On Nearness Measures in Fuzzy Relational Data Models

Elke A. Rundensteiner,* Lois W. Hawkes, and Wyllis Bandler

Department of Computer Science
Florida State University
Tallahassee, Florida

ABSTRACT

It has been widely recognized that the imprecision and incompleteness inherent in real-world data suggests a fuzzy extension for information management systems. Various attempts to enhance these systems by fuzzy extensions can be found in the literature. Varying approaches concerning the fuzzification of the concept of a relation are possible, two of which are referred to in this article as the generalized fuzzy approach and the fuzzy-set relation approach. In these enhanced models, items can no longer be retrieved by merely using equality-check operations between constants; instead, operations based on some kind of nearness measures have to be developed. In fact, these models require such a nearness measure to be established for each domain for the evaluation of queries made upon them. An investigation of proposed nearness measures, often fuzzy equivalences, is conducted. The unnaturalness and impracticality of these measures leads to the development of a new measure: the resemblance relation, which is defined to be a fuzzified version of a tolerance relation. Various aspects of this relation are analyzed and discussed. It is also shown how the resemblance relation can be used to reduce redundancy in fuzzy relational database systems.

KEYWORDS: ***fuzzy relational data model, fuzzy relations, nearness measure, tolerance relation, resemblance relation***

1. INTRODUCTION

The relational database model developed by Codd [1] is one of the most extensively studied models of an information management system and has found

* Currently at Information and Computer Science Department, University of California, Irvine, California 92701.

Address correspondence to Elke A. Rundensteiner at address given in footnote above.

International Journal of Approximate Reasoning 1989; 3:267–298

655 Avenue of the Americas, New York, NY 10010 0898-613X/89/$3.50

widespread use in industry. Unfortunately, most of the available implementations of relational database systems model the real world in a hard and deterministic manner and allow for only exact retrieval; in other words, the imprecision, vagueness, and incompleteness inherent in the real world have been totally ignored.

The concept of fuzzy sets proposed by Zadeh [2] has been recognized as a potential mathematical tool and a logical framework for uncertainty management. The emphasis is on explicit representation of fuzziness in a system rather than on trying to eliminate or disguise it by some clever trick or to simply ignore it and oversimplify the modeling process unrealistically. Thus, the claim can be made that stored "approximate information" is in fact more precise than "traditional crisp data," since it models more realistically the present state of knowledge. If some crisp decisions need to be made, then this can still be achieved in these new models—now by controlling the effect of the fuzzy data on the decisions in an explicit and conscious manner.

There are two general approaches to fuzzy extensions of such systems. The first considers the problem of approximate retrieval on precisely known values—the retrieval of items that are *sufficiently close* to those requested. This is an approach of practical importance for the time being, since it could be an "add-on" to existing conventional systems (Kacprzyk and Ziolkowski [3]). The second approach addresses the handling of values that are not precisely known, which also implicitly entails the problem of fuzzy retrieval (Prade and Testemale [4]). Since crisp data are simply special cases of fuzzy data, the second approach is the more general one and thus should gain more popularity in the future. This second route is therefore taken here; we concentrate primarily on the representation issue underlying this approach.

Various attempts at enhancing the relational database models by fuzzy extensions can be found in the literature (Prade and Testemale [4], Buckles et al. [5], Rundensteiner et al. [6], Zemankova and Kandel [7]). These promise to capture real-world data more realistically and hence broaden the area of possible applications for these models. In these enhanced models, items can no longer be retrieved by merely using equality-check operations between constants; operations based on some kind of *similarity measure* have to be developed. The need for the development of less strict nearness measures than the equality relation for a meaningful evaluation of queries in the context of fuzzy data has been recognized. An investigation of proposed nearness measures, however, reveals that not enough attention has been devoted to the development of an adequate one in the context of fuzzy relational databases. Since the concept of such a measure is essential for query evaluation purposes (Rundensteiner et al. [8]), it is worthwhile to study the desired properties of this measure. The unnaturalness and impracticality of existing measures leads to the proposal of a new measure, the *resemblance relation*, which appears to be an adequate tool for the evaluation of approximate queries based on fuzzy data.

The resemblance relation, which is defined as the fuzzified version of a tolerance relation (Schreider [9]), a reflexive and symmetric relation, is considered an enrichment to the theory of fuzzy sets and relations in general. In spite of its naturalness the tolerance relation has not been recognized outside of abstract modern algebra. This stands in sharp contrast to the situation of the equivalence relation, which has its place in the relation theory but which also is widely applied in various areas.

Consequently, the attempt is made here to introduce the concept of resemblance into fuzzy set theory. Various concepts related to the resemblance relation are being developed, and their relationship with graph theoretical ideas is pointed out.

2. THE CLASSICAL RELATIONAL REPRESENTATION

This section introduces the basic concepts of the classical relational database model (Codd [1]) in order to clarify our terminology.

A relational database consists of a set of attributes A_i, a set of domains U_i, which are sets of values upon which the attributes are defined, and a set of relations R_i.

DEFINITION 1 *A relation on a set of attributes is defined as a subset of the Cartesian product of the respective domains. More precisely, a relation R on the set of n attributes* $\{A_1, \cdots, A_n\}$ *is defined on the respective domains* $U_1, U_2, \cdots, U_n$ *if it is a subset of the Cartesian product* $U_1 \times U_2 \times \cdots \times U_n$. *The Cartesian product is the set of all n-tuples* $\langle u_1, \cdots, u_n \rangle$ *such that* $u_i \in U_i$ *for all i. The relation R is then defined to have degree or arity n.*

The set of attributes $\{A_1, \cdots, A_n\}$ upon which the relation R is defined is called a relation schema $R(A_1, \cdots, A_n)$, or R.

It helps to view a relation as a table in which each row is a tuple and each column corresponds to an attribute. Each column, or attribute, is given a distinct name within a relation. All data items in a column consist of values from the same domain. Consequently, each tuple within a relation has the same set of attributes. All rows, or tuples, are distinct; duplicates are not allowed. The row and columns can be ordered in any sequence at any time without affecting the information content involved. We can view tuples as mappings from attribute names to values in the domains of the attributes. Each relation consists of a relation name, a nonempty set of attributes with corresponding domains (the relation schema), a key, and a (possibly empty) set of tuples (Codd [1]).

Once a database is designed, a specialized language is needed to interrogate and manipulate the content of the database. Languages for expressing queries in the relational model are called data manipulation languages (DMLs). A common DML is the relational algebra proposed by Codd [1]. Queries often refer to data

residing in several distinct relations, but there are no explicit and/or structural links such as pointers between these relations. The connections among these relations are implicit; asociations between relations are established solely by common values. Thus, these DMLs need to exploit this fact by matching data values in order to perform operations on data from more than one relation.

3. FUZZY EXTENSIONS TO THE RELATIONAL REPRESENTATION

This section introduces fuzzy set theory as proposed by Zadeh [2] and indicates how it can be applied as enhancement to the relational representation.

DEFINITION 2 *Let U be a universe of discourse. F is a* fuzzy subset *of U if there is a membership function* $\mu_F: U \rightarrow [0, 1]$, *which associates with each element* $u \in U$ *a membership value* $\mu_F(u)$ *in the interval [0, 1]. The membership value* $\mu_F(u)$ *for each* $u \in U$ *represents the grade of membership of the element u in the fuzzy set F.*

Zadeh [2] proposed the following notation for a fuzzy set F:

$$F = \{\mu_F(u)/u \mid u \in U\}$$

Within the framework of fuzzy set theory, the concept of a fuzzy relation has been defined. A fuzzy relation has been described as a generalization of a fuzzy set, that is, as a fuzzy subset of the Cartesian product of some universe of discourse.

DEFINITION 3 *Let U be the Cartesian product of n universes of discourse* $U_1, \cdots, U_n$, *that is,* $U = U_1 \times U_2 \times \cdots \times U_n$. *Then* an *n*-ary fuzzy relation R in U *is a relation that is characterized by a n-variate membership function ranging over U, that is,*

$$\mu_R : U \rightarrow [0, 1]$$

A close connection between fuzzy sets and possibility theory can be established (Prade and Testemale [4]). The grade of membership $\mu_F(u)$ of u in the fuzzy set F may be interpreted as the degree of compatibility of u with the concept represented by F or as the degree of possibility of u given F (Zadeh [10]). This is stated more precisely in the following definition.

DEFINITION 4 *Let F be a fuzzy subset of U characterized by a membership function* μ_F. *Let X be a variable that takes values in a universe U. Then the proposition "X is F" induces a possibility distribution* Π_x *that is equal to F, that is,*

$$\Pi_x = F$$

This definition states that the possibility (relative to the fuzzy set F) that the value u may be assigned to X is equal to the membership of u in the fuzzy set F. This can be formulated as Poss $\{X = u\} = \mu_F(u)$ for all $u \in U$. Thus, a possibility distribution over a set U can be used to define a corresponding fuzzy set of U, or vice versa.

Now, since the traditional relational database model is based on the foundation of set and relation theory (both crisp), the proposal has been made to adopt the concept of a fuzzy relation from fuzzy set theory as given in Definition 4 as the fuzzified version of the concept of a relation for the database model. This is a valid approach and has been proposed by several researchers (Prade and Testemale [4], Zadeh [10], Zvieli [11]). However, there is a different approach for a possible fuzzy extension of the relational representation that is also based on the sound theoretical foundation provided by Codd's relational database model [1] and theories of fuzzy sets and possibilities (Zadeh [2]).

Recall that in general the relational database model consists of a set of relations comprised of tuples t_i for $i = 1, \cdots, m$ of the form $\langle u_{i1}, u_{i2}, \cdots, u_{in} \rangle$, where each of these data values u_{ij} is selected from a given fixed domain U_j. Thus, in the traditional data model, each of these data values u_{ij} is a *single* value from the respective domain.

