Querying Uncertain Data in Heterogeneous Databases*

Pauray S.M. Tsai and Arbee L.P. Chen

Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 300, R.O.C.
Email: alpchen@cs.nthu.edu.tw

Abstract

In heterogeneous databases, the user may issue a query to join two relations in different databases on the keys. However, the keys may be incompatible. In this paper, we extend our results on probabilistic query processing to consider joining two relations on incompatible keys. A new approach to identify the "same" entities in different relations is proposed. Various data and schema conflicts such as missing data, inconsistent data and domain mismatch are considered in the identification process. Probabilistic techniques are used to estimate the sameness of two entities, to process queries, and to estimate the degree of uncertainty for the query results.

1 Introduction

Because of the rapid advance in networking technologies and the requirement of data sharing among multiple databases, heterogeneous distributed databases have become the trend of future database development. One of the important characteristics of the heterogeneous database is that the autonomy of its component databases is preserved; that is, in a component database the data can be created and manipulated independent of other databases.

There are two approaches to derive data in a heterogeneous database environment. One is to provide a global schema for the component databases by integrating their schemas. Dayal and Hwang[8] and Motro[13] adopted this approach based on functional model, while Breitbart et al.[2] and Deen et al.[9] were based on relational model. For a comprehensive survey on methodologies developed for schema integration, refer to [1] .

*Appeared in Proc. IEEE International Workshop on Research Issues on Data Engineering: Interoperability in Multidatabase Systems (1993). This work was partially supported by the Republic of China National Science Council under Contract No. NSC 82-0408-E-007-162.

The other approach is to provide users a multidatabase query language[7, 11]; namely, users can pose their queries against the local schemas by using a multidatabase manipulation language. In this approach, it is required that the users have sufficient information of the local schemas. In this paper, we adopt the latter approach to study query processing issues for the heterogeneous database.

DeMichiel[10] proposed an idea of resolving the interoperability problems in heterogeneous database systems by partial values. A partial value corresponds to a set of possible values in which exactly one is the true value. Tseng, Chen and Yang[15] extended the concept of partial values to probabilistic partial values, and developed a full set of extended relational operators to manipulate probabilistic partial values, which come from the resolution of various conflicts in the heterogeneous database. As a result, more informative query results can be provided.

Pu[14] considered the semantic heterogeneity in heterogeneous databases and formulated the *key equivalence* problem. Specifically, name matching was considered to decide whether two names from different databases refer to the same person. Another related work proposed by Wang and Madnick[16] used a rule-based approach to determine whether, for example, a name in one database and a nickname in another database refer to the same person.

chatterjee and Segev[4] categorized attribute incompatibility into two types: structural incompatibility and semantic incompatibility. The problem of joining two relations on structually or semantically incompatible keys was studied. The idea of comparing *useful attributes* was used to determine whether two tuples from different relations represent the same entity. Also, a probabilistic model was presented to estimate the accuracy of the comparison. Issues on missing data or domain mismatch which may exist between the two relations to be joined, however, are not considered.

Morrissey[12] proposed a method to deal with imprecise values, missing data and inapplicable attributes and suggested an information-theoretic ap-

proach to estimate the uncertainty. Join operations were not considered in the query, so the problem of identifying the same entities was not discussed.

In this paper, we extend our results on probabilistic query processing to consider joining two relations on incompatible keys. An approach to identify the same entities in different relations is proposed. Various data and schema conflicts such as missing data, inconsistent data and domain mismatch[3] are considered in the identification process. Probabilistic techniques are used to estimate the sameness of two entities, to process queries, and to estimate the degree of uncertainty for the query results.

This paper is organized as follows. We give an example to illustrate the probabilistic approach to processing queries with compatible keys in Section 2. Section 3 describes a method to identify the same entities when the keys are incompatible. An example is given to illustrate the whole query processing approach in Section 4. Finally, we conclude this work in Section 5.

2 Compatible Keys

In this section, we give an example to illustrate the probabilistic approach[15] on processing queries with compatible keys.

Consider the relations in Figure 1, where relation **TC** in one database records the data for students who belong to the tea club, and relation **TW** in another database records the data for students who take Technical Writing. The key attributes *id* in both relations are assumed compatible. Attribute A_1 of relation R_1 is compatible with attribute A_2 of relation R_2 if the domains of attributes A_1 and A_2 are semantically related.

