A Fuzzy Model for Relational Databases

Fuzzy Sets and Systems 7 (1982) 213–226
North-Holland Publishing Company

A FUZZY REPRESENTATION OF DATA FOR RELATIONAL DATABASES

Billy P. BUCKLES

General Research Corporation, Southern Division, Huntsville, AL 35805, USA

Frederick E PETRY

Department of Computer Science, Tulane University, New Orleans, LA 70118, USA

Received November 1979
Revised October 1980

A structure for representing inexact information in the form of a relational database is presented. The structure differs from ordinary relational databases in two important respects: Components of tuples need not be single values and a similarity relation is required for each domain set of the database. Two critical properties possessed by ordinary relational databases are proven to exist in the fuzzy relational structure. These properties are (1) no two tuples have identical interpretations, and (2) each relational operation has a unique result.

Keywords: Fuzzy sets, Relational databases, fuzzy relational databases, Similarity relations, Similarity threshold.

1. Introduction

A primary objective of fuzzy analysis has been its application to problems for which precise formulation is not possible and the uncertainty in problem parameters is not probabilistic uncertainty. If this can be done in a manner that allows the exact representation to be a special case of the fuzzy representation, then, clearly, the fuzzy representation is the more powerful. The field of database construction and subsequent use via query languages is an application area of great importance.

In many cases information is found to be naturally imprecise or fuzzy as when representing personality traits, physical features of individuals, and subjective opinions and judgments in situations such as policy preferences, medical decision making, economic forecasting, and personnel evaluation. In addition, even when quantitative information is available, it may be preferable to analyze it in a fuzzy context to facilitate qualitative comparisions. In an area such as economic statistics, for instance, numerical values of unemployment are available exactly as, say, 4.1%, 6.2%, 10.2%. The representation of these values on an ordinal scale, however, would permit them to be dealt with subjectively as low, moderate, and very high.

The work reported here describes a consistent approach to incorporating fuzzy information within the specification of a database and mechanisms for the manipulation of the information (via queries) that preserve not only the information but the nuances of fuzzy uncertainty. That is, precise responses will not be formulated to queries based on fuzzy data.

The idea of null values and incompletely or partially specified records in databases has been under study by Grant [4–6] and Lipski [8]. They generally attempt to formulate consistent schemes to permit the interpretation of queries from such incomplete data. However, the concept of fuzzy data was not used because they considered that while data could be missing (incomplete), the existing data was exact, i.e., nonfuzzy. Fuzzy data structures were studied by Mizumoto et al. [9] in which the membership grades were directly coupled with each datum and relation. The work reported here is quite distinct from that of Mizumoto in that membership grades are not associated with individual data items but with entire classes. In addition, the primitive operations of the fuzzy algebra have been incorporated into a higher level algebra.

Specifically, a fuzzy representation for relational databases is defined and examined here. A fuzzy relational algebra is described for which ordinary relational algebra is a special case. Throughout the definition, careful attention is given to compatibility with ordinary relational database fundamentals and important properties dealing with consistency are proven.

2. Organization of a fuzzy database

The organization of relational databases is based on set theory and relation theory. The concept was introduced by Codd [1] and an excellent description can be found in Date [2]. Essentially, relational databases consist of one or more relations in two-dimensional (row and column) format. Rows are called tuples and correspond to records; columns are called domains and correspond to fields. One or more of the domains are distinguished as the key domains. It is desirable to maintain a relation in the third normal form in order to avoid certain redundancy problems and storage anomalies. A relation is in third normal form if the key and nonkey domains possess two characteristics. First, each domain must be fully dependent on the entire key and not a portion of it if the key encompasses more than one domain. Second, each of the nonkey domains must be nontransitively dependent on the key. That is, they are direct attributes of the key and not of each other.

One means of access to a relational database into which high level query languages are often translated is the relational algebra. The operations of the algebra are applied procedurally. For example, consider a relation FAMILY that consists of the domains FATHER, MOTHER, SON, and DAUGHTER. To construct a new relation consisting of the domains FATHER and SON, perhaps restricted to the cases in which the mother's name is 'Beth', one projects the original relation.

