Fuzzy Rough Set Techniques for Uncertainty Processing in a Relational Database

Theresa Beaubouef,[1], * Frederick E. Petry[2], †
[1]*Computer Science Department, Southeastern Louisiana University, Hammond, Louisiana 70402*
[2]*Center for Intelligent and Knowledge-Based Systems, Tulane University, New Orleans, Louisiana 70125*

This paper concerns the modeling of imprecision, vagueness, and uncertainty in databases through an extension of the relational model of data: the *fuzzy rough relational database*, an approach which uses both fuzzy set and rough set theories for knowledge representation of imprecise data in a relational database model. The fuzzy rough relational database is formally defined, along with a fuzzy rough relational algebra for querying. Comparisons of theoretical properties of operators in this model with those in the standard relational model are discussed. An example application is used to illustrate other aspects of this model, including a fuzzy entity−relationship type diagram for database design, a fuzzy rough data definition language, and an SQL-like query language supportive of the fuzzy rough relational database model. This example also illustrates the ease of use of the fuzzy rough relational database, which often produces results that are better than those of conventional databases since it more accurately models the uncertainty of real-world enterprises than do conventional databases through the use of indiscernibility and fuzzy membership values. © 2000 John Wiley & Sons, Inc.

1. INTRODUCTION

The study of databases began with the design of efficient storage and data sharing techniques for large amounts of data. Once technology became available for these basic data operations, models of data were developed to conceptualize, define, and manipulate the data without regard to the underlying hardware. One of the most popular of these models, due to its simplicity and mathematical

*Author to whom correspondence should be addressed; e-mail: tbeaubouef@selu.edu
†e-mail: fep@eecs.tulane.edu

elegance, was the relational model of Codd.[1] This model has served as a useful tool for many years.

A current trend in databases is the usage by non-computer scientists, individuals with little or no knowledge of the technical aspects of database systems. Consequently, the external view of such systems is becoming more removed from the hardware technology and simple data models, and closer to human cognition. Although the relational database provides the necessary foundation for rigid database modeling, it does not directly support the modeling of "human-related" concepts such as ambiguity, imprecision, and uncertainty. Therefore, extensions to the relational database such as can be found in Refs. 2–6 should be considered to provide the necessary mechanisms for a higher-level, more human-like model of data.

Consider, for example, the *Public Concerns Database*, which contains demographic information and commentary for people living near potentially hazardous nuclear and chemical plants. Data obtained from surveys may be precise. However, data recorded by interviewers, especially for uncooperative subjects, may be only reasonable approximations to the truth. If the interviewer must approximate traits such as "age" or "race," uncertainty must be taken into account. It is difficult to model these types of uncertainty using the standard relational model.

In this paper we develop an approach to using both fuzzy set and rough set theories for knowledge representation of imprecise data in a relational database model. Fuzzy set and rough set techniques should not be viewed as competitive but as complementary approaches as discussed extensively by Dubois and Prade.[7] This is just the viewpoint we take in our database modeling in this paper. As pointed out in Ref. 7, rough sets capture the idea of indiscernibility among members of a set and utilize a discrete formalism of set partitions. Correspondingly, fuzzy sets can be viewed as capturing imprecision by the nature of a vaguely defined set boundary and represented as a generalization of a discrete set membership function by a continuous function.

Consider the two major approaches that have been taken to modeling uncertainty in fuzzy databases.[8] In one approach, ill-described data has been modeled by possibility distributions over the attribute domains by many researchers.[9–11] Another approach, taken by Buckles and Petry and others, basically captures indiscernibility of values of an attribute using the idea of fuzzy similarity relationships.[3,12] We have used rough set theory for uncertain data representation in a relational data model.[13] This model has an analogous structure to the fuzzy set approach using similarity relationships and we have been able to prove similar desirable properties hold. In the development of the representation, certain aspects naturally lent themselves to fuzzy set modeling. In this paper we review our basic rough database approach and present the extension to the model with fuzzy set techniques. We utilize the notions of indiscernibility and possibility from rough set theory coupled with the idea of membership values from fuzzy set theory to represent uncertain information in a manner that maintains the degree of uncertainty of information for each tuple of the original database and also those resulting from queries.

The following section provides background on the various components of our model: relational databases, rough sets, the rough relational database, and fuzzy rough sets. Subsequently, we introduce our fuzzy rough data definition language and use it to define the Public Concerns Database, an application that embodies the various types of uncertainty that our fuzzy rough techniques are capable of modeling. Next we introduce the fuzzy rough relational database model and define its operators in terms of a fuzzy rough relational algebra. Sample queries to the Public Concerns Database are expressed using the fuzzy rough relational algebra and in a later section with SQL-type queries to complete the picture from a user's point of view. Finally, we discuss our conclusions.

2. BACKGROUND

2.1. Relational Databases

The relational database model introduced by Codd[1] uses the mathematical concept of a *relation* as its data structure. Data can be depicted in these relations as rows and columns of values in a table. *Tuples*, or rows in the table, are the elements of the relation. Each tuple contains a value for each of the attributes of the relation (each column), such that the tuple has some semantic meaning as belonging to the relation. All values within a given column are elements of the *domain* of the attribute for that column. A domain is a set of atomic values from which attributes draw their values, often specified in terms of data type and format.

A relation is described by a *relation schema* of the form $R(A_1, A_2, \ldots, A_n)$, where R is a set of tuples of the form $\{A_1, A_2, \ldots, A_n\}$ and A_i is an *attribute* which names a role played by the domain D_i defined for each column. A relation is a set whose elements are its tuples. Consequently, the ordering of the tuples is irrelevant and typical set operations such as UNION ($\cup$), INTERSECTION ($\cap$), CARTESIAN PRODUCT ($\times$), and SET DIFFERENCE ($-$) can be applied to relations.

The relational algebra defines the operations that can be used in the relational model to express queries, requests to retrieve data from relations in the database. This algebra contains the commonly used SELECT (σ), PROJECT (π), and JOIN ($\bowtie$) operators in addition to the set operators mentioned previously. SELECT retrieves a subset of tuples from a relation, PROJECT retrieves a subset of columns from a relation, and JOIN, a binary operator, retrieves a relation containing tuples which are combinations of related tuples from the two original relations. Many other operators such as DIVISION, and aggregation and partitioning operators have also been defined. Commercial products often include additional operations for such things as counting and sorting which make the database system more convenient for the user.

The relational algebra has the algebraic property of closure. Any operation applied to one or more relations produces a new relation. It also has the property that the operations DIFFERENCE, UNION, PROJECT, PRODUCT, and SELECT form a complete set. All other relational operations can be defined in terms of these. Therefore, these five operators are sufficient to specify data associated with any relationship in the database design, and query languages having these operations are called *relationally complete*.

Because the relational model has a strong mathematical foundation based on simple structures that are easy to understand and manipulate, it has become very popular and many commercial database packages based on the relational model are available. The relational model has also been a popular topic of research. Many properties of this model have been researched and formally proven and several extensions to the basic relational model exist. Further discussion of the relational data model and relational algebra can be found in Refs. 14–16.

2.2. Rough Sets

Rough set theory, introduced by Pawlak[17] and discussed in greater detail in Refs. 18 and 19, is a technique for dealing with uncertainty and for identifying cause–effect relationships in databases as a form of database learning. Rough sets involve the following concepts:

U is the nonempty *universe*.
R is an *indiscernibility relation*, or equivalence relation.
$A = (U, R)$, an ordered pair, is the *approximation space*.
$[x]_R$ denotes the equivalence class of R containing x, for an element x of U.
Elementary sets in A are the equivalence classes of R.
A *definable set* in A is any finite union of elementary sets in A.

Given the approximation space A defined on some universe U with equivalence relation R imposed upon A, U is partitioned into equivalence classes called elementary sets. The unions of combinations of these elementary sets define other sets in A. Given that $X \subseteq U$, X can be defined in terms of the definable sets in A by the following:

The *lower approximation of X in A* is the set $\underline{R}X = \{x \in U \mid [x]_R \subseteq X\}$.
The *upper approximation of X in A* is the set $\overline{R}X = \{x \in U \mid [x]_R \cap X \neq \emptyset\}$.

The set approximations $\underline{R}X$ and $\overline{R}X$ may also be described as follows: the R-positive region of X is $\mathrm{POS}_R(X) = \underline{R}X$, the R-negative region of X is $\mathrm{NEG}_R(X) = U - \overline{R}X$, and the boundary or R-borderline region of X is $\mathrm{BN}_R(X) = \overline{R}X - \underline{R}X$. X is *R-definable* if and only if $\underline{R}X = \overline{R}X$. Otherwise, $\underline{R}X \neq \overline{R}X$ and X is *rough* with respect to R. Consider the following example: Let

$$U = \{\text{CHILD, PRE-TEEN, TEEN, YOUTH, TEENAGER, YOUNG-ADULT,}$$
$$\text{ADULT, SR, SR-CITIZEN, ELDERLY}\}.$$

Let the equivalence relation R be defined as follows:

$$R^* = \{[\text{CHILD}, \text{PRE-TEEN}], [\text{TEEN}, \text{YOUTH}, \text{TEENAGER}], [\text{YOUNG-ADULT}], [\text{ADULT}], [\text{SR}, \text{SR-CITIZEN}, \text{ELDERLY}]\}.$$

Let

$$X = \{\text{CHILD}, \text{PRE-TEEN}, \text{YOUTH}, \text{YOUNG-ADULT}\}.$$

We can define X in terms of its lower and upper approximations:

$$\underline{R}X = \{\text{CHILD}, \text{PRE-TEEN}, \text{YOUNG-ADULT}\}, \qquad \text{and}$$

$$\overline{R}X = \{\text{CHILD}, \text{PRE-TEEN}, \text{TEEN}, \text{YOUTH}, \text{TEENAGER}, \text{YOUNG-ADULT}\}.$$

The group of subsets of U with the same upper and lower approximations in A is a *rough set in A*. In this example the rough set is

$$\begin{aligned}\{&\{\text{CHILD}, \text{PRE-TEEN}, \text{TEEN}, \text{YOUNG-ADULT}\}\\ &\{\text{CHILD}, \text{PRE-TEEN}, \text{YOUTH}, \text{YOUNG-ADULT}\}\\ &\{\text{CHILD}, \text{PRE-TEEN}, \text{TEENAGER}, \text{YOUNG-ADULT}\}\\ &\{\text{CHILD}, \text{PRE-TEEN}, \text{TEEN}, \text{YOUTH}, \text{YOUNG-ADULT}\}\\ &\{\text{CHILD}, \text{PRE-TEEN}, \text{YOUTH}, \text{TEENAGER}, \text{YOUNG-ADULT}\}\\ &\{\text{CHILD}, \text{PRE-TEEN}, \text{TEEN}, \text{TEENAGER}, \text{YOUNG-ADULT}\}\}.\end{aligned}$$

2.3. The Rough Relational Database Model

The rough relational database model[13] is an extension of the standard relational database model of Codd.[1] It captures all the essential features of the theory of rough sets including the notion of indiscernibility of elements through the use of equivalence classes and the idea of denoting an undefinable set by its lower and upper approximation regions.