A query "Find all twenty-year-old sophomores who belong to the tea club and take Technical Writing" can be expressed as

$$\sigma_{(age=20)and(class=sophomore)}(\mathbf{TC} \bowtie_{id} \mathbf{TW}),$$

where σ and $\bowtie$ denote selection and join, respectively. Figure 2 shows the result of joining **TC** and **TW** on *id*. Notice that we integrate tuples from both relations in the join operation, and assume equal probability for the conflict values.

Then, the selection is processed, and the query result obtained as shown in Figure 3. The column *poss* in Figure 3 denotes the possibilities of the answer tuples. For example, the *poss* value in the "Tony" tuple is computed as $1/2 \times 1/2 = 0.25$. The answer tuples can be ranked according to the *poss* values, which will provide users a more informative query result.

3 Incompatible Keys

We consider query processing involving joining two relations on imcompatible keys in this section.

Consider the relations in Figure 4, where relation **Teacher** in one database records the personal data of teachers in X University, and relation **Consultant** in another database records the data of consultants in the computer science division of Y Company. The key attribute *id* in **Teacher** represents the identification number for teachers in X University while that in **Consultant** represents the identification number for consultants in Y Company. It would be meaningless to compare the *id* value in **Teacher** with the *id* value in **Consultant**. In other words, the keys in **Teacher** and **Consultant** are incompatible.

Now, suppose we want to find out which teachers in X University consult at (the computer science division of) Y Company. If the keys are compatible, we can join the two relations on the keys to identify the teachers in X University who are also consultants in Y Company. When the keys are incompatible as the example shows, in order to process the query, we consider joining these two relations on the set of compatible attributes to identify the entities who are teachers in X University as well as consultants in Y Company. Since these attributes are nonkey attributes but not identifying attributes, they only represent properties of the entities in each relation. Therefore, there exists uncertainty on the "sameness" of two entities, each from a relation. For example, an entity in **Teacher** may not be an entity in **Consultant** even when all the values of their nonkey compatible attributes match. We consider various data and schema conflicts which may exist in the set of compatible attributes, to estimate the uncertainty of the joining result, and concentrate on processing queries of the form $\sigma(R_1 \bowtie R_2)$, where the join attributes are the key attributes which are incompatible.

Let R_1 and R_2 be the two relations to be joined, and the set of compatible attributes for them be $\{a_1, a_2, ..., a_k\}$. We assign a weight w_i to each compatible attribute a_i according to its *importance* such that

$$0 \leq w_i \leq 1 \quad and \quad \sum_{i=1}^{k} w_i = 1.$$

The importance of a compatible attribute depends on its semantic meaning. For example, since two tuples having the same *name* value have a better chance to represent the same entity than having the same *degree* value, attribute *name* is considered more important than attribute *degree*, and will be assigned a higher weight.

Let t_1 be a tuple of R_1 and t_2 a tuple of R_2. Denote the sameness for the values of attribute a_i in t_1 and t_2 as S_i, where $0 \leq S_i \leq 1$. Then the

TC

id	*name*	*age*	*class*	*phone*
2	John	21	junior	4335
30	Mary	20	sophomore	4863
41	Jane	22	senior	3776
46	Paul	18	freshman	2375
53	John	22	senior	6447
55	Tony	19	freshman	6450

TW

id	*name*	*age*	*dept*	*class*
30	Mary	20	CS	junior
43	Joe	23	CE	senior
46	Paul	18	CS	freshman
55	Tony	20	CS	sophomore
57	Joe	19	CS	freshman

Figure 1: Relations **TC** and **TW** in two different databases.

id	*name*	*age*	*dept*	*class*	*phone*
30	Mary	20	CS	$[\text{sophomore}^{1/2}, \text{junior}^{1/2}]$	4863
46	Paul	18	CS	freshman	2375
55	Tony	$[19^{1/2}, 20^{1/2}]$	CS	$[\text{freshman}^{1/2}, \text{sophomore}^{1/2}]$	6450

Figure 2: The result of $\mathbf{TC} \bowtie_{id} \mathbf{TW}$.