It is proposed to extend the set of possible domains to include domains such as membership function values. Then, the resulting enhanced relational representation allows data values to take different forms as well as being constants (Dubois and Prade [12]). The data values for the fuzzy relational representation are extended to be

1. A single scalar (e.g., Aptitude = *good*)
2. A single number (e.g., Age = 24)
3. A set of scalars (e.g., Aptitude = {*average, good*})
4. A set of numbers (e.g., Age = {20, 21, 25})
5. A possibilistic distribution of scalar domain values (e.g., Aptitude = {0.4/*average,* 0.7/*good*})
6. A possibilistic distribution of number domain values (e.g., Age = {0.4/23, 1.0/24, 0.8/25})
7. A real number from [0, 1] (e.g., Heavy = 0.9)
8. A designated null value (e.g., Age = unknown)

Note that here the domain of the attribute Aptitude is also called Aptitude and is defined to be the set {*very-good, good, average, bad*}, and the domains of the attributes Age and Heavy are, respectively, the positive integers and the unit interval.

It can easily be seen that all eight of these possible data value types can be described by some form of a possibility distribution (Zemankova and Kandel [7]). The first two cases correspond obviously to the crisp conventional form; for example, for Aptitude we have {1.0/*good*}. The third and fourth can be viewed as representing uncertainty in the data. An example of uncertain

information is a proposition such as "Joe is 20 or 21 years old." This statement implies that Joe's Age is either 20 or 21 and not both at the same time and that it is not possible that his Age is other than 20 or 21. Thus the data value {20, 21} will be used to represent Joe's Age, which corresponds to the possibility distribution 1.0/20 + 1.0/21. Note that the larger the set of elements in a data value of type 3 or 4 is, the less precise our knowledge is. Thus, if all values of the universe have a possibility of 1, then this signifies a total "unknown." This is, in general, not practical, so a special symbol "unknown" is being used instead. This proposal of eight different data types corresponds to the approach of Zemankova and Kandel [7, 13]. Most other approaches in the literature restrict their models to a subset of the above. Buckles and Petry [4] and Oezsoyoglu et al. [15] allow only data types 1–4, while Umano [16] permits 1–4, 7, and 8. Many other use on 7 in addition to 1 and 2 (e.g., Zvieli [11] and Raju and Majumdar [17]. Now, the following can be defined:

DEFINITION 5 *Let A_i, for i from 1 to n, be attributes defined on the domain sets U_i, respectively. Let $\pi(A_i)$, for i from 1 to n, stand for possibilistic distributions on U_i (any of the eight possible data types named previously). A* fuzzy relation *defined on these n sets U_i is a set of n-tuples $t_j = \langle \pi_j(A_1), \pi_j(A_2), \cdots, \pi_j(A_n) \rangle$.*

As in the classical relational database theory, it helps to view a relation (fuzzy or not) as a table in which each row corresponds to a tuple and each column to an attribute. The relation schemes are the same for both proposd fuzzy extensions. The tableau form, however, demonstrates the difference between two models of fuzzy extensions proposed here.

The first approach proposes the fuzzy extension of the concept of a relation based on the fuzzy set concept (fs-type). This proposal is first considered in its simplest form, the unary relation. A fuzzy subset F in the universe of discourse U, as given in Definition 2, is, in fact, a unary fuzzy relation R on the one attribute U, according to the fuzzy relation concept given in Definition 3. This unary relation R is characterized by the membership function $\mu_F: U \rightarrow [0, 1]$. Thus if $U = \{u_1, u_2, \cdots, u_n\}$, then a fuzzy set F on U can be described by the following:

$$F = \{\mu_F(u_1)/u_1, \mu_F(u_2)/u_2, \cdots, \mu_F(u_n)/u_n\}$$

Consequently, the unary relation R—that is, the fuzzy set F—can be captured by a tableau with two columns of the form shown in Figure 1.

At first, it may seem peculiar that this *unary* relation R actually has *two* columns (attributes). This is easily explained, though, with the concept of fuzziness, which introduces an extra nonconventional attribute. The last attribute is a special column, which in the crisp case can be omitted since then this information is provided implicitly (i.e., if a tuple appears in the relation, this corresponds to the membership value of this tuple being 1; and if a tuple does not

U	F
u_1	$\mu_F(u_1)$
u_2	$\mu_F(u_2)$
...	...
u_n	$\mu_F(u_n)$

Figure 1. A Unary Relation R (or a Fuzzy Set F).

appear in the relation, then its membership is 0). The above can now easily be extended to the n-ary case. Recall that a n-ary fuzzy relation R over attributes $A_1, A_2, \cdots, A_n$ as defined in Definition 4 corresponds to a fuzzy subset of $U_1 \times U_2 \times \cdots \times U_n$, where U_i is the domain of A_i for all i. A tuple t_j of the relation R can thus be expressed as

$$t_j = \langle u_{j1}, u_{j2}, \cdots, u_{jn}, \mu_R(u_{j1}, u_{j2}, \cdots, u_{jn}) \rangle$$

Consequently, the relation R is captured by a tableau of the form shown in Figure 2. Various semantics have been associated with these tuple membership values $\mu_R(\cdots)$ in the literature (Prade and Testemale [4], Buckles et al. [5], Zemankova and Kandel [7], Buckles and Petry [14], Anvari and Rose [18], Rundensteiner [19]). The previous paragraph has assumed that the membership value represents the degree to which the tuple belongs in the relation. Other possible semantics are the degree of accuracy of the represented information or the degree to which functional dependency holds.

Next we show how the second proposal of a fuzzy extension of the concept of a relation can be described in tableau format. Let $U_1, U_2, \cdots, U_n$ again be the universes of discourse upon which the fuzzy relation R is defined. Let $\pi(A_i)$ be the possibility distribution of the attribute A_i defined on the universe U_i for all i. Then a tuple t_j of R has the form $\langle \pi_j(A_1), \pi_j(A_2), \cdots, \pi_j(A_n) \rangle$. The relation R

U_1	U_2	...	U_n	μ_R
u_{11}	u_{12}	...	u_{1n}	$\mu_R(u_{11}, u_{12}, ..., u_{1n})$
...	...	...	...	...
u_{j1}	u_{j2}	...	u_{jn}	$\mu_R(u_{j1}, u_{j2}, ..., u_{jn})$
...	...	...	...	...

Figure 2. An n-ary Fuzzy Relation R in Tableau Format (fs-type).

A_1	A_2	...	A_n
$\pi_1(A_1)$	$\pi_1(A_2)$	...	$\pi_1(A_n)$
$\pi_2(A_1)$	$\pi_2(A_2)$	...	$\pi_2(A_n)$
...	...	...	...
$\pi_j(A_1)$	$\pi_j(A_2)$	...	$\pi_j(A_n)$
...	...	...	...

Figure 3. An n-ary Fuzzy Relation R in Tableau Format (**gf**-type).

can thus be represented by a tableau with n columns as shown in Figure 3 (gf-type).

Both types of fuzzy extensions (shown in Figures 2 and 3) are valid forms of fuzzifying the relational representation, and both have been used by researchers of fuzzy set theory, although usually without acknowledging the existence of other approaches of fuzzifying the data model. They both have their justification and are suitable for the representation of certain types of applications. We proposed that the first type be referred to as "the fuzzy-set relation" (*fs-relation*) because of its origin in fuzzy set theory and the second one as the "generalized fuzzy relation" (*gf-relation*) since it captures a variety of different possible data value types. An example of the first of these two types follows.

EXAMPLE 1. Suppose we want to capture the set of "intelligent" students in a university by a fuzzy relation of *fs type*. Thus, this fuzzy relation could be identified by attributes identifying the student, such as Name and SSN, and by attributes that help determine their level of intelligence, such as GRE score, GPA, and perhaps Status in school. The relation in Figure 4 represents a description of a fuzzy set of "intelligent" students, where the last attribute characterizes the degree to which the respective student is actually considered intelligent. (Other semantics could have been chosen for this tuple membership value.) The emphasis in this relation is on the $(n + 1)$th attribute, the membership value of the particular student in the fuzzy set of intelligent students. If this is the only information that is supposed to be conveyed, then a relation just identifying the student and representing his grade of membership in this fuzzy set would be sufficient. (See Figure 5.)

Next, an example of a relation of the *gf type* of fuzzy extension is given. This relation again is designed to store the intelligent students in a university. It is an n-ary relation, and thus it requires only n columns.

Name	SSN	GRE	GPA	Status	F
Jack	123456789	760	3.5	sr	0.8
Frank	112233445	800	3.99	gr	1.0
Dave	999933445	600	3.01	fm	0.7

Figure 4. Intelligent Student Relation (**fs**-type).

EXAMPLE 2. The information about the "intelligent" students of a university, measured here as "Aptitude," is stored in a relation of type *gf*. All characteristics known about these students are similar to those in the previous example. The values for any of these attributes might not be precisely known, and thus this lack of certainty has to expressed by allowing possibility distributions over the respective domains. (See Figure 6.)

The relation of *fs type* requires that all values but the last one in the relation be crisp values. The fuzziness is indeed an evaluation of how the values of the first n attributes determine the $(n + 1)$th characteristic, the tuple membership value. So, this $(n + 1)$th value refers back to the set of all previous values of the tuple within the context of certain semantics, for example, the concept of intelligence in Example 1. Viewed from this perspective, attributes that describe the object but do not contribute to the decision of whether the object is intelligent or not appear as extraneous information. An example of unnecessary information in Figure 4 would be the SSN attribute. Hence, in certain contexts a relation presented in Figure 5 may be sufficient to capture the essential information of Figure 4. Clearly, this $(n + 1)$th attribute has to be treated differently than the rest. Allowing the type 7 membership values as a possible data type in the *gf-relation* permits us to also capture fuzzy sets. This implies that the fs type is a specialized subset of the gf type of relation.

