LinkDB: A Probabilistic Linkage Database System

Ekaterini Ioannou
Technical Univ. of Crete
ioannou@softnet.tuc.gr

Wolfgang Nejdl
L3S Research Center
nejdl@L3S.de

Claudia Niederée
L3S Research Center
niederee@L3S.de

Yannis Velegrakis
University of Trento
velgias@disi.unitn.eu

ABSTRACT

Entity linkage deals with the problem of identifying whether two pieces of information represent the same real world object. The traditional methodology computes the similarity among the entities, and then merges those with similarity above some specific threshold. We demonstrate *Link*DB, an original entity storage and querying system that deals with the entity linkage problem in a novel way. *Link*DB is a probabilistic linkage database that uses existing linkage techniques to generate linkages among entities, but instead of performing the merges based on these linkages, it stores them alongside the data and performs only the required merges at run-time, by effectively taking into consideration the query specifications. We explain the technical challenges behind this kind of query answering, and we show how this new mechanism is able to provide answers that traditional entity linkage mechanisms cannot.

Categories and Subject Descriptors

H.2.m [**Database Management**]: Miscellaneous - linkage

General Terms

Algorithms

Keywords

Data Integration, Entity Linkage, Entity Resolution

1. INTRODUCTION

A major challenge during integration and cleaning of heterogeneous data is detecting whether two pieces of information correspond to the same real world object. The literature contains a plethora of techniques with effective and efficient solutions to this challenge [3]. Typically, these techniques compute some similarity between the data, and merge the data with a similarity exceeding a pre-decided threshold. This process is performed offline. At run-time, query answering is simply performed over the merged data.

Unfortunately, this traditional approach does not always yield the best results. Its success depends heavily on the right choice on the similarity metric and the careful selection of the threshold value, which remain the same for the whole process. This static approach to the problem leads to some serious limitations in scenarios involving Web data or integrated data that is characterized by large levels of heterogeneity, noise, missing values, uncertainty, and a high rate of information evolution [3]. A single threshold value may be sufficient for some groups of data but not for others.

In an effort to deal with these limitations, we have developed *Link*DB that is based on an idea we have recently presented [5] and aims at bringing together two worlds: the world of entity linkage, and that of probabilistic database. Actually, *Link*DB is the first *probabilistic linkage database system*. The novelty of *Link*DB is that it uses a generic entity-based representation model for highly heterogeneous data that supports the simultaneous representation of possible linkages (i.e., stating that two entities describe the same real world object) alongside the original data. This means that no data merging is performed in advance, but the outcome of the entity linkage algorithms, i.e., the pairs of entities possibly representing the same real world object with the belief of that being true, are stored in the data. The outcome is a database that contains uncertainty not only on the attributes of the entities, but also on their linkages. At query time, *Link*DB analyzes the query and based on its conditions generates the answers. These answers may have been produced by considering entity merges on-the-fly. As such, *Link*DB is able to generate answers that may not exist in the database but inferred through the entity merges, which is one of the characteristic features of the system. And since linkages are coming from the linkage algorithms with some degree of belief, answers will naturally be probabilistic.

Probabilistic databases have been used as a way to deal with the inherited uncertainty and heterogeneity of integrated or Web data. A probabilistic database is a database in which every attribute value (or tuple) comes with some probability reflecting the belief that the fact it represents is true. The limitation of probabilistic databases is that they consider uncertainty only at the tuple and the attribute level. Agrawal et al. [1] is an example of an approach that uses uncertainty to represent the possible alternative values of a single attribute. Andritsos et al. [2] used uncertainty at the tuple level to capture the records that could describe the same real world objects, a work that is closely related to entity linkage. However, the various possible records are mutually exclusive and there is no notion of merging. *Link*DB takes the idea of probabilistic databases further. It extends it to consider uncertainty at the linkage level and brings additional challenges to the existing querying mechanisms.

The remaining paper is structured as follows. Section 2 describes a motivating example. Section 3 provides the technical foundation and description of *Link*DB, and Section 4 describes the demonstration.

SIGMOD'11, June 12–16, 2011, Athens, Greece.
ACM 978-1-4503-0661-4/11/06.

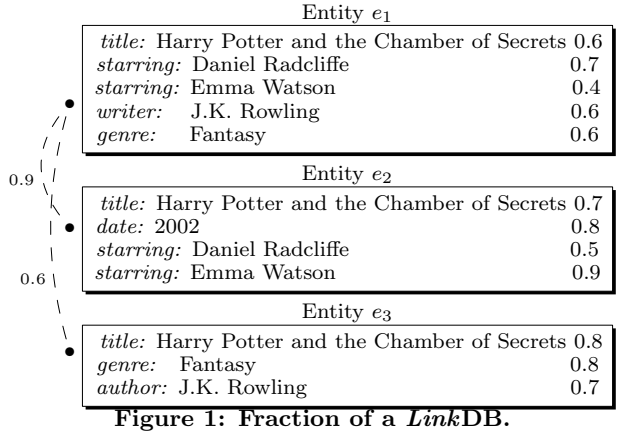

Figure 1: Fraction of a *Link*DB.