PROJECT (FAMILY: FATHER, SON) WHERE MOTHER = 'BETH'

Thus, relational algebra operation consists of (1) an operation name, (2) one or more relation names, (3) one or more domain names, and (4) an optional conditional expression.

Upon completion of a PROJECT or other operation, there is an implied elimination of redundant tuples from the result. A tuple is redundant if it is identical to another tuple in the relation. Other operations (e.g., INTERSECTION) depend even more directly on equivalence using identity. It is the weakening of dependence on this form of equivalence that leads to the concept of fuzzy databases to be presented here.

2.1. *Fuzzy data*

The idea of representing imprecise information for purposes of retrieval and analysis is not new. As previously noted, Grant [4–6] has investigated the impact of null and multiple-valued domains, his purpose being to find appropriate responses to queries in which these conditions exist. Mizumoto et al. [9] have defined and implemented a collection of set operations and data structures capable of coping with analysis performed over fuzzy sets. Much earlier, Kling [7] designed a LISP-like language for manipulating fuzzy relations stored in a structure resembling a semantic net. The work reported here attempts to generalize the concept of null and multiple-valued domains for implementation within an operational environment consistent with the relational algebra. In fact, the nonfuzzy relational database will be seen to a special case of the fuzzy relational database.

For each domain, j, in a relational database, a domain base set, D_j, is understood. Domains for fuzzy relational databases will be either discrete scalars or discrete numbers drawn from either a finite or infinite set. An example of a finite scalar domain is {poor, average, good, excellent}. The domain values of a particular tuple may be single scalars or numbers (including null) or a sequence of scalars or numbers. For example:

STUDENT	APTITUDE	AGE
{TOM}	{AVERAGE, GOOD}	{19, 20, 21}

The identity relation used in nonfuzzy relational databases induces equivalence classes (most frequently singleton sets) over D_j which affect the result of certain operations and the removal of redundant tuples. The identity relation is replaced in the fuzzy relational database by an explicitly declared similarity relation [11] of which the identity relation is a special case.

A similarity relation, $s(x, y)$, for given domain, D_j, is a mapping of every pair of elements in the domain onto the unit interval [0, 1]. A similarity relation is reflexive (with $s(x, x) = 1$) and symmetric as in an equivalence relation. Special forms of transitivity prevail, however.

T1: $$s(x, z) \geq \max_{y \in D_j}\{\min[s(x, y), s(y, z)]\},$$

T2: $$s(x, z) \geq \max_{y \in D_j}\{s(x, y) * s(y, z)\}$$

where $*$ is arithmetic multiplication.

An example of a similarity relation for a finite scalar domain base set satisfying T1 is:

	w	x	y	z
w	1	0	0	0
x	0	1	0.5	0.3
y	0	0.5	1	0.3
z	0	0.3	0.3	1

where $D_j = \{w, x, y, z\}$. Note that w is similar to only itself; thus, a subset of a domain base set can have e-similarity (i.e., be the identity relation). An example of a similarity relation satisfying T2 is:

$$s(x, y) = \exp(-\beta\ |y - x|)$$

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

A domain base set of a fuzzy database may consist of either

(1) a finite set of scalars and a similarity relation satisfying T1 which may be simply e-similarity;

(2) an infinite set of scalars and the e-similarity relation;

(3) a finite set of numbers with a similarity relation having property T1 or T2, again including e-similarity; or

(4) an infinite set of numbers with a similarity relation having property T2 or e-similarity.

The model developed in this paper is principally based on the first two alternatives.

2.2. *Fuzzy tuples and interpretations*

A key aspect of the fuzzy relational database is that domain values need not be atomic. A domain value, d_{ij} where i is the tuple index, is defined to be a subset of its domain base set, D_j. That is, any member of the powerset, 2^{D_j}, may be a domain value except the null set. Let $\mathbf{2}^{D_j}$ denote any non-null member of the powerset of D_j.

Definition. A *fuzzy relation, R*, is a subset of the set cross product $\mathbf{2}^{D_1} \times \mathbf{2}^{D_2} \times \cdots \times \mathbf{2}^{D_m}$.

