Journal of Advance Computational Research
Vol. 2 No. 2 (July-December, 2017)

Imprecise Database Security and Information Measures

[1]Theresa Beaubouef and [2]Frederick E. Petry
[1]Computer Science Department, Southeastern Louisiana State University, Hammond, LA 70402
E-mail: tbeaubouef@selu.edu
[2]Naval Research Laboratory, Stennis Space Center, MS 39529, *E-mail: fpetry@nrlssc.navy.mil*

Abstract: *In this paper we consider security issues that arise in imprecise databases based on rough set theory. The aspect of security that is considered is similar to that in statistical databases for which a combination of queries cannot reveal exact values of attributes. Information theory measures are used to characterize security for imprecise databases.*

Keywords: *rough sets; database security; information theory; entropy; rough relational database*

INTRODUCTION

Databases continue to grow in size and complexity, and they are used in many diverse applications. For many real world applications, it is necessary to incorporate some type of uncertainty management into the underlying data model. One characteristic of many imprecise databases is that they allow sets of values in their tuples. This is referred to as a non-first form or nested database [1, 2]. If the value of an attribute is non-atomic, i.e. set-valued, then there is uncertainty as to which one of the values in the set corresponds to the attribute. There are specific aspects in different uncertain database models but all share use of set values. Of particular interest here is database modeling using fuzzy set and rough set approaches to represent uncertainty.

Security is becoming more and more of an issue with database applications [3], especially considering the widespread problems associated with identity theft and fraud, website visit history trackers, privacy and data mining applications, and the plethora of SPAM. In this paper we investigate the area of security for imprecise databases, which have security issues similar to that of statistical databases. We are not referring to the general protection of the data from unauthorized use, but rather controlling the type of data that may be accessed by a valid user. For example, a user may be prevented from deducing a *specific* non-key attribute value associated with the key value, but could be allowed to retrieve an averaged value.

There will always be some tradeoff between the benefits of information sharing and that of privacy, and while we often want to maximize the sharing and use of data, we cannot allow protected data to be compromised. In this paper we first overview relevant security issues and then the general form for imprecise databases. Then fuzzy and rough database models are described and security issues and information measures of security in such databases are discussed.

SECURITY AND STATISTICAL DATABASES

There are many advantages to database technology such as the ability to share data and information and to allow controlled access to data for the purpose of data mining. However, with these advantages also come disadvantages, particularly, there are security issues. Security, which has been commonly defined as the protection of the database against unauthorized use, has several aspects [4-7]. Safeguarding against the illegal modification or destruction of data is one aspect of security, but our concern here is with protection against unauthorized viewing of data. In particular for imprecise databases, permitting access to some information if the exact correlations of data items remain unknown is considered. This use of security is similar to the idea of security in statistical databases. For example statistical information such as the average salary of a large group of individuals may be available, but not the exact salary of any one individual.

To discuss security in imprecise databases we will review the relevant concepts that have been developed for statistical databases. Several researchers have studied issues related to this type of database security [8-15]. In statistical databases we can assume that any one query cannot reveal specific data values that should be protected. If someone can deduce confidential data from one or more queries we say the database has been

compromised. The primary approaches to compromising these databases are to isolate a specific data value by intersecting a set of queries. Solving a system of equations based on these results may reveal a specific value of an attribute. . Security violations can occur when one or more queries reveal confidential information. Most of these are based on the isolation of a single data value at the intersection of several query sets. Four methods for protecting against such violations have been studied [9] which we describe next.

(1) Minimum query size controls violations using very large or small query sets by a formula to giving a lower bound on allowable query set size. A query set is not allowed if the size is less than this bound.

(2) Minimum overlap control is based on preventing replies to queries based on the number of records overlapping prior queries. Here queries are not allowed if they have more than a predetermined number of tuples in common with each of the previous queries. This is impractical in a realistic database as the potential number of queries over a significant period of time requires a checking that is computationally infeasible.

(3) Distortion of the data or the query response is another approach. For example in statistical queries this might be achieved by rounding an answer to a query. This trades accuracy of an answer for security of the data.

(4) Random sampling attempts to prevent violations by not allowing the user to specially craft query sets to overcome the controls described above. This approach prevents inference of specific data by no longer allowing a user to select specific records to query for statistical queries. It can be seen that random sampling using large samples can reduce risk but maintain high statistical accuracy.

We shall see that some of the security issues to be discussed for fuzzy and rough set data models are analogous to the ones described above for statistical databases.

GENERAL FORM FOR IMPRECISE DATABASES

In this section we review the non-first form (NF^2) for databases that is the structure used in both the fuzzy set and rough set data models. Then we introduce the concept of interpretations for NF^2 tuples which will play a key role in discussion of security for such databases.

An NF^2 relation is simply an unnormalized relation scheme and an instance over this scheme. This form was first suggested by Makinouchi [1] who relaxed the 1NF assumption to allow attributes to become set-valued. A number of researchers [16-18] generalized the NF^2 database model by allowing elements of a tuple to be either atomic or set-valued entries or even relations themselves. This is called the NF^2 assumption. Therefore, the NF^2 data model can be considered as a generalized relational data model which treats flat relations and hierarchical structures in a uniform way. A formal definition of a NF^2 data model can be stated as follows:

Definition: Let a relation scheme be a collection of rules of the form $R = \{A_1, A_2,..., A_n\}$ and $D_1, D_2,..., D_n$ be corresponding domains from which values for attributes $A_1, A_2,..., A_n$ are selected. A NF^2 scheme may contain any combination of zero or higher order attributes on the right hand side of the rules whereas a first normal form (1NF) scheme can contain only zero order attributes on the right hand side. (An attribute Ai is a higher order attribute if it appears on the left hand side of some rule; otherwise it is zero order.)

A tuple of a NF^2 relation is denoted as $(a_1, a_2,..., a_n)$ where the ith component is ai and each component may be an atomic or null value or another tuple a_j. Furthermore, when A_i is a higher order attribute, the value for attribute A_i may not be just a single value from its zero order attributes, but an element of the subset of the cross product of associated domains D_i from which the corresponding attribute values are drawn. So a NF^2 relation is a subset of the Cartesian product $(D_{i1} \times D_{i2} \times ... \times D_{in}) \times 2^{D}i$.

