

Tuple Merging in Probabilistic Databases

Fabian Panse and Norbert Ritter

Universität Hamburg, Vogt-Kölln Straße 33, 22527 Hamburg, Germany
{panse,ritter}@informatik.uni-hamburg.de
<http://vsis-www.informatik.uni-hamburg.de/>

Abstract. Real-world data are often uncertain and incomplete. In probabilistic relational data models uncertainty can be modeled on two levels. First by representing the uncertain instance of a tuple by a set of possible instances and second by assigning each tuple with its degree of membership to the considered relation. To overcome incompleteness, data from multiple sources need to be combined. In order to combine data from autonomous probabilistic databases, an integration of probabilistic data has to be performed. Until now, however, data integration approaches have focused on the integration of certain source data (relational or XML). There has been only less attention on the integration of uncertain (esp. probabilistic) source data so far. In this paper, we consider probabilistic tuple merging being an essential step in the integration of probabilistic data. We present techniques for merging uncertain instance data as well as for merging different degrees of tuple membership.

Key words: data integration, tuple merging, probabilistic database

1 Introduction

The increasing need for applications that produce uncertain data (e.g. in the area of astronomy [24]) has attracted high attention of uncertain data management in the database research community in recent years. Thus, several probabilistic data models have been proposed [9, 4, 16, 5, 26] and several probabilistic database prototypes have been designed [2, 17, 8].

Nevertheless, current approaches for data integration mostly consider a probabilistic handling of uncertainty emerging during the integration of certain source data (e.g. [13, 25, 26]). Integration of uncertain (esp. probabilistic) source data has only been rarely addressed so far [1]. However, to combine probabilistic data from multiple sources, for example for unifying data produced by different space telescopes, an integration of probabilistic source data is required.

In general, a data integration process consists of two steps: (a) duplicate detection [15] for identifying multiple representations of same real-world entities and (b) tuple merging [18, 23, 6] (also known as data fusion [7]) to consolidate multiple representations to a single one.

In [21] we already adapted existing techniques for duplicate detection to probabilistic data. From this duplicate detection process a set of tuple cluster

(one cluster for one real-world entity) results. In this paper, we present techniques for merging the tuples of each cluster to a single probabilistic representation.

The contributions of this paper are: (i) we define requirements for an ideal tuple merging, (ii) we present an ideal merging of instance data and (iii) we present strategies for merging tuple memberships.

The paper is structured as follows. First we present the concept of probabilistic tuples (Section 2). Then we formalize the problem of merging multiple probabilistic tuples in Section 3. In Section 4, we introduce a strategy for tuple merging in probabilistic databases. We present techniques for merging the possible instances of two probabilistic tuples (Section 4.1) as well as techniques for merging the tuples' degrees of membership (Section 4.2). Related work is presented in Section 5. Section 6 concludes the paper. Finally, in Section 7 open problems and future challenges are discussed.

2 Probabilistic Tuples

In probabilistic relational models, uncertainty is modeled on two levels: (a) each tuple t is assigned with a probability $p(t)_{\mathcal{R}} \in (0, 1]$ denoting the likelihood, also called membership degree, that t belongs to the corresponding relation \mathcal{R} (membership level), and (b) alternatives for attribute values are given (instance level).

In earlier approaches, alternatives of different attribute values are considered to be independent (e.g. [4]). In these models, each attribute value can be considered as a separate random variable with its own probability distribution. Newer models like ULDB [5] or MayBMS [17] support dependencies by introducing new concepts like ULDB's x-tuple and MayBMS's U-relation.

To get a general representation of the uncertain information captured by a single tuple, we consider a representation model which is defined within the *possible world semantics*. Theoretically, a probabilistic tuple can be considered as a set of possible instances together with the probability of each instance to be the true instance (uncertainty on instance level) and the tuple's degree of membership to the considered relation (uncertainty on membership level).

Definition 1 (Probabilistic Tuple): *Let \mathcal{R} be a probabilistic relation and $\mathcal{A} = \{A_1, A_2, \dots, A_n\}$ its finite set of attributes where each attribute $A_i \in \mathcal{A}$ has a finite domain D_i . A probabilistic tuple $t \in \mathcal{R}$ is defined as $t = ((\mathcal{I}, P), p(t)_{\mathcal{R}})$ where $\mathcal{I} = \{I_1, \dots, I_k\} \subseteq D_1 \times D_2 \times \dots \times D_n$ is the set of possible instances of t , $P : \mathcal{I} \rightarrow (0, 1]$, $\sum_{I \in \mathcal{I}} P(I) = 1$ is the probability distribution over these instances, and $p(t)_{\mathcal{R}}$ is degree of membership of t to \mathcal{R} .*

Each possible instance can be considered as a certain tuple. Along with its probabilities the set of possible instances is in the following denoted as instance data. Maybe-tuples are tuples for which the membership to the considered relation is uncertain and hence have a membership degree smaller than 1.

In the following, we present a probabilistic tuple by an own table, where each row represents a possible instance together with its probability. An extra column specifies the membership of the whole tuple. An example is depicted in Figure 1.

	$A_1 : D_1$	$A_2 : D_2$	\dots	$A_n : D_n$	$P(I)$	$p(t)_{\mathcal{R}}$
I_1	v_{11}	v_{12}	\dots	v_{1n}	0.6	0.8
I_2	v_{21}	v_{22}	\dots	v_{2n}	0.4	

Fig. 1. Representation of a probabilistic tuple $t = ((\{I_1 = (v_{11}, v_{12}, \dots, v_{1n}), I_2 = (v_{21}, v_{22}, \dots, v_{2n})\}, P : \{I_1 \mapsto 0.6, I_2 \mapsto 0.4\}), 0.8) \in \mathcal{R}$ based on the *possible world semantics*

2.1 Tuple Membership

In certain and complete data, to each entity an exact position within the real-world is assigned (not necessarily the correct one) and hence its membership to any specific extension can be exactly derived (as an example see the red dot in Figure 2(i)). In contrast, in uncertain and/or incomplete data, an entity's position only can be restricted on a subarea of the real-world (as an example see the red area in Figure 2(ii)). Consequently, for some extensions, the entity's membership cannot be exactly determined.