possibility for t_1 and t_2 to represent the same entity is expressed as

$$P_{same}(t_1, t_2) = \sum_{i=1}^{k} (S_i \times w_i).$$

If $P_{same}(t_1, t_2) > 0$, tuples t_1 and t_2 will be integrated into a new tuple in the "join" relation by using the technique of probabilistic partial values[15], and the value $P_{same}(t_1, t_2)$ attached to the integrated tuple as the degree of their sameness. When there are select operations in the query, they will be processed according to the approach in [15]. A possibility *poss* of a tuple that satisfies the select predicates will be computed. Since the events of identifying the sameness of two tuples and performing the select predicates are independent, the possibility of an answer tuple is computed by

$$poss* = P_{same} \times poss.$$

There are two rules which can be used to determine whether $P_{same}(t_1, t_2)$ is equal to one or zero.

Rule 1: If all the values in the compatible attributes are non-null and identical, then $P_{same}(t_1, t_2)$ is equal to one.

In some cases we do not allow the values of a compatible attribute, say d, to be inconsistent for the consideration of P_{same}, i.e., $P_{same}(t_1, t_2)$ will be set to zero when $t_1.d \neq t_2.d$. We call such an attribute a *dominant attribute*. Conversely, a compatible attribute d' is called a *nondominant attribute* if $P_{same}(t_1, t_2)$ is allowed to be greater than zero when $t_1.d' \neq t_2.d'$. For example, the attribute *name* may be designated as a dominant attribute while *degree* a nondominant attribute. Usually, dominant attributes will have higher weights.

Rule 2: If one of the dominant attributes has different values in t_1 and t_2, then $P_{same}(t_1, t_2)$ is equal to zero.

Now we consider other cases where $0 \leq P_{same}(t_1, t_2) < 1$, and pay attention to the subject of estimating the sameness between two values.

3.1 Missing Data and Inconsistent Data

Consider a compatible attribute a_i of relations R_1 and R_2. Let the set of elements in the domain of a_i be $\{v_1, ..., v_n\}$. We propose an approach to estimate the sameness of two a_i values, one from R_1 and the other from R_2, and an approach to integrate these values for the join processing. The null values we consider are applicable null values as defined in Codd[6], which are denoted as '$\sim$'. When a null value is expressed as a probabilistic partial value, we assume probabilities are uniformly distributed over all possible values. Let t_1 and t_2 be tuples of R_1 and R_2, respectively, and consider the following cases.

1. $t_1.a_i = t_2.a_i$
   If tuples t_1 and t_2 have an identical value, say v_j, on attribute a_i, then the sameness is one and the integrated data is v_j.
2. missing data
   (a) a_i is a nondominant attribute
   - **case 1**: the values of a_i in t_1 and t_2 are v_j and $\sim$, respectively. Since the size of the domain for a_i is n, and a null value is expressed as a probabilistic partial value with probabil-

id	*name*	*age*	*dept*	*class*	*phone*	*poss*
30	Mary	20	CS	[sophomore$^{1/2}$, junior$^{1/2}$]	4863	0.5
55	Tony	[$19^{1/2}$, $20^{1/2}$]	CS	[freshman$^{1/2}$, sophomore$^{1/2}$]	6450	0.25

Figure 3: The result of $\sigma_{(age=20)and(class=sophomore)}(\mathbf{TC} \bowtie_{id} \mathbf{TW})$.

Teacher

id	*name*	*department*	*degree*	*age*
7001	Mary	CS	$\sim$	30
7002	Paul	EE	PhD	29
7003	John	CS	PhD	29
7004	John	CS	MS	26
7005	Lina	EE	PhD	34
7006	Paul	CS	PhD	30

Consultant

id	*name*	*specialty*	*degree*	*age*	*city*
101	John	CN	MS	25	B
102	Jose	DB	MS	36	$\sim$
103	James	DB	BS	24	A
104	Mary	CN	MS	29	B
105	John	AI	MS	26	B
106	Dick	DB	BS	24	A
107	Tony	IP	MS	28	$\sim$
108	Paul	$\sim$	MS	25	B
109	Mary	AI	PhD	30	B
110	Paul	DB	PhD	30	$\sim$

Figure 4: Relations **Teacher** and **Consultant** in two different databases.

ities uniformly distributed over all possible values, the sameness between v_j and $\sim$ is computed by

$$S_i = \frac{1}{n},$$

and the integrated value = $[v_1^{\frac{1}{2n}}, v_2^{\frac{1}{2n}}, ..., v_j^{\frac{n+1}{2n}}, ..., v_n^{\frac{1}{2n}}]$.

