Duplicate Detection in Probabilistic Data

Fabian Panse #1, Maurice van Keulen *2, Ander de Keijzer *3, Norbert Ritter #4

# *Computer Science Department, University of Hamburg*
Vogt-Koelln Straße 33, 22527 Hamburg, Germany
1 panse@informatik.uni-hamburg.de
4 ritter@informatik.uni-hamburg.de

* *Faculty of EEMCS, University of Twente*
POBox 217, 7500 AE Enschede, The Netherlands
2 m.vankeulen@utwente.nl
3 a.dekeijzer@utwente.nl

Abstract— **Collected data often contains uncertainties. Probabilistic databases have been proposed to manage *uncertain* data. To combine data from multiple 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 is no work on the integration of *uncertain* source data so far. In this paper, we present a first step towards a concise consolidation of *probabilistic* data. We focus on duplicate detection as a representative and essential step in an integration process. We present techniques for identifying multiple *probabilistic* representations of the same real-world entities.**

I. Introduction

In a large number of application areas (e.g., astronomy [1]), the demand for storing *uncertain* data grows increasingly from year to year. As a consequence, in the last decades several probabilistic data models have been proposed (e.g., [2], [3], [4], [5]) and recently several probabilistic database prototypes have been designed (e.g., [6], [7], [8]).

In current research on data integration, probabilistic data models are only considered for handling uncertainty in an integration of *certain* source data (e.g., relational [9], [10] or XML [11]). Integration of *uncertain* (esp. *probabilistic*) source data has not been considered so far. However, to consolidate multiple probabilistic databases to a single one, for example for unifying data produced by different space telescopes, an integration of *probabilistic* source data is necessary.

In general, an integration process mainly consists of four steps: (a) schema matching [12] and (b) schema mapping [13] to overcome schema and data heterogeneity; (c) duplicate detection [14] (also called entity resolution or record linkage) and (d) data fusion [15] to reconcile data about the same real-world entities. In this paper, we focus on duplicate detection as a representative step in the data integration process and show how to adapt existing techniques to *probabilistic* data. For an extended version of this paper see [16], including first considerations of search space reduction.

The paper is structured as follows. First we present related work (Section II). In Section III, we examine current techniques of duplicate detection. Then we introduce duplicate detection for probabilistic databases in Section IV. Section V concludes the paper and gives an outlook on future research.

II. Related Work

In general, probability theory is already applied in methods for duplicate detection (e.g., decision models), but current approaches only consider *certain* relational ([17], [18], [19]) or XML data [20]. *Probabilistic* source data is not considered in these works. On the other hand, many techniques that focus on data preparation [21] and verification [22] as well as fundamental concepts of decision model techniques [22] can be adopted for duplicate detection in *probabilistic* data. Furthermore, existing comparison functions [14] can be incorporated into techniques for comparing *probabilistic* values.

There are several approaches that explicitly handle and produce *probabilistic* data in schema integration, duplicate detection and data fusion. Handling the uncertainty in schema integration requires *probabilistic* schema mappings [10], [23]. Van Keulen and De Keijzer ([5], [11]) use a semi-structured probabilistic model to handle ambiguities arising during deduplication in XML data. Tseng [9] already used *probabilistic* values in order to resolve conflicts between two or more *certain* relational values. None of the studies, however, allows *probabilistic* data as source data.

III. Fundamentals of Duplicate Detection

The data sets to be integrated may contain data on the same real-world entities. Often it is even the purpose of integration: to combine data on these entities. In order to integrate two or more data sets in a meaningful way, it is necessary to identify representations belonging to the same real-world entity. Therefore, duplicate detection is an important component in an integration process. Due to deficiencies in data collection, data modeling or data management, real-life data is often incorrect and/or incomplete. This principally hinders duplicate detection. Therefore, duplicate detection techniques have to be designed for properly handling dissimilarities due to missing data, typos, data obsolescence or misspellings.

In general, duplicate detection consists of five steps [22]:

A. Data Preparation

Data is standardized (e.g., unification of conventions and units) and cleaned (eleminiation of easy to recognize errors) to obtain a homogeneous representation of all source data [21].

B. Search Space Reduction

Since a comparison of all combinations of tuples is mostly too inefficient, the search space is usually reduced using heuristic methods (e.g., the sorted neighborhood method [22]).