Once the extension of the representation has been properly defined, the question of retrieval has to be addressed. The classical relational algebra has to be extended to apply to these enhanced forms of representation.

Operations for the fs type of fuzzy relation have to take care of the membership value, the $(n + 1)$th attribute of the relation, since it is the only characteristic that distinguishes the fs-relation from crisp relations. Thus, Zadeh [2] extends the concept of the projection and join operation of the classical relational algebra by simply specifying how the membership value of the resulting tuple is to be determined from the two initial tuples.

This approach cannot, of course, be taken for gf-relations, since its data items may have the form of any of the eight different data types previously discussed. Consequently, no simple equality check between data items can be performed. The relational algebra operations defined for the gf-type relations have to take

Name	Intelligent Student
Jack	0.8
Frank	1.0
Dave	0.7

Figure 5. Intelligent Student Relation (**fs**-type).

into consideration that any of the n attributes of a relation could be a fuzzy set. Thus, issues such as the comparison of two possibility distributions and a measure of their similarity have to be studied.

A relational database has no explicit links between its pieces of information; the relations, and therefore links denoting implicit relationships, are established during the evaluation phase of queries by matching data values. In the crisp model, this matching of values is determined by the identity function, which denotes values to be matchable if and only if they are *identical*. This identity function is also needed for the removal of identical, that is, redundant, tuples. Consequently, the extension of data values from single discrete values of type 1 or 2 to sets of values or possibility distributions over the domain implies that this measure of a perfect matching of data values has to be relaxed to a *measure of nearness*. Hence, associated with each domain set in the gf-type approach is a nearness relationship that is used to perform operations that entail the comparison of two values. This relationship can also be used to identify and remove redundant tuples, since it determines for two elements of a domain the degree to which they are considered to resemble each other. It is interesting to note here that these nearness measures are in general expressed as fs-relations.

The rest of this paper will concentrate on an investigation of these nearness measures, which is a necessary basis for a sound defintion of an extended relational algebra.

4. THE FUZZY RELATIONAL REPRESENTATION MODEL

Considerations of the previous observations led to the first attempt at the representation of a conceptual model for a fuzzy relational representation (FRR). This model is presented in Figure 7.

Name	SSN	GRE	Age	Aptitude
Jack	123456789	{780, 800}	{23, 24, 25}	$\{0.4/good, 0.7/very-good\}$
Frank	112233445	790	{31, 32}	$\{0.9/very-good\}$

Figure 6. Intelligent Student Relation (**gf**-type).

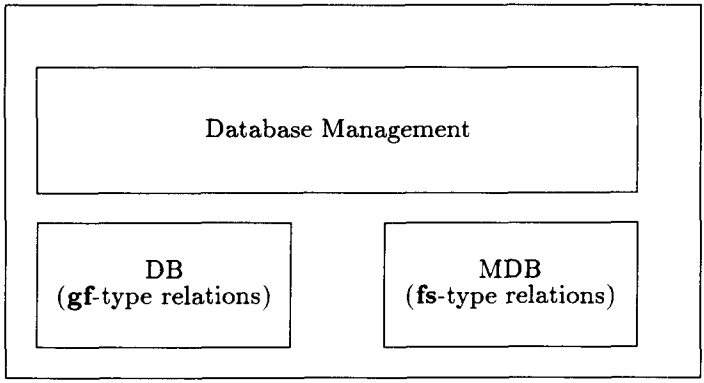


Figure 7. The FRR Data Model.

This model assumes a separation of its structure into two major data areas, the conventional database (DB) and the meta-database (MDB). The DB contains the actual data, such as the relations, the attributes, and the data values. Hence, the DB comprises most of what can be found in a crisp database. The MDB stores schema definitions and the like, as well as information that would not be found in a crisp database, such as definitions of fuzzy sets (e.g., OLD) and descriptions of fuzzy relations (e.g., nearness relations), which are needed for the interpretation and evaluation of queries. In short, the knowledge stored in the MDB helps to interpret the fuzzy information stored in the DB.

This model implies that the DB component should posses the power to capture all facets of data and associated certainties and uncertainties. A richness in representation is required. Hence, the relation schema employed in the DB is of the *gf* type. In contrast to this, the relationships to be represented in the MDB lend themselves to the *fs*-type format of a relation. For the most part, they are literally descriptions of fuzzy sets, and thus the fuzzy set type of relation is appropriate.

This model suggests that any definition of a query language ought to take into account whether the information should reside in the DB or the MDB. Research in this area is in progress. Existing proposals of fuzzy query languages have to be evaluated, and ideas from the fs- and the gf-type approaches to fuzzifying relations must be combined to form one coherent solution for a sound fuzzy relational algebra for this FRR data model.

5. DISCUSSION OF VARIOUS NEARNESS MEASURES

Several approaches to fuzzy databases [Buckles et al. [5], Buckles and Petry [14], Anvari and Rose [18]) can be found in the literature that are characterized by allowing data values to be *elements of the power set* of the strongly typed

domain base sets instead of just single values, that is, type 3 and 4. In our terminology, they could be classified as subsets of the gf-relation approach, allowing types 1–4 instead of all eight data types. In general, these approaches define nearness measures for discrete, finite domain sets to be a similarity relationship in the following sense.

DEFINITION 6 *Let u1, u2, u3* $\in U$. *A* similarity relation *s is a fuzzy binary relation on a discrete finite domain U that maps every pair of elements in the domain onto the unit interval* [0, 1]*:* $s: U \times U \rightarrow [0, 1]$, *such that the following properties hold for s:*

1. *Reflexive:* $\mu_s(u1, u1) = 1\ (\forall u1)$
2. *Symmetric:* $\mu_s(u1, u2) = \mu_s(u2, u1)\ (\forall u1, u2)$
3. *Transitive:* $\mu_s(u1, u3) \geq max_{\forall u2 \in U}\ \{min\ [\mu_s(u1, u2), \mu_s(u2, u3)]\}\ (\forall u1, u3)$

The $\mu_s(u1, u2)$ denotes the strength of the relationship s between $u1$ and $u2$, or in the case of the similarity relation, the similarity between $u1$ and $u2$. The $\mu_s(u1, u2)$ can be referred to as $s(u1, u2)$. These similarity relations can be represented in the natural data structure of the FRR model, in a relation of fs type. Note that these relations are part of the MDB of the model in Figure 7. A similarity relation for the discrete domain U is represented by 3-ary relations called SIM-U, where the first two attributes denote all possible combinations of value pairs from U, and the third attribute expresses the similarity value between the corresponding pairs. Since the reflexivity and symmetry of these special relations are assumed, all reflexive and symmetric pairs of values are redundant and hence are omitted.

An example of such a similarity relation is given in Figure 8. If the similarity relation given for the Aptitude domain in Figure 8 is observed more closely, most people would intuitively object to the chosen similarity values. This is so because humans have connotations associated with the different elements of the domain Aptitude. To consider the characteristics of 'good' and 'bad' to be

Aptitude	S-Aptitude	SIM
very-good	good	0.5
very-good	average	0.1
good	average	0.1
good	bad	0.1
average	bad	0.3

Figure 8. A Similarity Relation (fs-type) on the Domain Aptitude.

similar to the degree 0.1 is acceptable, but then it seems a natural consequence to demand that the similarity of the pair 'good' and 'average' should have a higher value, that is, they are more similar. Hence, one might propose to change this similarity relation to produce a better, more natural and intuitive interpretation of the similarity associated between elements of the domain Aptitude by setting the similarity of 'good' and 'average' to 0.2, that is, s(good, average) > s(good, bad) is achieved. This creates an inconsistency, because the third property of a similarity relation, the transitivity property, is violated, as demonstrated by the following example.

EXAMPLE 3. Given the similarity relation on the domain Aptitude as in Figure 8 except for $s_{Aptitude}(good, average) = 0.2$. Then, the following demonstrates that the value of similarity between good and bad is also enforced to be altered due to the transitivity property:

$$\begin{aligned} s_{Aptitude}(good, bad) &\geq \max_{y \in Aptitude}\{\min[s_{Aptitude}(good, y), s_{Aptitude}(y, bad)]\} \\ &\geq \min[s_{Aptitude}(good, average), s_{Aptitude}(average, bad)] \\ &= \min(0.2, 0.3) \\ &= 0.2 \end{aligned}$$

But $s_{Aptitude}(good, bad) = 0.1$.

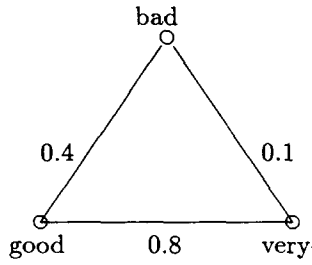
Note that the transitivity property forces the similarity value of the *(good, bad)* pair to increase in order to be able to increase the similarity of the *(good, average)* pair. This clearly contradicts human intuition, since it does not allow the database user to distinguish between the similarities of the two respective value pairs. It does not seem appropriate to enforce certain properties if they are not natural.

Recently, Potoczny [20] analyzed the similarity measure as presented previously and developed the following characterization of it.