Membership in a specific relation, $\mathfrak{R}$, is determined by the underlying semantics of the relation. For instance, if D_1 is the set of major cities and D_2 is the set of countries, then (Paris, Belgium) $\in \mathbf{2}^{D_1} \times \mathbf{2}^{D_2}$ but is not a member of the relation $\mathfrak{R}$ (capital, country).

Any member of a relation is called an ordered m-tuple or, more simply, a tuple.

Definition. A *fuzzy tuple, t*, is any member of both R and $\mathbf{2}^{D_1} \times \mathbf{2}^{D_2} \times \cdots \times \mathbf{2}^{D_m}$.

An arbitrary tuple is of the form $t_i = (d_{i_1}, d_{i_2}, \ldots, d_{i_m})$ where $d_{ij} \subseteq D_j$.

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

In summary, the space of interpretations is the set cross product $D_1 \times D_2 \times \cdots \times D_m$. However, for any particular relation, the space is limited by the set of valid tuples. Valid tuples are determined by an underlying semantics of the relation. Note that in an ordinary relational database, a tuple is equivalent to its interpretation.

2.3. *Similarity thresholds and fuzzy relational algebra operations*

As previously defined, domain values, d_{ij}, consist of one or more elements from the domain base set, D_j. That is, $d_{ij} \subseteq D_j$ where $i = 1, 2, \ldots, n$, the tuple index and $j = 1, 2, \ldots, m$, the domain index. Given a domain, D_j, in a relation, the similarity threshold is defined to be:

$$\text{THRES}(D_j) = \min_{\forall i}\left\{\min_{x, y \in d_{ij}} [s(x, y)]\right\}.$$

Note that in a nonfuzzy database, the cardinality of $d_{ij} = 1$ and $s(x, x) = 1$ so $\text{THRES}(D_j) = 1$ for all j. It will be shown next that 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.

A fuzzy relational algebra operation consists of the same four parts as an ordinary relational algebra operation. In addition, there is a clause defining minimum similarity thresholds. Consider the operation:

PROJECT(REL 1: APTITUDE, AGE) WITH THRES(APTITUDE) ≥ 0.75,
THRES(AGE) ≥ 0.80.

The relation REL 1 has any number of domains but one of them is name APTITUDE and a second is named AGE. The relation created by the PROJECT operation contains only the domains APTITUDE and AGE. That is, the new relation is created by removing all domains (i.e., columns) from REL 1 except these two. 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) either the minimum threshold for APTITUDE (0.75) or the minimum threshold for AGE(0.80). The minimum threshold constraints will be subsequently referred to as the *level* values.

If the operation were 'intersection' of two identically formatted relations, then any result tuple would be achieved by merging one or more tuples from *each* of the original relations such that the level values are not violated. A result tuple acquired through a 'union' operation would be achieved by merging one or more tuples from *either or both* of the original relations. Anytime a level value is missing, it is assumed to be one (1), that is, the same as assumed for a nonfuzzy 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.

3.* Redundancy and determinancy properties

In a nonfuzzy database, a tuple is redundant if it is exactly the same as another tuple. Any operation over a nonfuzzy 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 on the similarity thresholds.

Definition. $t_i = (d_{i1}, d_{i2}, \ldots, d_{im})$ and $t_k = (d_{k1}, d_{k2}, \ldots, d_{km})$, $i \neq k$, are *redundant* if

$$\text{LEVEL}(D_j) \leq \min_{x,y \in d_{ij} \cup d_{kj}} [s(x, y)]$$

for $j = 1, 2, \ldots, m$ and $\text{LEVEL}(D_j)$ given a priori.

In a fuzzy database, each tuple can potentially represent a large number of interpretations, each an element of the cross product of the domain values. 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, can the relation

green	soft
red	bright
blue	cool
black	dark

be reduced according to some pair of level values to

{green, blue}	{soft, cool}
{red, blue, black}	{bright, cool, dark}

where (blue, cool) 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 by updating. Fortunately, the situation illustrated above is impossible. Let T_i be the set of possible interpretations for tuple t_i.