C. Attribute Value Matching

Similarity of tuples is usually based on the similarity of the corresponding attribute values. Despite data preparation, syntactic as well as semantic irregularities remain. Thus, attribute value similarity is quantified by syntactic (e.g., n-grams, edit- or jaro distance [14]) and semantic (e.g., glossaries or ontologies) means. From comparing two tuples, we obtain a *comparison vector* $\vec{c} = [c_1, \ldots, c_n]$, where c_i represents the similarity of the values from the ith attribute.[1]

D. Decision Model

The *comparison vector* is input to a decision model which determines to which set a tuple pair (t_1, t_2) is assigned: matching tuples (M), unmatching tuples (U) or possibly matching tuples (P). In the following, the decision's result is stored in the *matching value* $\eta(t_1, t_2) \in \{m, p, u\}$, where m represents the case that (t_1, t_2) is assigned to M (resp. to P or U).

The most common decision models are based on domain knowledge or probability theory.

Knowledge-based techniques. In knowledge-based approaches for duplicate detection [22], domain experts define *identification rules* (see Figure 1). These rules specify conditions when two tuples are considered duplicates with a given confidence (*certainty factor*). Ultimately, if the resulting certainty is greater than two thresholds seperating M, P and U, the tuple pair is considered to be a duplicate.

```
IF name > threshold₁ AND job > threshold₂
   THEN DUPLICATES with CERTAINTY=0.8
```

Fig. 1. Identification rule

Probabilistic techniques. In the theory of fellegri and sunter ([18], [22]), two conditional probabilities $m(\vec{c})$ (*m-probability*) and $u(\vec{c})$ (*u-probability*) are defined for each tuple pair (t_1, t_2).

$$m(\vec{c}) = P(\vec{c} \mid (t_1, t_2) \in M) \quad (1)$$
$$u(\vec{c}) = P(\vec{c} \mid (t_1, t_2) \in U) \quad (2)$$

Based on the *matching weight* $R = m(\vec{c})/u(\vec{c})$ and the thresholds T_μ and T_λ, the tuple pair (t_1, t_2) is considered to be a match, if $R > T_\mu$ or a non-match, if $R < T_\lambda$. Otherwise, the tuples are a possible match and clerical reviews are required.

In general, the decision whether a tuple pair (t_1, t_2) is a match or not, can be decomposed into two steps (see Figure 2). In the first step, a single similarity degree $sim(t_1, t_2)$ is determined by a *combination function*:

$$\varphi : [0,1]^n \to \mathbb{R} \qquad sim(t_1, t_2) = \varphi(\vec{c}) \quad (3)$$

[1]If multiple comparison functions are used, we even obtain a matrix. Without loss of generality, we restrict ourselves to a *comparison vector*. Furthermore, we restrict on normalized comparison functions ($\Rightarrow \vec{c} \in [0,1]^n$).

The resulting degree is normalized, if a knowledge-based technique is used (*certainty factor*) and non-normalized if a probabilistic technique is applied (*matching weight*). In a second step, based on $sim(t_1, t_2)$ the tuple pair is assigned to one of the sets M, P or U by using two thresholds.

```
Input: tuple pair $(t_1, t_2)$, comparison vector ($\vec{c} = [c_1, \ldots, c_n]$)
1.  Execution of the combination function $\varphi(\vec{c})$
    ⇒ Result: $sim(t_1, t_2)$
2.  Classification of $(t_1, t_2)$ into $\{M, P, U\}$ based on $sim(t_1, t_2)$
Output: Decision whether $(t_1, t_2)$ is a duplicate or not
```

Fig. 2. General representation of existing decision models

E. Verification

The effectiveness of the applied identification is checked in terms of recall, precision, false negative percentage and false positive percentage [22]. If the effectiveness is not satisfactory, duplicate detection is repeated with other, better suitable thresholds or methods (e.g., other comparison functions).

IV. Duplicate Detection in Probabilistic Data

Theoretically, a probabilistic database is defined as *PDB* $= (W, P)$ where $W = \{I_1, \ldots, I_n\}$ is the set of possible worlds and $P : W \to (0, 1]$, $\sum_{I \in W} P(I) = 1$ is the probability distribution over these worlds. Because the data of individual worlds often considerably overlaps and it is sometimes even impossible to store them separately (e.g., if $|W| \to \infty$) a succinct representation has to be used.