Definition 2 (Real-World): *The real-world, denoted \mathfrak{W} , is the set of all existing real-world entities. The mapping $\omega : \mathcal{R} \rightarrow \mathfrak{W}$ maps each tuple of any relation \mathcal{R} on an entity of \mathfrak{W} .*

Definition 3 (Relation's Extension): *The extension of a relation \mathcal{R} , denoted $Ext(\mathcal{R})$, is the part of the real-world which is actually modeled by \mathcal{R} : $Ext(\mathcal{R}) = \bigcup_{t \in \mathcal{R}} \omega(t) \subseteq \mathfrak{W}$*

The membership of a tuple to a specific relation generally depends on the relation's intended universe of discourse.

Definition 4 (Reference Extension): *The reference extension of a relation \mathcal{R} , denoted $E_{\mathcal{R}}$, is the part of the real-world which has to be intendedly modeled by \mathcal{R} : $E_{\mathcal{R}} \subseteq \mathfrak{W}$*

Note, data can be incorrect and/or uncertain. Thus, the actual set of real-world entities modeled by a relation's tuples is not necessarily a subset of the relation's *reference extension* ($Ext(\mathcal{R}) \not\subseteq E_{\mathcal{R}}$).

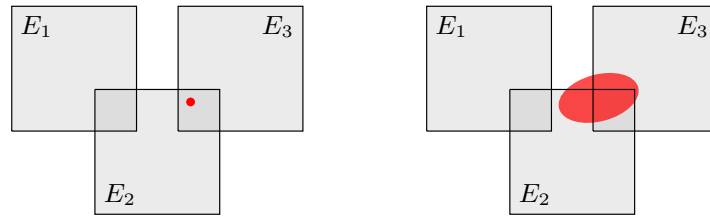


Fig. 2. Real-world position of an entity modeled in a certain and complete data source (left) and modeled in an uncertain and/or incomplete data source (right)

3 Problem Description

In the first data integration step duplicate representations of same real-world entities have been detected. The result of the duplicate detection step is a clustering of the tuples of all source relations (one cluster for each real-world entity).

Definition 5 (Clustering): Let \mathcal{R} be a relation. A clustering \mathcal{C} of \mathcal{R} is a partition $\{C_1, C_2, \dots, C_k\}$ of \mathcal{R} (meaning that the C_i are disjoint and their union equals \mathcal{R}), where each C_i is called a cluster, such that for each cluster all its tuples refer to the same real-world entity: $(\forall C_i \in \mathcal{C}) : (\forall t_1, t_2 \in C_i) : \omega(t_1) = \omega(t_2)$.

The goal of tuple merging is to combine all tuples of one cluster into a single one. As we will discuss in Section 4.2, tuple merging essentially depends on the integration context. Let $C = \{t_1, t_2, \dots, t_k\}$ be the considered cluster, where each probabilistic tuple t_i belongs to a source relation \mathcal{R}_i (the source relations of different tuples can be equal). For simplification, we consider the schema of all source relations to be identical. Let $\mathcal{S} = \{A_1 : D_1, \dots, A_n : D_n\}$, $D = D_1 \times \dots \times D_n$ be the common instance schema and E' the *reference extension* of the integration result. Thus, merging the probabilistic tuples of C can be formalized as (Please note: $\mathcal{P}(S)$ is the power set of the set S):

$$t_C = \mu(C, E'), \quad \mu : \mathcal{P}(\mathcal{P}(D \times (0, 1]) \times (0, 1]) \times \mathcal{E} \rightarrow \mathcal{P}(D \times (0, 1]) \times (0, 1]) \quad (1)$$

For reasons of clarity and comprehensibility, in the following examples, the index of a merged tuple is an ordered concatenation of the indexes of the tuples it is merged from. For example, $\mu(\{t_1, t_2, t_3\}, E')$ is denoted by t_{123} . Moreover, given the tuple $t_C = \mu(C, E')$, the tuples $\{t \in C\}$ are denoted as the base-tuples of t_C .

3.1 Requirements for an Ideal Merging

The tuple resulting from merging multiple base-tuples should properly combine the information of all these tuples. For that reason, we define a set of requirements for an ideal tuple merging. We denote a merging function μ to be ideal, if the following four conditions hold:

- (1) The merging result is independent from the representation of the source data and hence independent of the used probabilistic data model.
- (2) The function μ is associative. Thus, the tuple resulting from merging the base-tuples t_1, t_2 and t_3 w.r.t. the considered *reference extension* E' is independent of the merging order:

$$\mu(\{\mu(\{t_1, t_2\}, E'), t_3\}, E') = \mu(\{\mu(\{t_1, t_3\}, E'), t_2\}, E') = \mu(\{t_1, t_2, t_3\}, E')$$

This requirement is important if data integration is considered in a pairwise fashion.

- (3) The function is idempotent ($\mu(\{t \in \mathcal{R}\}, E') = t$), if it is considered within the tuple's original context ($E' = E_{\mathcal{R}}$). This requirement ensures that the result from deduplicating a duplicate free relation is the relation itself.

- (4) The information of all base-tuples is sufficiently captured in the merged tuple. In this case, sufficiently means that no information is lost and no information is incorrectly introduced by the merging function.