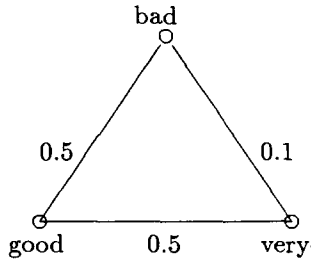
THEOREM 1 *The following conditions on U and s are equivalent:*

1. *s is a similarity relation as defined in Definition 7.*
2. *For any three values u1, u2, and u3 ∈ U, either*
   - *(a) the three similarity values s(u1, u2), s(u1, u3), and s(u2, u3) are equal, or*
   - *(b) two of the three values are equal and the third is larger.*

This theorem can easily be shown to be true because of the transitivity property of a similarity relation, but the conclusions to be drawn from it appear to be premature. Potoczny [20] uses this theorem to show that only a few similarity values of a similarity relation have to be known to determine all the others automatically. This is an important consideration, since it allows for the efficient storage of such a relation. But we see these results as an additional



(All values are unequal.)



(One value is smaller than the other two.)

Figure 9. Not Permitted Similarity Values.

support of our claim that this concept of a similarity relation is too rigid and unnatural, since, as Potoczny's theorem [20] points out, there is a strong restriction on the values of similarities between any three elements of the universe. For example, consider the universe $U = \{very\text{-}good, good, bad\}$. Most of the similarities between these three elements are already predetermined by the syntactic rules of the similarity definition. For instance, the setting of similarity values as in Figure 9 will not be allowed by Theorem 1. Both of them fulfill neither condition (2a) nor (2b) of Theorem 1.

The similarity relationship—being based on geometric models of similarity (Tversky [21])—is not appropriate for all possible domains (Zemankova and Kandel [13]). The max-min transitivity property can, for example, not be met by continuous domains. Hence, other approaches (Zvieli [11]) suggest a modified nearness measure for numeric domains; that is, they modify the similarity measure defined above by replacing the max-min transitive property by the product-transitive property. This nearness measure is then called a proximity relation.

DEFINITION 7 *Let p_j be a* proximity relation *defined for continuous domain sets. Given u1, u2, u3 $\in U_j$, p_j is defined to have the reflexivity and symmetry properties of a similarity relation, but another form of transitivity appropriate for number domains holds:*

$$3'.\ p_j(u1, u3) \geq \max_{\forall u2 \in U_j} [(p_j(u1, u2) \times p_j(u2, u3)]\ (\forall u1, u3)$$

Several examples of functions fulfilling this max-product transitivity can be found in the literature (Zemankova and Kandel [7], Zadeh [22]). These

functional descriptions for proximity relations have several favorable features; for example, they are easily described, they are definable for infinite domains, and they are also efficiently storable. A major advantage of the functional description of a nearness measure is that this description of a nearness measures does not require a complete relational format but only some parameters, such as the lower bound, in order to determine all its values. Note that again these types of measures leave little flexibility for the user to set up values totally to his desire. This loss of flexibility produced by the predefined form of the function is compensated for by the ease of use.

The use of the max-product transitivity is considered more appropriate than the use of the max-min transitivity (Zemankova and Kandel [7]). However, since neither can it be applied to all types of domains nor does it allow enough flexibility in defining similarity values according to intuition, it is not a completely satisfying solution.

6. THE CHARACTERISTICS OF A NEARNESS MEASURE

In the following, we investigate what properties a reasonable nearness measure needs to possess. First, reflexivity seems like a natural condition any nearness measure should have to fulfill, because any object should trivially be considered to be entirely similar (equal) to itself (Tversky [21]). Besides, it corresponds to the nearness measure implicitly chosen for the crisp data model, the identity relation. After all, a goal for this work is to guarantee that the classical relational database model is a special case of its extensions whenever possible in order to obtain a consistent fuzzy generalization. It is also reasonable to require that two objects either do or do not resemble each other, independently of the order in which they are considered. This property is explicated by the symmetry of the nearness measure. Since the transitivity of the nearness measure is by no means obligatory (Tversky [21]) and is often against human intuition, this transitivity property need not be enforced for a nearness measure. An even more convincing argument for the inappropriateness of the transitivity relationship in this context is the observation that distance measures come into conflict with the similarity measure inequality. This is demonstrated by Example 4.

EXAMPLE 4. Let the domain of interest be the real line R. A useful and generally accepted nearness measure on R is the distance between two points on R. Let $x, y \in R$. Let the distance d up to which points are considered to be close enough to each other to be called similar be 1 unit, that is, $d = 1$. Then the nearness measure of x, y is defined by

$$s(x, y) = \begin{cases} d - |x - y| & \text{if } |x - y| \leq d \\ 0 & \text{otherwise} \end{cases}$$

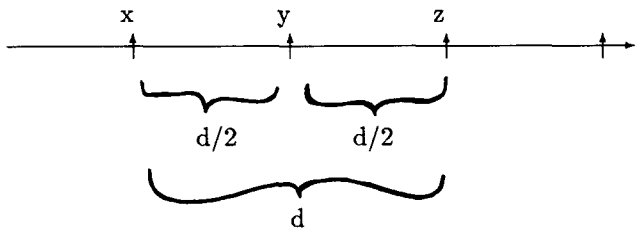


Figure 10. Points on the Real Line.

Without loss of generality, let $x, y \in R$ with $|x - y| = d/2$ and $x < y$. Note that there is an element z in R at a distance $d/2$ from y with $z > y$. This situation is depicted in Figure 10.

By the transitivity property of the similarity relation of Definition 7, it can now be concluded

$$\begin{aligned} s(x, z) &\geq \max_{w \in R} \{\min[s(x, w), s(w, z)]\} \\ &\geq \min[s(x, y), s(y, z)] \\ &= \min(d/2, d/2) \\ &= d/2 \end{aligned}$$

But note that the distance between x and z is $|z - x| = |y - z| + |y - x| = d$. This is a contradiction to the definition of the nearness measure, which demands two points to have a distance less than or equal to d in order to be considered similar at all. In addition, the transitivity property would produce the absurd conclusion that *all* elements of the real line are similar. Thus, we must use the above definition of a nearness measure using distance to determine the similarity of x and z even if the transitivity property is violated:

$$s(x, z) = d - |x - z| = d - d = 0$$

7. DISCUSSION OF THE SIMILARITY RELATION

The reasons for selecting the strong version of a nearness measure, the similarity relation, in various approaches to fuzzy extensions of the relational representation are discussed in this section.

Potoczny [20] claims that the transitivity property has to be enforced in order to avoid anomalous situations in a database. He uses the term anomaly in the sense that distinct tuples have the same interpretation, meaning that two distinct tuples can take on the same instantiation of values. The concept of interpretation of a tuple $t_i = \langle d_{i1}, d_{i2}, \cdots, d_{in} \rangle$ with $d_{ij} \subset U_j$ has been defined by Buckles and Petry [14] to be any assignment of values $A = \langle u_{i1}, u_{i2}, \cdots, u_{in} \rangle$ such that $u_{ij} \in$

d_{ij} for all j. For example, $\langle \{red\}, x \rangle$ as well as $\langle \{blue\}, x \rangle$ are interpretations of the tuple $\langle \{red, blue\}, x \rangle$. An example of an anomalous situation would be the two tuples $\langle \{red, blue\}, x \rangle$ and $\langle \{green, red\}, x \rangle$ since these two have the common interpretation $\langle \{red\}, x \rangle$. This situation, however, does not cause us too much concern. Instead, we propose that this problem should lead to an attempt to study the concepts of anomalies for a fuzzy database in order to generate an appropriate framework. It appears that new definitions for the concepts of anomaly have to be developed, since if we allow the capture of imprecise and vague information in a database, we cannot necessarily demand that no redundant information is to appear anywhere.

The question why the similarity relation has been used as a nearness measure is answered by analyzing Buckles and Petry's work [14]. They restrict the possible domains that the system is able to appropriately represent to either finite sets that fulfill the previously discussed form of similarity relationship or infinite sets with the use of the identity relations as a nearness measure. This indicates that the fuzzification process of a database is only half-heartedly executed by applying the crisp nearness measure, the identity relation, for numeric and/or infinite domains. This is not very satisfactory, apart from the fact that some finite domains do not naturally fulfill the characteristics of a similarity relation and hence have to be forced artificially to fit it. Furthermore, it can be observed that this approach is based nearly entirely on the three properties of the similarity relation. What is meant by the claim that the fuzzy data model is based on these similarity relations? The similarity relation is a fuzzy version of the concept of an *equivalence* relation, since an equivalence relation is a crisp binary relation characterized by the three properties reflexivity, symmetry, and transitivity. An equivalence relation is known to induce a partition on the domain it is defined upon. This means that the domain can be partitioned into classes (blocks) so that the union of these blocks of the partition forms a covering of the domain and every object of the domain is in exactly one block. This partition is defined in such a way that any two objects of the domain are in the same block of this partition if and only if they are equal—also called equivalent or interchangeable. Now, the similarity relation is a generalization of the notion of an equivalence relation with the same three properties, though, of course, with fuzzified versions of these properties.

Zadeh ([22], page 188) defines similarity classes induced by a similarity relation in accordance to (equivalence) classes in the case of an equivalence relation.

DEFINITION 8 *Let s be a similarity relation in the domain $U = \{u1, u2, \cdots, un\}$ characterized by $s(ui, uj)$. Each $ui \in U$ has associated a fuzzy set on U denoted by $[ui]$ whose membership values are $\mu_{[ui]}(uj) = s(ui, uj)$ for all $j \in \{1, \cdots, n\}$. These $[ui]$ are* the similarity classes *induced by the similarity relation s.*

An example is given next to demonstrate the form of these similarity classes defined by Zadeh [22].