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

In earlier approaches, alternatives of different attribute values are considered to be independent (e.g., [2]). In these models, each attribute value can be considered as a separate random variable with its own probability distribution. Newer models like Trio [6] or MayBMS [7] support dependencies by introducing new concepts like Trio's x-tuple and MayBMS's world set descriptor. For ease of presentation, we focus on duplicate detection in probabilistic data models without dependencies first, before considering x-tuples.

In general, tuple membership in a relation (uncertainty on tuple level) results from the application context. For example, a person can be stored in two different relations: one storing adults, the other storing people having a job. If we assume that the considered person is certainly 34 years old and jobless with a confidence of 90%, then the probability that a tuple t_1 representing this person belongs to the first relation is $p(t_1) = 1.0$, but the probability that a corresponding tuple t_2 belongs to the second relation is only $p(t_2) = 0.1$. Note that both tuples represent the same person despite the significant difference in probabilities. This illustrates that not tuple membership but only uncertainty on attribute value level should influence the duplicate detection process.

	name	**job**	$p(t)$
t_{11}	Tim	{machinist: 0.7, mechanic: 0.2}	1.0
t_{12}	{John: 0.5, Johan: 0.5}	{baker: 0.7, confectioner: 0.3}	1.0
t_{13}	{Tim: 0.7, Kim: 0.3}	mechanic	0.8

Fig. 3. probabilistic Relation $\mathcal{R}_1$

A. Duplicate detection in models without dependencies

Consider the probabilistic relation $\mathcal{R}_1$ as shown in Figure 3. The relation contains uncertainty on tuple level and attribute value level. Note that the person represented by tuple t_{11} is jobless with a probability of 10%. In the following, this notion of non-existence is denoted by $\perp$.

Since no dependencies exist, similarity can still be determined on an attribute-by-attribute basis. A non-existent value is definitely not similar with any existing one. Thus, we define $sim(\perp,\perp) = 1$ and $sim(a,\perp) = sim(\perp,a) = 0$ $(a \neq \perp)$. Assuming error-free data, the similarity of two uncertain attribute values a_1 and a_2 each defined in the domain D $(\hat{D} = \{D \cup \perp\})$ can be defined as the probability that both values are equal:

$$sim(a_1,a_2) = P(a_1 = a_2) = \sum_{d\in\hat{D}} P(a_1 = d, a_2 = d) \qquad (4)$$

In erroneous data, the similarity of domain elements has to be additionally taken into account:

$$sim(a_1,a_2) = \sum_{d_1\in\hat{D}}\sum_{d_2\in\hat{D}} P(a_1{=}d_1, a_2{=}d_2)\cdot sim(d_1,d_2) \qquad (5)$$

For instance, the similarity of t_{11}.name and t_{13}.name is either $sim(\text{Tim},\text{Tim}) = 1$ (with probability 0.7) or $sim(\text{Tim},\text{Kim}) = \alpha$ (with probability 0.3), where α depends on the chosen comparison function (e.g., $\alpha = 2/3$ if the normalized hamming distance is used).

Common decision models can be used without any adaption, because uncertainty is handled on the attribute value level and matching invariably results in a comparison vector $\vec{c}$.

B. Duplicate detection in models with x-tuples

To model dependencies between attribute values, the concept of x-tuples is introduced in the ULDB model of Trio [6]. An x-tuple t consists of one or more alternative tuples $(t^1,\ldots,t^n)$ which are mutually exclusive. The ULDB model does not support an infinite number of alternatives (e.g., uncertainty in a continuous domain). In these cases, and to avoid high numbers of alternatives, a probability distribution can sometimes still be associated with the attribute value. For example the value 'mu*' (see t_{21}^2.job) represents a uniform distribution over all possible jobs starting with the characters 'mu' (e.g., musician). *Maybe* x-tuples (tuples for which non-existence is possible, i.e., for which the probability sum of the alternatives is smaller than 1) are indicated by '?'. Relations containing one or more x-tuples are called x-relations. In the following, we consider a consolidation of the two x-relations $\mathcal{R}_2$ and $\mathcal{R}_3$ of Figure 4.

