Int. Computer Symposium, Dec. 15-17, 2004, Taipei, Taiwan.

Integrating Fuzzy Databases with Inconsistent Data Redundancy*

Julie Yu-Chih Liu
Department of Information Management
Yuan Ze University
imyuchih@saturn.yzu.edu.tw

Jun-Lin Lin
Department of Information Management
Yuan Ze University
jun@saturn.yzu.edu.tw

Abstract–*Data integration is highly complex in fuzzy relational databases, partially because of the involvement of the resemblance relation matrix. Inconsistent data redundancy may occur when the fuzzy databases to be integrated are associated with different relation matrices on a given domain. This work presents a solution for integrating fuzzy relational databases with inconsistent data redundancy.*

Keywords: similarity relations, proximity relations, fuzzy relational databases, data integration

1. Introduction

Data integration has been extensively studied in the context of classical databases [1], [2], [3], [4]. Several works [5], [6], [7], [8] have also discussed data integration in fuzzy databases, but they only focus on joining database relations within one fuzzy database. Integrating database relations from multiple fuzzy databases has not yet been addressed, and it is more complex because of more heterogeneity involved. Heterogeneity may initially occur in the resemblance relation matrices among multiple fuzzy databases.

A fuzzy relational database consists of a set of database relations. A fuzzy database relation r of a relation schema $R(A_1, A_2, \ldots, A_m)$ is a subset of the set cross product $\Phi(D_1)\times\Phi(D_2)\times\cdots\times\Phi(D_m)$, where $\Phi(D_j) := 2^{D_j} - \emptyset$. Each D_j is the domain of attribute A_j, and is either discrete scalars or discrete numbers drawn from a finite or infinite set. An example of a finite scalar domain is {poor, average, good, excellent}. Let t_i represent the i-th tuple of r, and t_i be of the form $(d_{i1}, d_{i2}, \ldots, d_{im})$. Each component d_{ij} of t_i is a non-empty subset of the corresponding domain D_j. That is, $d_{ij} \subseteq D_j$ and $d_{ij} \neq \emptyset$. An interpretation of t_i is a tuple $\theta = (a_1, a_2, \ldots, a_m)$ where $a_j \in d_{ij}$ for each domain D_j.

Buckles and Petry use the *similarity relation* to describe the degree of resemblance between elements in a scalar domain, and defined redundant tuples [9]. A similarity relation is a mapping $s_j : D_j \times D_j \to [0, 1]$ such that for $x, y, z \in D_j$,

$$\begin{aligned} &s_j(x, x) = 1 && \text{(reflexivity)}\\ &s_j(x, y) = s_j(y, x) && \text{(symmetry)}\\ &s_j(x, z) \geq \max_{y\in D_j}\{\min\{s_j(x, y), s_j(y, z)\}\} && \text{(max-min transitivity)}\end{aligned}$$

Different similarity relations and different associated thresholds may induce different equivalence classes. Equivalence classes serves as the basis of data redundancy. Two fuzzy databases might employee different similarity relations and/or different thresholds, and thus have different data redundancy, termed *inconsistent* database relations. It is difficult to integrate data from inconsistent database relations without violating the principle of fuzzy relational database, namely, database relation contains no redundant tuple.

This paper first demonstrates the problem of integrating data from inconsistent database relations in the resemblance-based database models [9], [10], and then presents a solution for integrating inconsistent database relations. Accordingly, the integration result does not violate the principle of fuzzy relational database, and can be interpreted soundly.

2. Preliminaries

The fuzzy database relation removes redundant tuples by tuple merging. Tuples $t_i = (d_{i1}, d_{i2}, \ldots, d_{im})$ and $t_k = (d_{k1}, d_{k2}, \ldots, d_{km})$ are *redundant* if

$$\min_{x,y\in d_{ij}\cup d_{kj}} s_j(x, y) \geq \alpha_j$$

for $j = 1, 2, \ldots, m$, and each α_j given apriori is the threshold of similarity relation s_j on domain D_j.