The last requirement is based on intuitive perceptions and hence its formulation is rather vague. A goal of future work is to formalize this requirement (maybe by a quantification of information loss [10]). In order to satisfy the first requirement, we generally consider tuple merging within the *possible world semantics*. The other requirements are discussed during the following sections.

4 Merging of Probabilistic Tuples

In relational data, each attribute value represents a property of the real-world entity modeled by the tuple this attribute value belongs to. For simplification, we consider the properties of a real-world entity to be independent from the membership of this entity to a specific extension. Thus the instance data of a tuple is considered to be independent from its membership to a specific relation (this is actually not always the case, see discussion in Section 7). As a consequence, we consider the merging of possible instance data and the merging of tuple memberships each as a separate process and divide the probabilistic tuple merging into two independent steps:

- (1) Merging of instance data (see Section 4.1).
- (2) Merging of tuple memberships depending on the given source context and the considered target context (see Section 4.2).

For that purpose, we decompose the merging function μ into a function for merging instance data (μ_{ID}) and a function for merging tuple memberships (μ_{TM}). The function $\mu(C, E') = (\mu_{\text{ID}}(C), \mu_{\text{TM}}(C, E'))$ is ideal, if μ_{ID} and μ_{TM} are ideal.

$$\begin{aligned} t_C = \mu(C, E') &= \mu\left(\bigcup_{t_i \in C} \{(\mathcal{I}_i, P_i), p(t_i)_{\mathcal{R}_i}\}, E'\right) = ((\mathcal{I}_C, P_C), p(t_C)_{\mathcal{R}_C}) \\ &= (\mu_{\text{ID}}(\bigcup_{t_i \in C} \{(\mathcal{I}_i, P_i)\}), \mu_{\text{TM}}(\bigcup_{t_i \in C} \{p(t_i)_{\mathcal{R}_i}\}, E')) \end{aligned}$$

As a running example throughout this paper, we consider the two probabilistic tuples $t_1 = ((\mathcal{I}_1, P_1), p(t_1)_{\text{student}})$ and $t_2 = ((\mathcal{I}_2, P_2), p(t_2)_{\text{author}})$ as presented in Figure 3. Both tuples are maybe-tuples. Moreover, each tuple has three possible instances. Two of these instances are the same in both tuples.

I_1	name	surname	location	$P_1(I)$	$p(t_1)$	→	I_2	name	surname	location	$P_2(I)$	$p(t_2)$
	John	Do	New York	0.3		←		John	Doe	Albany	0.4	
I_2	John	Doe	Albany	0.25		←	I_3	Johan	Doe	New York	0.35	
I_3	Johan	Doe	New York	0.45		←	I_4	Jon	Ho	York	0.25	

$t_1 = ((\mathcal{I}_1, P_1), p(t_1)_{\text{student}} = 0.8)$

$t_2 = ((\mathcal{I}_2, P_2), p(t_2)_{\text{author}} = 0.5)$

Fig. 3. Probabilistic tuples $t_1 \in \text{student}$, $t_2 \in \text{author}$

4.1 Merging of Instance Data

In certain data, instance merging is considered on an attribute by attribute basis. Since in probabilistic data dependencies between attribute values can exist, we consider merging techniques always for whole instances. In contrast to a merging of certain tuples [7], conflict resolution by choosing one of the conflicting items (deciding strategy) or by creating a new representative (mediating strategy) is generally not required. Instead, each kind of uncertainty can be stored in the resulting data by taking multiple possible instances into account.

An ideal instance merging results from the union of all possible instances of all base-tuples. Since the merging is not associative, if a simple average of the individual probabilities is calculated, we assign a weight q_i to each tuple t_i and define that the weight of a merged tuple results from the sum of the weights of its base-tuples ($q_{ij} = q_i + q_j$). If tuple merging is considered within the context of data integration, the reliabilities of the corresponding sources can be used as tuple weights. In conclusion, an ideal function for merging instance data can be formalized as:

$$\mu_{\text{id}}\left(\bigcup_{i \in [1, k]} \{(\mathcal{I}_i, P_i)\}\right) = \left(\bigcup_{i \in [1, k]} \mathcal{I}_i, \frac{\sum_{i \in [1, k]} q_i P_i}{\sum_{i \in [1, k]} q_i}\right) \quad (2)$$

It is obvious, that μ_{id} is idempotent and associative. Moreover, the function takes each instance into account which is possible for at least one base-tuple and does not add an instance which is impossible for all base-tuples. Thus, intuitively this function also satisfies the requirement of sufficiently capturing the information of the given set of base-tuples.

Running Example: The instance data of the tuple $t_{12} = \mu(\{t_1, t_2\}, E')$ resulting from merging the two base-tuples t_1 and t_2 of Figure 3 by using the tuple weights $q_1 = 0.6$ and $q_2 = 0.4$ is presented in Figure 4.

	name	surname	location	$P_{12}(I)$
I_1	John	Do	New York	0.18
I_2	John	Doe	Albany	0.31
I_3	Johan	Doe	New York	0.41
I_4	Jon	Ho	York	0.1

$$(\mathcal{I}_{12}, P_{12}) = \mu_{\text{id}}(\{(\mathcal{I}_1, P_1), (\mathcal{I}_2, P_2)\})$$

Fig. 4. Instance data $(\mathcal{I}_{12}, P_{12})$ resulting from merging (\mathcal{I}_1, P_1) and (\mathcal{I}_2, P_2)

User-defined Aggregation Functions. To enable the usage of additional domain knowledge, for each attribute a specific aggregation function can be defined by the user (resp. a domain expert). This is an important property, because in some scenarios a simple union of all possible instance values of an attribute is not adequate (e.g. in merging sales data) or other information for reducing the set of possible instances is available (e.g. the domain expert knows that the address field of a specific tuple is the correct one).