Theorem. *Given a fuzzy relation with no redundant tuples and each domain similarity relation formulated according to* T1 *then* $T_i \cap T_j = \emptyset$ *if* $i \neq j$.

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

Let $x, y \in d_{ih}$ be such that $s(x, y) = \min_{u,v \in d_{ih}}[s(u, v)]$ and $x', y' \in d_{jh}$ be such that $s(x', y') = \min_{u,v \in d_{jh}}[s(u, v)]$. Then, in particular for a_h

$$s(x, a_h) \geqslant s(x, y),\ s(x', a_h) \geqslant s(x', y').$$

Taking the minimum on each side and using the symmetric and T1 transitivity properties

$$\min[s(x, a_h), s(a_h, x')] \geqslant \min[s(x, y), s(x', y')]$$

and thus

$$s(x, x') \geqslant \min[s(x, y), s(x', y')].$$

By definition, $\text{LEVEL}(D_h) \leqslant s(u, v)$ where u, v are in the same domain value. Thus, $\text{LEVEL}(D_h) \leqslant \min[s(x, y), s(x', y')] \leqslant s(x, x')$. Again, applying the definition of level value and applying transitivity, for all $u \in d_{ih}$, $v \in d_{jh}$

$$s(x, v) \geqslant \min[s(x, x'), s(x', v)] \geqslant \text{LEVEL}(D_h)$$

and thus

$$s(u, v) \geqslant \min[s(u, x), s(x, v)] \geqslant \text{LEVEL}(D_h).$$

So 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. □

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, $\text{THRES}(D_j)$, are one (1), the relation is nonfuzzy 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 remains.

It would also be very desirable if the removal of redundancy could have but one outcome. To illustrate by exhibiting the converse, assume the relation

green	soft
red	bright
blue	cool
black	dark

had redundant tuples according to some pair of minimum similarity thresholds, $\text{LEVEL}(D_1)$ and $\text{LEVEL}(D_2)$. Would it be possible to derive two different relations

{green, red}	{soft, bright}
{blue, black}	{cool, dark}

{green, blue}	{soft, cool}
{red, black}	{bright, dark}

that have no redundant tuples? Fortunately, this situation is also impossible.

Theorem. *A fuzzy relation derived by merging redundant tuples is unique if each similarity relation satisfies property* T1.

Proof. Let $R = \{t_1, t_2, \ldots, t_n\}$. A subset of tuples, $\{t_i, t_j, \ldots, t_k\}$, each of which is redundant with every other tuple in the set is merged into a single tuple, $(t_i \cup t_j \cup \cdots \cup t_k)$, through the set union of the respective domain values. If the order of the redundant subset is simply one, it is copied directly to the new relation, R'. In general, it will be shown that redundancy, denoted $t_i \sim t_j$, is an equivalence relation and induces a unique partition of R. The uniqueness of the partition is sufficient to prove the theorem as each class is merged to form a single tuple in R'.

A tuple is redundant with itself, $t_i \sim t_i$, thus redundancy is reflexive. Since similarity measures from which $\text{LEVEL}(D_h)$ is derived are symmetric, $t_i \sim t_j$ implies $t_j \sim t_i$ assuring that redundancy is also symmetric. Given $t_i \sim t_j$ then for any domain h, $u \in d_{ih}$, $v \in d_{jh}$, then $s(u, v) \leqslant \text{LEVEL}(D_h)$. Given $t_j \sim t_k$, $w \in d_{kh}$, then $s(v, w) \leqslant \text{LEVEL}(D_h)$. By transitivity of similarity, $s(u, w) \geqslant \min[s(u, v), s(v, w)] \geqslant \text{LEVEL}(D_h)$ implying $t_i \sim t_k$, or, in other words, redundancy is transitive. Thus, redundancy is an equivalence relation and induces a unique partition in R. Each block of the partition, $\{t_i, t_j, \ldots, t_k\}$, is a subset of the tuples of R that is merged by set union of domain values to form one tuple, $(t_i, t_j, \ldots, t_k)$, in R'. R' is thus uniquely derived. □