Every attribute domain is partitioned by some equivalence relation designated by the database designer or user. Within each domain, a group of values that are considered indiscernible form an equivalence class. The query mechanism uses class equivalence rather than value equality in retrievals. A user may not know the particular attribute value, but might be able to think of a value that is equivalent to the value required. For example, if the query requests "COLOR = 'RED'," the result will contain all colors that are defined as equivalent to RED, such as SCARLET, CRIMSON, or ROUGE. Therefore, the exact wording of a query is less critical.

Recall is also improved in the rough relational database. A rough relation will be seen shortly to represent imprecision by the use of the indiscernibility relationship among the domain values of the attributes. So rough relations provide *possible* matches to the query in addition to the *certain* matches, which are obtained in the standard relational database. This is accomplished by using set containment in addition to equivalence of attributes in the calculation of lower and upper approximation regions of the query result.

The rough relational database has several features in common with the ordinary relational database. Both models represent data as a collection of *relations* containing *tuples*. These relations are sets. The tuples of a relation are its elements, and like the elements of sets in general, are unordered and nonduplicated. A tuple $\mathbf{t_i}$ takes the form $(d_{i1}, d_{i2}, \ldots, d_{im})$, where d_{ij} is a *domain value* of a particular *domain set* D_j. In the ordinary relational database, $d_{ij} \in D_j$. In the rough relational database, however, as in other non-first normal form extensions to the relational model,[20,21] $d_{ij} \subseteq D_j$, and although it is not required that d_{ij} be a singleton, $d_{ij} \neq \emptyset$. Let $P(D_i)$ denote the powerset(D_i) $-\emptyset$.

DEFINITION. *A rough relation R is a subset of the set cross product $P(D_1) \times P(D_2) \times \cdots \times P(D_m)$.*

A rough tuple $\mathbf{t}$ is any member of R, which implies that it is also a member of $P(D_1) \times P(D_2) \times \cdots \times P(D_m)$. If $\mathbf{t_i}$ is some arbitrary tuple, then $\mathbf{t_i} = (d_{i1}, d_{i2}, \ldots, d_{im})$, where $d_{ij} \subseteq D_j$. A tuple in this model differs from that of ordinary databases in that the tuple components may be sets of domain values rather than single values. For notational convenience, the set braces are omitted from singletons.

DEFINITION. *An interpretation $\alpha = (a_1, a_2, \ldots, a_m)$ of a rough tuple $\mathbf{t_i} = (d_{i1}, d_{i2}, \ldots, d_{im})$ is any value assignment such that $a_j \in d_{ij}$ for all j.*

The interpretation space is the cross product $D_1 \times D_2 \times \cdots \times D_m$, but is limited for a given relation R to the set of those tuples which are valid according to the underlying semantics of R. In an ordinary relational database, because domain values are atomic, there is only one possible interpretation for each tuple $\mathbf{t_i}$, the tuple itself. In the rough relational database, this is not always the case.

Let $[d_{xy}]$ denote the equivalence class to which d_{xy} belongs. When d_{xy} is a set of values, the equivalence class is formed by taking the union of equivalence classes of members of the set; if $d_{xy} = \{c_1, c_2, \ldots, c_n\}$, then $[d_{xy}] = [c_1] \cup [c_2] \cup \cdots \cup [c_n]$.

DEFINITION. *Tuples $\mathbf{t}_i = (d_{i1}, d_{i2}, \ldots, d_{im})$ and $\mathbf{t_k} = (d_{k1}, d_{k2}, \ldots, d_{km})$ are redundant if $[d_{ij}] = [d_{kj}]$ for all $j = 1, \ldots, m$.*

In rough relations, there are no redundant tuples. The merging process used in relational database operations removes duplicate tuples since duplicates are not allowed in sets, the structure upon which the relational model is based. However, it is possible for more than one tuple to have the same interpretation. The very nature of roughness in using both lower and upper approximations precludes the restriction of uniqueness in interpretations.

Indiscernibility can be represented in the rough relational database by an additional relation. (See Table VI of the Appendix for an example.) The tuples of this relation represent all of the possible singleton values d_{ij} for every domain D_j. Each tuple also contains an arbitrary indiscernibility identifier that associates the value d_{ij} with the equivalence class to which it belongs. This indiscernibility relation is an integral part of the rough relational database. All database retrieval operations implicitly access the indiscernibility relation in addition to these rough relations expressed in the query.

In addition to indiscernibility, the rough relational database must incorporate lower and upper approximations into the querying in order to retrieve a rough set. We can view rough querying as follows. *First, retrieve the elements of the lower approximation as previously described, utilizing the notion of indiscernibility. Next, retrieve the elements of the upper approximation, also utilizing the indiscernibility relation, and return those tuples that are not also in the lower approximation.* The lower approximation includes all tuples whose individual attribute values are equivalent to those expressed by the query. The upper approximation is based on set containment of values.

Rough Relational Operators

There are two basic types of relational operators. The first type arises from the fact that relations are considered sets of tuples. Therefore, operations that can be applied to sets also apply to relations. The most useful of these for database purposes are *set difference*, *union*, and *intersection*. Operators which do not come from set theory, but which are useful for retrieval of relational data, are *select*, *project*, and *join*.

In the rough relational database, relations are rough sets as opposed to ordinary sets. Therefore, new rough operators ($-, \cup, \cap, \times, \sigma, \pi, \bowtie$), which are comparable to the standard relational operators, must be developed for the rough relational database. Moreover, a mechanism must exist within the database to mark tuples of a rough relation as belonging to the lower or upper approximation of that rough relation. Because the definitions for the rough relational operators that follow are independent of any implementation details, only issues related to the determination of the approximation area to which a tuple belongs will be discussed.

The definition for the set operations for rough relations are comparable to those defined for ordinary relations in the standard relational database model. These binary operations require that the argument relations be "union compatible." Two relations $X(A_1, A_2, \ldots, A_n)$ and $Y(B_1, B_2, \ldots, B_n)$ are union compatible if they have the same number of attributes in their relation schemas and if the domain of A_i is equal to the domain of B_i for all $i = 1, n$.

Difference. The relational difference operator is a binary operator that returns those elements of the first relation which are not elements of the second relation. Let X and Y be two union compatible rough relations.

DEFINITION. *The rough difference,* **X − Y**, *between X and Y is a rough relation T where*

$$\underline{R}T = \{t \in \underline{R}X \mid t \notin \underline{R}Y\} \quad \text{and} \quad \overline{R}T = \{t \in \overline{R}X \mid t \notin \overline{R}Y\}.$$

The lower approximation of $T = X - Y$ contains those tuples belonging to the lower approximation of X which are not redundant with a tuple in the lower approximation of Y. The upper approximation of the rough relation T contains those tuples in the upper approximation of X which are not included in the upper approximation of Y.

For example, consider the sample relations X and Y which contain the tuples below where tuples of the lower approximation region are denoted with an *:

$X =$	(RED, SMALL)*	$Y =$	(RED, SMALL)*
	(BLUE, MEDIUM)*		(YELLOW, SMALL)*
	(YELLOW, SMALL)		(BLUE, MEDIUM)
	(YELLOW, MEDIUM)		(BLUE, LARGE)

Then the difference $X - Y$ contains the tuples (BLUE, MEDIUM)* and (YELLOW, MEDIUM). Other operators are similar.

Union. Let X and Y be two union compatible rough relations.

DEFINITION. *The rough union of X and Y, $X \cup Y$, is a rough relation T where*

$$\underline{R}T = \{t \in \underline{R}X \cup \underline{R}Y\} \quad \textit{and} \quad \overline{R}T = \{t \in \overline{R}X \cup \overline{R}Y\}.$$

The lower approximation of the resulting rough relation T contains those tuples which are a member of either or both of the lower approximations of X and Y, and the upper approximation of T contains tuples which belong to either or both of the upper approximations of X and Y.

Intersection. Rough intersection is defined similarly.

DEFINITION. *The rough intersection of X and Y, $X \cap Y$, is a rough relation T where*

$$\underline{R}T = \{t \in \underline{R}X \cap \underline{R}Y\} \quad \textit{and} \quad \overline{R}T = \{t \in \overline{R}X \cap \overline{R}Y\}.$$

In rough intersection, comparison of tuple values is based on redundancy, as opposed to the standard relational model, which bases comparisons on equality of values. The lower approximation of the resulting rough relation T contains those tuples of the lower approximation of X which have corresponding redundant tuples in the lower approximation of Y, and the upper approximation of T contains tuples of the upper approximation of X which have redundant tuples in the upper approximation of Y.

Other rough relational database operators, which are analogous to the select, project, and join for the ordinary relational database,[15] are also defined.

Selection. The select operator for the rough relational database model, σ, is a unary operator which takes a rough relation X as its argument and returns a rough relation containing a subset of the tuples of X, selected on the basis of values for one or more specified attributes. The operation $\sigma_{A=\mathbf{a}}(X)$, for example, returns those tuples in X where the value for attribute A is equivalent to the value **a**, or more precisely, a member of the equivalence class [**a**].