Several researchers have extended the ordinary relational algebra for the NF^2 model by extending the basic set operators and introducing two new restructuring operators, called the nest and unnest operators [19, 20] (additionally, the pack and unpack operators in [21]). These operators are used to transform a 1NF relation into a NF^2 relation and vice versa.

After applying the nest operator, we obtain an NF^2 relation nested along a set of attributes A_i of a relation r, if tuples in r agree in the remaining components. The unnest operator inverts this process; that is, it takes a relation nested on some set of attributes and disaggregates it making a "flatter" structure. The formal definitions and the properties of these operators along with ordinary relational algebra operators extended for the NF^2 relational model are given in [19-21].

A related approach to a more limited uncertainty representation is that of range values [22]. For example we may not know exact the age of an individual but we know it is in the range of 25 to 30 years. So we have an interval of values and know one is correct but do not

necessarily know exactly which one. For range values that are a discrete set this leads to the concept of the possible interpretations of the range as a set. In the above example of an age range this is the set: {25, 26, 27, 28, 29, 30} It is the idea of interpretations that we investigate carefully in relationship to security concerns in imprecise databases.

For our consideration of uncertainty data representations in a fuzzy or rough database we will only need to consider the form of NF^2 relations for which an attributes may have a set of values. Since we are concerned with security violations we must discuss the possible meaning or interpretation of such relations. For example consider a simple tuple with two attributes A_1, A_2 having the values: $A_1 = a$; $A_2 = \{b_1, b_2, \ldots b_m\}$. Then there are m possible meanings or interpretations for this tuple:

$$\{[a, b_1], [a, b_2] ,\ldots [a, b_m]\}$$

Assume we are concerned with not allowing some exact values of attribute A_2 corresponding to A_1 to be known. In this case there are m possible correspondences and a query that returned this one tuple, $(a, \{b_1, b_2, \ldots b_m\})$, would not directly violate our security concern. If the value of attribute A_1 were also a set of values, the possible relationship between the attribute values of A_1 and A_2 would be even more uncertain.

However it may be possible that multiple querying of such set valued tuples could still lead to security violations as described in the previous section. We will now carefully consider under what conditions in NF^2 databases, security could still be violated. To do this we must consider the idea of tuple interpretations more formally.

For a given tuple $t(A_1, A_2, \ldots A_n)$ let the value of attribute A_i be the set $d_i \subseteq D_i$. Then each interpretation of the tuple t has a specific value v_i for each attribute A_i:

$$I = [v_1, v_2, \ldots v_n] , v_i \in d_i$$

In general for every interpretation, I_j,

$$I_j \in d_1 \times d_2 \times \ldots \times d_n$$

To count the number of interpretations P_k of the tuple t_k, let the cardinality of the value of the ith attribute be $|d_{ki}| = p_i$. So then

$$P_k = \prod_{i=1}^{n} p_i.$$

Our discussion of interpretations allows us to address the question of whether a sequence of queries can isolate a single interpretation, thus violating security. Consider the general case in which we want to prevent an exact association between values of two attributes, A_j and A_k, when a set of r tuples $\{t_1, t_2, \ldots t_r\}$ are retrieved by querying. Then we have for each tuple the set of interpretations for the two attributes:

$$I_1(j, k) = \{I_{11}(j, k), I_{12}(j, k), \ldots I_{1n_1}(j, k)\}$$

$$I_2(j, k) = \{I_{21}(j, k), I_{22}(j, k), \ldots I_{2n_2}(j, k)\}$$

$$\ldots\ldots$$

$$I_r(j, k) = \{I_{r1}(j, k), I_{r2}(j, k), \ldots I_{rn_r}(j, k)\}$$

where $I_{pq}(j, k)$ is a specific interpretation of tuple t_p for these attributes.

Let us consider an example of how interpretations can be related to violations of security in retrieved set-valued tuples with the two attributes A_j and A_k. Assume the following three tuples are retrieved:

$t_1 = (\{a, b, c\}, \{r, s, t\})$

$t_2 = (\{a, d, e\}, \{r, v, x\})$

$t_3 = (\{e, f\}, \{s, t, z\})$

So these tuples have the interpretations :

$$I_1(j, k) = \{[a, r], [a, s], [a, t], [b, r], \ldots.[c, s], [c, t]\}$$

$$I_2(j, k) = \{[a, r], [a, v], [a, x], [d, r], \ldots.[e, v], [e, x]\}$$

$$I_3(j, k) = \{[e, s], [e, t], [e, z], [f, s], \ldots, [f, z]\}$$

Now these interpretations can be pairwise intersected:

$$I_1(j, k) \cap I_2(j, k) = \{[a, r]\}$$

and the other intersections are null:

$$I_1(j, k) \cap I_3(j, k) = I_2(j, k) \cap I_3(j, k) = \varnothing$$

Since the intersection of tuples t_1 and t_2 produces a set of cardinality 1, we can definitely say the value a for attribute A_j is uniquely associated with the value r of attribute A_k which represents a security violation with respect to these attributes. So in general there will be a security violation if for any p, q

$$|I_p(j, k) \cap I_q(j, k)| = 1, p \neq q.$$

These sort of security violations are similar to the minimum overlap control (2) discussed previously. Also if a query can return a large query set size (1), then it is more likely some tuple interpretations might overlap resulting in a security violation as described.

FUZZY AND ROUGH RELATIONAL DATABASES AND SECURITY

In this section we will discuss security issues for two specific imprecise data models – fuzzy and rough set database models. An overview of the appropriate fuzzy and rough set concepts is first given and the data models and their properties are described. Finally the security issues relevant to these typical imprecise data models are presented.

Fuzzy Database Model

There are a number of approaches to fuzzy databases that have developed [23, 24]. The approach we consider in this paper uses fuzzy similarity relationships in an imprecise relational model [25, 26]. The ordinary or non-fuzzy relational database is a special case of this fuzzy relational database approach. The identity relationship used in non-fuzzy relational databases induces equivalence classes (most frequently singleton sets) over a domain, D, which affect the results of certain operations and the removal of redundant tuples. The identity relationship is replaced in this fuzzy relational database by an explicitly declared similarity relationship [27] of which identity is a special case. Next we present the basics of similarity relationships.