The above result can be extended to show that the relational algebra commands PROJECT, UNION, and INTERSECTION give unique results in a fuzzy environment. Specifically, fuzzy PROJECT and UNION differ from their nonfuzzy counterparts only in the manner of removing redundant tuples. The above theorem is, thus, sufficient to assure their result is always unique. Fuzzy INTERSECTION produces a relation for which each tuple is the outcome of merging one (or more) tuples from each of two argument relations according to given similarity criteria. That is, a tuple in the first argument relation is represented in the output relation if and only if it is redundant with a tuple in the second argument relation. Therefore, the same approach used in the above proof is used in proving uniqueness of result for INTERSECTION. Unfortunately, fuzzy relations with one or more domains formulated with transitivity property T2 do not share the benign characteristics of unique representation and no common interpretation of distinct tuples. If some such relation originally had these two characteristics, then at least the no common interpretation property can be preserved through carefully formulated operation rules.

4. Application

On an athletic team there are usually a number of specialized positions. Various similarities and dissimilarities exist between pairs of positions. On a baseball team, for example, a manager needing a right fielder might select a center fielder to fill that position. Assume that most managers consider outfielders completely

	P	C	LF	CF	RF	FB	SB	TB	SS
P	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
C	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
LF	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0
CF	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0
RF	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0
FB	0.0	0.0	0.0	0.0	0.0	1.0	0.6	0.6	0.6
SB	0.0	0.0	0.0	0.0	0.0	0.6	1.0	0.8	0.9
TB	0.0	0.0	0.0	0.0	0.0	0.6	0.8	1.0	0.8
SS	0.0	0.0	0.0	0.0	0.0	0.6	0.9	0.8	1.0

P Pitcher CF Centerfielder SB Secondbaseman
C Catcher RF Rightfielder TB Thirdbaseman
LF Leftfielder FB Firstbaseman SS Shortstop

Fig. 1. Similarity relation for baseball team positions.

interchangeable while pitchers and catchers are unique with respect to the skills required. With respect to the infielders, assume a manager believes there are varying degrees of interchangeability. Let first basemen be the least interchangeable, shortstops be very (but not perfectly) interchangeable with the second basemen, and third basemen slightly less interchangeable. A similarity relation reflecting these beliefs is shown in Fig. 1. In the example that follows, it is assumed the reader is familiar with the ordinary relational algebra from which the fuzzy algebra is derived. This example will illustrate a typical correspondence between classes into which data is partitioned and the level values required in a specific application.

If a manager wished to trade one or more players in order to strengthen his team, he would probably deal with another manager with the same purpose in mind. In order to identify teams whose managers are willing to discuss a trade, a fuzzy relational database maintained by the league would be helpful. This database would have a relation AGENTS with domains PLAYER (domain set is 'person'), POSITION (domain set is 'position'), and CLUB (domain set 'team'). AGENTS contains all players, their positions, and their teams. At any one time, some teams would be in search of new talent. This would be maintained in a relation called BUYERS with domain CLUB (domain set 'team') and domain NEEDS (domain set 'position'). At any one time, some teams would have an excess of talent at some positions. This information would be contained in a relation called EXTRAS with domain POSITION (domain set 'position') and domain EXCESS-AT (domain set 'team'). This database is illustrated in Fig. 2. (The values for the database were chosen randomly, using a table of random numbers.)

Assume that periodically the league president sends a list of potential player trades to each manager. The high level query is

"Which pairs of teams could *benefit* from player trades?"

AGENTS

PLAYER	POSITION	CLUB
Jones	C	Braves
Black	TB	Giants
Conley	SS	Braves
Gonzales	FB	Reds
Dunn	LF	Astros
Oliver	TB	Reds
Helms	P	Padres
DeCamp	CF	Padres
Smith	P	Angels
Powers	SS	Braves
Mix	P	Astros
Moore	LF	Astros
Davis	LF	Angels
Brown	TB	Padres
Pugh	RF	Angels
Young	RF	Reds
Wales	C	Angels
Varner	SS	Angels
Walker	CF	Padres
Sisco	C	Giants
Shannon	TB	Braves
Hicks	FB	Giants

BUYERS

CLUB	NEEDS
Braves	CF
Reds	SS
Angels	SB
Reds	FB
Astros	SB
Giants	TB
Angels	P
Giants	FB
Reds	P
Reds	LF
Padres	LF
Braves	RF
Giants	RF
Padres	C

EXTRAS

POSITION	EXCESS-AT
TB	Padres
CF	Angels
FB	Angels
P	Giants
RF	Astros
SS	Padres
C	Astros
SB	Padres
LF	Angels
C	Reds
FB	Braves
SB	Braves

Fig. 2. Relational database for baseball league.