EXAMPLE 5. Let s be the similarity relation given in Figure 8 on the domain Aptitude. Then there are four similarity classes induced by s according to Definition 8. They are

[very-good] = {*(very-good, 1.0), (good, 0.5), (average, 0.1), (bad, 0.1)*}
[good] = {*(very-good, 0.5), (good, 1.0), (average, 0.1), (bad, 0.1)*}
[average] = {*(very-good, 0.1), (good, 0.1), (average, 1.0), (bad, 0.3)*}
[bad] = {*(very-good, 0.1), (good, 0.1), (average, 0.3), (bad, 1.0)*}

These classes $[ui]$ are fuzzy sets over the domain U that are obtained by conditioning the similarity relation on ui. They do not express much about the similarity of their objects in general, or, in other words, they express a "one-sided" similarity from their *one* standard object to the rest of the objects in the block. So, for two objects uj, uk to be in the same block $[ui]$ does not necessarily have any meaning, that is, it cannot be concluded that uj resembles uk to a particular degree. Altogether, these similarity classes do, of course, express the similarities between all pairs of objects in the domain. In fact, a lot of redundant information is contained in these classes, because, for example, the degree to which two objects uj and uk are similar to each other is explicitly represented twice, once as a membership value of uk in the class $[uj]$, $\mu_{[uj]}(uk) = s(uj, uk)$, and once as a membership value of uj in the fuzzy set $[uk]$, $\mu_{[uk]}(uj) = s(uk, uj)$. By the symmetry of s it is known that $\mu_{[uk]}(uj) = s(uj, uk) = s(uk, uj) = \mu_{[uj]}(uk)$.

These similarity classes as defined by Zadeh are not, in general, disjoint, as they are in the case of an equivalence relation, as has been shown by the previous example. Hence they do not seem to be a useful tool for the treatment of partitioning a relation of the FRR into unique pieces of information (tuples). It is interesting to mention, though, that the α-level sets of the resolution of a similarity relation, a fuzzy binary relation, are equivalence relations on the domain (Zadeh [22]).

THEOREM 2 *Let s be a similarity relation on the set U. Then for α satisfying $0 < \alpha \leq 1$, the α-level sets S_α are equivalence relations in U.*

Recall that an α-level set S_α of a fuzzy binary relation s on $U \times U$ is a nonfuzzy relation on $U \times U$ defined by $S_\alpha = \{(u1, u2) | s(u1, u2) \geq \alpha$ and $(u1, u2) \in U \times U\}$. The proof of Theorem 2 is presented by (Zadeh [22, page 186]). Clearly, the partition induced on the domain by S_α is a refinement of the one induced by $S_{\alpha'}$ if $\alpha \geq \alpha'$ (Zadeh [22]).

Based on these latter results, Buckles and Petry [14] define two tuples to be redundant with respect to an α-level if corresponding domain value pairs of both are within one block of the partition induced by an α-level set of the domain. Hence, this approach does not work with the similarity classes per se but reduces

the fuzziness inherent in those by just dealing with α-level sets (which are crisp sets); therefore, a "nonfuzzy" nearness measure, the equivalence relation, applies. The goal is that all objects contained in one block of this α-level set have to be within one tuple. This is, in fact, a desirable property for a fuzzy database, and it can be proved that a fuzzy relation derived by merging redundant tuples according to some α-level is unique. This makes use of the fact that only multiple values—crisp subsets of the domain (corresponding to data types 3 and 4)—are allowed as domain values and the fact that α-level sets induce partitions. It is a favorable goal to guarantee this property in a fuzzy database approach, but it can be gained only at the cost of unrealistic assumptions. Unfortunately, this goal can no longer be met when the transitivity property is dropped.

Models that allow more than the first four data types for their values can no longer make use of the properties of the similarity relation. The reason is that in order to use the fact that the α-level sets of a similarity relation induce a partition, all domains of a relation must have this similarity relation as a nearness measure. The decision to allow more expressiveness in these models by admitting membership values (type 7), possibility distributions (data types 5 and 6), and so on as domain values rules this out. Topics such as the determination of redundant tuples and the merging of tuples to get rid of redundancy are in general not addressed in these models. They are open problems.

8. THE TOLERANCE RELATION

The notion of an acceptable nearness measure is hence defined as a relation that is reflexive and symmetric but not necessarily transitive. This measure is referred to from now on as a *tolerance relation* in order to avoid conflicts. The beauty of this definition of a tolerance relation is that the similarity relation and the proximity relation are both special cases.

The rule of thumb for the development of the fuzzy data model FRR is to pose as few restrictions as possible and, of course, to have no rules contradicting human intuition. Fuzzy set theory has the goal of representing the real world by creating a model as realistic as possible, which implies it should fit with a human's conception of the world.

An in-depth discussion of tolerance relations follows. Schreider [9], in his studies of the algebra of relations, discusses the notion of resemblance. He denotes a tolerance relation, a reflexive and symmetric relation, to be explicable of the concept of resemblance. Schreider is concerned with "normal," that is, crisp, relation theory, and hence the reflexivity and symmetry properties referred to are crisp properties. For a relation τ on the set U to be reflexive means that, for all x in U, $x\tau x$ holds, and to be symmetric means that if $x\tau y$ holds then $y\tau x$ also has to hold. Recall that for a crisp relation $x\tau x$ to hold means in fuzzy set theory that $\mu_\tau(x, y) = 1$. The concept of tolerance of objects is

described as their partial interchangeability—the possibility of mutual replacements with certain (permissible) losses. Consider the important claim made by Schreider [9, page 81], "If we are given only resemblances for some objects, then we cannot partition them into clearly defined classes, so that the objects within a class resemble each other, but there is no resemblance between objects from different classes. In the case of resemblance, a hazy situation with no clear boundaries arises." Each object of the domain carries some information about the objects resembling it, but not all such information. In other words, complete information, as in the case of equivalence relations, can no longer be assumed.

In order to discover and understand the relationships between resembling objects, the properties of crisp resemblance relations, here termed tolerance relations, are reviewed in the following. This discussion is based on the work of Schreider [9]. Note that this section addresses crisp tolerance relations, but a relationship between this crisp measure and fuzzy set theory should be anticipated. Indeed, in a later section the results presented here are extended to the fuzzy context.

DEFINITION 9 *A set U with a tolerance relation τ given in it, $\langle U, \tau \rangle$, is called a* tolerance space. *A set $P \subset U$ is called a* preclass *in $\langle U, \tau \rangle$ if $\forall x, y \in P$, x and y are tolerant, that is, if $x \tau y$ holds.*

Note that the set of all preclasses of a tolerance space is always covering, because $\forall x \in U$, $\{x\}$ is a preclass by reflexivity. It is also true that in order for x and y to be tolerant, it is necessary and sufficient that there exists a preclass P in $\langle U, \tau \rangle$ containing both. The definition of classes given so far allows for a lot of redundancy, which should be diminished by the following notion.

DEFINITION 10 *A set $T \subset U$ is called a* tolerance class *in $\langle U, \tau \rangle$ if T is a maximal preclass.*

This means that for all objects z of U outside of T there exists an object x in T that is not tolerant to z, and hence no object z can be added to the tolerance class T without destroying the preclass property of this class. It can be shown that every preclass P of $\langle U, \tau \rangle$ is contained in at least one tolerance class T of $\langle U, \tau \rangle$. This implies that every object of the set U is contained in some tolerance class T of $\langle U, \tau \rangle$, that is, the set of tolerance classes is a covering of U. The following is an example of a set of tolerance classes for a tolerance space $\langle U, \tau \rangle$.

EXAMPLE 6. Let U be the powerset of $\{1, 2, 3\}$ minus $\varnothing$; and let τ on U be defined to be the following: Two elements x and y of U are considered to be tolerant, that is, $x \tau y$, iff $x \cap y \neq \varnothing$, that is, if the sets x and y contain at least one common element. Then examples of preclasses are

$$P1 = \{\{1\}, \{1, 2\}\}$$

$$P2 = \{\{1, 3\}, \{3, 2\}\}$$

etc.

And there are many more preclasses. Next, the tolerance classes are given:

$$T1 = \{\{1\}, \{1, 2\}, \{1, 3\}, \{1, 2, 3\}\}$$

$$T2 = \{\{2\}, \{1, 2\}, \{2, 3\}, \{1, 2, 3\}\}$$

$$T3 = \{\{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

$$T4 = \{\{1, 2\}, \{2, 3\}, \{1, 3\}, \{1, 2, 3\}\}$$

These four classes are tolerance classes of $\langle U, \tau \rangle$ because none is a subset of one of the others and there are no other classes in $\langle U, \tau \rangle$ that contain any of these classes as proper subsets and are preclasses at the same time.

This leads to the following lemma.

LEMMA 1 *In order for two objects x and y of $\langle U, \tau \rangle$ to be tolerant, it is necessary and sufficient that there exist a tolerance class T of $\langle U, \tau \rangle$ that contains both x and y.*

This lemma is a natural consequence of the fact that tolerance classes form a covering of U. Finally, an interesting theorem can be stated.

THEOREM 3 *Given a tolerance space $\langle U, \tau \rangle$, let H be the set of all its tolerance classes, and let S^H be the powerset of H excluding the empty set. Define the elements h1, h2 $\in$ H to be tolerant iff h1 $\cap$ h2 $\neq \varnothing$. Then there exists a mapping φ: $U \to S^H$ such that x, y $\in U$ are tolerant iff their images are tolerant in S^H.*

This theorem can be shown by letting φ be the mapping that assigns to each $x \in U$ the subset of H consisting of all tolerance classes containing x.

Finally, the attempt is made to reduce the amount of redundancy inherent in the set of these classes even further. This leads to the notion of a *basis* which is a minimal collection of "sufficient" tolerance classes.