Let R be a relation schema, X a rough relation on that schema, A an attribute in R, and $\mathbf{a} = \{a_i\}$ where $a_i, b_j \in \operatorname{dom}(A)$. $\cup_x$ denotes "the union over all x"; $t(A)$ denotes a tuple's value for attribute A.

DEFINITION. *The rough selection, $\sigma_{A=\mathbf{a}}(X)$, of tuples from X is a rough relation Y having the same schema as X and where*

$$\underline{R}Y = \{t \in X \mid \cup_i [a_i] = \cup_j [b_j]\}, \qquad a_i \in \mathbf{a}, \quad b_j \in t(A),$$
$$\overline{R}Y = \{t \in X \mid \cup_i [a_i] \subseteq \cup_j [b_j]\}, \qquad a_i \in \mathbf{a}, \quad b_j \in t(A).$$

The lower approximation of $Y = \sigma_{A=\mathbf{a}}(X)$ contains those tuples where the value of attribute A for that tuple is indiscernible from the members of **a**, as indicated in the select condition. The upper approximation contains those tuples where the members of **a** form a subset of the values of attribute A for that tuple.

Projection. Project is a unary operator that takes a relation as its argument and returns a relation containing a subset of the columns of the original relation. Let X be a rough relation with schema A, and let B be a subset of A. The rough projection of X onto schema B is a relation Y obtained by omitting the columns of X which correspond to attributes in $A - B$, and removing redundant tuples.

DEFINITION. *The rough projection of X onto B, $\pi_B(X)$, is a relation Y with schema $Y(B)$ where*

$$Y(B) = \{t(B) \mid t \in X\}.$$

Each $t(B)$ is a tuple retaining only those attributes in the requested set B. Additionally, the rough project must maintain which tuples belong to the lower approximation and which belong to the upper approximation. When comparing tuples for redundancy, if redundant tuples both belong to the lower approximation or both belong to the upper approximation, either can be deleted. In cases where one tuple is from the lower approximation and the other from the upper approximation, the tuple from the lower approximation is retained.

Join. The join operator is a binary operator that takes related tuples from two relations and combines them into single tuples of the resulting relation. It uses common attributes to combine the two relations into one, usually larger, relation. Let $X(A_1, A_2, \ldots, A_m)$ and $Y(B_1, B_2, \ldots, B_n)$ be rough relations with m and n attributes, respectively, and let $AB = C$, the schema of the resulting rough relation T.

DEFINITION. *The rough join, $X \bowtie_{\langle \text{JOIN CONDITION} \rangle} Y$, of two relations X and Y, is a relation $T(C_1, C_2, \ldots, C_{m+n})$ where*

$$T = \{t \mid \exists t_X \in X, t_Y \in Y \textit{ for } t_X = t(A), t_Y = t(B)\}, \textit{ and where}$$
$$t_X(A \cap B) = t_Y(A \cap B), \textit{ for } \underline{R}T$$
$$t_X(A \cap B) \subseteq t_Y(A \cap B) \textit{ or } t_Y(A \cap B) \subseteq t_X(A \cap B), \textit{ for } \overline{R}T$$

⟨JOIN CONDITION⟩ is a conjunction of one or more conditions of the form $A = B$.

Properties of the rough relational operators can be found in Ref. 13. They are not included here because comparable properties of fuzzy rough relational operators will be discussed in a later section.

2.4. Fuzzy Rough Sets

Because there are advantages to both fuzzy set and rough set theories, several researchers have studied various ways of combining the two theories.[7,22,23] Others have investigated the interrelations between the two theories.[24–26] Fuzzy sets and rough sets are not equivalent, but complementary.

It has been shown in Ref. 26 that rough sets can be expressed by a fuzzy membership function $\mu \to \{0, 0.5, 1\}$ to represent the negative, boundary, and positive regions. In this model, all elements of the lower approximation, or positive region, have a membership value of one. Those elements of the upper approximation that are not also in the lower approximation region, i.e., those elements of the boundary region, are assigned a membership value of 0.5. Elements not belonging to the rough set have a membership value of zero. Rough set definitions of union and intersection were modified so that the fuzzy model would satisfy all the properties of rough sets.[25] This allowed a rough set to be expressed as a fuzzy set.

Our purpose in integrating the fuzziness into the rough relational database model is not as a means for expressing rough relations in an alternate manner, but to quantify levels of roughness in boundary region areas through the use of fuzzy membership values. Therefore, the fuzzy rough set should not require membership values of elements of the boundary region to equal 0.5, but allow them to take on values anywhere within the range of real numbers between zero and one, not including zero and one. Additionally, the union and intersection operators for fuzzy rough sets are comparable to those for ordinary fuzzy sets, where MIN and MAX are used to obtain membership values of redundant elements.

Let U be a *universe*, X a rough set in U.

DEFINITION. *A fuzzy rough set Y in U is a membership function $\mu_Y(x)$ which associates a grade of membership from the interval $[0,1]$ with every element of U where*

$$\mu_Y(\underline{R}X) = 1$$

$$\mu_Y(U - \overline{R}X) = 0$$

$$0 < \mu_Y\left(\overline{R}X - \underline{R}X\right) < 1.$$

All elements of the positive region have a membership value of one and elements of the boundary region have a membership value between zero and one.

DEFINITION. *The union of two fuzzy rough sets A and B is a fuzzy rough set C where*

$$C = \{x \mid x \in A \text{ OR } x \in B\}$$

$$\mu_C(x) = \text{MAX}[\,\mu_A(x), \mu_B(x)].$$

DEFINITION. *The intersection of two fuzzy rough sets A and B is a fuzzy rough set C where*

$$C = \{x \mid x \in A \text{ AND } x \in B\}$$

$$\mu_C(x) = \text{MIN}[\,\mu_A(x), \mu_B(x)].$$

3. THE FUZZY ROUGH RELATIONAL DATABASE MODEL

3.1. Introduction and Definitions

The fuzzy rough relational database, as in the ordinary relational database, represents data as a collection of *relations* containing *tuples*. Because a relation is considered a set having the tuples as its members, the tuples are unordered. In addition, there can be no duplicate tuples in a relation. A tuple $\mathbf{t}_i$ takes the form $(d_{i1}, d_{i2}, \dots, d_{im}, d_{i\mu})$, where d_{ij} is a *domain value* of a particular *domain set* D_j and $d_{i\mu} \in D_\mu$, where D_μ is the interval $[0,1]$, the domain for fuzzy membership values. In the ordinary relational database, $d_{ij} \in D_j$. In the fuzzy rough relational database, except for the fuzzy membership value, however, $d_{ij} \subseteq D_j$, and although d_{ij} is not restricted to be a singleton, $d_{ij} \neq \emptyset$. Let $P(D_i)$ denote any nonnull member of the powerset of D_i.

DEFINITION. *A fuzzy rough relation R is a subset of the set cross product* $P(D_1) \times P(D_2) \times \cdots \times P(D_m) \times D_\mu$.

For a specific relation, R, membership is determined semantically. Given that D_1 is the set of names of nuclear/chemical plants, D_2 is the set of locations, and assuming that RIVERB is the only nuclear power plant that is located in VENTRESS,

$$\begin{gathered}(\text{RIVERB}, \text{VENTRESS}, 1)\\(\text{RIVERB}, \text{OSCAR}, .7)\\(\text{RIVERB}, \text{ADDIS}, 1)\\(\text{CHEMO}, \text{VENTRESS}, 3)\end{gathered}$$

are all elements of $P(D_1) \times P(D_2) \times D_\mu$. However, only the element (RIVERB, VENTRESS, 1) of those listed above is a member of the relation $R(\text{PLANT}, \text{LOCATION}, \mu)$, which associates each plant with the town or community in which it is located. A *fuzzy rough tuple* $\mathbf{t}$ is any member of R. If $\mathbf{t_i}$ is some arbitrary tuple, then $\mathbf{t_i} = (d_{i1}, d_{i2}, \dots, d_{im}, d_{i\mu})$, where $d_{ij} \subseteq D_j$ and $d_{i\mu} \in D_\mu$.

DEFINITION. *An interpretation* $\alpha = (a_1, a_2, \dots, a_m, a_\mu)$ *of a fuzzy rough tuple* $\mathbf{t_i} = (d_{i1}, d_{i2}, d_{i2}, \dots, d_{im}, d_{i\mu})$ *is any value assignment such that* $a_j \in d_{ij}$ *for all* j.

The interpretation space is the cross product $D_1 \times D_2 \times \cdots \times D_m \times D_\mu$, but is limited for a given relation R to the set of those tuples which are valid according to the underlying semantics of R. In an ordinary relational database, because domain values are atomic, there is only one possible interpretation for each tuple $\mathbf{t_i}$. Moreover, the interpretation of $\mathbf{t_i}$ is equivalent to the tuple $\mathbf{t_i}$. In the fuzzy rough relational database, this is not always the case.

Let $[d_{xy}]$ denote the equivalence class to which d_{xy} belongs. When d_{xy} is a set of values, the equivalence class is formed by taking the union of equivalence classes of members of the set; if $d_{xy} = \{c_1, c_2, \dots, c_n\}$, then $[d_{xy}] = [c_1] \cup [c_2] \cup \cdots \cup [c_n]$.

DEFINITION. *Tuples* $\mathbf{t_i} = (d_{i1}, d_{i2}, \dots, d_{in}, d_{i\mu})$ *and* $\mathbf{t_k} = (d_{k1}, d_{k2}, \dots, d_{kn}, d_{k\mu})$ *are redundant if* $[d_{ij}] = [d_{kj}]$ *for all* $j = 1, \dots, n$.

If a relation contains only those tuples of a lower approximation, i.e., those tuples having a μ value equal to one, the interpretation α of a tuple is unique. This follows immediately from the definition of redundancy. In fuzzy rough relations, there are no redundant tuples. The merging process used in relational database operations removes duplicate tuples since duplicates are not allowed in sets, the structure upon which the relational model is based.