As an example, we consider a relation *inventory* with the three attributes *name*, *producer* and *stock* (see Figure 5). The two tuples $t_3 = ((\mathcal{I}_3, P_3), p_3)$ and $t_4 = ((\mathcal{I}_4, P_4), p_4)$ represent the same product, but the stock information of each tuple belongs to different orders. Therefore, in this scenario neither 15, 20 nor 6 items of this product but rather 21 or 26 items are available. As a consequence, the true stock value of this product results from the sum of stocks of both base-tuples instead of being the stock of one of them.

In general, the values of each possible combination of instances belonging to different base-tuples have to be aggregated. If for all attributes an aggregation function is defined, for each of these combinations a single possible instance of the merged tuple results by aggregating the values for each attribute. Otherwise, for each combination two instances result (one for each of the combined instances).

In our example, two aggregation functions are defined. As mentioned above, the true stock value is specified by the *sum*-function (mediating strategy). Moreover, the user knows that the *producer* value of the second instance data (\mathcal{I}_4) is correct. For that reason, always the *producer* of this tuple is chosen (deciding strategy). For the attribute *name*, no function is specified. Thus, all possible values are taken into account (see Figure 5). Note, instance merging is not ideal, if at least one non-associative aggregation function (e.g. average) is used.

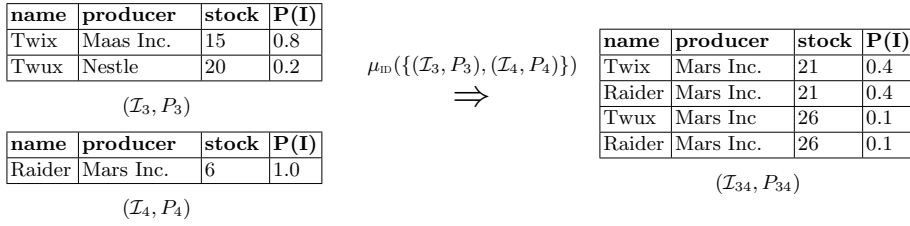


Fig. 5. Example for instance merging with user-defined aggregation functions

4.2 Merging of Tuple Memberships

The degree of membership of a merged tuple to the result relation depends on two factors: (i) the overlap scenario of the source relations' *reference extensions* (source context) and (ii) the intended scope of the result relation (target context).

Source Context: The *reference extensions* $E_{\mathcal{R}_1}$ and $E_{\mathcal{R}_2}$ of two relations \mathcal{R}_1 and \mathcal{R}_2 can be in four different overlap situations (see Figure 6(i)-(iv)).

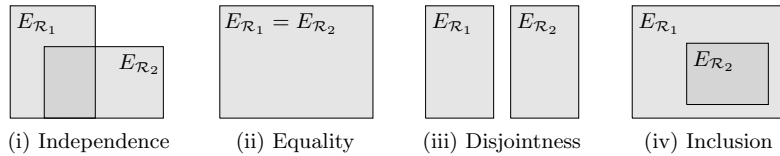


Fig. 6. The four different overlap situations of two *reference extensions*

Both *reference extensions* can be independent from each other (partially overlapping), equal, disjoint or one extension can be a subset of the other one (quantified overlap is supposed to be considered in future work). A set of pairwise overlap situations is called an overlap scenario.

To detect the given overlap scenario of different *reference extensions*, additional meta information on the individual sources and general information on real-world relationships is required. In our research, based on semantic concepts as ontologies and thesauri, we aim to identify overlap scenarios only by using the source relations' names (see example shown in Figure 7).

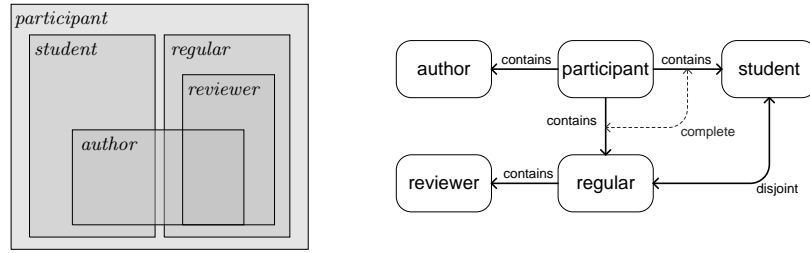


Fig. 7. Overlap Scenario of *reference extensions* (left), corresponding ontology (right)

Target Context: The membership of a merged tuple to the result relation generally depends on the integration's intended universe of discourse. In the following, the *reference extension* of the integration's result relation is denoted as *target reference extension*.

Let \mathcal{R}_3 be the relation which results from integrating the relations \mathcal{R}_1 and \mathcal{R}_2 . In general, each part of the real world ($E_{\mathcal{R}_3} \subseteq \mathfrak{W}$) can be considered as *target reference extension* (e.g., $E_{\mathcal{R}_3} \supseteq E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2}$). However, the most cases are very unusual and there is not enough information available for predicting an adequate membership degree of the resulting tuples. Moreover, for the most integration processes an intuitive *target reference extension* can be implied by the used merge operators. For example, in [20] the four merge operators *merge join* \sqcap , *left outer merge join* \sqsubset , *right outer merge join* \sqsupset and *full outer merge join* \sqcup are introduced. An intuitive *target reference extension* $E_{\mathcal{R}_3}$ for each of these operators can be defined as:

$$\frac{\mathcal{R}_3}{E_{\mathcal{R}_3}} = \left\| \begin{array}{c|c|c|c} \mathcal{R}_1 \sqcap \mathcal{R}_2 & \mathcal{R}_1 \sqcup \mathcal{R}_2 & \mathcal{R}_1 \sqsubset \mathcal{R}_2 & \mathcal{R}_1 \sqsupset \mathcal{R}_2 \\ \hline E_{\mathcal{R}_1} \cap E_{\mathcal{R}_2} & E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2} & E_{\mathcal{R}_1} & E_{\mathcal{R}_2} \end{array} \right\|$$