- **case 2**: $t_1.a_i$ and $t_2.a_i$ are both null values. Consider any value v_j of a_i, the probability for t_1 and t_2 to have the identical value v_j is $1/n^2$. Since there are n possible values with equal probability in a_i, the sameness is computed by

$$S_i = \frac{1}{n},$$

and the integrated value = $[v_1^{\frac{1}{n}}, v_2^{\frac{1}{n}}, ..., v_j^{\frac{1}{n}}, ..., v_n^{\frac{1}{n}}]$.

(b) a_i is a dominant attribute

- **case 1**: the values of a_i in t_1 and t_2 are v_j and $\sim$, respectively. The sameness is computed as before by

$$S_i = \frac{1}{n}.$$

Since t_1 and t_2 are possible to represent the same entity, the only possible value for the dominant attribute has to be the known value v_j. Therefore, the integrated value is v_j.

- **case 2**: $t_1.a_i$ and $t_2.a_i$ are both null values. The sameness and the integrated value can be computed as in the case when a_i is nondominant.

3. inconsistent data

By Rule 2, we know if the values in the dominant attributes are different, $P_{same}(t_1, t_2)$ is equal to zero. Hence, we only consider nondominant attributes here.

Consider a nondominant attribute a_i. Let the values of a_i in t_1 and t_2 be v_1 and v_2, respectively. Let the *difference distance* between v_1 and v_2 be v' and the *maximum difference distance* which can be tolerated be D_i for attribute a_i. The sameness can be computed by

$$S_i = \begin{cases} 1 - \frac{v'}{D_i} & \text{if } 1 - \frac{v'}{D_i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Rule 3: When the difference distance between values of attribute a_i in t_1 and t_2, respectively, is greater than maximum difference distance, we assume the two entities represented by t_1 and t_2 are different, and $P_{same}(t_1, t_2)$ is equal to zero.

In the case where the difference distance between values of a_i in t_1 and t_2, respectively, is

less than or equal to maximum difference distance, the integrated value is $[v_1^{\frac{1}{2}}, v_2^{\frac{1}{2}}]$ by assuming v_1 and v_2 have equal probability to be the true value.

3.2 Domain Mismatch

Naming conflicts, data scaling conflicts and data representation conflicts are considered as the *domain mismatch* problems [3]. The notion of *canonical virtual attributes* [10] can be used to resolve both naming conflicts and data representation conflicts. For a data scaling conflict where a bijective transformation rule can be defined to resolve the conflict, the sameness and the integrated value can be computed as in the cases 1 and 3 in Section 3.1. However, for example, although there exists a data scaling conflict between **Teacher**.*department* and **Consultant**.*specialty* in Figure 4, we cannot find a bijective transformation rule to resolve the conflict. We discuss the computation of the sameness and the integrated value for this case as follows.

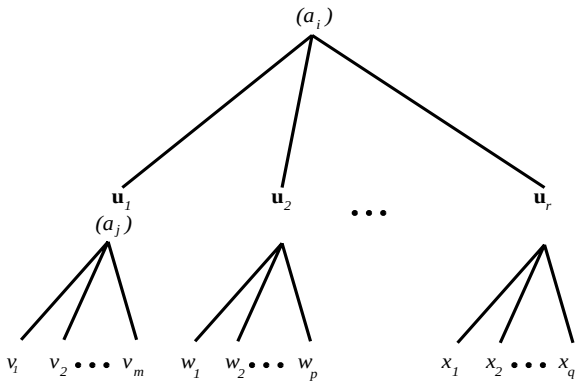

Figure 5: The domain hierarchy.

Let a_i and a_j be the attributes in relations R_1 and R_2, respectively. Suppose a_i and a_j are semantically related in the following manner. The domains of a_i and a_j are $\{u_1, u_2, ..., u_r\}$ and $\{v_1, v_2, ..., v_m\}$, respectively. Moreover, the domain of u_1 is $\{v_1, v_2, ..., v_m\}$, the domain of u_2 is $\{w_1, w_2, ..., w_p\}$, ..., and the domain of u_r is $\{x_1, x_2, ..., x_q\}$. Sets $\{v_1, v_2, ..., v_m\}$, $\{w_1, w_2, ..., w_p\}$, ... , and $\{x_1, x_2, ..., x_q\}$ are assumed to be mutually exclusive. We can express the relationship between these domains as a *domain hierarchy* as depicted in Figure 5. Notice that the domains of a_j and u_1 are the same.