Tuples may be redundant in all values except μ. As in the union of fuzzy rough sets where the maximum membership value of an element is retained, it is the convention of the fuzzy rough relational database to retain the tuple having the higher μ value when removing redundant tuple during merging. If we are supplied with identical data from two sources, one certain and the other uncertain, we would want to retain the data that is certain, avoiding loss of information.

Recall that the rough relational database is in non-first normal form; there are some attribute values which are sets. Another definition, which will be used for upper approximation tuples, is necessary for some of the alternate definitions of operators to be presented. This definition captures redundancy between elements of attribute values that are sets.

DEFINITION. *Two sub-tuples $X = (d_{x1}, d_{x2}, \ldots, d_{xm})$ and $Y = (d_{y1}, d_{y2}, \ldots, d_{ym})$ are roughly redundant, $\approx_{\mathrm{R}}$, if for some $[p] \subseteq [d_{xj}]$ and $[q] \subseteq [d_{yj}]$, $[p] = [q]$ for all $j = 1, \ldots, m$.*

In order for any database to be useful, a mechanism for operating on the basic elements and retrieving specified data must be provided. The concepts of redundancy and merging play a key role in the operations defined in Section 3.3.

3.2. An Example Application: The Public Concerns Database

The example we present in this section is a database used in the documentation of public concerns. For simplicity and space limitations, only a subset of the complete database is included. However, it is a nontrivial real-world example that illustrates the necessity of incorporating uncertainty in databases.

A government agency is studying the concerns of citizens who reside near or are employed by one of several potentially hazardous nuclear or chemical plants. A study is being conducted which documents these concerns and stores all the data in a database. Several public meetings and rallies were conducted to promote public involvement in the project and to gather information from the participants about their concerns. The study requires the gathering and analysis of demographic data to determine issues related to senior citizens, minorities, and families.

We must first design our database using some type of semantic model. We use a variation of the entity–relationship diagram that we call a fuzzy rough E-R diagram. This diagram is similar to the standard E-R diagram in that entity types are depicted with rectangles, relationships with diamonds, and attributes with ovals. However, in the fuzzy rough model, it is understood that membership values exist for all instances of entity types and relationships. Attributes which allow values where we want to be able to define equivalences are denoted with an asterisk (*) above the oval. These values are defined in the indiscernibility relation, which is not actually part of the database design, but inherent in the fuzzy rough model.

Our fuzzy rough E-R model is similar to the second and third levels of fuzziness defined by Zvieli and Chen.[29] However, in our model, all entity and relationship occurrences (second level) are of the fuzzy type so we do not mark an ‘f’ beside each one. Zvieli and Chen’s third level considers attributes that may be fuzzy. They used triangles instead of ovals to represent these attributes. We do not introduce fuzziness at the attribute level of our model in this paper, only roughness, or indiscernibility, and denote those attributes with the “*.” A fuzzy rough E-R diagram for our example appears in Figure 1.

From the fuzzy-rough E-R diagram, we design the structure of the fuzzy rough relational database. If we have *a priori* information about the types of queries that will be involved, we can make intelligent choices that will maximize computer resources. In the example, most of our queries deal with demographic information of participants at the various events such as Rally 1. We also assume that there are not very many people who attend more than one event. Therefore, for this illustration, we will store age, sex, and race attribute values with events and link the information via the ID attribute to the fuzzy rough relation PEOPLE, rather than storing these attributes within PEOPLE. Under other conditions, a different design may be more appropriate.

We introduce a fuzzy rough data definition language (DDL) to define the fuzzy rough relations and the indiscernibility relation. The fuzzy rough DDL is similar to that of SQL, having such commands as CREATE, DROP, etc., but for fuzzy rough relations.

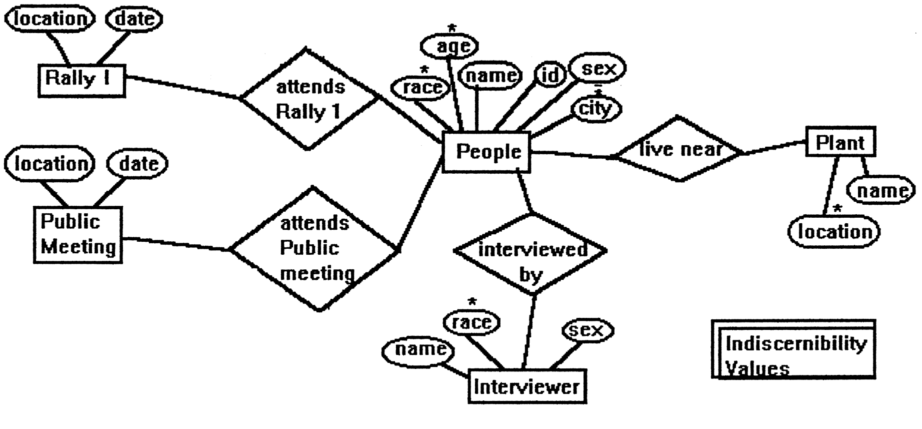

Figure 1. A fuzzy-rough E-R diagram for the Public Concerns Database.

The FRCREATE DDL command creates a base table that is fuzzy rough in the following ways. First of all, it contains an additional attribute called MU which draws values from the range $[0,1]$, tuple membership values. This attribute does not have to be specified. It is automatically included as part of all fuzzy rough relations. Additionally, we can specify for each attribute whether or not we allow indiscernibility of values. This is defined by including "IND" along with the attribute line of the table definition. The FRCREATE command is used to define the fuzzy rough relations of this example. The representative tables defined and explained in this section can be found in the Appendix.

(1)
```
FRCREATE TABLE PEOPLE
          (ID          DECIMAL(4),
           NAME        CHAR(20),
           CITY        CHAR(20)      IND,
          PRIMARY KEY (ID));
```

PEOPLE (Table I): associates a unique id number for each person about whom we have some data. Each tuple contains an ID, the person's NAME, and the CITY in which he or she resides.

This information is usually easy to obtain from surveys, or from the public meetings since every speaker is required to state his or her name and address. It is possible, however, that some of this information may be uncertain or unavailable. Perhaps handwritten words may be illegible or spoken words unclear. Sometimes it is possible to make a "good guess" at the uncertain part, at the same time acknowledging the fact that uncertainty is present. Another possibility is that a person may provide a street address, but no city. If we know that "Oakwood Drive" is the name of a street in one of the cities of interest, we may be reasonably sure that the person lives in that city. However, we cannot be certain since several towns have common street names. Still, a reasonable guess associated with some measure of uncertainty provides more information than does no information at all. It may also be the case in rural areas that a person does not live in any town at all. Often he or she will name one of several nearby towns when asked "Where do you live?"

(2)
```
FRCREATE TABLE RALLY1
          (ID          DECIMAL(4),
           SEX         CHAR(1),
           RACE        CHAR(10)      IND,
           AGE         CHAR(15)      IND,
          PRIMARY KEY (ID));
```

RALLY1 (Table II): stores information about the SEX, RACE, and AGE of all participants at the first Rally that can be identified by name. The ID field associates the information in this fuzzy rough relation with that in PEOPLE.

(3) PUBLIC-MEETING (Table III): like RALLY1, this fuzzy rough relation contains the SEX, RACE, and AGE information for individuals. However, these individuals were participants at the public meeting.

Accurate data can be obtained for the relations RALLY1 and PUBLIC-MEETING if it is willingly and truthfully provided by the participants as part of a survey or as responses to questions in an interview. However, many people choose not to answer these (usually optional) questions or to answer them in a way that is either inconsistent or inaccurate.

Sometimes this data has to be approximated by a trained observer, and some traits are easier to approximate than others. In this study, it is not necessary to know the exact age in years of a person, only the approximate age for classification as CHILD, ADULT, or SR-CITIZEN, for example. This makes it easier for an observer to approximate a person's age unless the observer is uncertain about which age group is most appropriate. In this case, we may want to include both age groups, along with some information about uncertainty.

Another problem arises when there are multiple observers recording data about participants. If there are inconsistencies in categorizing data, we may not know which observation, if either, is more correct than the other. In the fuzzy rough relational database model, we can incorporate all of these types of uncertainty, rather than discarding the data as invalid.

We also need to store information in the database about the interviewers. The data may be required in studies involving observer bias or discrimination. If the interviewers are trusted colleagues or employees, we can be reasonably certain about the data in this relation.

```
(4) FRCREATE TABLE INTERVIEWERS
    (NAME CHAR(20),
     SEX         CHAR(1),
     RACE        CHAR(10)       IND,
    PRIMARY KEY (NAME));
```

INTERVIEWERS (Table IV): contains the NAME, SEX, and RACE of each interviewer.

Information on the various chemical and nuclear plants is also of interest. In this example, we show only information related to the location of the plant. This information may not be as "uncertain" as that of some of the other relations since information about the plants' locations is readily available.

```
(5) FRCREATE TABLE PLANTS
    (COMPANY       CHAR(10),
     LOCATION      CHAR(20)       IND,
    PRIMARY KEY (COMPANY));
```

PLANTS (Table V): contains the NAMEs and LOCATIONs of plants of concern to citizens.

We may find that several possible values for a given attribute assigned by an observer or participant may be equivalent in terms of the study. For example, one observer may note that a participant is a TEEN, whereas another may record the age TEENAGER. We want to allow the incorporation of data from different sources and let the database model keep track of these equivalences. The indiscernibility relation of the rough set theory is ideally suited for this purpose. In the fuzzy rough relational database, the designer sets up a special relation that contains information about indiscernibility:

(6)
```
FRCREATE TABLE INDISCERNIBILITY
         (IND         DECIMAL(3),
          VALUE       CHAR(20));
```

INDISCERNIBILITY (Table VI): contains legal values for attributes and an indiscernibility identifier IND that denotes the equivalence class to which values belong. All VALUE components of tuples in this relation that have the same IND components belong to a common equivalence class.