Membership Merging in Consistent and Complete Data: Given the definitions above, the membership merging function μ_{TM} can be formalized as:

$$\mu_{\text{TM}} : \mathcal{P}((0, 1]) \times \mathcal{P}(\mathfrak{W}) \rightarrow (0, 1], \quad p(t_C)_{\mathcal{R}_C} = \mu_{\text{TM}}\left(\bigcup_{t_i \in C} \{p(t_i)_{\mathcal{R}_i}\}, E'\right) \quad (3)$$

where $\mathcal{P}(\mathfrak{W})$ is the set of all possible parts of the real-world and E' is the considered *target reference extension*. As mentioned above, the quality of membership

merging can be enhanced if the overlap scenario of the source relations' *reference extensions* is given. Otherwise, situations of independence have to be assumed.

Target reference extension:	Membership merging function: $p(t_{12})_{\mathcal{R}_3} = \mu_{\text{TM}}(\{p(t_1)_{\mathcal{R}_1}, p(t_2)_{\mathcal{R}_2}\}, E_{\mathcal{R}_3})$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_1}) + P(e \in E_{\mathcal{R}_2}) - P(e \in E_{\mathcal{R}_1} \wedge e \in E_{\mathcal{R}_2})$ $\Rightarrow p(t_{12})_{\mathcal{R}_3} = p(t_1)_{\mathcal{R}_1} + p(t_2)_{\mathcal{R}_2} - p(t_1)_{\mathcal{R}_1} \cdot p(t_2)_{\mathcal{R}_2}$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_1} \cap E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_1} \wedge e \in E_{\mathcal{R}_2})$ $\Rightarrow p(t_{12})_{\mathcal{R}_3} = p(t_1)_{\mathcal{R}_1} \cdot p(t_2)_{\mathcal{R}_2}$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_1}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_1})$ $\Rightarrow p(t_{12})_{\mathcal{R}_3} = p(t_1)_{\mathcal{R}_1}$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_2})$ $\Rightarrow p(t_{12})_{\mathcal{R}_3} = p(t_2)_{\mathcal{R}_2}$

Fig. 8. Membership merging in case of two independent *reference extensions*

As a demonstrating example, we consider all four merge operators ($\sqcup, \sqcap, \sqcup, \sqcap$) w.r.t. the two independent *reference extensions* $E_{\mathcal{R}_1}$ and $E_{\mathcal{R}_2}$ (see Figure 8). Both base-tuples t_1 and t_2 represent the same real-world entity $e = \omega(t_1) = \omega(t_2)$. If for example, the full outer join merge is used ($E_{\mathcal{R}_3} = E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2}$), the probability that e belongs to the *target reference extension* is equal to the probability that one of the two base-tuples t_1 and t_2 belongs to their corresponding relation ($P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_1}) + P(e \in E_{\mathcal{R}_2}) - P(e \in E_{\mathcal{R}_1} \wedge e \in E_{\mathcal{R}_2})$).

In probability theory, the situation of independence can be specialized to the situations of equality, inclusion or disjointness by introducing some additional dependencies:

(Equality) $E_{\mathcal{R}_1} = E_{\mathcal{R}_2}$	Dep. 1.: $\omega(t_1) \in E_{\mathcal{R}_1} \Leftrightarrow \omega(t_2) \in E_{\mathcal{R}_2}$ $\Rightarrow p(t_1)_{\mathcal{R}_1} = p(t_2)_{\mathcal{R}_2}$
(Disjointness) $E_{\mathcal{R}_1} \cap E_{\mathcal{R}_2} = \emptyset$	Dep. 1.: $\omega(t_1) \in E_{\mathcal{R}_1} \Rightarrow \omega(t_2) \notin E_{\mathcal{R}_2}$ Dep. 2.: $\omega(t_2) \in E_{\mathcal{R}_2} \Rightarrow \omega(t_1) \notin E_{\mathcal{R}_1}$ $\Rightarrow p(t_1)_{\mathcal{R}_1} + p(t_2)_{\mathcal{R}_2} \leq 1$
(Inclusion) $E_{\mathcal{R}_1} \supset E_{\mathcal{R}_2}$	Dep. 1.: $\omega(t_2) \in E_{\mathcal{R}_2} \Rightarrow \omega(t_1) \in E_{\mathcal{R}_1}$ $\Rightarrow p(t_1)_{\mathcal{R}_1} \geq p(t_2)_{\mathcal{R}_2}$

If for the given membership degrees some of these dependencies are not valid, elementary principles of probability theory can be violated. As for example:

$$(\exists e \in \mathfrak{W}) : (\exists E_{\mathcal{R}} \subseteq \mathfrak{W}) : P(e \in E_{\mathcal{R}}) + P(e \notin E_{\mathcal{R}}) \neq 1$$

In this case, we call the given membership degrees to be inconsistent to each other. In a single probabilistic database we can assume that all membership degrees are defined in a consistent way. In data integration, however, we operate with degrees specified by different independent sources. Thus such an assumption is not reasonable.

As a consequence, deriving membership merging functions for the situations of equality, inclusion or disjointness from the functions defined for the situation of independence (see Figure 8) is not suitable.