Principally, we consider the similarity of two x-tuples $t_1 = \{t_1^1,\ldots,t_1^k\}$ and $t_2 = \{t_2^1,\ldots,t_2^l\}$ as the expected similarity

	name	**job**	$p(t)$	
t_{21}	John	pilot	0.7	
	Johan	mu*	0.3	
t_{22}	Tim	mechanic	0.3	?
	Jim	mechanic	0.2	
	Jim	baker	0.4	

	name	**job**	$p(t)$	
t_{31}	John	pilot	0.8	
	Johan	pianist	0.2	
t_{32}	Tom	mechanic	0.8	?
t_{33}	John	$\perp$	0.2	?
	Sean	pilot	0.6	

Fig. 4. X-relations $\mathcal{R}_2$ (left) and $\mathcal{R}_3$ (right)

of their alternative tuples. Therefore, in the attribute value matching step, the attribute values of all alternative tuples of t_1 and all alternatives tuples of t_2 are pairwise compared. Since individual attribute values (e.g., t_{21}^2.job) can be uncertain, we use the formulas of Section IV-A. In this way, instead one single vector $\vec{c}$, $k \times l$ *comparison vectors* are obtained. Therefore, decision models for assigning the pair (t_1,t_2) to one of the sets M, P or U need to be adapted.

We define two approaches (see Figure 5). For each approach, the input consists of the considered x-tuple pair (t_1,t_2) and a *comparison matrix* containing the *comparison vector* of each alternative tuple pair (t_1^i,t_2^j). In the first approach (Figure 5, left side), the similarity of the x-tuples is based on the similarity of their alternative tuples ($\vartheta : \mathbb{R}^{k\times l} \to \mathbb{R}$). In the second approach (Figure 5, right side), it is derived from their matching results ($\vartheta : \{m,p,u\}^{k\times l} \to \mathbb{R}$).

In more detail, the first, more intuitive approach is based on the *similarity vector* $\vec{s}(t_1,t_2)$ containing the similarity of each alternative tuple pair (t_1^i,t_2^j) which is determined by $\varphi(\vec{c}_{ij})$ (Step 1). The final similarity $sim(t_1,t_2)$ results from a *derivation function* $\vartheta(\vec{s}(t_1,t_2))$ (Step 2). Ultimately, the x-tuple pair is classified into $\{M,U\}$ or $\{M,P,U\}$ by comparing $sim(t_1,t_2)$ with one or two thresholds (Step 3).

One adequate derivation is to calculate the expected value of the alternative tuple similarities $(\vartheta(\vec{s}(t_1,t_2)) = E(sim(t_1^i,t_2^j)))$. Since tuple membership is not relevant for duplicate detection, the probability of each alternative tuple t^i has to be normalized w.r.t. the probability of the corresponding x-tuple $(p(t^i)/p(t))$, where $p(t) = \sum_{j\in[1,n]} p(t^j)$. As a consequence, $E(sim(t_1^i,t_2^j))$ and hence the similarity of the two x-tuples t_1 and t_2 are defined as:

$$sim(t_1,t_2) = \sum_{i\in[1,k]}\sum_{j\in[1,l]} \frac{p(t_1^i)}{p(t_1)}\cdot\frac{p(t_2^j)}{p(t_2)}\cdot sim(t_1^i,t_2^j) \qquad (6)$$

Note that equations 5 and 6 are equivalent to the expected value of the corresponding similarity over all possible worlds.

Unfortunately, if the values resulting from Step 1 are not normalized, the expected value $E(sim(t_1^i,t_2^j))$ can become unrepresentative. For example, if the two alternative tuples t_1^i and t_2^j are similar to a large extent $(\varphi(\vec{c}_{ij}) \to \infty)$, the similarity $sim(t_1,t_2)$ becomes infinite, too, independent from the probability of these alternatives. As a consequence, this approach is more fitting for knowledge-based than for probabilistic techniques.