2. MOTIVATING EXAMPLE

Consider an entity-based database representing a materialization of data collected by multiple heterogeneous data sources. Each entity consists of an identifier and a number of attribute name-value pairs that describe the properties of the real world object the entity represents. A small fraction of the database is illustrated in Figure 1. It contains three entities: e_1, e_2, and e_3. Since these entities have been collected from multiple distributed sources, they may contain incomplete, outdated, or inconsistent information, thus, the attributes of each entity are coming with some degree of uncertainty. This uncertainty is reflected by a probability value that annotates each attribute. To associate data collected across the different sources, an entity linkage mechanism is run on this data. Due to the lack of global schemas and universal keys, the entity linkage algorithm employs structural and semantic similarities to reach a decision. The three entities seem to refer to the same episode of Harry Potter. Yet, the first may be modeling the movie, the second the DVD, and the third the corresponding book. Based on their similarities, the entity linkage algorithm returns that with some high confidence entities e_1 and e_2 refer to the same real world object, and with some lower confidence so do entities e_1 and e_3. On the figure, this outcome is modeled through the dotted lines across the respective entities and their numeric annotations.

Let us now consider the query "writer='J.K. Rowling' and date='2002'". As can be seen from Figure 1, none of the three entities satisfies both conditions. Therefore, if the integration system did not perform any entity linkage and merging process, or had a threshold of 0.95, an empty set would have been the answer to this query. If, on the other hand, an entity linkage mechanism was in place and its results had been used to merge data based on a 0.7 threshold, then entities e_1 and e_2 would have been merged, and e_3 would have remained independent. Merging two or more entities means replacing them in the database by a new entity that contains as attributes the union of their respective attribute sets. In the merged database, the system would have returned one answer to the above query consisting of only the merged entity of e_1 with e_2. If instead of 0.7, the threshold was 0.5, then e_3 would have also be merged with the e_1 and e_2 forming one big entity. The answer to the above query would still be one entity, but this time a different (richer) one, the one resulting from the merging of e_1, e_2, and e_3. Thus, in a traditional entity linkage setting, any of the three situations may happen, but in a mutually exclusive way.

We believe that there are no strong evidences to argue towards any of the three possible thresholds and this is a situation that is met very often in practice. As such, we advocate that we should accept that all three situation are possible options. The first situation is the one in which none of the two linkages suggested by the linkage algorithm is accepted. Since the first linkage has a probability 0.9 and the second 0.6, the probability of neither of them being true is (1-0.9)×(1-0.6)=0.04. The probability of the linkage between e_1 and e_2 to be true, but not the one between e_1 and e_3, is 0.9×(1-0.6)=0.36. On top of this probability, we need to add the probability of the attributes, i.e., the entity resulting from the merging of e_1 and e_2 has the attributes to satisfy the two query conditions, but this is conditional since it is not 100% sure that these attributes exist (each has some probability). Similar observations can be made for the third case.

Our position is that a system should perform no merges in advance and simply keep the computed linkages. The system should return as results for the above query all these possible answers, each one with the respective confidence (probability). Note that since no merge action takes place in advance, the system will be returning answers that may not appear in the database, i.e., they are inferred based on the respective merges. Opting for no merges in advance comes with the additional advantage that it deals with dynamic data. One of the main challenges in entity linkage is to avoid remaking or changing merges that have already taken place when data is modified. Our approach is able to overcome this issue, since merges are computed and executed on-the-fly at query time.

3. TECHNICAL CHALLENGES

One of the data models that is increasingly gaining popularity is the entity-based data model which forms the basis of Dataspaces [4]. We adopt in *Link*DB the same model. The world is modeled through a set of entities, each one partially describing a specific real world object. An entity e is a tuple $\langle id, A\rangle$ where id is the entity's identifier and A is a finite set of attributes. Each attribute provides some characteristic of the entity, and it is given by a pair $\langle n,v\rangle$ where n is the name and v the value of this attribute. In addition, an l_{e_1,e_2} denotes that entity e_1 is linked to entity e_2 as defined by an entity linkage technique. Each l_{e_1,e_2} is accompanied with a probability that indicates the belief of the entity linkage technique for the specific linkage. A set of linked entities is called a linkage assignment.