Fuzzy Similarity Relationships

A similarity relationship, $s(x, y)$, for given domain, D, is a mapping of every pair of elements in the domain onto the unit interval [0, 1] with the following three properties, $x, y, z \in D$ [28]:

1. Reflexive: $s_D(x, x) = 1$
2. Symmetric: $s_D(x, y) = s_D(y, x)$
3. Transitive: $s_D(x, z) \geq \text{Max}(\text{Min}[s_D(x, y), s_D(y, z)])$: (T1)

This particular max-min form of transitivity is known as T1 transitivity. Another useful form is T2 also known as max-product:

3'. Transitive: $s_D(x, z) = \text{Max}([s_D(x, y) * s_D(y, z)])$: (T2)

where * is arithmetic multiplication.

An example of a similarity relation satisfying T2 transitivity is:

$$s_D(x, y) = e^{-\beta^*} | y - x |$$

where $\beta > 0$ is an arbitrary constant and $x, y \in D$.

A typical similarity relation for a finite scalar domain base set satisfying T1 transitivity is shown in figure 1, where $D = \{A, B, C, D, E\}$.

Sim(x,y)	A	B	C	D	E
A	1.0	0.8	0.4	0.5	0.8
B	0.8	1.0	0.4	0.5	0.9
C	0.4	0.4	1.0	0.4	0.4
D	0.5	0.5	0.4	1.0	0.5
E	0.8	0.9	0.4	0.5	1.0

Figure 1: T1- Similarity Relation

Equivalence Classes and Partitions for Similarity Relations

If S is a similarity relation on a domain X, the α-level set S_α is an equivalence relation on X

$$S_\alpha = \{(x, y) \mid s(x, y) \geq \alpha\}$$

This provides the following equivalence classes for the similarity relation of figure 1.

$\{\{A\}, \{B\}, \{E\}, \{D\}, \{C\}\} : S_{1.0}$

$\{\{A\}, \{B, E\}, \{D\}, \{C\}\} : S_{0.9}$

$\{\{A, B, E\}, \{D\}, \{C\}\} : S_{0.8}$

$\{\{A, B, E, D\}, \{C\}\} : S_{0.5}$

$\{\{A, B, E, D, C\}\} : S_{0.4}$

It is useful to illustrate these classes in a tree structure, called a partition tree in figure 2. Clearly as α increases partitions get smaller.

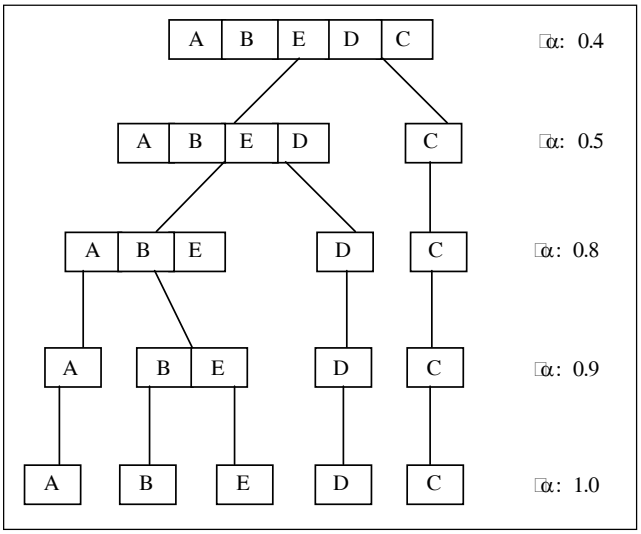

Figure 2: Partition Tree

Fuzzy Database Definitions

The basic concepts of fuzzy tuples and interpretations follow the discussion we have given about set valued domains of imprecise databases. A domain value, d_i, where i is the index of the attribute in the tuple, is a subset of its domain base set, D_i. That is, any member of the power set may be a domain value except the null set. The actual values in such a set, d_i, are determined by the similarity of the values as given the similarity relationship associated with the domain D_i.

A fuzzy tuple is a non-first normal form tuple, $t_i = (d_{i1}, d_{i2}, ..., d_{im})$ where $d_{ij} \subseteq D_j$.

As discussed before these have interpretations, $I = [a_1, a_2, ..., a_m]$ which is any value assignment such that $a_j \in d_{ij}$ for all j.

Similarity Thresholds

Given a domain, D_j, in a relation, the similarity threshold is defined to be:

$$\text{Thres}(D_j) = \min\{\min[s(x, y)] \mid \forall i\, x, y \in d_{ij}\}$$

Note that in a non-fuzzy database, the cardinality of $d_{ij} = 1$ and $s(x, x) = 1$, so Thres$(D_j) = 1$ for all j. A minimal threshold value given a priori can be used to determine which tuples may be combined by direct set union of the respective domain values. This essentially corresponds to the elimination of redundant tuples in non-fuzzy databases.

Typically a fuzzy database involves some attribute domains consisting of subjective or linguistic terms that have varying degrees of similarity. For example consider an opinion/evaluation survey in which a group of evaluators assess the quality of various products using a specified set of descriptive terms from which to select their evaluations illustrated in figure 3. The domain of such descriptors might range such as {poor, inferior, …, excellent, superior} with a similarity relationship provided over the domain.

ASSESSMENTS

PRODUCT	EVALUATOR	RATINGS
Prod X	Eval 1	{poor, inferior}
Prod X	Eval 2	mediocre
Prod Y	Eval 1	excellent
Prod Y	Eval 3	superior
.....	.....	.....

Figure 3: Assessments Relation Example

As we shall discuss further the security issue of concern is confidentiality of the evaluators' ratings, i.e., can a querying of such a database reveal the exact ratings of any specific evaluator.

A fuzzy relational query can use the same SQL structure as an ordinary relational database. In addition, there is a clause defining minimum similarity thresholds. Consider the SQL type query

Select PRODUCT, RATINGS

From ASSESSMENTS

Where Thres (RATINGS) ≥ 0.75

In this simple projection operation over the relation we have extended the Where clause to provide the threshold desired for the similarity of the Ratings values to be used in the query result.