Theorem 1[9]**.** *Let r be a database relation generated by merging the redundant tuples according to the level constraints on similarity relations. If T_i represents the set of interpretations of a tuple t_i, then for $t_i, t_k \in r$,*

$$T_i \cap T_k = \emptyset \quad \textit{whenever} \quad t_i \neq t_k. \qquad \square$$

Shenoi and Melton [10] noted that Theorem 1 depends on the fact that a similarity relation induces disjoint subsets (also called *equivalence classes*) by *α-similarity*. Given any $\alpha \in [0, 1]$

*Supported by the National Science Council under Grant NSC 92-2416-H-155-007.

and a similarity relation s on domain D, two elements $x, y \in D$ are α-*similar* (denoted by $xS_\alpha y$) if $s(x, y) \geq \alpha$. And, a subset $C \subseteq D$ is an equivalence class [10] in the partition determined by S_α if and only if C is a maximal subset of D that satisfies the constraint

$$\min_{x,y \in C} s(x, y) \geq \alpha.$$

Shenoi and Melton [11] generalized the model of Buckles and Petry [9] by replacing similarity relation with *proximity relation*, and using α-*proximity* to induce equivalence classes. A *proximity relation* is reflexive and symmetric but not necessarily transitive. Let s be a proximity relation on D and $\alpha \in [0, 1]$. Two elements $x, z \in D$ are α-*proximate* (denoted by $xS^+_\alpha z$) if $xS_\alpha z$ or there exists a sequence $y_1, y_2, \ldots, y_r \in D$, such that

$$xS_\alpha y_1 S_\alpha y_2 \ldots S_\alpha y_r S_\alpha z.$$

Note that $xS_\alpha z$ is called α-*similarity* and holds if $s(x, z) \geq \alpha$.

3. The Problems of Inconsistent Integration

Data redundancy over a fuzzy database influences the elimination of redundant tuples through tuple merging. In [9], two redundant tuples $t = (d_1, d_2, \ldots, d_m)$ and $t' = (d'_1, d'_2, \ldots, d'_m)$ are merged into $t'' = (d''_1, d''_2, \ldots, d''_m)$ where $d''_j = d_j \cup d'_j$, $1 \leq j \leq m$. An example is given below.

Example 1. Consider a database relation $R(A_1, A_2)$, and let $\mathcal{D}_i$ denote the set of equivalence classes used on domain D_i of attribute A_i in R. The following figures reveal that, given the set r of tuples, the result of tuple merging in R varies with $\mathcal{D}_1$ and $\mathcal{D}_2$.

r

A_1	A_2
a	e
a	f
b	d
c	f

case 1:
$\mathcal{D}_1 = \{\{a, b\}, \{c\}\}$
$\mathcal{D}_2 = \{\{d, e\}, \{f\}\}$ $\Longrightarrow$

$r(R)$

A_1	A_2
$\{a, b\}$	$\{d, e\}$
a	f
c	f

case 2:
$\mathcal{D}_1 = \{\{a\}, \{b, c\}\}$
$\mathcal{D}_2 = \{\{d\}, \{e, f\}\}$ $\Longrightarrow$

$r(R)$

A_1	A_2
a	$\{e, f\}$
b	d
c	f

With classical databases, two *union compatible* database relations can be integrated using the *union* operation [12], which removes redundant tuples (namely, duplicates in the case of classical databases) from the integrated result. When generalizing the union operation to fuzzy databases, two union compatible database relations can be integrated via the traditional union operation, followed by tuple merging to remove redundant tuples from the integrated result.

Definition 1. Let $t \diamond t'$ denote the result of merging tuples t and t', and let r and r' be two union compatible fuzzy database relations. The union of r and r', denoted by $r \widetilde{\cup} r'$, then is given by

$$\begin{aligned} r \widetilde{\cup} r' = \quad & r'' \setminus \{t, t' \in r'' : t \cong t'\} \\ & \cup \{t \diamond t' : t, t' \in r'', t \cong t'\} \end{aligned}$$

where $r'' = r \cup r'$, and $t \cong t'$ denotes that t and t' are redundant. □

Notably, neither r nor r' contains redundant tuples since they comply with the principle of the fuzzy database, and consequently the union result contains no redundant tuples.

The prerequisite of the above definition is that r and r' agree with each other regarding data redundancy so that the union result can be based on the data redundancy of both operands. However, when two fuzzy database relations disagree with each other in terms of data redundancy, the data redundancy for the union result remains undefined. The union result cannot simultaneously agree with both operands unless the two operands agree with each other on data redundancy. If integrating two inconsistent database relations, the union result cannot simultaneously agree with both of them on data redundancy, and could violate Theorem 1.