In the second approach, after calculating the similarity of all alternative tuple pairs (Step 1.1), each of these pairs is classified into $\{M,P,U\}$ (Step 1.2). From the resulting *matching vector* $\vec{\eta} = \{m,p,u\}^{k\times l}$, the similarity of the

```
Input: x-tuple pair ($t_1 = \{t_1^1, \ldots, t_1^k\},\ t_2 = \{t_2^1, \ldots, t_2^l\}$)
        comparison matrix ($\vec{c}(t_1, t_2) = [\vec{c}_{11}, \ldots, \vec{c}_{kl}]$)
1.  For $\vec{c}_{ij}$ of each pair of alternative tuples $(t_1^i, t_2^j)$
    1.1  Execution of the combination function $\varphi(\vec{c}_{ij})$
         ⇒ Result: $sim(t_1^i, t_2^j)$
    ⇒ Result: $\vec{s}(t_1, t_2) = [sim(t_1^1, t_2^1), \ldots, sim(t_1^k, t_2^l)] \in \mathbb{R}^{k \times l}$
2.  Execution of the derivation function $\vartheta(\vec{s}(t_1, t_2))$
    ⇒ Result: $sim(t_1, t_2)$
3.  Classification of $(t_1, t_2)$ into $\{M, P, U\}$ based on $sim(t_1, t_2)$
Output: Decision whether $(t_1, t_2)$ is a duplicate or not
```

```
Input: x-tuple pair ($t_1 = \{t_1^1, \ldots, t_1^k\},\ t_2 = \{t_2^1, \ldots, t_2^l\}$)
        comparison matrix ($\vec{c}(t_1, t_2) = [\vec{c}_{11}, \ldots, \vec{c}_{kl}]$)
1.  For $\vec{c}_{ij}$ of each pair of alternative tuples $(t_1^i, t_2^j)$
    1.1  Execution of the combination function $\varphi(\vec{c}_{ij})$
         ⇒ Result: $sim(t_1^i, t_2^j)$
    1.2  Classification of $(t_1^i, t_2^j)$ into $\{M, P, U\}$ based on $sim(t_1^i, t_2^j)$
         ⇒ Result: matching value $\eta(t_1^i, t_2^j) \in \{m, p, u\}$
    ⇒ Result: $\vec{\eta}(t_1, t_2) = [\eta(t_1^1, t_2^1), \ldots, \eta(t_1^k, t_2^l)] \in \{m, p, u\}^{k \times l}$
2.  Execution of the derivation function $\vartheta(\vec{\eta}(t_1, t_2))$
    ⇒ Result: $sim(t_1, t_2)$
3.  Classification of $(t_1, t_2)$ into $\{M, P, U\}$ based on $sim(t_1, t_2)$
Output: Decision whether $(t_1, t_2)$ is a duplicate or not
```

Fig. 5. General representations of decision models adapted to the x-tuple concept: approach 1 (left) and approach 2 (right)

corresponding x-tuples is derived (Step 2) and the tuple pair is assigned to one of the three sets M, P and U (Step 3).

The derivation function ϑ of Step 2 can be based on probability theory, e.g., by defining the tuple similarity as a kind of matching weight: $sim(t_1, t_2) = P(m)/P(u)$, where the two probabilities $P(m)$ and $P(u)$ are defined as:

$$P(m) = \sum_{(t_1^i, t_2^j) \in M} p(t_1^i) \cdot p(t_2^j) \qquad P(u) = \sum_{(t_1^i, t_2^j) \in U} p(t_1^i) \cdot p(t_2^j)$$

Since in this approach the similarity of two x-tuples is based on values defined in the discrete domain $\{m, p, u\}$, the x-tuple similarity is more imprecise than in the first approach. In contrast, in spite of unnormalized results of Step 1, cases of total unrepresentative similarity values can be avoided.

In summary, the first approach is more suitable for knowledge-based techniques (for example by calculating the expected certainty in Step 2) and the second one is more adequate for probabilistic techniques.

V. Conclusion

Since many applications naturally produce *uncertain* data, 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. However, an integration of *uncertain* (esp. *probabilistic*) source data has not been considered so far and hence is still an unexplored area of research.

In order to obtain concise integration results, duplicate detection is an essential activity. In this paper, we investigate how duplicates can be detected in *probabilistic* data.

We consider probabilistic data models representing uncertainty on tuple and attribute value level with and without using the x-tuple concept. We introduce methods for attribute value matching and decision models for both types of models.

