ABOUT FUZZY DATABASE QUERY LANGUAGES AND THEIR RELATIONAL COMPLETENESS THEOREM

Cristina-Maria Vladarean
Software Development Team, S.C. WATERS Romania S.R.L., Romania
cristina_vladarean@waters.com

Two fuzzy database query languages are proposed. They are used to query fuzzy databases that are enhanced from relational databases in such a way that fuzzy sets are allowed in both attribute values and truth values. A fuzzy calculus query language is constructed based on the relational calculus, and a fuzzy algebra query language is also constructed based on the relational algebra. In addition, a fuzzy relational completeness theorem such that the languages have equivalent expressive power is proved.

Keywords: fuzzy database query languages; relational completeness theorem; fuzzy sets; attribute values; truth values; fuzzy calculus query language; relational calculus; fuzzy algebra query language; fuzzy set theory; information retrieval; query languages; relational databases.

1. INTRODUCTION

Database technology has been advanced up to the relational database stage with the purpose that user interfaces with databases may approach a level of human interfaces. It is recognized that the fuzzy theory is suitably applied to some human-oriented engineering fields, one of which is information processing, in particular database retrievals. In fact, fuzzy database models that allow fuzzy attribute values and fuzzy truth values in enhanced relational databases have been studied in [3] and [4]. However, these studies are restricted to just some particular applications and not grounded on theories of fuzzy database query languages. The fuzzy database systems would not be systematically developed on the basis of these studies; it is due to Codd's relational database theory that relational database systems have been systematically developed. It is desirable that theoretical foundations of fuzzy databases be established in order to systematically develop fuzzy database systems.

There was an excelent work done in the field of the fuzzy database theory; it develops a theoretical foundation for the fuzzy functional dependencies of

fuzzy databases [1]. The work encourages further research for the rest of theoretical foundation of fuzzy databases. This paper aims to present a theoretical foundation of query languages to fuzzy databases. It proposes two fuzzy database query languages: a fuzzy calculus query language and a fuzzy algebra query language. In adition, it proves a relationalcompletness theorem such that both the languages are equivalent in expressive power to each other. With these theoretical foundations, fuzzy database query systems will be developed systematically.

2. A FUZZY DATABASE MODEL

A fuzzy database is defined as an enhanced relational database that allows fuzzy attribute values and fuzzy truth values; both of these are expressed as fuzzy sets.

Fuzzy data model. A fuzzy database consists of relations: a relation is a relation $R(t_1, \ldots, t_n)$ in a Cartesian product $P_1 \times \cdots \times P_n$ of domains P_i; each P_i is a set of fuzzy sets t_i over an attribute domain $D_i (1 \leq i \leq n)$. It is assumed that key attributes take ordinary nonfuzzy values. For the notational convenience, fuzzy sets are identified with their representative membership functions; for example, t_i also denotes a membership function.

Fuzzy attributes. Attribute values such as age have nonfuzzy values such as 20 in the relational database; attribute values are defined as fuzzy predicates such as „young" and „about forty" in the fuzzy database. For example, a fuzzy attribute value of „age of Dr. X is young" is expressed as a possibility function $P(age\ of\ X)$ =YOUNG; here YOUNG denotes a fuzzy set that represents the fuzzy predicate „young". Thus attribute values are identified with fuzzy sets such as YOUNG.

Fuzzy truth values. Truth values of any tuples are either 1 (=true) or 0 (=false) in the relational database; truth values of any tuples are defined as fuzzy predicates such as „0.7" and „completely true" in the fuzzy database. Consider, for example, a tuple t that asserts a fuzzy proposition: „It is completely true that Dr. X is very much older than twenty". The truth value of t is expressed as a possibility distribution $P[T(t)] = N$; $T(t)$ denotes truth value of t and N denotes a fuzzy set that represents the fuzzy predicate „completely true". Thus the true values $T(t)$ are identified with fuzzy sets such as N over $z \in [0, 1]$; the value $z \in [0, 1]$ has the following meaning:

1) $z = 0$ means that the tuple t is completely false;

2) $0 < z < 1$ means that the tuple t is true to the degree expressed by the real number z;

3) $z = 1$ means that the tuple t is completely false.