Example 2. Consider the database relation $r(R)$ and its equivalence classes $\mathcal{D}_1$ and $\mathcal{D}_2$ of case 1 in Example 1, and a database relation $r'(R)$ described below, using equivalence classes $\mathcal{D}_1$ and $\mathcal{D}_2$ of case 2 in Example 1.

$r'(R)$

A_1	A_2
{b,c}	f
a	f

As $r(R)$ and $r'(R)$ are inconsistent, forcibly integrating them by Definition 1 using the equivalence classes $\mathcal{D}_1$ and $\mathcal{D}_2$ of case 1 to determine redundant tuples yields the following result.

	$r(R) \widetilde{\cup} r'(R)$	
	colors	temperatures
t_1 :	{a,b}	{d,e}
t_2 :	a	f
t_3 :	c	f
t_4 :	{b,c}	f

□

The merging result in Example 2 violates Theorem 1 owing to $T_3 \subset T_4$, where T_i denotes the set of all possible interpretations of t_i. That is because Definition of redundant tuple implies that all elements in a component of a tuple must belong to the same equivalence class. Otherwise, even identical tuples may not be considered as redundant. For example, tuples t_4 and t_4 are not redundant based on $\mathcal{D}_1$ and $\mathcal{D}_2$ of case 1. On the other hand, the same problems occur as using the equivalence classes of case 2 to determine redundant tuples during database integration. To avoid the heterogeneity of equivalence classes between database, the *consistency* on similarity relations is introduced as constraint for the fuzzy database relations to be integrated [13]. Similarity relations that satisfy the consistency constraint, shown below, can induce the identical sets of equivalence classes via appropriate thresholds.

Definition 2. Let $p_1, p_2, \ldots, p_k$ be k Similarity relations on domain D. These k relations $p_1, p_2, \ldots, p_k$ are *consistent* if the following conditions are satisfied,

$$p_i(x, y) > p_i(z, u) \Leftrightarrow p_j(x, y) > p_j(z, u)$$

for every $1 \leq i, j \leq k$, where $x, y, z, u \in D$. □

However, in multi-databases or data warehouses, each component database was created independently and thus, the similarity relations on the same domain were likely defined by different experts with slightly different opinions regarding this domain. It is worth providing an integral view of these database relations in the multi-databases or data warehouses. The following section discusses how to identify an appropriate set of equivalence classes for the union operation to make the result meaningful.

4. Selection of Equivalence Classes

According to the implication of Definition of redundant tuple, each equivalence class used for the union result must be a super set of some of the equivalence classes used for each operand. An extreme example is as follows: for each attribute of the union result, using an entire domain as an equivalence class, the result will not violate Theorem 1. However, the result contains only one tuple after tuple merging, and consequently much information is lost. This study introduces the notion of the *general set* to preserve more information in the union result after tuple merging.

Definition 3. Let $E_1, E_2, \ldots, E_k$ and E' be k+1 sets of equivalence classes on the same domain. E' is a *general set* of $E_1, E_2, \ldots, E_k$, denoted by $E' \succeq \{E_1, E_2, \ldots, E_k\}$, if for every $e \in E_i$, there exists $e' \in E'$ such that $e \subseteq e'$, for $i = 1, 2, \ldots, k$. □

To integrate two inconsistent fuzzy database relations, we first choose a general set as a set of equivalence classes (discussed further below) for each attribute, and then proceed tuple merging based on the general set. Using the general set, the database relation may contain redundant tuple regarding the new data redundancy. After merging redundant tuples for each of involved database relations based on the general sets chosen, these database relations can be integrated by Definition 1.

But, the question is which general set is better than the others. Notably, more than one general set could exist for given sets of equivalence classes on the same domain. A general set containing fewer elements than other general sets on the same domain has vaguer semantic meaning. Consequently, selecting a general set containing more elements can preserve more semantic meaning of the original sets of equivalence classes, and thus can conserve more information of the original fuzzy database relations.

