation system is expressed. The features, scheme and syntax of fuzzy relational algebraic language FdBASE are also given here.

Data representation

There are three types of fuzzy data in fuzzy relational DB(FRDB): character (discrete and finite), numerical data and null value(Unknown represents existing, Undefined not existing and NULL not knowing whether it exists or not), singleton as a special case.

Character data is the mostly used data in FRDB, so it is necessary to be represented powerfully. There are sementic relations between and in data.

Numerical data is ordered by itself.

Null value is attached to two above-mentioned data, it is important to distinct it. The process of null value is almost the same with other papers discussed.

There are two levels to represent data, they are external (logical) and internal (physical) ones. External level data is just like natural language, which can be easily used and understood by users, while internal level data can be processed by computer. In internal level, we use 4-tuples(e.g. (19,21,1,1) presents fuzzy data "about-20") to present numerical data(which measured by trapezoidal possibility distributions) and new relations to present character data (subsets, which measured by similarity relations). Two levels data is equivalent and must have a converter.

In one character data, there exists sementic relationship among elements of the subset such as exclusive OR restriction etc. This kind of relationship is very general in real world. All these we have discussed.

Now an example is given.

External level representation of the relation: PERSON:

name	age	math score	child name	cloth color
A	young	75	A1	NULL
B	20	about-80	{B1,B2,B3|<=2}	red
C	[35-40]	Undefined	{C1,C2,C3|>=2}	{yellow, blue}
D	Unknown	rather-good	{D1,D2|=1}	{red, blue}

In the relation PERSON, such as {C1,C2,C3|>=2} indicates C has at least two (>=2) children (name C1,C2 or C3), {yellow, blue} means C's cloth-color may be yellow or blue etc. Of course these sementic relations in data have many kinds and only part of them is discribed here.

Corresponding to the external level data of relation PERSON, its internal one is shown as follows. Character null value and singleton can be identified by an ID (e.g. "¥"), other character data can be presented by new relations and numerical data can be presented by 4-tuples.

Internal level representation of the relation: PERSON:

name	ageA	ageB	age-δ_1	age-δ_2	mathA	mathB	math-δ_1	math-δ_2	child-name	cloth-color
¥A	20	35	3	5	75	75	0	0	¥A1	¥NULL
¥B	20	20	0	0	78	82	2	2	Bchild	¥red
¥C	35	40	0	0	1	0	0	0	Cchild	Ccloth
¥D	2	0	0	0	65	75	5	3	Dchild	Dcloth

Query

In relational algebraic manipulations, union, set difference, Cartesian product, projection and selection(which are completeness) are discussed. It is more pentinent and suitful to use similarity relation measure in character data and possibility distribution measure in numerical data. The definition of similarity relation given by Zadeh is too strong. In practical use the weaker similarity relation or even proximity relation may be enough.

Set difference and union

The main difference of these two manipulations between FRDB and RDB is the definition of redundancy. The threshold is used and the formulae are omitted.

Cartesian product and projection

These two manipulations in FRDB is almost the same to that in RDB.

Selection

Selection manipulation plays the core role in relational algebra query. Fuzzy data and comparators in query statement are also considered. In fact, at last it is always the comparision between two data. The measure formulae of character and numerical data are also omitted here. Because of the varieties of fuzzy data and comparators, interactive system is more effective and easily used than compiler one.

Application to Expert Consultation System(ES)

Rule-based ES take an important role in ES. While in fact, DB can support rule of ES, and fact can be stored in tuple directly. It must be mentioned that the representation ability of FRDB is very powerful, esp. fuzzy information, which is widely used in ES.

Under the prevailing circumstances, ES consultation level is higher than DB query level. When knowledge already has been in FRDB. the ES consultation result can be given by FRDB query. Theoretically speaking, it is obvious that we only need a translator which converts ES consultation to FRDB queries and get result.

$$\begin{tikzcd}
\text{ES consultation} \arrow[rr, bend left] & \text{translator} & \text{a series of FRDB queries} \arrow[d] \\
 & & \text{query results} \arrow[ull, bend left]
\end{tikzcd}$$

There are two ways to apply FRDB to ES. One way is that when FRDB exists, to expand interface of FRDB into ES consultation level is simple (i.e. to develop a translator) and it can given expert level consultation. The other way is that when an ES needs to be developed, fuzzy relational DBMS can be used to create a FRDB and then develop a translator, that is ES has already been generated.

Fuzzy Relational Algebraic Language FdBASE

FdBASE is based on dBASE. FdBASE uses data in conventional RDB but improves the query interface of RDB thoroughly. FdBASE can deal with fuzzy comparator and fuzzy constant. Its efficiency is satisfied and its syntax is shown in appendix.