Define a *select attribute* as an attribute involved in a select operation. In the following, the select attributes in the query are used to estimate the sameness and deal with data integration. t_1 and t_2 are assumed tuples of R_1 and R_2, respectively. Denote S_n as the sameness for the values of $t_1.a_i$ and $t_2.a_j$.

1. a_j is a select attribute

   - **case 1**: the values of a_i in t_1 and a_j in t_2 are u_1 and v_1, respectively. Since u_1 is assumed to be one of the elements in $\{v_1, v_2, ..., v_m\}$, the sameness between u_1 and v_1 is computed by
   $$S_n = \frac{1}{m},$$
   and the integrated value is $[v_1^{\frac{m+1}{2m}}, v_2^{\frac{1}{2m}}, ..., v_m^{\frac{1}{2m}}]$.
   - **case 2**: the values of a_i in t_1 and a_j in t_2 are u_1 and $\sim$, respectively. Since both u_1 and$\sim$ can be expressed as a probabilistic partial value with probabilities uniformly distributed over $\{v_1, v_2, ..., v_m\}$, the sameness is computed by
   $$S_n = \frac{1}{m},$$
   and the integrated value is $[v_1^{\frac{1}{m}}, v_2^{\frac{1}{m}}, ..., v_m^{\frac{1}{m}}]$.
   - **case 3**: the values of a_i in t_1 and a_j in t_2 are $\sim$ and v_1, respectively. Since the probability for t_1 to have u_1 as the value of a_i is 1/r, and the sameness between u_1 and v_1 is 1/m as obtained in case 1, the sameness is computed by
   $$S_n = \frac{1}{r} \times \frac{1}{m}.$$
   Since only the value u_1 of a_i in t_1 is concerned when a_j is a select attribute, the integrated value is $[v_1^{\frac{m+1}{2m}}, v_2^{\frac{1}{2m}}, ..., v_m^{\frac{1}{2m}}]$.
   - **case 4**: $t_1.a_i$ and $t_2.a_j$ are both null values. Since the probability for t_1 to have u_1 as the value of a_i is 1/r, and the sameness between u_1 and $\sim$ is 1/m as in case 2, the sameness is computed by
   $$S_n = \frac{1}{r} \times \frac{1}{m},$$
   and the integrated value is $[v_1^{\frac{1}{m}}, v_2^{\frac{1}{m}}, ..., v_m^{\frac{1}{m}}]$.
   - **case 5**: the values of a_i in t_1 and a_j in t_2 are u_i and v_1, respectively, where $i \neq 1$. Assume i equals 2 for the discussion. Since u_2 is assumed to be one of

the elements in $\{w_1, w_2, ...w_p\}$, and sets $\{v_1, v_2, ..., v_m\}$ and $\{w_1, w_2, ..., w_p\}$ are assumed to be disjoint, t_1 and t_2 are considered different entities.

- **case 6**: the values of a_i in t_1 and a_j in t_2 are u_i and $\sim$, respectively, where $i \neq 1$. Assume i equals 2 for the discussion. t_1 and t_2 are considered different entities. The reason is the same as that for case 5.

2. a_i is a select attribute

   - **case 1**: the values of a_i in t_1 and a_j in t_2 are u_1 and v_1, respectively. The sameness is computed by

     $$S_n = 1,$$

     and the integrated value is u_1 because v_1 is in the domain of u_1.

   - **case 2**: the values of a_i in t_1 and a_j in t_2 are u_1 and $\sim$, respectively. The sameness is computed by

     $$S_n = 1,$$

     and the integrated value is u_1 because any possible value of a_j is in the domain of u_1.

   - **case 3**: the values of a_i in t_1 and a_j in t_2 are $\sim$ and v_1, respectively. Since the probability for t_1 to have u_1 as the value of a_i is 1/r, and the sameness between u_1 and v_1 is 1 when a_i is a select attribute, the sameness is computed by

     $$S_n = \frac{1}{r},$$

     and the integrated value is u_1.