The word '*benefit*' is a fuzzy term in this query which in this context implies both teams should benefit to some degree. Assume the league president assigns a benefit level of 0.7. If it is assumed that he must work with the fuzzy algebra rather than a higher query language, then he would begin by constructing a relation of teams that can release players.

R1 ← JOIN (AGENTS, EXTRAS)
WITH THRES(POSITION) ≥ 0.7,
THRES(NAME) ≥ 0.0.

Because both POSITION and CLUB in AGENTS are distinguished columns, this operation chooses precisely which players can be traded. Positions are combined if they are similar to the degree stated. Players are combined with no other constraint than that they are within the same position category. The result is shown in Fig. 3.

Next, the league president must determine which teams are interested in acquiring which players. BUYERS contains a list of teams with the positions for which acquisitions are desired. The league president needs to JOIN the relation BUYERS with R1. Only the position column in each relation should be treated as distinguished even though both have a column for which the domain set is team.

R1

SELLER	POSITION	PLAYER
Angels	CF,LF,RF	Davis, Pugh
Astros	LF,RF	Dunn, Moore
Padres	SS,TB,SB	Brown
Braves	SS,TB,SB	Conley, Powers, Shannon

Fig. 3. Teams with extra players.

An obvious augmentation to the syntax used thus far is used below to designate the column over which JOIN is to be performed.

R2 ← JOIN(R1, BUYERS: POSITION)
WITH THRES(POSITION) ≥ 0.7.

The result is shown in Fig. 4.

Relation R2 contains the teams having compatible requirements, but it is not

R2

BUYER	SELLER	POSITION	PLAYER
Braves	Angels	CF,LF,RF	Davis, Pugh
Braves	Astros	CF,LF,RF	Dunn, Moore
Reds	Padres	SS,TB,SB	Brown
Reds	Braves	SS,TB,SB	Conley, Powers, Shannon
Angels	Padres	SS,TB,SB	Brown
Angels	Braves	SS,TB,SB	Conley, Powers, Shannon
Astros	Padres	SS,TB,SB	Brown
Astros	Braves	SS,TB,SB	Conley, Powers, Shannon
Giants	Padres	SS,TB,SB	Brown
Giants	Braves	SS,TB,SB	Conley, Powers, Shannon
Reds	Angels	CF, LF, RF	Davis, Pugh
Reds	Astros	LF,RF	Dunn, Moore
Padres	Angels	CF,LF,RF	Davis, Pugh
Padres	Astros	LF,RF	Dunn, Moore
Giants	Angels	CF,LF,RF	Davis, Pugh
Giants	Astros	LF,RF	Dunn, Moore

Fig. 4. Potential player acquisitions.

R3

BUYER	SELLER	S-POSITION	S-PLAYER	B-POSITION	B-PLAYER
Angels	Padres	SS,TB,SB	Brown	CF,LF,RF	Davis, Pugh
Angels	Braves	SS,TB,SB	Conley, Powers, Shannon	CF,LF,RF	Davis, Pugh
Astros	Padres	SS,TB,SB	Brown	LF,RF	Dunn, Moore
Astros	Braves	SS,TB,SB	Conley, Powers, Shannon	LF,RF	Dunn, Moore
Braves	Angels	CF,LF,RF	Davis, Pugh	SS,TB,SB	Conley, Powers, Shannon
Braves	Astros	CF,LF,RF	Dunn, Moore	SS,TB,SB	Conley, Powers, Shannon
Padres	Angels	CF,LF,RF	Davis, Pugh	SS,TB,SB	Brown
Padres	Astros	LF,RF	Dunn, Moore	SS,TB,SB	Brown

Fig. 5. The league president's final relation.

limited to the teams having compatible resources. That is, each BUYER team and each SELLER team do not have players of the proper position category to trade. This can be rectified by taking the PRODUCT of R1 and R2 while simultaneously requiring that the SELLER of R1 (R1. SELLER) be equal to the BUYER of R2 (R2. BUYER). That is, two tuples, one from each operand, can produce a result tuple only if they satisfy this equality.