Once the database schema has been defined, we may begin to store actual data in the fuzzy rough relations. Often database packages have utility programs to expedite this process. Alternatively, we can directly enter data into a relation with the SQL INSERT command. In the fuzzy relational database, the command is similar. Data values for all attributes including the membership value MU are inserted into the specified relation. If a value for MU is not included, it is automatically assigned a default value of 1. This saves considerable data entry time since many of the data items typically have a MU equal to 1. The fuzzy rough counterpart to SQL's INSERT is FRINSERT:

```
FRINSERT
INTO PLANTS
VALUES (RIVERB, VENTRESS, 1);
```

The designer/user may use the FRINSERT command to enter tuples in the INDISCERNIBILITY relation since it is, after all, a fuzzy rough relation. However, the fuzzy rough INDISCERNIBILITY relation is a special one, used only for the grouping of similar attribute values into equivalence classes. Therefore, we introduce some new commands that will facilitate the creation of classes of equivalent values. All membership values for tuples in this relation are automatically set to 1, since we are not introducing fuzziness or similarities among attribute values into the database at this time. The indiscernibility identifier ID serves to specify values that are indiscernible. The actual value of ID is irrelevant, as long as all tuples belonging to a given class have identical values for the attribute ID. Therefore, all the user needs to specify are values to be grouped into a class, and the system can set up the tuples in the INDISCERNIBILITY relation, providing system-supplied identifiers for classes. In order to create a new equivalence class, the FRCLASS command is used:

```
FRCLASS (TEEN, YOUTH, TEENAGER);
```

After verifying that none of 'TEEN,' 'YOUTH,' or 'TEENAGER' already exist in the INDISCERNIBILITY relation, three new tuples will be inserted, all three having identical values for the attribute IND, which for the example relation of Table VI is 102.

There may be times when we want to add a value to a class, rather than create a new class. For example, if we want to add the value 'ADOLESCENT' to the class we just created, we use the FRADD command, specifying any of the three values that already belong to the class. For example,

```
FRADD CLASS TEEN
    (ADOLESCENT) ;
```

We may also remove a value from a class using FRREMOVE, or delete an entire class, including all its members, with FRDELETE as shown below:

```
FRREMOVE CLASS TEEN          FRDELETE CLASS TEENAGER;
    (ADOLESCENT) ;
```

The fuzzy rough relational database commands FRCLASS, FRADD, FR-REMOVE, and FRDELETE are special commands to facilitate operations involving indiscernibility and the updating of this special relation of equivalence classes. The commands FRINSERT and FRCREATE are analogous to their SQL counterparts in standard relational databases. The fuzzy rough relational database also has the usual SQL DDL and update commands for deleting or updating tuples (FRDELETE, FRUPDATE) and dropping tables (FRDROP). These operate on fuzzy rough relations as DELETE, UPDATE, and DROP TABLE commands operate on ordinary relations. Refer to Refs. 14, 28, 29 for more information on SQL and data definition languages.

Uncertainty, ambiguity, and indiscernibility are all prevalent in the Public Concerns Database. In the next section we formally define the fuzzy rough relational database operators and discuss issues relating to the real-world problems of data representation and modeling. Informally, however, we view indiscernibility as being modeled through the use of the indiscernibility relation, imprecision through the use of non-first normal form constructs, and degree of uncertainty and fuzziness through the use of tuple membership values, which are given as the value for the MU attribute in every fuzzy rough relation.

3.3. Fuzzy Rough Relational Operators

In the background material on the rough relational database model, we defined several operators for the rough relational algebra. We now define similar operators for the fuzzy rough relational database and demonstrate the expressive power of the model through its fuzzy rough relational algebra with example queries to the "Public Concerns" Database. Recall that for all of these operators, the indiscernibility relation is used for equivalence of attribute values rather than equality of values.

3.3.1. Difference

The fuzzy rough relational difference operator is very much like the ordinary difference operator in relational databases and in sets in general. It is a binary operator that returns those elements of the first argument that are not contained in the second argument.

In the fuzzy rough relational database, the difference operator is applied to two fuzzy rough relations and, as in the rough relational database, indiscernibility, rather than equality of attribute values, is used in the elimination of redundant tuples. Hence, the difference operator is somewhat more complex. Let X and Y be two union compatible fuzzy rough relations.

DEFINITION. *The fuzzy rough difference, $X - Y$, between X and Y is a fuzzy rough relation T where*

$$T = \{t(d_1, \ldots, d_n, \mu_i) \in X \mid t(d_1, \ldots, d_n, \mu_i) \notin Y\}$$
$$\cup \{t(d_1, \ldots, d_n, \mu_i) \in X \mid t(d_1, \ldots, d_n, \mu_j) \in Y \text{ and } \mu_i > \mu_j\}$$

The resulting fuzzy rough relation contains all those tuples which are in the lower approximation of X, but not redundant with a tuple in the lower approximation of Y. It also contains those tuples belonging to upper approximation regions of both X and Y, but which have a higher μ value in X than in Y. For example, let X contain the tuple (ELDERLY, 1) and Y contain the tuple (ELDERLY, .02). It would not be desirable to subtract out certain information with possible information, so $X - Y$ yields (ELDERLY, 1).

Consider the fuzzy rough relation RALLY1 and PUBLIC-MEETING in Tables II and III in the Appendix. The query "*Retrieve the demographic information on individuals who participated in the first rally, but did not take part in the public meeting*" can be expressed as the fuzzy rough difference of the two relations: RALLY1 − PUBLIC-MEETING, which yields

ID	SEX	RACE	AGE	MU
4601	M	W	ADULT	1
4602	M	{W, H}	ADULT	.9
4603	F	H	SR	1
4604	F	W	ELDERLY	1
4606	F	B	ADULT	1
{4603, 4608}	{M, F}	H	ADULT	.6

3.3.2. *Union*

Because relations in databases are considered as sets, the union operator can be applied to any two union compatible relations to result in a third relation which has as its tuples all the tuples contained in either or both of the two original relations. The union operator can be extended to apply to fuzzy rough relations. Let X and Y be two union compatible fuzzy rough relations.

DEFINITION. *The fuzzy rough union* of X and Y, $X \cup Y$, is a fuzzy rough relation T where

$$T = \{t \mid t \in X \text{ OR } t \in Y\} \qquad \text{and} \qquad \mu_T(t) = \text{MAX}[\mu_X(t), \mu_Y(t)].$$

The resulting relation T contains all tuples in either X or Y or both, merged together and having redundant tuples removed. If X contains a tuple that is redundant with a tuple in Y except for the μ value, the merging process will retain only that tuple with the higher μ value.

The fuzzy rough union of the relations RALLY1 and PUBLIC-MEETING satisfies the high level query "*List all demographic information for individuals who participated in either the rally or the public meeting or both*," which results in the following:

ID	SEX	RACE	AGE	MU
4601	M	W	ADULT	1
4602	M	{W, H}	ADULT	.9
4603	F	H	SR	1
4604	F	W	ELDERLY	1
4605	M	WHITE	ADULT	1
4606	F	B	ADULT	1
4607	M	B	SR	1
{4603, 4608}	{M, F}	H	ADULT	.6
5100	F	B	ADULT	1
5101	F	B	{CHILD, TEEN}	.8
4603	F	H	{ADULT, SR}	.6

3.3.3. Intersection

The fuzzy rough intersection, another binary operator on fuzzy rough relations, can be defined similarly.

DEFINITION. *The fuzzy rough intersection of X and Y, $X \cap Y$, is a fuzzy rough relation T where*

$$T = \{t \mid t \in X \text{ AND } t \in Y\} \qquad \text{and} \qquad \mu_T(t) = \text{MIN}[\mu_X(t), \mu_Y(t)].$$

In intersection, the MIN operator is used in the merging of equivalent tuples having different μ values and the result contains all tuples that are members of both of the original fuzzy rough relations.

The query "*Retrieve all demographic information for those citizens who are so concerned about the hazards of the plants that they have attended both the rally and the public meeting*" can be formulated as a fuzzy rough intersection of RALLY1 and PUBLIC-MEETING, yielding

ID	SEX	RACE	AGE	MU
4602	M	{W, H}	ADULT	.7
4605	M	WHITE	ADULT	1
4607	M	B	SR	1

Notice in the above example that data about the person having ID 4603 was not included in the result of this intersection operation, although there was data involving this person in both of the original fuzzy rough relations. Because of this loss of information and because of some problems with combinations of other operations that will be discussed later, we propose an alternate definition for intersection. This intersection will be distinguished from the previous operator by an A subscript to denote the *Alternate* form of intersection.

DEFINITION. *The fuzzy rough intersection of X and Y, $X \cap_{\mathrm{A}} Y$, is a fuzzy rough relation T where*

$$T = \{t \mid t \in X, \text{ and } \exists s \in Y \mid t \approx_{\mathrm{R}} s\} \cup \{s \mid s \in Y, \text{ and } \exists t \in X \mid s \approx_{\mathrm{R}} t\} \quad \text{and}$$
$$\mu_T(t) = \mathrm{MIN}[\mu_X(t), \mu_Y(t)].$$

The result of the previous example is different when this alternate definition of intersection is applied. T = RALLY1 $\cap_{\mathrm{A}}$ PUBLIC-MEETING yields the following result:

ID	SEX	RACE	AGE	MU
4602	M	{W, H}	ADULT	.7
4603	F	H	SR	1
4605	M	WHITE	ADULT	1
4607	M	B	SR	1
{4603, 4608}	{M, F}	H	ADULT	.6
4603	F	H	{ADULT, SR}	.6

3.3.4. *Select*

The select operator for the fuzzy rough relational database model, σ, is a unary operator which takes a fuzzy rough relation X as its argument and returns a fuzzy rough relation containing a subset of the tuples of X, selected on the basis of values for a specified attribute. The operation $\sigma_{A=\mathbf{a}}(X)$, for example, returns those tuples in X where attribute A is equivalent to the class $[\mathbf{a}]$. In general, select returns a subset of the tuples that match some selection criteria.

Let R be a relation schema, X a fuzzy rough relation on that schema, A an attribute in R, $\mathbf{a} = \{a_i\}$ and $\mathbf{b} = \{b_j\}$, where $a_i, b_j \in \operatorname{dom}(A)$, and $\cup_x$ is interpreted as "the union over all x."