In conclusion, this paper gives first insights in the large area of identifying duplicates in probabilistic databases. Individual subareas, e.g., search space reduction or detecting duplicates in complex *probabilistic* data, have to be examined in future reflections. Furthermore, for realizing an integration of *probabilistic* data: schema matching, schema mapping and data fusion have to be considered w.r.t. *probabilistic* source data.

References

[1] D. Suciu, A. Connolly, and B. Howe, "Embracing Uncertainty in Large-Scale Computational Astrophysics," in *MUD*, 2009, pp. 63–77.

[2] D. Barbará *et al.*, "The Management of Probabilistic Data," *IEEE Trans. Knowl. Data Eng.*, vol. 4, no. 5, pp. 487–502, 1992.

[3] N. Fuhr and T. Rölleke, "A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems," *ACM Trans. Inf. Syst.*, vol. 15, no. 1, pp. 32–66, 1997.

[4] R. Cavallo and M. Pittarelli, "The theory of probabilistic databases," in *VLDB*, 1987, pp. 71–81.

[5] M. van Keulen, A. de Keijzer, and W. Alink, "A Probabilistic XML Approach to Data Integration," in *ICDE*, 2005, pp. 459–470.

[6] M. Mutsuzaki *et al.*, "Trio-One: Layering Uncertainty and Lineage on a Conventional DBMS," in *CIDR, Asilomar, USA*, 2007, pp. 269–274.

[7] J. Huang *et al.*, "MayBMS: a probabilistic database management system," in *SIGMOD Conference*, 2009, pp. 1071–1074.

[8] J. Boulos *et al.*, "MYSTIQ: a system for finding more answers by using probabilities," in *SIGMOD Conference*, 2005, pp. 891–893.

[9] F. S.-C. Tseng, A. L. P. Chen, and W.-P. Yang, "Answering Heterogeneous Database Queries with Degrees of Uncertainty," *Distributed and Parallel Databases*, vol. 1, no. 3, pp. 281–302, 1993.

[10] X. L. Dong, A. Y. Halevy, and C. Yu, "Data integration with uncertainty," *VLDB J.*, vol. 18, no. 2, pp. 469–500, 2009.

[11] M. van Keulen and A. de Keijzer, "Qualitative effects of knowledge rules and user feedback in probabilistic data integration," *VLDB J.*, vol. 18, no. 5, pp. 1191–1217, 2009.

[12] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *VLDB J.*, vol. 10, no. 4, pp. 334–350, 2001.

[13] M. A. Hernández, R. J. Miller, and L. M. Haas, "Clio: A Semi-Automatic Tool For Schema Mapping," in *SIGMOD Conference*, 2001, p. 607.

[14] A. K. Elmagarmid *et al.*, "Duplicate Record Detection: A Survey," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 1, pp. 1–16, 2007.

[15] J. Bleiholder and F. Naumann, "Data fusion," *ACM Comput. Surv.*, vol. 41, no. 1, 2008.

[16] F. Panse, M. van Keulen, A. de Keijzer, and N. Ritter, "Duplicate Detection in Probabilistic Data," Enschede, Technical Report TR-CTIT-09-44, Dec. 2009.

[17] O. Benjelloun *et al.*, "Swoosh: a generic approach to entity resolution," *VLDB J.*, vol. 18, no. 1, pp. 255–276, 2009.

[18] I. Fellegi and A. Sunter, "A Theory for Record Linkage," *Journal of the American Statistical Association*, vol. 64, p. 11831210, 1969.

[19] M. A. Hernández and S. J. Stolfo, "The Merge/Purge Problem for Large Databases," in *SIGMOD Conference*, 1995, pp. 127–138.

[20] M. Weis and F. Naumann, "Detecting Duplicates in Complex XML Data," in *ICDE*, 2006, p. 109.

[21] H. Müller *et al.*, "Problems, Methods, and Challenges in Comprehensive Data Cleansing," Humboldt Universitt Berlin, Tech. Rep., 2003.

[22] C. Batini *et al.*, *Data Quality: Concepts, Methodologies and Techniques*, ser. Data-Centric Systems and Applications. Springer, 2006.

[23] M. Magnani and D. Montesi, "Uncertainty in data integration: current approaches and open problems," in *MUD, Vienna, Austria*, ser. CTIT Workshop Proceedings, no. WP07-08, Sept. 2007.