DEFINITION 11 *A collection $H_B = \{K^1, K^2, \cdots, K^m\}$ of tolerance classes in $\langle U, \tau \rangle$ is called a* basis *if*

1. *For all tolerant pairs x, y in U there exists a tolerance class K in H_B such that x, y $\in$ K.*
2. *The deletion of any class from H_B leads to the loss of property* 1*; that is, for every K $\in$ H_B, there exists a tolerant pair x, y for which K is the only common tolerance class in H_B.*

This concept of a basis of a tolerance space is what can be compared to a partition of an equivalence relation, where the first property guarantees the covering of the domain by the tolerance classes (the completeness of information) and the second the minimality of redundancy. Hence, the first refers to the sufficiency and the second to the necessity of selected classes. A basis contains only as much information as is necessary to express the tolerance relation and not any more; that is, any repetitive information is omitted

whenever possible. This concept of a basis might be useful to define when tuples can be considered to be redundant in the FRR database, and hence when tuples should be merged to decrease redundancy. It is possible to find a basis given a tolerance space by starting from the initial set of all preclasses (condition 1). First, these preclasses could be reduced to tolerance classes by checking for subset and superset relationships between them. If, after that, all superfluous classes (condition 2) are successively deleted, then a basis of the tolerance space is obtained at the end. It is always possible to obtain a basis by the described process, but it is obviously an inefficient approach. The complexity issue is further discussed in a later section. The following is an example of the selection process of a basis for a tolerance space.

EXAMPLE 7. Let U and τ be defined as in the previous example. Assume that the set of all preclasses has already been reduced to the collection of all tolerance classes, $\{T1, T2, T3, T4\}$.

Note that none of these classes Ti can be enlarged without destroying this preclass property. Furthermore, note that the class $T1$ has to be included in any basis of the domain. The class $T1$ contains all sets that have the 1 as one element, and hence the sets of $T1$ are all tolerant on account of this common element 1. Since the singleton $\{1\}$ is not in any other class, the class $T1$ is the only tolerance class that contains the tolerant pairs where one of the objects is $\{1\}$. Consequently, $T1$ is needed in any basis. A similar argument applies to the classes $T2$ and $T3$, which contain the singletons $\{2\}$ and $\{3\}$, respectively. Hence, $T1$, $T2$, and $T3$ are *necessary classes*. This cannot be said about class $T4$, which by condition 2 of a basis is *superfluous* and could be removed from any basis without a loss of information about the resemblance between objects. This is so because the information contained in $T4$ is dispersed over the other classes $T1$–$T3$. Hence, $\{T1, T2, T3\}$ is a basis.

Note that we refer to "a" basis instead of "the" basis. This is not a mistake, as there is not, in general, one unique basis per tolerance space. There can be various sets that form a basis of a tolerance space, and, what is more, the number of classes in a basis is not invariant with respect to the choice of the basis. These are unwelcome facts that suggest that caution should be taken when exploiting these concepts for the FRR model. Algorithms have to be developed to, for example, find the basis with the minimal number of classes or to determine the number of different bases per tolerance space. The concepts presented here have been developed within the framework of relation theory (Schreider [9]). It is, however, possible to recast them in terms of graph theory. This is of advantage since graph theory problems have been well studied in the literature (Garey and Johnson [23]). Transforming a problem from one domain to another may help to shed light on the existence of algorithms and their complexity.

A tolerance space, $\langle U, \tau \rangle$, corresponds to a graph $G = (V, E)$, which is defined as follows. The set U—the objects in the tolerance space—corresponds

to the set of vertices V of the graph G. Clearly, $|U| = |V|$. There is a 1-1 mapping m such that each object x of U is mapped to a vertex v of V. The tolerance relation τ defined on the set U is represented by the set of edges E on V of G. It is assumed that each vertex $v \in V$ has an edge to itself (a loop) to model the reflexivity of the tolerance relation. The symmetry of τ is guaranteed by restricting the graph G to undirected edges, since an undirected edge can conceptually be viewed as consisting of two directed edges pointing in opposite directions. If two objects x and y of U are represented by the vertices $v1$ and $v2$ of V, respectively, then there is an edge e between $v1$ and $v2$ if and only if x and y are tolerant with respect to τ. In short, there is a 1-1 mapping m between a tolerance space and an undirected graph with loops.

Now, the relation theory concepts introduced in this section can be expressed in graph theory terminology. The concept of a preclass P in a tolerance space corresponds to a subgraph G' of G, where $G' = (V', E')$, the mapping m maps all objects of P to the set of vertices V', and all vertices of V' are connected in G'. Such a complete subgraph has been called a *clique* in graph theory. Consequently, a tolerance class is a maximal clique. In these terms, a basis corresponds to the problem of finding a minimal cover of G in terms of maximal cliques.

The problems of cliques and covers have been studied in the context of graph theory and are often computationally expensive (Garey and Johnson [23]). Hence, it is anticipated that the same is true for the relational problems. Consequences of this complexity issue are discussed at the end of the next section. For the following, the fact that a set of tolerance classes of a tolerance space with the properties of a basis exists and can be found is considered satisfying. Next, we propose fuzzy extensions of the concepts just presented.

9. THE NEW NEARNESS MEASURE: THE RESEMBLANCE RELATION

In what follows, we discuss how the framework just presented can be extended to handle the possibilistic representation found in the FRR model. We define a nearness measure as a fuzzy version of a tolerance relation. The process of fuzzifying the tolerance relation was given implicitly by Zadeh [22] when he defined properties such as reflexivity and symmetry for fuzzy relations. We propose that the fuzzy version of a tolerance relation be called a *resemblance relation* because it expresses, as previously discussed, the resemblance of two or more objects. The resemblance relation is defined as follows.

DEFINITION 12 *A* resemblance relation, res *of U, is a fuzzy binary relation on* $U \times U$ *that fulfills the following properties:*

1. *Reflexive:* $\mu_{res}(x, x) = 1\ (\forall x)$

2. *Symmetric:* $\mu_{res}(x, y) = \mu_{res}(y, x)$ $(\forall x, y)$
The strength of a resemblance relation $\mu_{res}(x, y)$ *is referred to as* $res(x, y)$, *that is, the degree of the resemblance between* x *and* y.

An α-level set can be defined on any fuzzy set; consequently, a lemma concerning the α-level sets of resemblance relations can be constructed.

LEMMA 2 *Let res be a resemblance relation on a set U. For all* α *with* $0 < \alpha \leq 1$, α-*level sets* RES_α *are* tolerance relations *on U.*

Proof. Recall that an α-level set RES_α is a crisp binary relation on U defined by $\text{RES}_\alpha = \{(u1, u2) | \mu_{\text{res}}(u1, u2) \geq \alpha \text{ and } (u1, u2) \in U \times U\}$ for $\alpha \in (0, 1]$

By assumption, the resemblance relation res is reflexive, that is, $\mu_{\text{res}}(u, u) = 1.0$ for all $u \in U$. So we have $(u, u) \in \text{RES}_\alpha$ for all α; that is, RES_α is reflexive.

Also, for all $(u1, u2) \in \text{RES}_\alpha$ we know that $\mu_{\text{res}}(u1, u2) \geq \alpha$. By the symmetry property of the resemblance relation res, we get $\mu_{\text{res}}(u2, u1) = \mu_{\text{res}}(u1, u2) \geq \alpha$. This implies that $(u2, u1) \in \text{RES}_\alpha$ and thus RES_α is symmetric for all α. ■

To summarize, Lemma 2 results from the fact that α-level sets are crisp sets and that a tolerance relation is the crisp version of a resemblance relation. This result is an important observation since it guarantees the reduction of resemblance relations and associated concepts to tolerance relations.

An α-level set, RES_α, contains all pairs of values from U that resemble each other (at least to degree α). We now propose that the collection of the definitions and theorems related to the crisp tolerance relation and its properties be extended in a rather natural manner to those for its fuzzy counterpart, the resemblance relation. This is done with respect to an arbitrary but fixed α-value, however. These results are presented in a condensed version in the following.

DEFINITION 13 *Given a set U with a resemblance relation* ρ *as previously defined. Then,* $\langle U, \rho \rangle$ *is called a* resemblance space. *An* α-*level set* RES_α *induced by* ρ *is termed an* α-resemblance set. *Define the relationship of two values x, y,* $\in U$ *that resemble each other with a degree larger than or equal to* α, *that is,* $\rho(x, y) \geq \alpha$, *as* α-resemblant. *The following notation is proposed for the notion of two values x, y being* α-*resemblant:* $x \rho_\alpha y$. *A set* $P \subset U$ *is called an* α-preclass *on* $\langle U, \rho \rangle$ *if* $\forall x, y \in P$, x *and* y *are* α-*resemblant, that is,* $x \rho_\alpha y$ *holds.*

The concepts introduced in Definition 13 are demonstrated in Example 8.

EXAMPLE 8. Let Aptitude be the set {*very-good, good, average, bad, very-bad*} or, for short, $\{vg, g, a, b, vb\}$. Let a resemblance relation res be defined on the set Aptitude as shown in Figure 11. Then the two values b and vb are α-

res	*vg*	*g*	*a*	*b*	*vb*
vg	1.0	0.8	0.3	0.1	0.0
g		1.0	0.7	0.2	0.1
a			1.0	0.7	0.3
b				1.0	0.8
vb					1.0

Figure 11. Resemblance Relation on the Domain Aptitude.

resemblant with $\alpha = 0.75$, because they resemble each other with a degree larger than or equal to 0.75. Also, $\alpha = 0.75$ induces the following α-resemblance set:

$$\text{RES}_\alpha = \{(vg, vg), (g, g), (a, a), (b, b), (vb, vb),$$
$$(vg, g), (vb, b), (g, vg), (b, vb)\}$$

The following are 0.75 preclasses:

$$P1 = \{vg\} \quad P2 = \{g\} \quad P3 = \{a\} \quad P4 = \{b\} \quad P5 = \{vb\}$$
$$P6 = \{b, vb\} \quad P7 = \{g, vg\}$$

Note that the set of all α-preclasses of a resemblance space for a fixed α is always covering, because $\forall x \in U$, $\{x\}$ is an α-preclass by reflexivity of the resemblance relation. It is also trivial that in order for x and y to be α-resemblant, it is necessary and sufficient that there exist an α-preclass P in $\langle U, \rho \rangle$ containing both.