RESULT

PRODUCT	RATINGS
Prod X	{poor, inferior, mediocre}
Prod Y	{excellent, superior}
.....	.....

Figure 4: Query Result Relation

The relation RESULT in figure 4 created by the query contains only the domains Product and Ratings. The final form of the relation is obtained by merging tuples via the set union of respective domain values until no additional tuples can be merged without violating (falling below) the minimum threshold for RATINGS (0.75). This corresponds to the removal of duplicate tuples such as in an ordinary relational algebra Project operation, but is based on the similarity relationships for the respective domains as opposed to identity in a crisp domain. The minimum threshold constraints will be subsequently referred to as the level values.

Anytime a level value is missing, it is assumed to be one (1), that is, the same as assumed for a non-fuzzy relational algebra command. It should be noted that in practice, the numerical specification of level values can be abandoned in favor of linguistic terms for which there are precise meanings.

Redundancy and Uniqueness Properties

In a non-fuzzy database, a tuple is redundant if it is exactly the same as another tuple. Any operation over a non-fuzzy relation at least implicitly entails removing redundant tuples. That is, any interpretation of the domains can be found in at most one tuple in the relation. In a fuzzy database, a tuple is redundant if it can be merged with another through the set union of corresponding domain values. The merging of tuples, however, is subject to constraints based on the similarity thresholds.

Definition. The tuples $t_i = (d_{i1}, d_{i2}, ..., d_{im})$ and $t_k = (d_{k1}, d_{k2}, ..., d_{km})$ are called redundant if

$$\text{Level}(D_j) \leq \min [s(x, y)]; \; x, y \; d_{ij} \cap d_{kj}$$

for $j = 1, 2,..., m$ and Level(D_j) given a priori.

In a fuzzy database, each tuple can potentially represent a large number of interpretations. Despite this, it would be extremely satisfying if this definition of redundant tuples were, in some sense, compatible with the one for ordinary databases. The lack of redundant tuples in an ordinary database is tantamount to the absence of multiple occurrences of the same interpretation. Therefore, given any interpretation of the domains, a fuzzy relation should contain at most one tuple with that interpretation. For example consider a subjective interpretation of infra-red remotely sensed images in a database. These images may be colour-coded to enhance interpretation. Experts take into account the colour-codes as well as the terrain and other information about a particular site to provide their interpretations. Here we illustrate the interpretation of colours as corresponding to some imprecise linguistic terms that represent temperatures in the relation IR_IMAGE of figure 5.

IR_IMAGE

Colours	Temperatures
Yellow	Warm
Blue	Cool
Black	Cold
Red	Hot

R'

Colours	Temperatures
{ Yellow, Red }	{ Warm, Hot }
{ Blue, Black, Yellow }	{ Cool, Cold, Warm }

Figure 5: Example of Multiple Interpretations

Can the relation IR_IMAGE be reduced according to some pair of level values to *R*' in figure 5, where {Yellow, Warm} is an interpretation of both tuples? The question is important when one considers the impact of the answer on the design of query languages and the possibility of creating anomalies during updating. Also it would then be possible to intersect such tuples' interpretations risking security violations. Fortunately, the situation illustrated above is impossible. Let T_i be the set of possible interpretations for tuple t_i.

Redundancy Theorem

Given a fuzzy relation with no redundant tuples and each domain similarity relation formulated according to Tl then

$$T_i \cap T_j = \emptyset \text{ if } i \neq j.$$

Proof: Assume $T_i \cap T_j \neq \emptyset$ and let $I = (a_1, a_2, ..., a_m) \in T_i \cap T_j$, where $a_h \in d_{ih} \cap d_{jh}$. Now if it can be demonstrated that for any h, the domain value d_{ih} can be merged into a single domain value without violating the Level(D_h), then the tuples t_i and t_j are redundant.

Let $x, y \in d_{ih}$ be such that $s(x, y) = \min(s(u, v))$, $u, v \in d_{ih}$; also let $x' \in y' \in d_{jh}$ be such that $s(x', y') = \min(s(u, v))$, $u, v \in d_{ih}$ Then, in particular for a_h

$$s(x, a_h) \geq s(x, y), s(x', a_h) \geq s(x', y')$$

Taking the minimum on each side and using the symmetric and Tl transitivity properties of similarity

$$\min(s(x, a_h), s(a_h, x')) \geq \min(s(x, y), s(x', y'))$$

and thus

$$s(x, x') \geq \min(s(x, y), s(x', y')).$$

By definition, Level(D_h) $\leq s(u, v)$ where u, v are in the same domain value. Thus,

$$\text{Level}(D_h) \leq \min(s(x, y), s(x', y')) \leq s(x, x')$$

Again, applying the definition of level value and applying transitivity, for all $u \in d_{ih}$, $v \in d_{jh}$

$$s(x, v) \geq \min(s(x, x'), s(x', v)) \geq \text{Level}(D_h)$$

and so

$$s(u, v) \geq \min(s(u, x), s(x, v)) \geq \text{Level}(D_h).$$

Hence all corresponding domain values in tuples t_i and t_j can be merged without affecting their thresholds, producing a contradiction. Thus, a non-redundant fuzzy relation indeed is such that interpretations of any given tuple are unique. QED.

The converse of the above theorem is also true. If no two tuples can be interpreted in an identical manner, then there exist level values for the domains under which no two tuples are redundant.

If all domain similarity thresholds, Thres(Di), are one (1), the relation is non-fuzzy and each tuple has a unique interpretation with respect to all others. By the preceding theorem, if the similarity thresholds are less than one, the property of uniqueness of tuple interpretation still remains.

Security in Fuzzy Databases

In this section we will present a description of how the concept of a fuzzy database naturally lends itself to the protection of security of data [29]. In a fuzzy database with a non-first normal form (NF^2) structure, each tuple can potentially represent a large number of interpretations. The merging of data into sets, depending on the level values, provides a measure of inherent security in the similarity-based fuzzy database. The specific association of values, i.e., specific interpretations, is blurred in the merging. As described for statistical databases, distortion of data (3) provides security and the blurring in the fuzzy database is analogous. Security protection for a fuzzy database thus means that if some data item, $b \in D_i$, is protected the value $x \in D_j$ associated with b cannot be determined. This implies it should not be possible to derive a tuple with two singleton sets that contain only b and x respectively.