   - **case 4**: $t_1.a_i$ and $t_2.a_j$ are both null values. Since the probability for t_1 to have u_1 as the value of a_i is 1/r, and the sameness between u_1 and $\sim$ is 1 as in case 2, the sameness is computed by

     $$S_n = \frac{1}{r},$$

     and the integrated value is u_1.

   - **case 5**: the values of a_i in t_1 and a_j in t_2 are u_i and v_1, respectively, where $i \neq 1$. Assume i equals 2 for the discussion. Since disjoint sets $\{v_1, v_2, ..., v_m\}$ and $\{w_1, w_2, ..., w_p\}$ consist of all the possible elements for u_1 and u_2, respectively, t_1 and t_2 are considered different entities.

   - **case 6**: the values of a_i in t_1 and a_j in t_2 are u_i and $\sim$, respectively, where $i \neq 1$. Assume i equals 2 for the discussion. t_1 and t_2 are considered different entities. The reason is the same as that for case 5.

   If attributes a_i and a_j are not select attributes, we use the approach where a_j is a select attribute (i.e., the attribute in the lower level of the domain hierarchy) such that more detailed information can be considered.

4 Example

In this section, we give an example to illustrate the whole query processing approach. Consider the relations in Figure 4 and the query

$$\sigma_{(specialty=AI\ or\ DB)\ and\ (age \leq 30)\ and\ (city=B)}$$
$$(\textbf{Teacher} \bowtie_{id} \textbf{Consultant}).$$

Before join processing, we first resolve the naming conflict between the compatible attributes *department* and *specialty* in relations **Teacher** and **Consultant**, respectively. We designate *specialty* as the canonical virtual attribute, and rename the attribute *department* to *specialty*. Some necessary information for the query processing is listed as follows:

(1) the set of compatible attributes = {name, specialty, degree, age}
(2) the set of dominant attributes = {name}
(3) the set of nondominant attributes = {specialty, degree, age}
(4) weights
The weight "w(a_i)" for attribute a_i is defined as

- w(name) = $\frac{1}{2}$
- w(specialty) = $\frac{1}{6}$
- w(degree) = $\frac{1}{6}$
- w(age) = $\frac{1}{6}$

(5) difference distance

- The domain of attribute *degree* is {BS, MS, PhD}. The difference distance "D(u, v)" between any two values u and v in attribute *degree* is defined as
  D(BS, MS) = 2
  D(MS, PhD) = 4
  D(BS, PhD) = 6.
- For any two values v_1 and v_2 in attribute *age*, the difference distance "D(v_1,v_2)" between v_1 and v_2 is defined as
  $D(v_1, v_2) = \mid v_1 - v_2 \mid$.

(6) maximum difference distance (MDD)

- for attribute *degree*, MDD= 6
- for attribute *age*, MDD = 5

(7) the domain hierarchy

- The domain of *department* consists of Computer Science(CS) and Electronic Engineering(EE).
- The domain of *specialty* (which is also the domain of CS) consists of Computer Network (CN), Database(DB), Image Processing(IP), and Artificial Intelligence(AI). (Recall that **Consultant** stores the data of the consultants in the computer science division of Y Company.)
- The domain of Electronic Engineering (EE) consists of Communication Electronics(CE) and IC Design (IC).

The domain hierarchy is depicted in Figure 6.

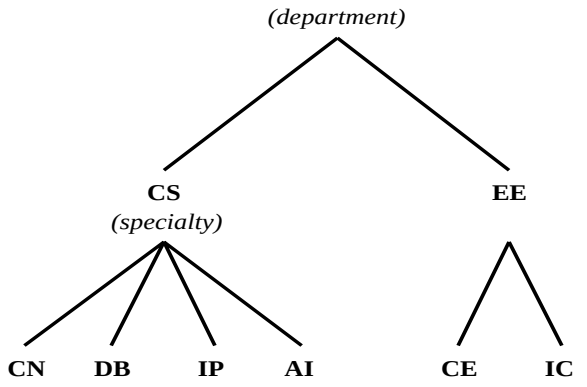

Figure 6: The domain hierarchy for attributes *department* and *specialty*.