Two main points of FDB are data representation and query, the second one is discussed here. It is important to spread up the use of FRDB, so some work of query interface has been done. Because of the popularity of dBASE, FdBASE is implemented at first. The interface of FdBASE is very friendly(in some extension just like the opinion of QBE)and the spirit of FDB can be easily feeled by users. The input of query is system-directed. According to data (RDB) used now, it is obvious that some data measures must be simplified, such as the numerical fuzzy constant, which is now represented by rectangle (not trapezoid) etc.

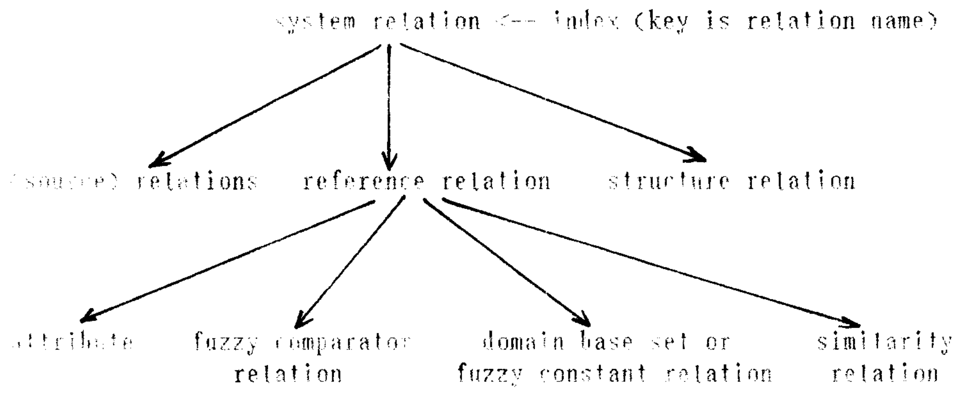
There are several features in FdBASE.
(1). It can deal with fuzzy condition and has intelligence in some extension.
(2). It is interactive(any non-professional user can use it without background).
(3). In it all dBASE commands can be used(it's useful to specialist).
(4). All fuzzy comparators and constants can be defined by users themselfs or pre-defined in system.
(5). It make full use of existing resources and it can be developed and widely used easily.
(6). The fuzzy comparators and constants may be appended step by step.

The main menu of FdBASE is following:

```
                WELCOME TO USE FDBASE (1986)
                        MAIN MENU

0 ... end query                  1 ... two relations intersection
2 ... two relations minus        3 ... two relations union
4 ... projection                 5 ... two relations product
6 ... select-projection          7 ... select-join-projection
8 ... dBASE command

waitting (0-8) ......
```


system relation <-- index (key is relation name)
(source) relations
reference relation
structure relation
attribute
fuzzy comparator relation
domain base set or fuzzy constant relation
similarity relation

RELATION STRUCTURE

Relation scheme.

system relation (relation name, reference relation, structure relation)
reference relation (attribute name, fuzzy comparator relation, domain base set or fuzzy constant relation, similarity relation)
structure relation (attribute name, attribute type, ...)
numerical fuzzy comparator relation (fuzzy comparator name, precise comparator, scope comparator, scope)
character fuzzy comparator relation (fuzzy comparator name, similarity comparator, similarity)
domain base set relation (element)
fuzzy constant relation (constant name, low-value, high-value)
similarity relation (element-1, element-2, similarity)

Appendix

Syntax of FdBASE (EBNF)

```
<fuzzy relational algebra> ;;=
  <relation name> <relational algebra operator> <relation name> to <relation name>[1]
 | project <relation name> to <relation name> attributes <attribute list>
 | select from <relation name> to <relation name> [attributes <attribute list>][for <expression 1>]
 | join <relation name>, <relation name> to <relation name> [attributes <attribute list>][for <expression 2>]
 |<all dBASE commands>
<relational algebra operator> ;;= minus | union | intersection | product
<attribute list> ;;= <attribute name>[{,<attribute name>}   ]
<expression 1> ;;= [<expression 1><logic operator>]<simple expression> | .not. <simple expression>
<logic operator> ;;= .and. | .or.
<simple expression> ;;= <fuzzy relational expression> | <precise relational expression>[2]
<fuzzy relational expression> ;;= <attribute name><comperator>(<attribute name> | <constant>)[3]
<comperator> ;;= <relational operator>[4] | <fuzzy comperator> [5]
<constant> ;;= <precise contant> | <fuzzy constant>[6]
<expression 2> ;;= <attribute name><comperator><attribute name | <precise relational expression>
```

REMARK

[1]. sementic issue; two source relations must be compatible.
[2]. precise relation expression is the same to that in dBASE, and its BNF is omitted here.
[3]. sementic issue; left and right side of comperator must be compatible.
[4]. relational operator is the same to that in dBASE and omitted.
[5]. fuzzy comperator is linguistic comperator, it is pre-defined by system or self-defined by user in his use.
[6]. fuzzy constant is linguistic value, its definition is just like [5].