Analytics over Probabilistic Unmerged Duplicates

Ekaterini Ioannou and Minos Garofalakis
Technical University of Crete, Chania, Greece
{ioannou, minos}@softnet.tuc.gr

Abstract. This paper introduces probabilistic databases with unmerged duplicates (DB\textsuperscript{ud}), i.e., databases containing probabilistic information about instances found to describe the same real-world objects. We discuss the need for efficiently querying such databases and for supporting practical query scenarios that require analytical or summarized information. We also sketch possible methodologies and techniques that would allow performing efficient processing of queries over such probabilistic databases, and especially without the need to materialize the (potentially, huge) collection of all possible deduplication worlds.

1 Introduction

Entity Deduplication is the task of processing a data set in order to create entities by merging the data set instances that describe the same real-world objects. Traditional deduplication techniques \cite{4} are based on an a-priori merging of instances: they first detect the possible matches between instances, and then, given a threshold, decide which instances to merge into entities. The entities resulting from the merges are then used for replacing the coreference instances in the original data set. Query processing is performed over the updated data set.

To handle the new resolution challenges, the recently introduced approaches (e.g., \cite{1, 6, 9}) moved towards databases that maintain and incorporate unmerged duplicates. These approaches perform only the first part of the resolution process, which is the identification of the possible matches between the instances. This is the deduplication information, and it corresponds to a set of possible linkages between instances. In some approaches each linkage is accompanied with a probability that reflects the belief of the deduplication technique that the specific two instances describe the same real-world object. The resulting information is not used for performing entity merges (using a given threshold), but is stored alongside the original data. The complete deduplication is performed during query processing, and thus answers reflect the different real-world situations that are encoded in the deduplication information. In case the deduplication information is probabilistic, as for instance in \cite{1} and \cite{6}, then the probabilities are used for computing the overall probability of each query answer.

Although answering queries over unmerged duplicates is important, it is still just a first step towards a complete solution to the problem. The typical situation is that the unmerged duplicates are part of a large database that of course contains other tables. Consequently, users would require retrieving information related to all data in the database, duplicated or not. However, this would require generating and considering all the possible worlds, which is typically huge \cite{2} and will overwhelm the user instead
of providing useful information. In addition, users might not care about the exact entities but rather on obtaining insights through analytical and summarizing queries, as for example performed in the online analytical processing.

In this paper, we introduce DB\textsuperscript{ud}: a database containing probabilistic information about instances found to describe the same real-world objects. DB\textsuperscript{ud} adopts the most expressive form of deduplication information (i.e., probabilistic linkages between instances – also accounting for transitivity), and significantly extends its scope by considering the deduplication information as part of a database with other tables providing entity-related data. In the following sections, we first introduce analytical queries for retrieving information of the entities in DB\textsuperscript{ud} (Section 2), and then sketch possible methodologies and techniques for efficiently processing queries over such a probabilistic database with unmerged duplicates (Section 3).

## 2 Modeling Data and Queries

A probabilistic database with unmerged duplicates DB\textsuperscript{ud} contains deterministic relational tables $T_1, ..., T_n$ as well as tables with duplicates $R_1, ..., R_k$, i.e., some instances of $R_i$ describe the same real-world objects. The deduplication information for table $R_i$ is given in table $L_i$. More specifically, $L_i$ contains probabilistic linkages over the instances in $R_i$: $l_{r_\alpha, r_\beta} \in L_i$ means that instances $r_\alpha$ and $r_\beta$ from $R_i$ describe the same real-world object with probability $p_i$.

![Fig. 1. A fragment of a probabilistic database with unmerged duplicates](image-url)

To process queries over DB\textsuperscript{ud} we must be able to support joins between the tables with unmerged duplicates and the deterministic tables. For example, answering queries over the DB\textsuperscript{ud} fragment shown in Figure 1 requires considering the join between table Buyer with Order. Since table Buyer contains duplicates, we must first derive the possible entities using the deduplication information provided in table Deduplication. Each linkage from the Deduplication table can be either accepted or rejected, e.g., we can accept $l_{r_1, r_3}$ with probability 0.55 or reject it with probability (1-0.55). Rejecting the linkage means that the database has two entities, one for each of the instances. Accepting the linkage implies a new entity, with identifier $e_{1,3}$, that replaces both $r_1$ and $r_3$. Creating a single entity given these two instances maybe performed using different semantics. For example, if we assume that we keep the instance with the highest value on the year attribute, the tuple for the merge between instances $r_1$ and $r_3$ is $(e_{1,3}, “Mary”, “Smith”, “DE”, “female”, “2011”).

For creating the possible entities of a table with unmerged duplicates $R_i$ we need to consider the acceptance and rejection of each linkage of $L_i$. Deciding which linkages from $L_i$ (e.g., table Deduplication from Figure 1) to accept or reject leads to a huge