Moreover, for queries defined on a single database, the *closed world assumption* (CWA) [22] applies. Thus, each relation is assumed to be complete and each real-world entity which is not represented by a relation's tuple is assumed to be definitely not belonging to the relation's *reference extension* ($p(t)_{\mathcal{R}} = 0 \Rightarrow \omega(t) \notin E_{\mathcal{R}}$). Nevertheless, one main purpose of integrating data is to join multiple incomplete sources to a complete result. Thus, for source relations completeness cannot be assumed and the hypothesis that each missing tuple definitely does not belong to the considered relation cannot be made. For dealing with missing membership degrees, we make a generalization of the CWA and assume that the membership of missing tuples is completely unknown.

In conclusion, the problem of membership merging is to determine an adequate probability that the considered entity belongs to the result relation despite of inconsistent and missing membership degrees given by the individual sources.

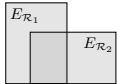
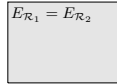
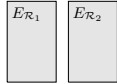
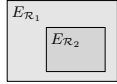
Overlap situation:		Membership merging function:
		$p(t_{12})_{\mathcal{R}_1 \sqcup \mathcal{R}_2} = \mu_{TM}(\{p(t_1)_{\mathcal{R}_1}, p(t_2)_{\mathcal{R}_2}\}, E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2})$
Independence		$p(t_{12})_{\mathcal{R}_1 \sqcup \mathcal{R}_2} = p(t_1)_{\mathcal{R}_1} + p(t_2)_{\mathcal{R}_2} - p(t_1)_{\mathcal{R}_1} \cdot p(t_2)_{\mathcal{R}_2}$
Equality		$p(t_{12})_{\mathcal{R}_1 \sqcup \mathcal{R}_2} = (q_1 \cdot p(t_1)_{\mathcal{R}_1} + q_2 \cdot p(t_2)_{\mathcal{R}_2}) / (q_1 + q_2)$
Disjointness		$p(t_{12})_{\mathcal{R}_1 \sqcup \mathcal{R}_2} = \min(1, p(t_1)_{\mathcal{R}_1} + p(t_2)_{\mathcal{R}_2})$
Inclusion		$p(t_{12})_{\mathcal{R}_1 \sqcup \mathcal{R}_2} = \begin{cases} p(t_1)_{\mathcal{R}_1} & , \text{ if } p(t_1)_{\mathcal{R}_1} \geq 0 \\ (q_2 \cdot p(t_2)_{\mathcal{R}_2}) / (q_1 + q_2) & , \text{ else} \end{cases}$

Fig. 9. Four different overlap situations together with possible membership merging functions defined for the case of inconsistent membership degrees

Inconsistent Membership Degrees: In Figure 9 we present four possible functions defined for merging tuple memberships in a *full outer merge join* adapted to inconsistent membership degrees. Even though we defined these functions to be binary, a definition for more than two base-tuples is also possible. In the situations of equality and inclusion, we use the two weights q_1 and q_2 already defined in Section 4.1.

Note, the merging functions defined for each situation are individually associative. From a global point of view (e.g. an overlap scenario of multiple source extensions as shown in Figure 7(i)), however, the merging is not associative and hence not ideal. Defining a globally associative merging of inconsistent membership degrees seems very hard and is an interesting challenge of future research

(see Section 7). Nevertheless, in many integration scenarios the reference extensions of all source relations are equal (e.g. in unifying data resulting from different observations of same real-world phenomena) , disjoint (e.g. in an integration of multiple local databases into a global one) , almost independent (e.g. in integrating data resulting from observations of several independent real-world phenomena) or pairwise include each other (e.g. in integrating data for enhancing the quality of a specific database’s part). Since each of the merging functions defined in Figure 9 is individually associative, for such scenarios an ideal membership merging can be guaranteed.

$E_{\mathcal{R}_3} = E_{\mathcal{R}_1} \cup E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) \geq P(e \in E_{\mathcal{R}_1}) \Rightarrow p(t_C)_{\mathcal{R}_3} \in [p(t_1)_{\mathcal{R}_1}, 1]$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_1} \cap E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) \leq P(e \in E_{\mathcal{R}_1}) \Rightarrow p(t_C)_{\mathcal{R}_3} \in [0, p(t_1)_{\mathcal{R}_1}]$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_2}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_2}) \Rightarrow p(t_C)_{\mathcal{R}_3} \in [0, 1]$
$E_{\mathcal{R}_3} = E_{\mathcal{R}_1}$	$P(e \in E_{\mathcal{R}_3}) = P(e \in E_{\mathcal{R}_1}) \Rightarrow p(t_C)_{\mathcal{R}_3} = p(t_1)_{\mathcal{R}_1}$

Fig. 10. Membership merging w.r.t. two independent source relations \mathcal{R}_1 and \mathcal{R}_2 in case of a missing membership degree of e to $E_{\mathcal{R}_2}$ ($P(e \in E_{\mathcal{R}_2}) \in [0, 1]$)

Missing Membership Degrees: Independent from the duplicate detection result, the membership degree of each tuple has to be recalculated, if the *target reference extension* deviates from the *reference extension* of its source relation. For that reason, in an integration process working on disparate *reference extensions*, membership merging has to be applied on each duplicate cluster, whether this cluster contains multiple tuples or not.

If a cluster does not contain a tuple for each of the different source relations’ *reference extensions*, the membership degree of the considered real-world entity to this extension is missing. Thus, in some contexts, the membership of the merged tuple to the *target reference extension* cannot be exactly determined as described above and instead of an exact value only a range of possible membership can be specified. Since, in this paper, we restrict ourselves to binary merging functions, we exemplary consider the case of an integration of two relations \mathcal{R}_1 and \mathcal{R}_2 and a cluster C that only contains a single tuple $C = \{t_1 \in \mathcal{R}_1\}$. Depending on the considered *target reference extension* ($E_{\mathcal{R}_3}$), the resulting membership degree $p(t_C)_{\mathcal{R}_3}$ is more or less uncertain. As an example, we consider the situation of independence which is shown in Figure 10. In three of four target contexts only a range of possible membership results. In the last case, however, the *target reference extension* is equal to $E_{\mathcal{R}_1}$. Thus, in this case, none of the required membership degrees is missing and the resulting degree of membership can be exactly determined. Note, if in the probabilistic target model, probability ranges cannot be stored, using the expected probability seems most suitable.