Example 3. Consider two sets of equivalence classes $E_1 = \{\{a, b\}, \{c\}, \{d, e\}\}$ and $E_2 = \{\{a\}, \{b, c\}, \{d\}, \{e\}\}$. Both $\{\{a, b, c\}, \{d, e\}\}$ and $\{\{a, b, c, d, e\}\}$ are a general set of E_1 and E_2, and the later is semantically vaguer. □

The general set that has the least vagueness semantic representation is formally defined below. For brevity, let $\mathcal{E}$ denote the set $\{E_1, E_2, \ldots, E_k\}$ for any given sets $E_1, E_2, \ldots, E_k$ of equivalence classes on the same domain.

Definition 4. A set E is the *minimal general set* of $\mathcal{E}$ if $E \succeq \mathcal{E}$ and no set $E' \succeq \mathcal{E}$ exists such that $E' \neq E$ and $E \succeq \{E'\}$. □

Notably, the minimal general set (MGS for short) of $\mathcal{E}$ is unique. For instance, only $\{\{a, b, c\}, \{d, e\}\}$ is the MGS of E_1 and E_2 in Example 3. Obviously, when E is the MGS of $\mathcal{E}$, $|E| > |E'|$ for every other general set E' of $\mathcal{E}$. Thus, using the MGS in the union of fuzzy database relations can minimize the information loss in the union result. The MGS of $E_1, E_2, \ldots, E_k$ on domain D of size N can be determined in $O(kN)$ using **Algorithm Find_MGS**. In the Algorithm, the underlined statement can be performed within a constant time for each iteration by storing each E_i in the *adjacent list* [14], in which each element in the domain represents a node, and the elements in the same equivalence classes are regarded as the adja-

cent nodes. This statement is executed only $k \times N$ times in total.

Algorithm Find_MGS
Input: The sets of equivalence classes
$E_1, E_2, \ldots, E_k$ on domain D;
Output: The minimal general set E of
$E_1, E_2, \ldots, E_k$.
begin
$E := \emptyset$; $V := \emptyset$; $V' := \emptyset$;
// V collects all elements in an E_i
while $D \neq \emptyset$ **do** {
choose an element d from D;
$V := \{d\}$; $V' = \emptyset$;
// $V' = \{e \in V : e$ is processed$\}$
while $V \setminus V' \neq \emptyset$ **do** {
choose an element d' from $V \setminus V'$;
// executed N times in total
$V' = V' \cup \{d'\}$;
for each $E_i \in \{E_1, E_2, \ldots, E_k\}$ **do**
$V := V \cup \hat{e}$ where $\hat{e} \in E_i$ and $d' \in \hat{e}$
} // end while $V \setminus V' \neq \emptyset$
$E := E \cup \{V\}$; $D := D \setminus V$;
}
end

Recall that a set of equivalence classes is determined by a similarity relation and its associated threshold, and thus, lowering the threshold can derive a new set of equivalence classes, which is vaguer than the original set of equivalence classes but still complies semantically with of the original similarity relations. One shortcoming of MGS is that it may not be obtained by reducing the thresholds on the original similarity relations, illustrated in Example 4. Consequently, MGS may conflict semantically the original similarity relations. The *minimal adapted general set* (MAGS for short) is introduced to avoid this situation.

Example 4. Consider E_1 and E_2 in Example 3. Suppose that E_2 is obtained by similarity relation s and threshold β, and $s(b,d) = s(c,d) > s(a,d) > s(d,e)$. Then, the MGS $\{\{a,b,c\},\{d,e\}\}$ of E_1 and E_2 can not be obtained by S_α (that is, α-similarity) for any $\alpha \in [0,1]$. Restated, the MGS conflicts with the similarity relation s on semantic meaning. □

Definition 5. Let $s_1, s_2, \ldots, s_k$ denote k similarity relations on the same domain. A set E of equivalence classes is an *adapted set* on $s_1, s_2, \ldots, s_k$ if there exist $\alpha_1, \alpha_2, \ldots, \alpha_k$ such that E can be obtained by S_{α_i} on s_i for each $i = 1, 2, \ldots, k$. □

Definition 6. A set E' is an *adapted general set* of $\mathcal{E}$ if E' is an adapted set on similarity relations $s_1, s_2, \ldots, s_k$ and $E' \succeq \mathcal{E}$, where $\mathcal{E} = \{E_1, E_2, \ldots, E_k\}$, and each E_i is a set of equivalence classes induced by S_{α_i} on s_i, $\alpha_i \in [0,1]$. An adapted general set E of $\mathcal{E}$ is *minimal* if no adapted general set E' of $\mathcal{E}$ exists in which $E' \neq E$ and $E \succeq \{E'\}$. □