DEFINITION. *The fuzzy rough selection, $\sigma_{A=\mathbf{a}}(X)$, of tuples from X is a fuzzy rough relation Y having the same schema as X and where*

$$Y = \{t \in X \mid \cup_i[a_i] \subseteq \cup_j[b_j]\},$$

where $a_i \in \mathbf{a}$, $b_j \in t(A)$, and where membership values for tuples are calculated by multiplying the original membership value by

$$\operatorname{card}(\mathbf{a})/\operatorname{card}(\mathbf{b})$$

where $\operatorname{card}(x)$ *returns the cardinality, or number of elements, in x.*

Assume we want to retrieve those elements where CITY = "ADDIS" from the following fuzzy rough tuples:

(ADDIS	1)
({ADDIS, LOTTIE, BRUSLY}	.7)
(OSCAR	1)
({ADDIS, JACKSON}	.9)

The result of the selection is the following:

(ADDIS	1)
({ADDIS, LOTTIE, BRUSLY}	.23)
({ADDIS, JACKSON}	.45)

where the μ for the second tuple is the product of the original membership value .7 and $1/3$.

Consider the relation PUBLIC-MEETING in Table III in the Appendix. The query "*List the demographic data for all adults who attended the public meeting*" can be expressed as a selection operation with the condition AGE = 'ADULT' on PUBLIC-MEETING, or $\sigma_{\text{AGE}=\text{'ADULT'}}$(PUBLIC-MEETING), which yields the following fuzzy rough relation:

ID	SEX	RACE	AGE	MU
4605	M	W	ADULT	1
5100	F	B	ADULT	1
4603	F	H	{ADULT, SR}	.3
4602	M	{W, H}	ADULT	.7

3.3.5. Project

Project is a unary fuzzy rough relational operator. It returns a relation that contains a subset of the columns of the original relation. Let X be a fuzzy rough relation with schema A, and let B be a subset of A. The fuzzy rough projection of X onto B is a fuzzy rough relation Y obtained by omitting the columns of X which correspond to attributes in $A - B$, and removing redundant tuples. Recall that the definition of redundancy accounts for indiscernibility, which is central to the rough sets theory, and that higher μ values have priority over lower ones.

DEFINITION. *The fuzzy rough projection of X onto B, $\pi_B(X)$, is a fuzzy rough relation Y with schema $Y(B)$ where $Y(B) = \{t(B) \mid t \in X\}$.*

The query "*List all ages represented at the public meeting*" can be expressed as a fuzzy rough projection on the attribute AGE of the PUBLIC-MEETING relation. This operation projects out all other attributes and eliminates redundant tuples. Note in the result that those tuples having higher μ values are retained during the merging process:

AGE	MU
ADULT	1
SR	1
{CHILD, TEEN}	.8
{ADULT, SR}	.6

3.3.6. Join

Join is a binary operator that takes related tuples from two relations and combines them into single tuples of the resulting relation. It uses common attributes to combine the two relations into one, usually larger, relation. Let $X(A_1, A_2, \dots, A_m)$ and $Y(B_1, B_2, \dots, B_n)$ be fuzzy rough relation with m and n attributes, respectively, and $AB = C$, the schema of the resulting fuzzy rough relation T.

DEFINITION. *The fuzzy rough join, $X \bowtie_{\langle\text{JOIN CONDITIONS}\rangle} Y$, of two relations X and Y, is a relation $T(C_1, C_2, \ldots, C_{m+n})$ where*

$$T = \{t \mid \exists t_X \in X, t_Y \in Y \text{ for } t_X = t(A), t_Y = t(B)\},$$

and where

(1) $t_X(A \cap B) = t_Y(A \cap B)$, $\mu = 1$
(2) $t_X(A \cap B) \subseteq t_Y(A \cap B)$ or $t_Y(A \cap B) \subseteq t_X(A \cap B)$, $\mu = \text{MIN}(\mu_X, \mu_Y)$

$\langle$JOIN CONDITION$\rangle$ is a conjunction of one or more conditions of the form $A = B$.

Only those tuples which resulted from the "joining" of tuples that were both in lower approximations in the original relations belong to the lower approximation of the resulting fuzzy rough relation. All other "joined" tuples belong to the upper approximation only (the boundary region), and have membership values less than one. The fuzzy membership value of the resultant tuple is simply calculated as in Ref. 4 by taking the minimum of the membership values of the original tuples. Taking the minimum value also follows the logic of Ref. 6, where in joins of tuples with different levels of information uncertainty, the resultant tuple can have no greater certainty than that of its least certain component.

Suppose it is necessary to relate individuals to the plants that are located in the communities where they live: "*Retrieve id, name, and city for people, and the plant's name and location such that a plant is located in the same community as the person.*" This query can be expressed as the fuzzy rough join, PEOPLE $\bowtie_{\text{CITY} = \text{LOCATION}}$ PLANT, yielding:

ID	NAME	CITY	COMPANY	LOCATION	MU
4601	Jay Reed	OSCAR	RIVERB	VENTRESS	1
4602	Joe Diaz	{VENTRESS, ADDIS}	RIVERB	VENTRESS	.5
4602	Joe Diaz	{VENTRESS, ADDIS}	CHEMO	ADDIS	.5
4603	Tera Lunt	NEW-ROADS	RIVERB	VENTRESS	1
4604	Anna Lone	OSCAR	RIVERB	VENTRESS	1
4605	Dave Hart	{OSCAR, ADDIS}	RIVERB	VENTRESS	.5
4605	Dave Hart	{OSCAR, ADDIS}	CHEMO	ADDIS	.5
4606	Dione Sand	NEW-ROADS	RIVERB	VENTRESS	1
4607	Frank Person	NEW-ROADS	RIVERB	VENTRESS	1
4608	Tim Lunt	NEW-ROADS	RIVERB	VENTRESS	1
5100	Lena Norr	ADDIS	CHEMO	ADDIS	1
5100	Lena Norr	ADDIS	KBUR	{SUN, BRULY}	.6
5101	Joan Mann	BRULY	CHEMO	ADDIS	1
5101	Joan Mann	BRULY	KBUR	{SUN, BRULY}	.6

Notice in the above example that there are some JOINs which did not occur, but which we may wish to include. For example, although Dave Hart may live in ADDIS where it is possible that KBUR is located, that tuple combination is not returned as a result of the JOIN. Therefore, we propose an alternate definition for JOIN that is based on the concept of fuzzy rough redundancy of tuples. As in the alternate definition for intersection, the alternate join operator will be denoted by a subscript A.

DEFINITION. *The fuzzy rough join,* $X \bowtie_{A\langle \text{JOIN CONDITION} \rangle} Y$, *of two relations* X *and* Y, *is a relation* $T(C_1, C_2, \ldots, C_{m+n})$ *where*

$$T = \{t \mid \exists t_X \in X, t_Y \in Y \text{ for } t_X = t(A), t_Y = t(B)\},$$

and where

(1) $t_X(A \cap B) = t_Y(A \cap B)$, $\mu = 1$
(2) $t_X(A \cap B)$ is roughly redundant with $t_Y(A \cap B)$, $\mu = \text{MIN}(\mu_X, \mu_Y)$

⟨JOIN CONDITION⟩ is a conjunction of one or more conditions of the form $A = B$.

The following fuzzy rough relation is the result of the previous query, but this time employing the alternate definition of join, $\bowtie_A$. Note that it contains all tuples that were in the previous result plus some other *possible* tuples. The designer of a database may decide which operation is more appropriate to incorporate when an alternate operation is available, or both operations could be included, allowing the experienced user to choose between the two.

ID	NAME	CITY	COMPANY	LOCATION	MU
4601	Jay Reed	OSCAR	RIVERB	VENTRESS	1
4602	Joe Diaz	{VENTRESS, ADDIS}	RIVERB	VENTRESS	.5
4602	Joe Diaz	{VENTRESS, ADDIS}	CHEMO	ADDIS	.5
4602	Joe Diaz	{VENTRESS, ADDIS}	KBUR	{SUN, BRULY}	.5
4603	Tera Lunt	NEW-ROADS	RIVERB	VENTRESS	1
4604	Anna Lone	OSCAR	RIVERB	VENTRESS	1
4605	Dave Hart	{OSCAR, ADDIS}	RIVERB	VENTRESS	.5
4605	Dave Hart	{OSCAR, ADDIS}	CHEMO	ADDIS	.5
4605	Dave Hart	{OSCAR, ADDIS}	KBUR	{SUN, BRULY}	.5
4606	Dione Sand	NEW-ROADS	RIVERB	VENTRESS	1
4607	Frank Person	NEW-ROADS	RIVERB	VENTRESS	1
4608	Tim Lunt	NEW-ROADS	RIVERB	VENTRESS	1
5100	Lena Norr	ADDIS	CHEMO	ADDIS	1
5100	Lena Norr	ADDIS	KBUR	{SUN, BRULY}	.6
5101	Joan Mann	BRULY	CHEMO	ADDIS	1
5101	Joan Mann	BRULY	KBUR	{SUN, BRULY}	.6

3.4. Properties of Fuzzy Rough Relational Operators

There are many properties of the relational algebra which are proven to hold under all conditions. Most of these deal with equivalence of results of various combinations of operations. Many of these properties remain valid when the model is extended to incorporate fuzzy rough sets as the structure upon which relations are based.

One property of interest is the ability to express intersection in terms of difference:

$$A \cap B = A - (A - B).$$

Let A = RALLY1, B = PUBLIC-MEETING. Previously, we have calculated both $A \cap B$ and $A - B$ so we can compute the right-hand side of the equation and compare it to $A \cap B$. $A - (A - B)$ results in the following fuzzy rough relation:

ID	SEX	RACE	AGE	MU
4605	M	WHITE	ADULT	1
4607	M	B	SR	1

which is certainly not equal to $A \cap B$. Let us see what happens when we reverse the original relations, letting A = PUBLIC-MEETING and B = RALLY1. The left-hand side of the equation will be unaffected since $A \cap B = B \cap A$. On the right-hand side, we must now calculate $B - (B - A)$. First we obtain the difference $B - A$, which is (PUBLIC-MEETING − RALLY1):

ID	SEX	RACE	AGE	MU
5100	F	B	ADULT	1
5101	F	B	{CHILD, TEEN}	.8
4603	F	H	{ADULT, SR}	.6

We now subtract this fuzzy rough relation from B to obtain

ID	SEX	RACE	AGE	MU
4602	M	{W, H}	ADULT	.7
4605	M	WHITE	ADULT	1
4607	M	B	SR	1

which is not the same result as before. This one, however, is equal to the left-hand side. Neither of the right-hand-side results equal the intersection when the alternate definition ($\cap_A$) is applied. Therefore, because of the varying levels of uncertainties in similar tuples and the properties of the difference operator, the property $A \cap B = A - (A - B)$ does not always hold in the fuzzy rough relational database.