Set Intersection Security Violations

Let us assume a query has produced a result relation, *r*, with no explicit violation of security for protected values, i.e., protected values were merged into sets in the resulting relation. However it might still be possible to manipulate data in the relation to obtain some explicit associations for protected values. In particular consider the intersection

of the sets of values in different domains across several tuples. For example, the intersection of the names and salary domains of two tuples

(...{ Baum, Seither, Perez} ... {68,000, 77,300})

(...{Adams, Perez, Badeux} ... {51,000, 77,300, 92,500})

produces a security violation if the exact salary of Perez should not disclosed

(... {Perez} ... {77,300}).

The reason that the intersection produced a security violation was that the protected name and salary had appeared, albeit merged with other data items, in two tuples. So we are led to the following theorem:

Theorem

Intersection of tuples of a single relation in a similarity-based fuzzy database cannot lead to a security violation.

Proof: Consider the intersection of tuples t_1, t_2, t_3, ... in relation r relative to the domains D_i and D_j. In order to have a security violation we must have for at least two tuples p and q

$$| d_{pi} \cap d_{qi} | = 1$$

and correspondingly

$$| d_{pj} \cap d_{qj} | = 1$$

The resulting set are singletons, e.g., $\{b\}$ and $\{x\}$, $b \in d_{pi}, d_{qi}$; $x \in d_{pj}, d_{qj}$. In other words, the interpretation associating b and x, $[b, x]$, must have been in the interpretations of both tuples that were intersected. However as we have shown a fuzzy relation based on similarity measures cannot have two or more tuples containing the same interpretation. Therefore the intersection of tuples cannot produce a security violation.

Fuzzy Security Violations

Now we can examine other possibilities for security violations. If two or more data items occur in the protected domain values for a single retrieved tuple there is no direct security violation, $d_{ki} = \{a, b\}$; $d_{kj} = \{x, y\}$:

$$t_k : (.... \{a, b\} .... \{x, y\} ...)$$

Security is not violated since it is not known which element, x or y, is to be associated with b. However if the Level (D_j) value used in the query is a large value, N, then the similarity of the elements of d_{kj} must be very high.

$$s_j (x, y) \geq N$$

From an external point of view this may constitute a security violation. Either value might be satisfactory to the unauthorized individual if she knows the values are very "similar." This will be called a fuzzy security violation. In general a fuzzy violation can occur with any number of elements, $d_{ki} = \{a, b, c, d,..\}$; $d_{kj} = \{v, w, x, y, z,\}$:

$$t_k = (...\{a, b, c, d,...\} ... \{v, w, x, y, z,... \} ...).$$

The pairwise similarity of all elements in d_{kj} is greater than N, the Level (D_j) specified, since they have been merged. Thus any value, say w, could be associated with b and so be considered as a fuzzy security violation. In order to prevent a fuzzy security violation, it would be necessary to have the highest level allowable value, N^*, specified along with the specific element, such as b, that is to be protected. Enforcing this as a control on querying can be related to the statistical approach of distorting data (3) in the query response.

Information-Theoretic Measures

Fuzzy databases are used in applications which involve some imprecision or uncertainty in the data and in decision-making utilization of the data. In order to help understand the impact of such imprecision, information-theoretic characterizations have been developed which measure the overall uncertainty in an entire relation. Additionally, a variation of fuzzy entropy has been used to determine how well a fuzzy query differentiates among potential responses [30].

Fuzzy entropy may be measured as a function of a domain value or as a function of a relation. Intuitively, the uncertainty of a domain value increases as its cardinality $| d_{ij} |$ increases or when the similarity $s_j (x, y)$ decreases. So if a domain value in a relational scheme, d_{ij}, consisting of a single element represents exact information and multiple elements are a result of fuzziness, then this uncertainty can be represented by entropy. DeLuca and Termini [31] have devised formulas for uncertainty based on fuzzy measures. Adapting their result to a fuzzy database, the entropy $H_{fz} (d_{ij})$, for a domain value $d_{ij} \subseteq D_j$ would be

$$H_{fz} (d_{ij}) = - \sum_{\{x,y\} \in d_{ij}} [s_j (x, y) \log_2 (s_j (x, y)) + (1 - s_j (x, y)) \log_2 (1 - s_j (x, y))]$$

Note that $H_{fz} (d_{ij})$ is directly proportional to $| d_{ij} |$ and inversely proportional to $s_j (x, y) > 0.5$.

This definition cannot be directly extended to tuples, so a probabilistic entropy measure after Shannon [32] is needed for an entire tuple. First recalling the concept of interpretation of a tuple, for the ith tuple, t_i, there are P_i possible interpretations, i.e., the cardinality of the cross product of the domain values, $| d_{i1} \times d_{i2} \times ... \times d_{im} |$. Viewing all interpretations as a priori equally likely, the entropy of tuple ti can be defined as

$$H_{pb}(t_i) = -\sum_{k=0}^{P_i} (1 / Pi) \log_2 (1 / P_i) = \log_2(P_i)$$

For a non-fuzzy database, clearly $P_i = 1$ and $Hpb(t_i) = 0$.

If the choice of a tuple in a relation r is independent of the interpretation of the tuple, the joint probabilistic entropy $Hpb(r, t)$ of a relation can be expressed as

$$Hpb(r, t) = -\sum_{i=1}^{n} \sum_{k=1}^{P_i} (n P_i)^{-1} \log_2 [(n P_i)^{-1}]$$

where there are n tuples.

Also, a query response measure can be given for a Boolean query with linguistic modifiers by using the membership value $\mu_Q(t)$ for each tuple in the relation r which is the response to a query Q. This membership value is not static but represents the best matching interpretation of the tuple t relative to the query. So the fuzzy entropy of a relation r with n tuples is

$$Hfz(r \mid Q) = -\sum_{i=1}^{n} [\mu_Q(t_i) \log_2 (\mu_Q(t_i)) + (1 - \mu_Q(t_i)) \log_2 (1 - \mu_Q(t_i))]$$

Note that $Hfz(r \mid Q) = 0$ if and only if $(\mu_Q(t_i) = 0)$ or $(\mu_Q(t_i) = 1)$ for all i. In every other case $Hfz(r \mid Q) > 0$ and is maximized when $\mu_Q(t_i) = 0.5$ for all i. This maximization condition is achieved when a query fails to distinguish the dominant truth value of any tuple.