In particular, each tuple t of the relation $R(t_1, \ldots, t_n)$ is given a unique truth value $T(t)$ by system designers at system generation time. In this case,

$T(t)$ determines a mapping $T : P_1 \times \cdots \times P_n \to P([0,1])$ where $P([0,1])$ is a set of fuzzy sets over $z \in [0,1]$.

3. QUERY BY TUPLE FUZZY CALCULUS

Tuple fuzzy calculus. A tuple fuzzy calculus (query language) is constructed as an enhancement of the tuple relational calculus. Formulas in the tuple fuzzy calculus are of the form $(t \mid f(t))$ where t is a fuzzy tuple variable; each ith component t_i is a fuzzy variable in P_i; f is a tuple fuzzy well-formed formula (WFF).

Tuple fuzzy WFF's are enhanced from those of the tuple relational calculus as follows.

1) **Atomic Tuple Fuzzy WFF's:** An atomic tuple fuzzy WFF consists of fuzzy sets and a fuzzy comparison operator $*$. The fuzzy comparison operator $*$ is one of the operators: equal; not equal; proper inclusion; inclusion. The fuzzy comparison operator $*$ is an enhancement from the arithmetic comparison operator $(=, \neq, <, >, \leq, \geq)$ in the relational calculus. Then the atomic tuple fuzzy WFF's are either of the following two types:

a) $(t_i) * (s_j)$; here, it is assumed that t and s are fuzzy tuple variables such that $D_i = D_j (1 \leq i, j \leq n)$.

b) $(t_i) * (c), (c) * (t_i)$; here, it is assumed that c is a fuzzy set over D_i.

2) **Logical Connectives and Quantifiers:** The logical connectives („AND", „OR", and „NOT") are used for tuple fuzzy WFF's.

Also, quantifiers („for all" and „there exists") are used for tuple fuzzy WFF's.

3) **Others:** Other definitions concerning tuple fuzzy WFF's are the same as in the tuple relational calculus.

Thus tuples in any relation $R(t_1, \ldots, t_n)$ that satisfy the formula $(t \mid f(t))$ form a set of Cartesian products of fuzzy sets.

It should be considered further whether or not to include fuzzy comparison operators $*$ expressed by fuzzy relations such as „much greater than", „is close to", „is similar to", and „is relevant to".

Query evaluation. Queries expressed in the tuple fuzzy calculus are evaluated by two steps as follows.

(Step1) Selecting resultant tuples: Consider that the query $(t \mid f(t))$ is issued to a relation $R(t_1, \ldots, t_n)$. Resultant tuples are those $r \in R(t_1, \ldots, t_n)$ each of which satisfies the formula $f(r)$.

(Step2) Calculating truth values of resultant tuples: Let any resultant tuple r be a projection of $t \in R(t_1, \ldots, t_n)$ onto the components $k_1, \ldots, k_j, \ldots, k_m$ where $1 \leq m \leq n$, $1 \leq k_1, \ldots, k_j, \ldots, k_m \leq n$. Then the truth value $T(r)$ is defined as a projection of $T(t)$ onto the components $k_1, \ldots, k_j, \ldots,$

k_m: $T(r) = Max.T(t)$, where the maximum is taken over those components $t_k (1 \leq k \leq n)$, such that $t_k \neq t_{k_j}$.

4. QUERY BY FUZZY ALGEBRA

Fuzzy algebra. A fuzzy algebra (query language) is constructed as an enhancement of the relational algebra. Fundamental fuzzy algebraic operations are union, set difference, Cartesian product, projection, and selection, which are defined as follows.

1) **Union:** Let R and S denote any relations in the fuzzy database. The union of R and S is a set of tuples that belongs to R or S. The union is equal to that in set theory.

Any resultant tuple t by the union of R and S inherits the truth value $T(t)$ from its original tuple in R or S.

2) **Set difference:** The difference $R - S$ of R from S is a set of tuples, each of which belongs to R and does not belong to S. The difference is equal to that in set theory.

Any resultant tuple t by the set difference $R - S$ inherits the truth value $T(t)$ from its original tuple in R.

3) **Cartesian product:** The Cartesian product $R \times S$ of R and S is a set of tuples, $\{r, s) \mid r$: tuple in R, s : tuple in $S\}$. The Cartesian product is equal to that in set theory.