A property of the rough projection operator is that for a string of projections upon a relation Y having schema R, where $X_1 \subseteq X_2 \subseteq \cdots \subseteq X_n \subseteq R$, only the outermost projection operator is necessary. Note, however, that because of indiscernibility, we are dealing with equivalence class values and not ordinary values in the removal of redundant tuples:

$$\pi_{X1}(\pi_{X2}(\dots(\pi_{Xn}(Y))\dots)) = \pi_{X1}(Y).$$

Because each set of attributes X_i is included in the set X_{i+1}, and because at every step of the sequence of projections on the left side of the equality a subset of the attributes is retained and redundant tuples removed until we reach the minimum subset X_1, the same rough relation would result by taking the subset of attributes X_1 to begin with and removing redundant tuples all at once.

The operations on both sides of the equality produce relations which are equal in the sense that every tuple in one rough relation has a corresponding tuple in the other rough relation such that the tuples are indiscernible from each other. In other words, every tuple of one relation is redundant with one and only one tuple of the other relation. Consider, for example, the operation

$$\pi_{\mathrm{RACE}}(\pi_{\mathrm{RACE,SEX}}(\text{PUBLIC-MEETING})).$$

This is equal to $\pi_{\mathrm{RACE}}(A)$, where A is the result of the inner projection shown in the following:

RACE	SEX	MU
W	M	1
B	M	1
B	F	1
H	F	.6
{W, H}	M	.7

The second projection operation results in the following:

RACE	MU
W	1
B	1
H	.6
{W, H}	.7

which is the same result that would have been obtained by the operation

$$\pi_{\mathrm{RACE}}(\text{PUBLIC-MEETING}).$$

Another interesting property is the distribution of the select operator over the Boolean set operators. This property states that for an operator γ, where $\gamma \in \{\cup, \cap, -\}$, and two relations X and Y,

$$\sigma_{A=\mathbf{a}}(X \gamma Y) = \sigma_{A=\mathbf{a}}(X)\, \gamma\, \sigma_{A=\mathbf{a}}(Y).$$

The proof of this property is given for distribution of selection over intersection.

Proof. $\sigma_{A=\mathbf{a}}(X \cap_{\mathrm{A}} Y) = \sigma_{A=\mathbf{a}}(T)$, where

$$\begin{aligned}
T &= \{t \mid t \in X \text{ and } \exists s \in Y \mid t \approx_{\mathrm{R}} s\} \cup \{s \mid s \in Y, \text{and } \exists t \in X \mid s \approx_{\mathrm{R}} t\} \\
&= \Big\{t' \in \big\{\{t \mid t \in X \text{ and } \exists s \in Y \mid t \approx_{\mathrm{R}} s\} \\
&\qquad \cup \{t \mid t \in Y \text{ and } \exists s \in Y \mid t \approx_{\mathrm{R}} s\}\big\} \Big| \cup_i [a_i] \subseteq \cup_j [b_j] \Big\}
\end{aligned}$$

$a_i \in \mathbf{a}, b_j \in t'(A)$

$$\begin{aligned}
&= \Big\{\{t \mid t \in X \text{ and } \cup_i[a_i] \subseteq \cup_j[b_j]\} \cap_{\mathrm{A}} \{t \mid t \in Y \text{ and } \cup_i[a_i] \subseteq \cup_j[b_j]\}\Big\} \\
&= \sigma_{A=\mathbf{a}}(\{t \mid t \in X\}) \cap_{\mathrm{A}} \sigma_{A=\mathbf{a}}(\{t \mid t \in Y\}) \\
&= \sigma_{A=\mathbf{a}}(X) \cap_{\mathrm{A}} \sigma_{A=\mathbf{a}}(Y).
\end{aligned}$$

■

Consider the following example, which refers to Tables II and III of the Appendix. Let X = RALLY1, Y = PUBLIC-MEETING, and let us first evaluate the left-hand side of the proof using the data from RALLY1 and PUBLIC-MEETING: $T1$ = RALLY1 $\cap_{\mathrm{A}}$ PUBLIC-MEETING yields

ID	SEX	RACE	WCAGE	MU
4602	M	{W, H}	ADULT	.9
4603	F	H	SR	1
4603	F	H	{ADULT, SR}	.6
4605	M	W	ADULT	1
4607	M	B	SR	1
{4603, 4608}	{M, F}	H	ADULT	.6

Now let us perform a selection operation on T1 to complete the left-hand side of the equation of the previous proof. LHS = $\sigma_{\mathrm{AGE}=\text{‘SR-CITIZEN’}}$(T1) yields the following rough set of tuples:

ID	SEX	RACE	AGE	MU
4603	F	H	SR	1
4603	F	H	{ADULT, SR}	.6
4607	M	B	SR	1

Working now on the right-hand side of the equation,

$T2 = \sigma_{\text{AGE}=\text{'SR-CITIZEN'}}$(RALLY1) yields

ID	SEX	RACE	AGE	MU
4603	F	H	SR	1
4603	F	H	{ADULT, SR}	.6
4607	M	B	SR	1

$T3 = \sigma_{\text{AGE}=\text{'SR-CITIZEN'}}$(PUBLIC-MEETING) yields

ID	SEX	RACE	AGE	MU
4603	F	H	SR	1
4603	F	H	{ADULT, SR}	.6
4607	M	B	SR	1

When the intersection of T2 and T3 is taken next, the result is the same as that computed for the left-hand side of the equation: RHS = T2 $\cap_A$ T3, which yields

ID	SEX	RACE	AGE	MU
4603	F	H	SR	1
4603	F	H	{ADULT, SR}	.6
4607	M	B	SR	1

Closure is another significant property of the relational model that can be extended to the fuzzy rough relational database. This property states that any operations applied to fuzzy rough relations result in another fuzzy rough relation. This follows from the definitions presented, where fuzzy rough relations result from all of the defined fuzzy rough operations.

4. SQL-LIKE QUERIES FOR THE FUZZY ROUGH RELATIONAL DATABASE

SQL, or SEQUEL, is one of the most popular languages for relational databases. As in other database query languages, there are often several ways of expressing a given query. In Section 3 of this paper, we based our data definition language for the fuzzy rough relational database on SQL. In this section, we present some SQL queries to our database. Many of the queries are the same as those discussed in Section 3.3 where we used the fuzzy rough relational algebra to express queries to the database.

(1) "*List all ages represented at the public meeting.*"

```
SELECT AGE
FROM PUBLIC-MEETING
```

There are no conditions, and hence, no WHERE clause for this query since it is a simple projection of the attribute AGE from PUBLIC-MEETING. All attribute values (columns) of the fuzzy rough relation PUBLIC-MEETING except AGE and MU are deleted and then redundant tuples eliminated to result in the following:

AGE	MU
ADULT	1
SR	1
{CHILD, TEEN}	.8
{ADULT, SR}	.6

(2) "*List the demographic data for all adults who attended the public meeting.*"

```
SELECT ID, AGE, RACE, SEX
FROM PUBLIC-MEETING
WHERE AGE = 'ADULT'
```

This query involves one fuzzy rough relation and the selection condition AGE = 'ADULT' followed by a simple projection operation. Note that in the fuzzy rough relational database, however, the attribute MU is implicitly included in the list of attributes whose values are returned, and that for projections, when deleting redundant tuples, the one having the higher MU value is retained.

ID	SEX	RACE	AGE	MU
4605	M	W	ADULT	1
5100	F	B	ADULT	1
4603	F	H	{ADULT, SR}	.3
4602	M	{W, H}	ADULT	.7

(3) "*Retrieve the IDs of individuals who participated in the first rally, but did not take part in the public meeting.*"

```
(SELECT ID
FROM PUBLIC-MEETING)
EXCEPT
(SELECT ID
FROM RALLY1)
```

This SQL expression involves two subexpressions. First the subexpressions are evaluated and then the difference operator is applied to result in a relation containing tuples of the first subexpression which are not redundant with a tuple in the second subexpression having an equivalent or greater membership value. This type of SQL query is often equivalently expressed with logical quantifiers such as NOT EXISTS as part of a WHERE condition. The result will be the same as that found in Section 3 of this paper.

(4) "*List all demographic information for individuals who participated in either the rally or the public meeting or both*."

```
(SELECT ID, SEX, RACE, AGE
FROM RALLY1)
UNION
(SELECT ID, SEX, RACE, AGE
FROM PUBLIC-MEETING)
```

This query illustrates a simple union of two union compatible fuzzy rough relations:

ID	SEX	RACE	AGE	MU
4601	M	W	ADULT	1
4602	M	{W, H}	ADULT	.9
4603	F	H	SR	1
4604	F	W	ELDERLY	1
4605	M	WHITE	ADULT	1
4606	F	B	ADULT	1
4607	M	B	SR	1
{4603, 4608}	{M, F}	H	ADULT	.6
5100	F	B	ADULT	1
5101	F	B	{CHILD, TEEN}	.8
4603	F	H	{ADULT, SR}	.6

(5) "*Retrieve all demographic information for those citizens who are so concerned about the hazards of the plants that they have attended both the rally and the public meeting*."

```
SELECT SEX, RACE, AGE
FROM RALLY1 R, PUBLIC-MEETING P
WHERE R.SEX = P.SEX AND R.RACE = P.RACE AND R.AGE
 = P.AGE AND R.ID = P.ID
```

or

```
(SELECT SEX, RACE, AGE
FROM RALLY1)
INTERSECT
(SELECT SEX, RACE, AGE
FROM PUBLIC-MEETING)
```

Both of the above queries result in a type of intersection and in conventional databases would produce equivalent results. However, in the fuzzy rough relational database, this may not always be the case. In the first query, we return those tuples which, based on indiscernibility, have equivalent values for all their attributes. This is equivalent to the original definition for fuzzy rough intersection of two relations. In the second example, because the SQL query specifies INTERSECTION, the result will depend on which of the definitions of INTERSECTION we have adopted for our application. If the alternate definition was used, then the results will include all those tuples returned from the first query. Moreover, because of rough-redundancy, additional tuples may satisfy the query and be returned.