DEFINITION 14 *A set $R \subset U$ is called an* α-resemblance class *in* $\langle U, \rho \rangle$ *if R is a maximal α-preclass.*

EXAMPLE 9. Given the resemblance relation as defined in Figure 11 and the Aptitude domain of the previous example. The following are α-resemblance classes for $\alpha = 0.75$:

$$R1 = \{a\} \quad R2 = \{b, vb\} \quad R3 = \{g, vg\}$$

For $\alpha = 0.65$, the following α-resemblance classes exist:

$$R1 = \{vg, g\} \quad R2 = \{g, a\} \quad R3 = \{a, b\} \quad R4 = \{b, vb\}$$

Trivially, it can be concluded that every α-preclass P of $\langle U, \rho \rangle$ is contained in at least one α-resemblance class R of $\langle U, \rho \rangle$. Since, as indicated in Lemma 2, the set RES_α on U is in fact a crisp relation, that is, a tolerance relation, every object

of the set U is contained in some α-resemblance class R of $\langle U, \rho \rangle$; that is, the set of α-resemblance classes is a covering of U.

LEMMA 3 *In order for two objects x and y of $\langle U, \rho \rangle$ to be α-resemblant, it is necessary and sufficient that there exists an α-resemblance class T of $\langle U, \rho \rangle$ that contains both x and y.*

In accordance with the notion of a basis for a tolerance space, the concept of an α-*basis* for a resemblance space is proposed next.

DEFINITION 15 *A collection $H_B = \{P^1, P^2, \cdots, P^m\}$ of α-resemblance classes on $\langle U, \rho \rangle$ is called an* α-basis *if*

1. *For all α-resemblant pairs x, y in U, there exists an α-resemblance class T in H_B such that $x, y \in T$.*
2. *The deletion of any class from H_B leads to the loss of property 1; that is, for every $T \in H_B$, there exists an α-resemblant pair x, y for which T is the only common α-resemblance class in H_B.*

EXAMPLE 10. Let res be the resemblance relation from Example 9. Then the α-basis consists of all resemblance classes as listed in Example 9 for $\alpha = 0.75$ and for $\alpha = 0.65$, respectively. In other words, they are both unique.

Again, an α-basis contains only as much information as is necessary to express the resemblance relation and not any more; that is, any repetitive information is omitted whenever possible. The property of a tolerance space of not having a unique basis does also hold for the fuzzy case, the α-resemblance space. Fortunately, however, in most practical cases the basis will be unique (as in Example 10). The results for a tolerance relation have been successfully transferred to those of a resemblance relation.

Next, we show how these extended concepts can be expressed in terms of graph theory. A resemblance space, $\langle U, \rho \rangle$, corresponds to a graph $G = (V, E)$ with undirected but labeled edges where each vertex has a loop. All edges that begin and end at the same vertex (loops) have a label of 1.0. All other edges are labeled by a real number taken from (0, 1]. Again, if two objects x and y of U are represented by two vertices $v1$ and $v2$, respectively, of V, then there is an edge e between $v1$ and $v2$ if and only if x and y are α-resemblant with $\alpha > 0$. The label of edge e is given by $\sup_\alpha \{\alpha | x \rho_\alpha y\}$. For a given $\alpha \in (0, 1]$, the α-resemblance set RES_α induced by ρ corresponds to an unlabeled graph $G_\alpha = (V_\alpha, E_\alpha)$, where $V_\alpha = V$ and E_α consists of those edges of E that have a label greater than or equal to α in G. An α-preclass of a resemblance space, $\langle U, \rho \rangle$, corresponds to a clique in the graph G_α (with G_α isomorphic to RES_α). An α-resemblance class is a maximal clique in G_α. Then, an α-basis corresponds to a minimal cover of G_α in terms of maximal cliques.

Given a user's perception of similarity, that is, an α-resemblance threshold, the issues related to the resemblance relation reduce to those of the correspond-

ing tolerance relation. It has already been indicated that the problems of cliques and covers have been studied in the context of graph theory and are known to be computationally expensive (Garey and Johnson [23]). Consequently, the calculation of a basis and other relational properties, which have just been shown to be isomorphic to these graph theoretic problems, is complex as well.

In the following, we investigate how these concepts could be used as tools for handling the redundancy of fuzzy tuples. Note that the database designer is faced with a trade-off between the time invested in reducing redundancy in the database and the wasted space and possible problem of inconsistency due to redundant pieces of information kept in the database. This is a design decision the database designer should be aware of since it is likely to considerably influence the performance of the database system. It is expected that the decision depends on the application at hand as well as the database features desired by the database users. There are several routes one may consider.

First, one may not be concerned about the redundancy that could accumulate in the database. An example of such a situation is an application domain that mainly requires retrieval operations and hardly any update operations. In this case, the proposed resemblance measure is used as a nearness measure in the retrieval process by, for example, determining how similar two tuples are, but it is not used to reduce redundancy.

On the other hand, one may be interested in reducing the redundancy of the information stored in the database in order to create a very compact format. Then it is advisable to precompute the basis with the *minimal number* of resemblance classes for each attribute. Note that conceptually all objects in a resemblance class model the same piece of information, and thus these objects can be reduced to one combined object. This solution requires a long preprocessing time, which, fortunately though, will not affect the actual performance of the system during query processing. To increase the performance it may further be of interest to develop indexing schemes that allow us to determine whether or not tuples belong to the same tolerance class of the precomputed basis. This approach not only avoids cluttering the database with redundant data but may also lead to performance improvement due to the fact that the retrieval operations have to deal with fewer data. This is an open question, however, that requires empirical evaluation.

Then there are intermediate solutions that are located on a scale between possibly a lot of redundancy and little processing time versus little redundancy and a lot of processing time. One may want to reduce some of the redundancy without, however, insisting on the most optimal reduction. In the light of the problem complexity, one would not attempt to find a "minimal" basis. Instead one could adopt the approach of merging tuples pairwise as long as there are no violations of the preclass condition. Since there is no unique basis, this approach will not necessarily result in the exact same relation. In fact, the result will depend on the order in which one attempts to merge tuples. This is acceptable as

long as the information content is preserved. Clearly, these issues should investigated further and the trade-off just described should be empirically evaluated. However, this work goes beyond the scope of this paper.

10. SOME EXAMPLES

To tie the theoretical discussion of the previous sections back into the framework of fuzzy relational database models, some simple examples of how a resemblance relation could be used are presented next.

For simplicity, let us first limit the discussion to fuzzy databases with data values of data types 1–4 as defined in Section 3. As discussed in a previous section, Buckles and Petry [14] have limited data values to similarity classes in order to deal with the concept of redundancy. Similarly we propose to limit data values to resemblance classes. Resemblance classes represent the criteria based on which we decide whether or not two tuples are redundant. The problem of redundancy corresponds to the question of when tuples are considered to be *resemblant* enough to be merged into one tuple. Two tuples being resemblant implies that all their corresponding values for all attributes are considered to be resemblant. Hence, the following definition can be given.

DEFINITION 16 *Let x, y be two tuples of the relation R. Let relation R be defined on the attributes A1, A2, ···, An with domains D1, D2, ···, Dn, respectively. Let* $\alpha_k \in [0, 1]$ *be the* resemblance threshold *and* res_k *be the resemblance relation for the attribute Ak, for* $k = 1, \cdots, n$. *Then tuple x is* redundant *if and only if it can be merged with another tuple y of R without violating the constraints of the resemblance threshold* α_k *for corresponding domain values xk and yk for* $k = 1, \cdots, n$, *which is*

$$xk \cup yk \text{ has to be a subset of an } \alpha_k\text{-resemblance class.}$$

By the definition of an α-resemblance class this is equivalent to the following lemma.

LEMMA 4 *Let all definitions be given from Definition 16. Then the tuple x is redundant if and only if some tuple y exists in the relation R such that*

$$\min_{z1, z2 \in xk \cup yk} \{res_k(z1, z2)\} \geq \alpha_k \qquad \text{for } k = 1, \cdots, n$$

A relation R is redundant if it contains at least one redundant tuple. A redundant relation R is reduced by merging all its redundant tuples through set union. An example of a redundant relation and the merging process follows.

EXAMPLE 11. Let the relation R be defined over the domains ANY $= \{A, B, C\}$ and Aptitude. Let the resemblance relation of the domain ANY be the

identity relation and that of Aptitude be the relation res given in Figure 11. Let R consist of the three tuples $x1$, $x2$, and $x3$:

$$x1 = (x11, x12) = (\{A\}, \{vg\})$$
$$x2 = (x21, x22) = (\{B\}, \{g\})$$
$$x3 = (x31, x32) = (\{C\}, \{a\})$$

For a resemblance threshold $\alpha_1 > 0.0$ for the domain ANY, the relation R is not redundant since the first part of the tuples cannot be merged. Now assume the resemblance threshold $\alpha_1 = 0.0$. Then, for $\alpha_2 = 0.75$ the relation R is redundant. R can be reduced to

$$x1' = (x11', x12') = (\{A, B\}, \{vg, g\})$$
$$x2' = (x21', x22') = (\{C\}, \{a\})$$

This can be done because $\{vg, g\}$ is a (proper) subset of an 0.75-resemblance class as shown in Example 9. For $\alpha_2 = 0.65$ the relation R is also redundant. In this case R can be reduced to

$$x1' = (x11', x12') = (\{A, B\}, \{vg, g\})$$
$$x2' = (x21', x22') = (\{B, C\}, \{g, a\})$$

Again, this is so because $\{vg, g\}$ and $\{g, a\}$ are subsets of 0.65-resemblance classes as shown in Example 9.