Now, consider the two tuples (7003, John, CS, PhD, 29) and (101, John, CN, MS, 25, B) in relations **Teacher** and **Consultant**, respectively. The sameness $S(a_i)$ for the two values under attribute a_i (a_i is a compatible attribute) is computed by

S(name) = 1
S(specialty) = $\frac{1}{4}$
S(degree) = 1-$\frac{4}{6}$ = $\frac{1}{3}$
S(age) = 1-$\frac{4}{5}$ = $\frac{1}{5}$

The associated integrated value $V(a_i)$ is expressed as

V(name) = John
V(specialty) = [CN$^{\frac{5}{8}}$, DB$^{\frac{1}{8}}$, IP$^{\frac{1}{8}}$, AI$^{\frac{1}{8}}$]
V(degree) = [MS$^{\frac{1}{2}}$, PhD$^{\frac{1}{2}}$]
V(age) = [25$^{\frac{1}{2}}$, 29$^{\frac{1}{2}}$].

We compute the sameness of these two tuples by
sameness = Σ S(a_i)$\times$ w(a_i) = $1 \times \frac{1}{2} + \frac{1}{4} \times \frac{1}{6} + \frac{1}{3} \times \frac{1}{6} + \frac{1}{5} \times \frac{1}{6} = 0.63$.
Then the select operation is processed, and the possibility *poss* of an integrated tuple that satisfies the select predicate is computed by using the extended select operator[15]. For the above example, the value of *poss* for the integrated tuple
(7003, 101, John, [CN$^{\frac{5}{8}}$, DB$^{\frac{1}{8}}$, IP$^{\frac{1}{8}}$, AI$^{\frac{1}{8}}$], [MS$^{\frac{1}{2}}$, PhD$^{\frac{1}{2}}$], [25$^{\frac{1}{2}}$, 29$^{\frac{1}{2}}$], B)
is computed by

$$poss = (\frac{1}{8} + \frac{1}{8}) \times 1 \times 1 = 0.25.$$

Finally, the possibility of the integrated tuple to be a query result is computed by
*poss** = sameness $\times$ *poss* = $0.63 \times 0.25 \approx 0.16$.
The domain of attribute *city* is assumed to be {A,B}. The result of the whole query processing is depicted in Figure 7. Notice that *id_1* and *id_2* represent the attribute *id* in **Teacher** and **Consultant**, respectively. It is interesting to see that the relationship between these two incompatible keys may be derived from the resultant relation. For example, teacher "7003" can be consultant "101" or "105" with the sameness being 0.63 and 0.66, respectively. Moreover, consultant "101" can also be teacher "7004" with the sameness being 0.84.

5 Conclusions and Future Work

In this paper, we propose an approach to process queries involving joining two relations on their incompatible keys. Interoperability problems – missing data, inconsistent data, and domain mismatch are considered in the process of query processing. Probabilistic techniques are used to estimate the sameness between two tuples, integrate data, and handle data manipulation.

In a heterogeneous database which consists of object databases, the same real-world entity can be stored as objects in different databases with incompatible object identifiers. We have applied similar techniques to identify and integrate these objects for query processing in [5].

Currently, we are working on extending this approach to handle queries involving more than two relations. Besides, how to derive the relationship between the incompatible keys based on this approach is under investigation. Finally, we think that a query with a threshold possibility is more flexible than the conventional queries. The users attach a possibility "α" to the query, and the system only presents those tuples with possibilities greater than or equal to α in the resultant relation. By this method, query