(6) "*List the names of all people who attended the rally*."

```
SELECT P.NAME
FROM RALLY1 R, PEOPLE P
WHERE R.ID = P.ID
```

A simple JOIN of the two fuzzy rough relations RALLY1 and PEOPLE is performed, then all attributes except NAME (and MU, of course) are projected out. As before, the join definition adopted will affect the outcome.

(7) "*Retrieve the id, name, and city of people, and plant's name and location such that a plant is located in the same community as the person*."

```
SELECT P.ID, P.NAME, P.CITY, C.COMPANY, C.LOCATION
FROM PEOPLE P, COMPANY C
WHERE P.CITY = C.LOCATION
```

This is the SQL equivalent of the JOIN example from Section 3.3.6.

(8) "*Did more senior citizens participate in the rally or the public meeting*?"

```
SELECT COUNT(*)        SELECT COUNT(*)
FROM RALLY1            FROM PUBLIC-MEETING
WHERE AGE = 'SR'       WHERE AGE = 'ELDERLY'
```

In these queries we use the aggregation function COUNT(*) to count the number of tuples of each result, rather than returning those tuples as the result. We may then compare the two numbers to discover which event had more senior citizen participation. Note for this example that the WHERE clause of the first query specified AGE = 'SR' and the other specified AGE = 'ELDERLY.'

These clauses are equivalent because SR and ELDERLY belong to the same equivalence class.

We discover that there were three senior citizens at the rally and two at the public meeting. However, we know that not all of the data tuples involving senior citizens are certain, or have a membership value of one. If we want to include in our count only those tuples which are certain, we may modify our WHERE clauses to include a condition on the MU attributes. For these modified queries which follow, we determine that we can be sure that three senior citizens took part in the rally, but we are only certain that one senior citizen was involved in the public meeting:

```
SELECT COUNT(*)
FROM RALLY1
WHERE AGE = 'SR' AND MU = 1
```

```
SELECT COUNT(*)
FROM PUBLIC-MEETING
WHERE AGE = 'ELDERLY'
  AND MU = 1
```

From a user's point of view, the data manipulation language of the fuzzy rough relational database closely resembles SQL. Its queries follow the same syntax so that no new query language must be learned in order to use the fuzzy rough relational database. The semantics of the queries are different, however, because the relations in this model are fuzzy rough.

There is no loss of information in using the fuzzy rough relational model. We can always examine the MU values of tuples to discover which are *certain* and which are *possible* responses to the query. The 'certain' tuples coincide with those tuples available in a conventional relational database. In addition, we are provided with 'possible' tuples, and membership values to represent degrees of belonging to the fuzzy rough relation.

5. CONCLUSIONS

This paper concerns the modeling of imprecision, vagueness, and uncertainty in databases through an extension of the relational model of data: the fuzzy rough relational database. The fuzzy rough relational database was formally defined, along with a fuzzy rough relational algebra for querying. Comparisons of theoretical properties of operators in this model with those in the standard relational model were discussed.

The usefulness of this model was illustrated with the Public Concerns Database where we showed how the various types of uncertainty can arise in a real-world example. Using this example, we designed our database with a fuzzy rough E-R diagram, created our fuzzy rough database schema using a fuzzy rough data definition language, and populated our database with sample data. Using an SQL-like language, we then illustrated from a user point of view how the fuzzy rough relational database may be queried and how the results are better than those of conventional databases.

The fuzzy rough relational database is a sound model, which incorporates the various types of uncertainty into the underlying data model and its algebra. The design of databases for this model is similar to that of ordinary databases except for the user-defined indiscernibility values. The data definition and manipulation languages (DDL and DML) for the fuzzy rough database are closely related to standard SQL for conventional databases. The user simply has to remember that the underlying model is based on fuzzy rough sets, which will be used in determining results of queries, and that MU membership values must be considered when populating or updating the database.

In conclusion, the fuzzy rough relational database model is easy to understand and to use. In addition, it more accurately models the uncertainty of real-world enterprises than do conventional databases through the use of indiscernibility and fuzzy membership values.

References

1. Codd, EF. Comm ACM 1970;13:377–387.
2. Beaubouef, T, Petry, FE. In Proc Third IEEE Int Conf on Fuzzy Systems, Orlando, FL, 1994.
3. Buckles, BP, Petry, FE. Fuzzy Sets Syst 1982;7:213–226.
4. Buckles, BP, Petry, FE. J Inf Sci 1985;11:77–87.
5. Lipski, W. ACM Trans Database Syst 1979;4:262–296.
6. Ola, A, Ozsoyoglu, G. IEEE Trans Knowledge Data Eng 1993;5:293–308.
7. Dubois, D, Prade, H. In Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, R Slowinski, Ed.; Kluwer Academic Publishers: Boston, 1992.
8. Petry, F. Fuzzy Databases: Principles and Applications; Kluwer Academic Publishers: Boston, 1996.
9. Prade, H, Testemale, C. Inf Sci 1984;34:115–143.
10. Umano, M. In Fuzzy Information and Decision Processes, M Gupta and E Sanchez, Eds.; Amsterdam, North Holland: 1982, 339–347.
11. Zemankova, M, Kandel, A. Inf Sci 1985;37:107–141.
12. Shenoi, S, Melton, A. Int J Fuzzy Sets Syst 1989:31:287–296.
13. Beaubouef, T, Petry, FE. Comput Intell 1995;11:233–245.
14. Elmasri, R, Navathe, S. Fundamentals of Database Systems; Benjamin/Cummings: New York, 1989.
15. Maier, D. The Theory of Relational Databases; Computer Science Press: Rockville, MD, 1983.
16. Ullman, J. Principles of Database Systems, 2nd ed.; Computer Science Press: Rockville, MD, 1982.
17. Pawlak, Z. Int J Computer Inf Sci, 1982;11:341–356.
18. Grzymala-Busse, J. Managing Uncertainty in Expert Systems; Kluwer Academic Publishers: Boston, 1991.
19. Pawlak, Z. Rough Sets: Theoretical Aspects of Reasoning about Data; Kluwer Academic Publishers: Norwell, MA, 1991.
20. Makinouchi, A. In Proc Third Int Conference on Very Large Databases, Tokyo, Japan, 1977:447–453.
21. Roth, MA, Korth, HF, Batory, DS. Inf Syst 1987;12:99–114.
22. Dubois, D, Prade, H. Fuzzy Sets Syst 1987;23:3–18.
23. Nana, S, Majumdar, S. Fuzzy Sets Syst 1992;45:157–160.
24. Chanas, S, Kuchta, D. Fuzzy Sets Syst 1992;47:391–394.
25. Pawla, Z. Fuzzy Sets Syst 1985;17:99–102.
26. Wygralak, M. Fuzzy Sets Syst 1989;29:241–243.
27. Zvieli, A, Chen, P. In Proc Int Conference on Data Engineering, Los Angeles, California, 1986;320–327.
28. Date, CJ. A Guide to the SQL Standard; Addison-Wesley: Reading, MA, 1989.
29. Date, CJ. An Introduction to Database Systems, 6th ed.; Addison-Wesley, Reading, MA, 1995.

APPENDIX

Table I. PEOPLE

ID	NAME	CITY	MU
4601	Jay Reed	OSCAR	1
4602	Joe Diaz	{VENTRESS, ADDIS}	.5
4603	Tera Lunt	NEW-ROADS	1
4604	Anna Lone	OSCAR	1
4605	Dave Hart	{OSCAR, ADDIS}	.5
4606	Dione Sand	NEW-ROADS	1
4607	Frank Person	NEW-ROADS	1
4608	Tim Lunt	NEW-ROADS	1
5100	Lena Norr	ADDIS	1
5101	Joan Mann	BRULY	1

Table II. RALLY1

ID	SEX	RACE	AGE	MU
4601	M	W	ADULT	1
4602	M	{W, H}	ADULT	.9
4603	F	H	SR	1
4604	F	W	ELDERLY	1
4605	M	WHITE	ADULT	1
4606	F	B	ADULT	1
4607	M	B	SR	1
{4603, 4608}	{M, F}	H	ADULT	.6

Table III. PUBLIC−MEETING

ID	SEX	RACE	AGE	MU
4605	M	W	ADULT	1
4607	M	B	SR	1
5100	F	B	ADULT	1
5101	F	B	{CHILD, TEEN}	.8
4603	F	H	{ADULT, SR}	.6
4602	M	{W, H}	ADULT	.7

Table IV. INTERVIEWERS

NAME	SEX	RACE	MU
Sue Bin	F	H	1
Jo Luck	F	B	1

Table V. PLANTS

COMPANY	LOCATION	MU
RIVERB	VENTRESS	1
CHEMO	ADDIS	1
KBUR	{SUN, BRULY}	.6

Table VI. INDISCERNIBILITY

IND	VALUE	MU
001	W	1
001	WHITE	1
001	ANGLO-AM	1
001	EURO-AM	1
002	B	1
002	BLACK	1
002	AFRO-AM	1
003	H	1
003	HISPANIC	1
004	N	1
004	NATIVE-AM	1
005	OTHER	1
101	CHILD	1
101	PRE-TEEN	1
102	TEEN	1
102	YOUTH	1
102	TEENAGER	1
103	YOUNG-ADULT	1
104	ADULT	1
105	SR	1
105	SR-CITIZEN	1
105	ELDERLY	1
201	ADDIS	1
201	BRULY	1
201	PORT-ALLEN	1
202	VENTRESSS	1
202	NEW-ROADS	1
202	OSCAR	1
202	MORGANZA	1
203	SUN	1