Example 11 shows that the concept of a resemblance relation as a nearness measure cannot guarantee that a relation has at most one tuple with any given interpretation of the domains. For instance, the interpretation $(d1, d2) = (\{B\}, \{g\})$ is contained in both tuples of the reduced relation for $\alpha = 0.65$. This does not seem really surprising. Given a nearness measure concept without transitivity, it cannot be expected that it will be possible to separate data values into crisp nonoverlapping partitions.

The discussion is now extended to the fuzzy relational database model, which allows all eight data types described in Section 3. The results presented concerning the resemblance relation in Section 9 work with α-level values, and thus it has been neglected that the actual data values are not necessarily crisp subsets but possibility distributions over the respective domains. These possibilities of the data values should have an impact on the evaluation of whether or not two tuples are considered to be redundant. Hence, we extend Definition 16 as follows.

DEFINITION 17 *Let the attribute Ai defined over the domain Di, for i = 1, ···, n, be an attribute of a fuzzy relation R (gf-type relation). Assume for a tuple x that the attribute Ai is described by the possibility distribution $\pi_{Ai}(x)$ over the domain Di. The notation x.Ai refers to the value of the*

tuple x for the attribute Ai. Let res_i *be a resemblance relation and* $\alpha_i \in [0, 1]$ *be a resemblance threshold for all* $i = 1, \cdots, n$. *Let x and y be tuples of the relation R. Let* $x.Ak = \Sigma_{xj \in Dk} \pi_{x.Ak}(xj)/xj$ *and* $y.Ak = \Sigma_{yj \in Dk} \pi_{y.Ak}(yj)/yj$ *for* $k = 1, \cdots, n$. *Then x is considered* redundant *if and only if a tuple y exists in R such that for all* $k = 1, \cdots, n$ *the following holds:*

1. $min_{z1,z2 \in xk \cup yk \text{ with } \pi_{x.Ak}(z1)>0 \text{ and } \pi_{y.Ak}(z2)>0} \{res_k(z1, z2)\} \geq \alpha_k$
2. $min_{zj \in Dk}(1 - |\pi_{x.Ak}(zj) - \pi_{y.Ak}(zj)|) \geq \alpha$

where α *expresses the matching threshold.*

The first condition states that the union of the domain values is a subset of an α_k-resemblance class—it guarantees that the domain values resemble each other sufficiently. The second condition makes sure that the possibility distributions associated with the respective domain values are similar—it limits the difference between respective possibility distribution values. If the second property is not to be enforced, then simply set α to zero. The importance of that second condition is demonstrated in the following.

EXAMPLE 12. Let $xi = \{0.01/vg, 0.99/g\}$ and $yi = \{0.01/g, 0.99/vg\}$. Then these two values fulfill condition 1 of Definition 17 for $\alpha = 0.75$, since $\{vg, g\}$ forms a resemblance class. They are not similar, however, since xi is close to $\{g\}$ and yi is close to $\{vg\}$. This conclusion is in fact derived from the second condition, since $\min\{1 - |0.99 - 0.01|, 1 - |0.99 - 0.01|\} = (1 - 0.98) = 0.02$. This means that the matching of possibilities, α, is as low as 0.02.

Again, a relation is defined to be redundant if it contains at least one redundant tuple. Given that two tuples are determined to be redundant, they are merged into one tuple in order to reduce the amount of information stored. Two tuples are merged by the union operation of fuzzy sets as defined by Zadeh [2]. To clarify the concepts introduced in the Definition 17, an example of a redundant relation and the resulting reduced relation is given in the following.

EXAMPLE 13. Let the relation student be the relation described in Figure 6. Assume that a project operation returns only the last column of that relation. Let the resemblance of the domain Aptitude be as given in Figure 11. For $\alpha_{\text{Aptitude}} \leq 0.7$, the two domain values of this derived relation are in the same α-resemblance set (condition 1 of Definition 17). For the second condition of Definition 17, we get

$$\begin{aligned}
&\min_{z \in Aptitude} (1 - |(\pi_{Aptitude(Jack)}(z) - \pi_{Aptitude(Frank)}(z)|) \\
&= \min(1 - |0.0 - 0.4|,\ 1 - |0.9 - 0.70|) \\
&= \min(1 - 0.4,\ 1 - 0.2) \\
&= \min(0.6,\ 0.8) \\
&= 0.6
\end{aligned}$$

Thus, for $\alpha \leq 0.6$, the relation is considered redundant and could be reduced by merging the two tuples to $\{0.4/g, 0.9/vg\}$.

11. SUMMARY AND CONCLUSION

We have presented a general framework for the fuzzy extension of the relational representation, called the FRR model. The inappropriateness of existing approaches for the development of nearness measures has been shown. We have demonstrated the minimal and maximal characteristics of an acceptable nearness measure. This analysis led to the proposal of a new nearness measure, the resemblance relation. Examples of how one may apply the proposed nearness measure are given at the end of the paper.

This investigation is considered important, since the nearness measure concept constitutes the basis for a sound development of the query language of any fuzzy relational database model.

Throughout this paper, several areas for extending the work described herein have been proposed. There is the application of algorithms for the proposed concepts, for example, how to find a preclass and how to determine whether a given set of objects is α-resemblant. Then an empirical study should be conducted on comparing the approaches presented in Section 9. Such a study may either reveal that one of these approaches is superior to the others or determine some criteria for choosing among them.

ACKNOWLEDGMENTS

We are thankful for the partial support by the National Science Foundation, through grants IST 8510894 and IST 8604575. We are also grateful to the reviewers for their helpful comments and constructive suggestions.

References

1. Codd, E. F., A relational model of data for large shared data banks, *Commun. ACM* 13(6), 337–387, 1970.

2. Zadeh, L. A., Fuzzy sets, *Inf. Control* 8, 338–353, 1965.

3. Kacprzyk, J., and Ziolkowski, A., Database queries with fuzzy linguistic quantifiers, *IEEE Trans. Syst. Man, Cybern.* SMC-16(3), 474–479, 1986.

4. Prade, H., and Testemale, C., Generalizing database relational algebra for the treatment of incomplete/uncertain information and vague queries, *Inf. Sci.* 34, 115–143, 1984.

5. Buckles, B. P., Petry, F. E., and Sachar, H. S., Retrieval and design concepts for similarity-based (fuzzy) relational databases, in *Proc. ROBEXS'86,* Houston, 243–251, June 1986.

6. Rundensteiner, E. A., Hawkes, L. W., and Bandler, W., A set-valued temporal

knowledge representation for fuzzy temporal retrieval in ICAI, *Proc. NAFIPS '87,* West Lafayette, Indiana, 37–65, 1987.

7. Zemankova, M., and Kandel, A., *Fuzzy Relational Database—A Key to Expert Systems,* Verlag TÜV Rheinland, Cologne, 1984.
8. Rundensteiner, E. A., Bandler, W., Kohout, L., and Hawkes, L. W., An investigation of fuzzy nearness measures, 2nd IFSA Congress, Meiji University, Tokyo, Japan, July 1987.
9. Schreider, J. A., *Equality, Resemblance, and Order,* Russian translation, Mir Publishers, Moscow, 1971.
10. Zadeh, L. A., Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets Syst.* 1(1), 3–28, 1979.
11. Zvieli, A., On complete fuzzy relational query languages, *Proc. NAFIPS '86,* New Orleans, 704–726, 1986.
12. Dubois, D., and Prade, H., The treatment of uncertainty in knowledge-based systems using fuzzy sets and possibility theory, *Int. J. Intell. Syst.* 3(2), 141–165, 1988.
13. Zemankova, M., and Kandel, A., Implementing imprecision in information systems, *Inf. Sci.* 37(3), 107–141, 1985.
14. Buckles, B. P., and Petry, F. E., A fuzzy representation of data for relational databases, *Fuzzy Sets Syst.,* 7, 213–226, 1982.
15. Oezsoyoglu, G., Oezsoyoglu, Z. M., and Matos, V., Extending relational algebra and relational calculus with set-valued attributes and aggregate functions, *ACM Trans. Database Syst.* 12(4), 566–592, 1987.
16. Umano, M., Freedom-0: a fuzzy database system, in *Fuzzy Information and Decision Processes* (M. M. Gupta and E. Sachez, Eds.), North-Holland, Amsterdam, 339–347, 1982.
17. Raju, K. V. S. V. N., and Majumdar, A. K., Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems, *ACM Trans. Database Syst.* 13(2) 129–166, 1988.
18. Anvari, M., and Rose, G. F., Fuzzy relational databases, in 1st Int. Conf. on Fuzzy Information Processing, Kuaui, Hawaii, 1984.
19. Rundensteiner, E. A., The development of a fuzzy temporal relational database (FTRDB): an artificial intelligence application, Master's thesis, Dept. of Computer Science, Florida State Univ., 1987.
20. Potoczny, H. B., On similarity relations in fuzzy relational databases, *Fuzzy Sets Syst.* 12, 231–235, 1984.
21. Tversky, A., Features of similarity, *Psychol. Rev.* 84(4), 327–353, 1977.
22. Zadeh, L. A., Similarity relations and fuzzy orderings, *Inf. Sci.* 3, 177–200, 1971.
23. Garey, M. R., and Johnson, D. S., *Computers and Intractability. A Guide to the Theory of NP-Completeness,* Freeman, San Francisco, 1979.