The truth value $T(t)$ of the resultant type $t = (r, s)$ by the Cartesian product $R \times S$ is the minimum of $T(r)$ and $T(s)$, where $T(r)$ and $T(s)$ are truth values of r and s, respectively.

4) **Projection:** The projection $\text{Proj}(k_1, \ldots, k_j, \ldots, k_m)(R)$ of R onto the k_jth attributes is a set of tuples of the k_jth attribute values. The projection is equal to that in set theory.

Let r denote any resultant tuple of the projection $\text{Proj}(i_1, \ldots, i_j, \ldots, i_m)(R)$ of $t \in R$. Then the truth value $T(r)$ is the maximum of $T(t)$ taken over those components T_k, such that $t_k \neq t_{k_j}$.

5) **Selection:** Let G denote a fuzzy WFF involving the following constituents:

i) operands that are constant fuzzy sets and attribute item numbers of the relation R;

ii) the fuzzy set comparison operators $*$ (equal, not equal, proper inclusion, inclusion);

iii) logical conectives „AND", „OR", and „NOT".

The selection $\text{Sel}_G(R)$ of the relation R is a set of tuples t in R each of which satisfies the fuzzy WFF G when any occurrences of the number i in G are replaced by the ith component of r in R.

When any resultant tuple r is made by the selection $\mathrm{Sel}_G(R)$, $t \in R$ inherits the truth value $T(t)$ from the original tuple t in R: $T(r) = T(t)$.

Some additional fuzzy algebraic operations such as intersection, quotient, θ-join, and natural join are defined as combinations of the fundamental fuzzy algebraic operations defined previously in the same way as in the relational algebra. For example, the θ-join and the natural join are defined as follows.

6) **θ-join:** The θ-join of R and S is defined as a combination of two fundamental fuzzy algebraic operations: the Cartesian product and the selection where θ is enhanced to a fuzzy comparison operator $*$ (equal, not equal, proper inclusion, inclusion). Truth values of resultant tuples by the θ-join are calculated as those of combinations of the two fundamental operations.

7) **Natural join:** The natural join of R and S is defined as a combination of three fundamental fuzzy algebraic operations: the Cartesian product, the selection, and the projection. Truth values of resultant tuples by the natural join are calculated as those of combinations of the three fundamental fuzzy algebraic operations.

Query evaluation. Any query by the fuzzy algebra is expressed as a combination of the fundamental fuzzy algebraic operations. Thus the resultant tuples r and their truth values $T(r)$ by this query are obtained as combinations of its constituent fundamental fuzzy algebraic operations.

Duplicate removal schemes and return methods of resultant tuples to users are the same as described in the fuzzy calculus.

5. RELATIONAL COMPLETENESS THEOREM FOR FUZZY DATABASE QUERY LANGUAGES

The relational database theory establishes the relational completeness theorem such that the relational calculus is equivalent in expressive power to the relational algebra [2]. A similar theorem in the fuzzy database is given.

Theorem: *The following three fuzzy database query languages have the same expressive power*:

1) *tuple fuzzy calculus;*

2) *domain fuzzy calculus;*

3) *fuzzy algebra.*

Proof: The fundamental idea of the proof of this theorem is given by Ullman [**?**, pp. 114-122]; it presents the proof of the relational completeness theorem for the relational database query languages. Ullman's proof techniques consist of the following three reduction techniques:

i) reduction of the relational algebra to the tuple relational calculus;

ii) reduction of the tuple relational calculus to the domain relational calculus;

iii) reduction of the domain relational calculus to the relational algebra.

The reduction technique ii) is just the transformation between variable expressions, and thus is not influenced by the enhancements of the fuzzy database query languages. Therefore, it should be proved here that the reduction techniques i) and iii) can also be extended to cover the enhancements of the fuzzy database query languages.

There are two essential enhancements in the fuzzy database query languages from the relational database.

1) The fuzzy database allows fuzzy sets as attribute values; the fuzzy comparison operators $*$ (equal, not equal, proper inclusion, inclusion) are used in the fuzzy database query languages instead of the arithmetic comparison operators ($=, \neq, <, >, \leq, \geq$) used in the relational database query languages.