R3 ← PRODUCT (R1, R2)
IF R1. SELLER = R2. BUYER.

R3 is shown in Fig. 5.

R3 is the paired list that the league president sought. With the aid of the fuzzy algebra operations, it required only three steps to produce. It could be produced with ordinary relational algebra but with greater difficulty.

There are many other queries for which the answers could easily be provided using fuzzy operations. Among them are:

"With which teams would Black have the best chance of succeeding?"

"Which group of players is in the greatest peril of being dropped from their teams and which group is in the least peril?"

"On which positions will there be concentration when the league holds a new player draft?"

Each of these queries depend on some measure of similarity between positions. Given a measure of similarity between teams perhaps based on league standing or similarity between players, the potential power of the database increases accordingly.

There are many extensions of this application to the scientific and commercial environments. Imagine a corporation that plans to establish an office or division at a new location. It is often the policy of corporations to transfer employees to establish a corps group which is then augmented by new employees. The corporation goals are to transfer employees that will be of most benefit to the new organization and least disrupt their previous organizations. This is a direct extension of the baseball player application.

Other applications include the comparision of a company's product line against perceived demand and the product lines of competitors. Instead of product lines, the question may involve the location of new warehouses or distributorships. It is true that many such applications may ultimately depend on favorable cost/benefit analyses. Yet, such analyses can best be performed after the number of candidate alternatives has been reduced.

5. Conclusions

The given formulation for fuzzy relational databases possesses the desirable property that the nonfuzzy database is a special case. The similarity measures required in the specification can be 'personalized'. That is, they could represent the subjective opinions of an individual where the individual is an actual person, group of persons, or an organization. In this sense, the database can capture the value systems of different individuals and thus respond differently to the same queries depending on the querier.

Future directions of inquiry include nonprocedural query languages, investigation of fuzzy functional dependency between domains, and measures of similitude between tuples. The authors have already begun such studies. In addition, this framework for database definition offers interesting insights into information content as data progresses from preciseness to impreciseness as measured by fuzzy entropy [3] and its probabilitistic counterpart [10]. This also is being investigated.

In summary, the concept of fuzzy relational database offers promise both as a tool of practical value and a vehicle for further study of fuzzy phenomena.

References

[1] E.F. Codd, A relational model of data for large shared data banks, Comm. ACM 13 (1970) 377–387.

[2] C.J. Date, An Introduction to Database Systems, 2nd edn. (Addison-Wesley, Reading, MA, 1977).

[3] A. Deluca and S. Termini, A definition of a non-probabilistic entropy in a setting of fuzzy sets theory, Inform. and Control 20 (1972) 301–312.

[4] J. Grant, Null values in a relational data base, Inform. Process. Lett. 6 (1977) 156–157.

[5] J. Grant, Partial values in a tabular database model, Inform. Process. Lett. 9 (1979) 97–99.

[6] J. Grant, Incomplete information in a relational database, Fund. Informaticae, to appear.

[7] R. Kling, Fuzzy-Planner: Reasoning with inexact concepts in a procedural problem-solving system, J. Cybernet. 4 (1974) 105–122.

[8] W. Lipski, On semantic issues connected with incomplete information databases, Trans. Database Systems 4 (1979) 262–296.
[9] M. Mizumoto, M. Umano and K. Tanaka, Implementation of a fuzzy-set-theoretic data structure system, Proc. Third Int. Conf. on Very Large Databases, Tokyo (1977) 59–69.
[10] C.E. Shannon and W. Weaver, The Mathematical Theory of Communication (Univ. of Ill. Press, Urbana, IL, 1949).
[11] L.A. Zadeh, Similarity relations and fuzzy orderings, Inform. Sci. 3 (1970) 177–200.