Ideality: Membership merging without inconsistent and missing membership degrees is based on probability theory and hence seems to be sufficient. In contrast, a sufficient capturing of the information represented by the individual memberships cannot be intuitively defined if inconsistent or missing membership

degrees exist. Nevertheless, in case of inconsistent membership degrees, a preserving of the information represented by all the base-tuples' memberships is not reasonable, because the information on memberships is definitely not accurate.

Running Example: With respect to our running example, the reference extensions of both source relations are independent (see Figure 7). Thus, by using the *join merge*, the degree of membership of the merged tuple to the result relation representing all student authors results in:

$$p(t_{12})_{student\ authors} = p(t_1)_{student} \cdot p(t_2)_{authors} = 0.4$$

5 Related Work

Tuple merging in certain data is considered in different works [11, 7, 19, 6]. Since in certain data only single values can be stored, conflicts can only be resolved by choosing one of the conflicting values (e.g. by using *max*) or by creating a new representative (e.g. by using *avg*). With respect to the most conflict resolution functions tuple merging is not associative and hence not ideal. Moreover, our approach is more general, which can be specialized to conflict resolution as defined for certain data if for each attribute an aggregation function is specified.

Robertson et al [23] consider tuple merging within a transposition of certain data. Merging of two tuples with contrary instance data is not provided (in such cases both tuple are denoted to be *non mergeable*).

DeMichiel [12] and Tseng [25] use *partial values* (resp. *probabilistic values*) to resolve conflicts between certain values by taking multiple possible instances into account. Consequently, these approaches already produce uncertain data as result data. This is similar to our ideal instance merging if each base-tuple is considered to be certain and no aggregation functions are used. Nevertheless, both approaches consider conflict resolution on an attribute by attribute basis. Dependencies between possible attribute values are not considered.

Andritsos et al [3] define queries on multiple conflicting duplicates. Thus instead of merging the tuples of each cluster into a single one, query results are derived from sets of mutual exclusive base-tuples. Since to each cluster's tuple a probability can be assigned, this approach is mostly identical to our ideal instance merging without the additional offering of attribute specific aggregation functions.

None of the studies, however, allows uncertain (esp. probabilistic) data as source data. Membership merging is consequently not handled in these works.

A merging of tuples representing uncertain information (on instance as well as membership level) is proposed by Lim et al [18]. Nevertheless, instead of probability theory this approach is based on the *Dempster-Shafer theory of evidence*. For membership merging in a union of two relations the authors do not take different *target reference extensions* or different overlap scenarios of source *reference extensions* into account. Moreover, the authors explicitly specify a *membership derivation function* only for the relational selection operator.

In the publication of Agrawal et al [1], deduplication is not considered.

6 Conclusion

Many applications naturally produce uncertain data. For that reason, probabilistic databases have become a topic of interest in the database community in recent years. In order to combine the data from different probabilistic data sources, an integration process has to be applied. To obtain concise integration results, merging of duplicate tuples is an essential activity. We consider duplicate detection in probabilistic data in [21]. In this paper, we have investigated how a set of probabilistic tuples designated as duplicates can be merged to a single one. We have considered probabilistic tuples representing uncertainty on instance level and uncertainty on membership level. We have defined a set of requirements for an ideal tuple merging. Moreover, we have divided probabilistic tuple merging into a merging of instance data as well as a merging of membership degrees. Without additional domain knowledge, instance merging is realized by the union of the tuples' possible instances. Otherwise user-defined aggregation functions can be used. For defining an adequate membership merging, we take the overlap scenario of the real-world scopes modeled by all source relations as well as the intended scope of the integration result into account. Whereas the instance merging is always ideal if solely associative aggregation functions are used, an ideal membership merging only results if either the underlying membership degrees are consistent to each other or a scenario with only a single kind of overlap situation (independence, equality, disjointness, or inclusion) exists.

In conclusion, this paper gives first ideas in the large area of merging duplicate tuples in probabilistic databases. Nevertheless, open problems still exist. Thus, we discuss some future challenges in the following section.

7 Open Problems and Future Challenges

As already mentioned in Section 4.2, the presented membership merging functions are only associative, if a scenario with same situations of overlap is given. Otherwise associativity only can be guaranteed if the underlying membership degrees are consistent to each other.

Challenge 1 *Definition of an associative merging of inconsistent membership degrees beyond scenarios with same situations of overlap.*

Usually relations to be integrated have heterogeneous schemas and their sets of attributes only partially overlap. In this case, merging of instance data could be considered as a kind of full outer union in relational data where missing values are filled up with uniform distributions on corresponding attribute domains.