Since the domains in a fuzzy database may be both ordinary and fuzzy sets, some combined information estimate is desirable. One possible approach would be an entropy combining Shannon information and fuzzy information similar to one that has been introduced for image evaluation [33].

Security Measures

Security in the sense we have discussed for fuzzy databases means uncertainty about the specific associations of data items. Information theory has been used to measure uncertainty in statistical databases [34] and also as described in the previous section to the analysis of the fuzzy relational database and queries

Let us now apply the entropy expression Hpb to an example relation to see its relationship to security. Consider the two relations in figure 6.

In R1, $n = 5$, and for the two tuples with set-valued entries there are two interpretations each. For the second tuple, these are $[b, v]$ and $[c, v]$, and for the next tuple the interpretations are, $[d, w]$ and $[d, x]$. So $P_2 = P_3 = 2$; the other P_i are 1. For R_2, $n = 2$ and $P_1 = 6$, $P_2 = 12$. Evaluating the entropy for these two we have

$$Hpb(\text{R1}, t) = 2.72$$

R2

ATTRIBUTE 1	ATTRIBUTE 2
{ a, b, c }	{ u, v }
{ d, e, f }	{ w, x, y, z }

R1

ATTRIBUTE 1	ATTRIBUTE 2
a	u
{ b ,c }	v
d	{ w, x }
e	y
f	z

Figure 6: Example Relations Illustrating Security Entropy

$$Hpb(\text{R2}, t) = 4.08$$

This is consistent with the intuitive idea of entropy since R1 contains more "information" than R2; that is, in R1 we have a more exact idea of how data items are related. Our concept of security is clearly in correspondence to these measures. So we have a quantitative evaluation of the security of any given relation for a database using this entropy measure that is related to blurring or distortion approach to data security (3) A database security monitor could then evaluate a result relation and assess whether Hpb exceeds a pre-set threshold before providing the query result to a user.

Rough Set Database Model

Another approach for uncertainty representation is the rough set database model using the rough set concept of indiscernibility of values. We first provide the overview needed of rough set theory and then discuss the rough set database model for imprecise data.

Rough Set Theory

Rough set theory [35] is a mathematical formalism for representing uncertainty. An approximation region in rough sets partitions some universe into equivalence classes. This partitioning can be adjusted to increase or decrease its granularity, to group items together that are considered indiscernible for a given purpose, or to "bin" ordered domains into range groups.

U is the *universe*, which cannot be empty,

R : *indiscernibility relation*, or equivalence relation,

A = (U,R), an ordered pair, called an *approximation space*,

$[x]_R$ denotes the equivalence class of *R* containing *x*, for any element *x* of *U*,

elementary sets in A - the equivalence classes of R,

definable set in A - any finite union of elementary sets in A.

Any finite union of these elementary sets is called a definable set. A *rough set* $X \subseteq U$, however, is defined in terms of the definable sets by specifying its lower ($\underline{R}X$) and upper ($\bar{R}X$) approximation regions:

$$\underline{R}X = \{x \in U \mid [x]_R \subseteq X\} \text{ and } \bar{R}X = \{x \in U \mid [x]_R \cap X \neq \varnothing\}.$$

$\underline{R}X$ is the positive region, $U - \bar{R}X$ is the negative region, and $\bar{R}X - \underline{R}X$ is the boundary or borderline region of the rough set *X*, allowing for the distinction between certain and possible inclusion in a rough set. The set approximation regions provide a mechanism for determining whether something *certainly belongs* to the rough set, *may belong* to the rough set, or *certainly does not belong* to the rough set. Given the upper and lower approximations $\bar{R}X$ and $\underline{R}X$, of *X* a subset of *U*, the R-positive region of *X* is $POS_R(X) = \underline{R}X$, the *R*-negative region of *X* is $NEG_R(X) = U - \bar{R}X$, and the boundary or *R*-borderline region of *X* is $BN_R(X) = \bar{R}X - \underline{R}X$. *X* is called *R*-definable if and only if $\underline{R}X = \bar{R}X$. Otherwise, $\underline{R}X \neq \bar{R}X$ and *X* is rough with respect to *R*. In Figure 7 the universe *U* is partitioned into equivalence classes denoted by the rectangles. Those elements in the lower approximation of *X*, $POS_R(X)$, are denoted with the letter "*p*" and elements in the *R*-negative region by the letter "*n*". All other classes belong to the boundary region of the upper approximation.

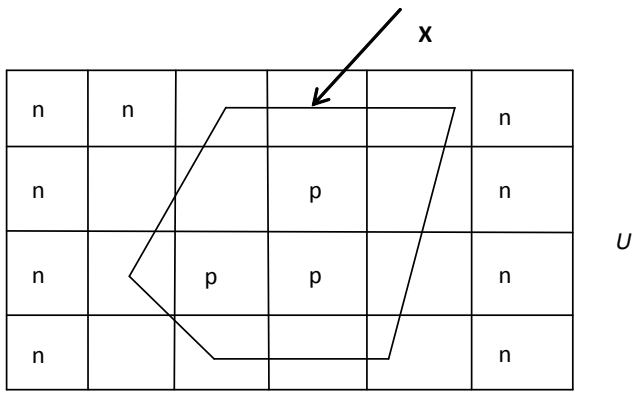

Figure 7: Example of a Rough Set X

As a specific example let *U* = {medium, small, little, tiny, big, large, huge, enormous}. Then let the equivalence relation R be defined as follows:

R = {[medium], [small, little, tiny], [big, large], [huge, enormous]}.

A given set *X* = {medium, small, little, tiny, big, huge}, can be defined in terms of its lower and upper approximations:

$\underline{R}X$ = {medium, small, little, tiny}, and $\bar{R}X$ = {medium, small, little, tiny, big, large, huge, enormous}.