2) The fuzzy database allows fuzzy sets as truth values $T(t), t \in R$; truth values $T(r)$ of resultant tuples r are inherited from $T(t)$ of original tuples $t \in R$, or calculated as combinations of Cartesian products or projection of $T(t)$ of original tuples $T \in R$.

The enhancements 1) is easily incorporated into the reduction techniques i) and iii) by replacing the arithmetic comparison operators with the fuzzy comparison operators.

Next, consider the enhancement 2). Remember that the truth value $T(t)$ of the tuple t is defined just depending on the fuzzy set and fuzzy set operations that have been established in the fuzzy theory; calculations of $T(r)$ are made independently of any of the definitions of the fuzzy database query languages. Thus the reduction techniques i) and iii) can be extended to incorporate the enhancement 2). This completes the proof.

6. CONCLUDING REMARKS

This paper presents two fuzzy database query languages (fuzzy relational calculus and fuzzy relational algebra) based on the relational database query languages. In addition, it proves the relational completeness theorem such that both the languages are equivalent in expressive power to each other. As in the case of the relational database, this relational completeness theorem in the fuzzy database is expected to provide a criterion for the minimum fuzzy database query capability that must be implemented in any reasonable real fuzzy database query languages.

Further interesting studies are induced for extending the existing real relational database query languages, such as the international standard database language SQL. Such an example is the fuzzy query language developed by Rasmussen and Yager in 1997 [5], named *SummarySQL*. The purpose of the

language is to let linguistic summaries be a part of a fuzzy query. In SummarySQL, we can evaluate a linguistic summary to find the measure of validity, but a linguistic summary can also be used as a predicate in a fuzzy query. When a predicate is evaluated in a query it will, like a linguistic summary, take a value in the unit interval; for this reason, SummarySQL will treat a linguistic summary as a fuzzy predicate.

Linguistic summaries are related to the class of operators called aggregate functions in the SQL, such as the average (AVG) and count (COUNT) functions. But, different from the usual aggregate functions, linguistic summaries always take truth values in the unit interval. The prototype interpreter built for the SummarySQL can be used to evaluate the following query examples.

A SummarySQL statement has the form: select attributelist from tablelist where conditions

The result from a query is a table where every tuple is associated with a truth value. The *from* clause defines the join-table we access through the query and is the result of joining the tables in the tablelist. A table in the tablelist can also be a subquery. The *select* clause defines a projection on the join-table by an attributelist. The conditions in the *where* clause represent a fuzzy expression over the attributes from the join-table. The predicates in the conditions can be „summaries" or have the form „attribute IS fuzzyterm" and can be conjuncted (AND), disjuncted (OR) or negated (NOT). The fuzzy expression is evaluated for each tuple in the join-table and the result is assigned to the associated truth value (μ).

A statement for summary has the form: summary quantifier from tablelist where conditions where the quantifier is a fuzzy quantifier and the *from* clause and the *where* clause are the same as in the fuzzy query. Compared to a summary of the form „Q objects in FDB are S", Q is equal to the quantifier in the *summary* clause and the summarizer S is equal to the fuzzy conditions defined in the *where* clause. The join-table defined in the *from* clause is equal to the fuzzy table FDB. As we mentioned earlier, a table in the tablelist can be a subquery, and the result of a subquery could be a fuzzy subset FDB. If we look at the fuzzy subset FDB as the subpopulation „the R objects in DB", we have the summary „Q R objects in DB are S".

References

[1] Raju, K. V. S. V. N., Majumdar, A. K., Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems, ACM Trans. Database Syst., **13**, 2 (1988), 129-166.

[2] Ullman, J. D., Principles of database systems, MD: Computer Science, Rockville, 1980.

[3] Umano, M., Relational algebra in fuzzy database, IEICE Tech. Rep. DE86-4, **86**, 192 (1986), 1-8.

[4] Zemankova-Leech, M., Kandel, A., Fuzzy relational databases - A key to expert systems, Verlag TUV Rheiland Gmbh, 1980.

[5] Rasmussen, D. and Yager, R. R., SummarySQL - A fuzzy tool for data mining, Intelligent Data Analysis, 1997.