Challenge 2 *Techniques for merging probabilistic tuples defined on heterogeneous schemas.*

In contrast to the assumption made in Section 4, membership merging and instance merging is not always independent from each other. The instance of an entity and its membership to a specific extension depends on each other, if (a) only entities of the considered extension have a specific property (e.g. only

students have a study path or a student number) or if (b) the membership to the considered extension restricts the value of at least one entity’s property on some special domain elements (e.g. each mathematics student has the study path ‘mathematics’ or each driver is older than 18 years). Given a relation \mathcal{R} , an attribute A defined in the domain D and the symbol \perp denoting the situation of nonexistence, corresponding functional dependencies can be specified as:

$$\textit{existence dependency: } e \notin \textit{Ext}(\mathcal{R}) \rightarrow (\forall t \in \omega^{-1}(e)) : t.[A] = \perp \quad (4)$$

$$\textit{value dependency: } e \in \textit{Ext}(\mathcal{R}) \rightarrow (\forall t \in \omega^{-1}(e)) : t.[A] \in X \subseteq D \quad (5)$$

Challenge 3 *Adaptation of the presented tuple merging to existence dependencies and value dependencies between instance data and membership degrees.*

We presented an ideal merging of instance data in Section 4.1. The requirements for ideality guarantee that the resulting instance data is as correct as possible. Nevertheless, by using an ideal merging function the instance data on a single real-world entity becomes more and more uncertain the more tuples are merged together. This, however, is most often not the actual purpose of data integration. Thus, in many applications using an ideal merging function is often not valuable and other merging strategies are required. Finding a merging strategy best fitting for a special application is generally a trade off between correctness and certainty. Most correct and also most uncertain data results, if all possible instances of all base-tuples are taken into account. In contrast, the result is most certain but most likely also incorrect, if only one of the possible instances is chosen. In many applications an adequate merging function has to be a compromise between these two extremes.

Challenge 4 *Definition of some non-ideal functions making a suitable trade-off between certainty and correctness possible.*

We do not address a merging of data lineage in this paper. In many probabilistic data models, e.g., ULDB, however, data lineage is an important concept which can be used for validating the consistence of given probabilities.

Challenge 5 *Techniques for merging the base-tuples’ lineage in a way that the merged membership degree can be consistently derived from the merged lineage.*

A non-consideration of dependencies between individual data sources can impair the quality of the merged tuple. Usually, data sources are not independent from each other. In contrast, often the data of one source is copied from another source. Thus false instances can be spread through copying and are considered in tuple merging with high certainty. Techniques for detecting dependencies between individual sources are proposed in [14].

Challenge 6 *To figure out the role of source dependencies in merging probabilistic tuples.*

Finally, one of the most important challenges is to adapt the proposed tuple merging strategies to more succinct representation models on which probabilistic databases usually are based.

Challenge 7 *Adaptation of the presented merging functions to probabilistic data-models not storing each possible instance of a tuple separately.*

References

1. P. Agrawal et al. Foundations of Uncertain-Data Integration. In *VLDB 2010*. Stanford.
2. P. Agrawal et al. Trio: A System for Data, Uncertainty, and Lineage. In *VLDB*, pages 1151–1154, 2006.
3. P. Andritsos et al. Clean Answers over Dirty Databases: A Probabilistic Approach. In *ICDE*, page 30, 2006.
4. D. Barbará et al. The Management of Probabilistic Data. *IEEE Trans. Knowl. Data Eng.*, 4(5):487–502, 1992.
5. O. Benjelloun et al. ULDBs: Databases with Uncertainty and Lineage. In *VLDB*, pages 953–964, 2006.
6. O. Benjelloun et al. Swoosh: a generic approach to entity resolution. *VLDB J.*, 18(1):255–276, 2009.
7. J. Bleiholder et al. Data fusion. *ACM Comput. Surv.*, 41(1), 2008.
8. J. Boulos et al. MYSTIQ: a system for finding more answers by using probabilities. In *SIGMOD Conference*, pages 891–893, 2005.
9. R. Cavallo et al. The Theory of Probabilistic Databases. In *VLDB*, pages 71–81, 1987.
10. T. M. Cover et al. *Elements of Information Theory*. Wiley & Sons, 2006.
11. U. Dayal. Processing Queries Over Generalization Hierarchies in a Multidatabase System. In *VLDB*, pages 342–353, 1983.
12. L. G. DeMichiel. Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains. *IEEE Trans. Knowl. Data Eng.*, 1(4):485–493, 1989.
13. X. Dong et al. Data Integration with Uncertainty. *VLDB J.*, 18(2):469–500, 2009.
14. X. Dong et al. Integrating Conflicting Data: The Role of Source Dependence. *PVLDB*, 2(1):550–561, 2009.
15. A. K. Elmagarmid et al. Duplicate Record Detection: A Survey. *IEEE Trans. Knowl. Data Eng.*, 19(1):1–16, 2007.
16. N. Fuhr et al. A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems. *ACM Trans. Inf. Syst.*, 15(1):32–66, 1997.
17. J. Huang et al. MayBMS: a probabilistic database management system. In *SIGMOD Conference*, pages 1071–1074, 2009.
18. E.-P. Lim et al. An Evidential Reasoning Approach to Attribute Value Conflict Resolution in Database Integration. *IEEE Trans. Knowl. Data Eng.*, 8(5):707–723, 1996.
19. A. Motro et al. Fusionplex: Resolution of Data Inconsistencies in the Integration of Heterogeneous Information Sources. *Information Fusion*, 7(2):176–196, 2006.
20. F. Naumann et al. Completeness of Integrated Information Sources. *Inf. Syst.*, 29(7):583–615, 2004.
21. F. Panse et al. Duplicate Detection in Probabilistic Data. In *NTII*, pages 179–182, 2010.
22. R. Reiter. On Closed World Data Bases. In *Logic and Data Bases*, pages 55–76, 1977.
23. E. Robertson et al. Optimal Tuple Merge is NP-Complete. Technical report, 2004.
24. D. Suciú et al. Embracing Uncertainty in Large-Scale Computational Astrophysics. In *MUD*, pages 63–77, 2009.
25. F. S.-C. Tseng et al. Answering Heterogeneous Database Queries with Degrees of Uncertainty. *Distributed and Parallel Databases*, 1(3):281–302, 1993.
26. M. van Keulen et al. A Probabilistic XML Approach to Data Integration. In *ICDE*, pages 459–470, 2005.