The major rough set concepts of interest are the use of an indiscernibility relation to partition domains into equivalence classes and the concept of lower and upper approximation regions to allow the distinction between certain and possible, or partial, inclusion in a rough set. The indiscernibility relation allows the grouping of items based on some definition of 'equivalence' as it relates to the application domain. The results in the lower approximation region are certain, corresponding to exact matches. The boundary region of the upper approximation contains results that are possible, but not certain.

Rough Relational Databases

The rough relational database model [36] has an imprecise tuple structure similar to that of a typical imprecise NF^2 database. It captures all the essential features of rough sets theory including indiscernibility of elements denoted by equivalence classes and lower and upper approximation regions for defining sets which are indefinable in terms of the indiscernibility.

Every attribute domain is partitioned by some equivalence relation designated by the database designer or user. Within each domain, those values that are considered indiscernible belong to an equivalence class. This information is used by the query mechanism to retrieve information based on equivalence with the class to which the value belongs rather than equality, resulting in less critical wording of queries [37].

Recall is also improved in the rough relational database because 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 equality of attributes in the calculation of lower and upper approximation regions of the query result.

The rough relational database has common structures with the fuzzy relational database discussed previously so rough tuples and their interpretations are similar in structure. However the nature of rough set data causes a different view of tuple redundancy.

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 $d_{xy} = \{c_1, c_2, ..., c_n\}$, then $[d_{xy}] = [c_1] \cup [c_2] \cup ... \cup [c_n]$.

Definition. Tuples $t_i = (d_{i1}, d_{i2}, ..., d_{im})$ and $t_k = (d_{k1}, d_{k2}, ..., d_{km})$ are *redundant* if $[d_{ij}] = [d_{kj}]$ for all $j = 1,..., m$.

So in the rough relational database, redundant tuples are removed in the merging process based on this definition. As described for fuzzy databases, in a rough relational database, the intersection of tuples in a single relation cannot produce a security violation. This follows because redundant tuples are not allowed in a rough relation, and so there cannot be two tuples having the same interpretation.

Rough Databases and Security Measures

In the rough relational database information-theoretic measures for uncertainty were defined for rough schemas and rough relations [38, 39]:

Definition: The *rough schema entropy* for a rough relation schema S is

$$H_{SE}(S) = -\Sigma_j [\Sigma\ Q_i \log(P_i)] \text{ for } i = 1,... n; j = 1,..., m$$

where there are n equivalence classes of domain j, and m attributes in the schema $R(A_1, A_2, ..., A_m)$. The schema entropy provides a measure of the uncertainty inherent in the definition of the rough relation schema taking into account the partitioning of the domains on which the attributes of the schema are defined

Definition: The *rough relation entropy* of a particular extension of a schema is

$$H_{RS}(R) = -\Sigma_j D\rho_j(R) [\Sigma\ DQ_i \log(DP_i)] \text{ for } i = 1,... n; \quad j = 1,..., m$$

where $D\rho_j(R)$ represents a type of database roughness for the rough set of values of the domain for attribute j of the relation, m is the number of attributes in the database relation, and n is the number of equivalence classes for a given domain for the database. DQ_i is the probability of a tuple in the database relation having a value from class i, and DP_i is the probability of a value for class i occurring in the database relation out of all the values which are given We obtain the $D\rho_j(R)$ values by letting the non-singleton domain values represent elements of the boundary region, computing the original rough set accuracy and subtracting it from one to obtain the roughness The entropy of an actual rough relation instance $H_{RS}(R)$ is an extension of the schema entropy obtained by multiplying each term in the product by the roughness of the rough set of values for the domain of that given attribute.

Consider the example relations in figure 8 where domains for soil color and size have been defined as

COLOR = {[black, ebony], [brown, tan, sienna], [white], [gray], [orange]}, and PARTICLE-SIZE = {[big, large], [huge, enormous], [medium], [small, little, tiny]}

SAMPLE-114

BIN-NO	COLOR	PARTICLE-SIZE
P21	brown	medium
P22	{black, tan}	large
P23	gray	{medium, small}
T01	black	tiny
T04	{gray, brown}	large

SAMPLE-115

BIN-NO	COLOR	PARTICLE-SIZE
M43	{black, tan, white}	{big, huge, medium}
M46	{brown, orange, white, gray}	{medium, small}

Figure 8: Rough Relations for Entropy Analysis

The rough relation entropy of the relations SAMPLE-114 and SAMPLE-115 shown in the tables are calculated as follows:

H_{RS} (SAMPLE-114) = -(4/7)[(2/5)log(2/7) + (3/5)log (3/7) + 0 + (2/5)log(2/7) + 0] - (2/6)[(2/5log(2/6) + 0+ (2/5)log(2/6) + (2/5)log(2/6)] = .56

H_{RS}(SAMPLE-115) = -(7/7)[(1/2)log(1/7) + (2/2)log (2/7) + (2/2)log(2/7) + (1/2)log(1/7) + (1/2)log(1/7)] – (5/5)[(1/2)log(1/5) + (1/2)log(1/5) + (2/2)log(2/5) + (1/2)log(1.5) = 3.7821

From this example it is clear that our concept of security in the rough relational database corresponds to uncertainty in this sense, so we can use these measures of entropy as a quantitative measure for security in a rough relational database.

CONCLUSIONS

We have described approaches analogous to data security in statistical databases for security in imprecise databases using a non-first normal from representation. This was illustrated specifically for fuzzy and rough set database models. Information theory measures for these databases were described and applied to data security control in such databases.

ACKNOWLEDGMENT

The authors would like to thank the Naval Research Laboratory's Base Program, Program Element No. 0602435N for sponsoring this research.

REFERENCES

[1] A. Makinouchi "A Consideration on Normal form of Not-necessarily Normalized Relation in the Relational Data Model".*Proc. of the 3rd Int. Conf.VLDB*, 1977, 447-453.

[2] A. Yazici, A. Soysal, B. Buckles and F. Petry, "Uncertainty in a Nested Relational Database Model", *Data and Knowledge Engineering*, 30, #3, pp275-301, 1999.

[3] W. Stallings and. L, Brown. *Computer Security: Principles and Practice*, Prentiss Hall, 2007.