id_1	id_2	name	specialty	degree	age	city	sameness	poss	poss*
7003	101	John	$[CN\frac{5}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{1}{8}]$	$[MS\frac{1}{2}, PhD\frac{1}{2}]$	$[25\frac{1}{2}, 29\frac{1}{2}]$	B	0.63	0.25	0.16
7004	101	John	$[CN\frac{5}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{1}{8}]$	MS	$[25\frac{1}{2}, 26\frac{1}{2}]$	B	0.84	0.25	0.21
7001	104	Mary	$[CN\frac{5}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{1}{8}]$	$[BS\frac{1}{6}, MS\frac{2}{3}, PhD\frac{1}{6}]$	$[29\frac{1}{2}, 30\frac{1}{2}]$	B	0.73	0.25	0.18
7003	105	John	$[CN\frac{1}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{5}{8}]$	$[MS\frac{1}{2}, PhD\frac{1}{2}]$	$[26\frac{1}{2}, 29\frac{1}{2}]$	B	0.66	0.75	0.50
7004	105	John	$[CN\frac{1}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{5}{8}]$	MS	26	B	0.87	0.75	0.65
7001	109	Mary	$[CN\frac{1}{8}, DB\frac{1}{8}, IP\frac{1}{8}, AI\frac{5}{8}]$	$[BS\frac{1}{6}, MS\frac{1}{6}, PhD\frac{2}{3}]$	30	B	0.76	0.75	0.57
7006	110	Paul	$[CN\frac{1}{8}, DB\frac{5}{8}, IP\frac{1}{8}, AI\frac{1}{8}]$	PhD	30	$[A\frac{1}{2}, B\frac{1}{2}]$	0.87	0.375	0.33

Figure 7: The result of $\sigma_{(specialty=AI or DB) and (age \leq 30) and (city=B)}$(**Teacher** $\bowtie_{id}$ **Consultant**).

results are presented to the users according to their requirement. Moreover, α can be used to save some query processing effort, it also serves as a query optimization strategy. Query optimization based on the threshold possibility is another subject of our future work.

References

[1] C. Batini, M. Lenzerini, and S.B. Navathe, A Comparative Analysis of Methodologies for Database Schema Integration, *ACM Computing Surveys,* 18 (4) (1986) pp.323-364.

[2] Y. Breitbart, P.L. Olson, and G.R. Thompson, Database Integration in a Distributed Heterogeneous Database System, *Proc. IEEE Int. Conf. Data Eng.,* (1986) pp.301-310.

[3] Y. Breitbart, Multidatabase Interoperability, *SIGMOD RECORD,* 19 (3) (1990) pp.53-60.

[4] A. Chatterjee and A. Segev, Data Manipulation in Heterogeneous Databases, *SIGMOD RECORD,* 20 (4) (1991) pp.64-68.

[5] A.L.P. Chen, P.S.M. Tsai, and J.L. Koh, Identifying Object Polymorphism in Heterogeneous Databases, *submitted for publication.*

[6] E.F. Codd, Missing Information (Applicable and Inapplicable) in Relational Databases, *SIGMOD RECORD,* 15 (4) (1986) pp.53-78.

[7] B. Czejdo, M. Rusinkiewicz, and D.W. Embley, An Approach to Schema Integration and Query Formulation in Federated Database Systems, *Proc. IEEE Int. Conf. Data Eng.,* (1987) pp.477-484.

[8] U. Dayal and H.Y. Hwang, View Definition and Generalization for Database Integration in a Multidatabase System, *IEEE Trans. Software Eng.,* 10 (6) (1984) pp.628-644.

[9] S.M. Deen, R.R. Amin, and M.C. Taylor, Data Integration in Distributed Databases, *IEEE Trans. Software Eng.,* 13 (7) (1987) pp.860-864.

[10] L.G. DeMichiel, Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains, *IEEE Trans. Knowledge and Data Eng.,* 1 (4) (1989) pp.485-493.

[11] W. Litwin and A. Abdellatif, An Overview of the Multi-Database Manipulation Language MDSL, *Proc. of the IEEE,* 75 (5) (1987) pp.621-632.

[12] J.M. Morrissey, Imprecise Information and Uncertainty in Information Systems, *ACM Trans. Information Systems,* 8 (2) (1990) pp.159-180.

[13] A. Motro, Superviews: Virtual Integration of Multiple Databases, *IEEE Trans. Software Eng.,* 13 (7) (1987) pp.785-798.

[14] C. Pu, Key Equivalence in Heterogeneous Databases, *Proc. of the First International Workshop on Interoperability in Multidatabase Systems,* Kyoto, Japan (1991) pp.314-316.

[15] F.S.C. Tseng, A.L.P. Chen, and W.P. Yang, Answering heterogeneous database queries with degrees of uncertainty, to appear in *Distributed and Parallel Databases: an International Journal,* Kluwer Academic Publishers.

[16] Y.R. Wang and S.E. Madnick, The Inter-Database Instance Identification Problem in Integrating Autonomous Systems, *Proc. of the Fifth International Conference on Data Engineering,* Los Angeles (1989) pp.46-55.