Though MAGS has vaguer semantic meaning and takes more time to determine than MGS, it presents a semantic meaning that is more concordant to the semantic meaning of the similarity relations used in the original database relations. MAGS thus is a better choice than MGS if MAGS does not cause unacceptable information loss, otherwise MGS is necessary. For example, the MAGS of E_1 and E_2 in Example 4 is no other than the set of the entire domain. Using such equivalence classes yields one-tuple result, and much information is consequently lost following integration. Thus, MGS is a better choice in this case. Also, note that no information is lost when integrating consistent database relations.

5. Conclusions

In the proximity-based data model, data redundancy is also based on equivalence classes, although, without max-min transitivity, a proximity cannot induce equivalence classes by applying the α-similarity operation. Shenoi and Melton [10] developed the α-proximity operation to induce equivalence classes on proximity relations. Thus, the solution provided in this paper can be applied to the proximity-based fuzzy databases.

To date, the discussion has been limited to using union operations for integration. To integrate two database relations which are not union-compatible, *outer join* or *outer union* operation is frequently used [15], [16], which extends each operand to include attributes that are peculiar to the other, deposits *null* values in every tuple for all such adding attributes, and then performs the normal union operation.

References

[1] B. Scotney, S. McClean, Efficient knowledge discovery through the integration of heterogeneous data, Information and Software Technology 41 (1999) 569–578.

[2] W.-S. Li, C. Clifton, Semint: A tool for indentifying attribute corrospondences in heterogeneous datadase using neural networks, Data and Knowledge Engineering 22 (2000) 49–84.

[3] M. Lenzerini, Data integration is harder than you thought, in: Proceedings of the 9th International Conference on Cooperative Information Systems, 2001, pp. 22–26.

[4] A. Cali, D. Calvanese, G. D. Giacomo, M. Lenzerini, Data integration under integrity constrains, Information Systems 29 (2004) 147–163.

[5] K. V. S. V. N. Raju, A. Majumdar, Fuzzy

function dependencies and lossless join decomposition of fuzzy relational database system, ACM Transactions on Database Systems 13 (2) (1988) 129–166.

[6] G. Q. Chen, J. Vandenbulcke, E. E. Kerre, On the lossless-join decomposition of relation scheme(s) in a fuzzy relational data model, in: Second International Symposium on Uncertainty Modeling and Analysis, Proceedings ISUMA'93, 1993, pp. 440–446.

[7] S. Joythi, M. S. Babu, Multivalued dependencies in fuzzy relational databases and lossless join decomposition, Fuzzy sets and Systems 88 (1997) 315–322.

[8] J. C. Cubero, J. M. Median, O. Pons, M. Vila, Fuzzy lossless decompositions in databases, Fuzzy sets and Systems 97 (1998) 145–167.

[9] B. P. Buckles, F. E. Petry, A fuzzy representation of data for relational databases, Fuzzy Sets and Systems 7 (3) (1982) 231–226.

[10] S. Shenoi, A. Melton, Proximity relations in the fuzzy relational database model, Fuzzy Sets and Systems 31 (3) (1989) 285–296.

[11] S. Shenoi, A. Melton, An extended version of the fuzzy relational database model, Information Sciences 51 (1990) 35–52.

[12] E. Codd, Relational Completeness of Database Sublanguages, in R. Rustin (ed.), Database Systems, Prentice-Hall, 1972.

[13] J. Y.-C. Liu, J.-L. Lin, Combination of similarity relations under consistency constraint, in: International Conference on Computer Science and its Applications (ICCSA-03), 2003.

[14] E. Horowitz, S. Sahni, S. Anderson-freed, Fundamentals of Data Structure in C, Computer Science Press, An imprint of W. H. Freeman and Company, New York, Sixth printing, 1997.

[15] E. Codd, Extending the database relational model to capture more meaning, ACM TODS 4 (4).

[16] C. J. Date, The outer join, in: Proceedings of the Second International Conference on Databases ICOD2, 1983.