[4] D. Denning and P. Denning, "Data Security", *ACM Computing Surveys*, **11**, 2-54, 1979.

[5] E. Fernandez, R. Summers and C. Wood. *Database Security and Integrity*. Addison-Wesley, Reading MA. 1981.

[6] E. Leiss. *Principles of Data Security*. Plenum Press, New York, 1982.

[7] R. Elmasri and S. Navathe, *Fundamentals of Database Systems, 5th ed*., Addison Wesley, 2007

[8] F. Chin, "Security in Statistical Databases for Queries with Small Counts", *Transactions onn Database Systems*,, **3**, 212-237, 1978.

[9] D. Denning,. "Secure Statistical Databases with Random Sample Queries," *Transactions onn Database Systems*, 5: 3, 1980, 291-315.

[10] Chin, F. and Ozsoyoglu, G., "Statistical Database Design, "*Transactions onn Database Systems*, 6: 1, 1981, 113-139.

[11] E. Leiss, "Randomizing: A Practical Method for Protecting Statistical Databases Against Compromise", *Proc. Very Large Databases Conference*, Mexico, 1982, 189-196.

[12] H. Wong, "Micro and Macro Statistical/Scientific Database Management," *Int. Conf. on Data Engineering,* 1984, 104-106.

[13] M. McLeish,. "Further Results on the Security of Partitioned Dynamic Statistical Database," *Transactions onn Database Systems*, 14: 1, 1989, 98-113.

[14] A. Motro., D. Marks, and S. Jajodia, "Aggregation in Relational Databases: Controlled Disclosure of Sensitive Information," in *Proceedings of ESORICS 94, Third European Symposium on Research in Computer Security*, Brighton, UK, November 1994. Lecture Notes in Computer Science No. 875, Springer-Verlag, 431–445.

[15] C. Clifton and D. Marks. "Security and Privacy Implications of Data Mining," *Proc. ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*, Canada, 1996, 15-19.

[16] H. Schek and M. Scholl, "Relational Model with Relational-Valued Attributes," *Information Systems,* **11**, 137-147, 1986.

[17] D. Van Gucht "On the Expressive Power of the Extended Relational Algebra", *Proc. 6th Princip. of Database Systems*, 302-312, 1987.

[18] L. Colby , "A Recursive Algebra for Nested Relations," *Information Systems*, **15**, 567-562 1990.

[19] M. Roth., H. Korth, and D. Batory, "SQL/NF: A Query Language for Non-1NF Relational Databases," *Information Systems,* **12**, 99-114, 1987.

[20] S. Thomas and P. Fischer, "Nested Relational Structures," *Advances in Computing Research* , 3, 269-307, JAI Press, Greenwich CT, 1989.

[21] G. Ozsoyoglu, z. Ozsoyoglu, and V. Matos, "Extending Relational Algebra and Relational Calculus with set valued Attributes", *Transactions onn Database Systems*,, 12,#4, pp. 566-592, 1987.

[22] J. Grant, " Incomplete Information in a Relational Database", *Fundamenta Informaticae*, **3**, 363-378, 1980.

[23] T. Beaubouef, F. Petry and G. Arora "Information-Theoretic Measures of Uncertainty for Rough Sets and Rough Relational Databases". *Information Sciences*, 109, 1998, 185-195.

[24] F. Petry, F. *Fuzzy Databases: Principles and Applications*. Kluwer Academic Publishers, Boston 1996.

[25] J. Galindo, A. Urrutia and M. Piattini, *Fuzzy Databases: Modeling, Design and Implemenataion*. Idea Group Publishing, Hershey PA, 2006.

[26] B. Buckles and F. Petry, "A Fuzzy Model for Relational Databases", *Int. Jour. Fuzzy Sets and Systems*, **7**, 213-226, 1982.

[27] B. Buckles and F. Petry, "Fuzzy Databases and their Applications", *Fuzzy Information and Decision Processes, Vol. 2*, (eds. M. Gupta and E. Sanchez), 361-371, North-Holland, Amsterdam, 1982.

[28] L. Zadeh, "Similarity Relations and Fuzzy Orderings.". *Information Sciences*, **3**, 177-200, 1970.

[29] B. Buckles and F. Petry, "Security and Fuzzy Databases," *Proceedings of 1982 IEEE International Conference on Cybernetics and Society*, 622-625, Seattle WA 1982.

[30] B. Buckles and F. Petry, "Information Theoretic Characterization of Fuzzy Relational Data Bases" *IEEE Transactions on Systems, Man and Cybernetics* , **12**, 74-77, 1983.

[31] A. DeLuca and S. Termini, "A Definition of Nonprobabilistic Entropy in the Setting of Fuzzy Sets Theory", *Information and Control*, **20**, 301-334, 1972.

[32] C. Shannon and W. Weaver, *The Mathematical Theory of Communication*, Univ. of Illinois, Urbana, IL, 1949.

[33] W. Zie and S. Bedrosian, "The Information in a Fuzzy Set and the Relation between Shannon and Fuzzy Information", *Proc. Conf. on Information Science and Systems*, 102-107, 1982.

[34] M. Thomason, "On Applications of Probabilistic Information Theory to Relational Databases", *Proc. SPIE Technical Symposium*, 104-109, 1979.

[35] Z. Pawlak,. *Rough Sets: Theoretical Aspects of Reasoning About Data*, Kluwer Academic Publishers, Norwell, MA 1991

[36] T. Beaubouef, F. Petry "Rough Set Model for Relational Databases",, *RKSD' 93, Lecture Notes in Computer Science*, Springer-Verlag, London, 1994.

[37] T. Beaubouef, F. Petry and B. Buckles "Extension of the Relational Database and its Algebra with Rough Set Techniques". *Computational Intelligence*. 11,1995, 233-245.

[38] T. Beaubouef, F. Petry and G. Arora, "Information-theoretic Measures of Uncertainty for Rough Sets and Rough Relational Databases", *Information Sciences*, 109, #1, 185-195, 1998.

[39] Y.Sui, Y. Xia and J. Wang, :"The Information Entropy of Rough Relational Databases", *RSFDGrC* 2003, LNAI 2639, pp. 320-324, Springer Verlag, Berlin, 2003.