Def. 1 (Probabilistic Linkage Database). *A probabilistic Linkage Database is a tuple $\langle \mathcal{E},\mathcal{L}, p^a,p^l\rangle$, where $\mathcal{E}$ is a set of entities, $\mathcal{L}$ is a linkage assignment on $\mathcal{E}$, and p^a, p^l are attribute and linkage probability assignment functions respectively. In particular, $p^l|\mathcal{L}\mapsto[0,1]$ and $p^a|B\mapsto[0,1]$ with $B=\{a \mid \exists\langle id,A\rangle\in\mathcal{E} \wedge a\in A\}$.* ∎

Since the linkages in a probabilistic linkage database have probabilities, to answer a query over such a database one needs to consider the different situations that may exists. This means deciding whether each of the linkages between two entities is actually true or not. Taking such a decision for each linkage, creates one possible situation. We call such a situation a possible l-world. Note that the possible l-world

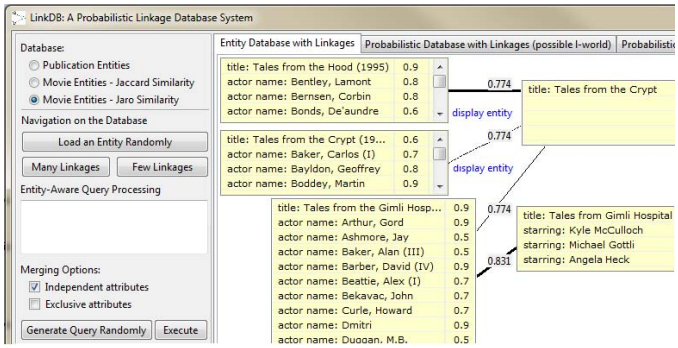

Figure 2: Part of a *Link*DB interface.

does not any more have probabilities on the linkages. It only has present the linkages that were selected. The rest have been removed. Given a possible l-world, one can perform the merging of the entities that have linkages present between them. The outcome is a new database with no linkages but probabilities on the attributes. This is actually a probabilistic database, i.e., a database with probabilities on the attributes. Deciding whether each such attribute actually is present or not, means creating a possible world, exactly as it has been defined in the area of probabilistic databases. Each l-world exists with some probability. The probability depends on the probabilities of the original linkages that are present in the specific world. Accordingly, each possible world has a probability of existence which depends on the probabilities of the attributes that are (and are not) present in the specific world.

The answer of a query over a probabilistic linkage database is the union of the answer of the query over each possible world of every possible l-world of the database. Since the l-worlds and the possible worlds exist with some probability, each element in that answer set will be returned with a probability. That probability will be the product of the probabilities of its l-world and its possible world.

Efficient Entity-Aware Query Processing. The naive approach for answering a query over a probabilistic linkage database is performed by computing all the possible l-worlds and all their possible worlds, and then execute the query over them. This is clearly not practical since the number can easily become exponential. We have instead developed an efficient algorithm, details can be found elsewhere [5], which starts from the query to identify and perform only the required merges. The algorithm uses the notion of a linkage assignment, i.e., the decision on what linkages to accept for each possible l-world. Off-line, we use the different linkage assignments to generate factors, which represent a maximal group of pairwise linked entities. For example, from $\{l_{e_1,e_2}, l_{e_1,e_3}, l_{e_4,e_5}\}$ we generate factors $\{e_1,e_2,e_3\}$ and $\{e_4,e_5\}$. These factors along with their entities are used for building an index, allowing us to efficiently detect the factor of an entity.

Given a query, we use the created factor index to efficiently detect which entities should be merged in order to provide a valid answer to the query. These entity merges give us the linkage assignments that should be considered, and from these assignments we can retrieve the possible l-worlds. Then, the probability of each possible l-world is computed. Finally, the possible worlds of each l-world are generated alongside their own probability, which is combined with the probability of the respective l-world, and then included in the answer set of the query.

4. DEMONSTRATION HIGHLIGHTS

The demonstration (part of the interface shown in Figure 2) will aim at communicating to the audience four main messages: (i) that the entity linkage on heterogeneous, incomplete, and volatile data can not be effectively managed using the current norma of a preselected threshold; (ii) that the suggested representation based on entities and linkages is suitable for modeling the entity linkage problem; (iii) that the on-the-fly entity-aware query processing is both effective and efficient, and generates answers that are more natural for highly heterogeneous data collections; and (iv) that the theory of probabilistic databases can be easily extended to support probabilistic linkages.

The suggested demonstration consists of two main parts. The focus of the first is to highlight the entity linkage challenges over the modern highly heterogeneous data, and the limitations of the way the problem is typically managed. This will be illustrated using the navigation functionality and three different *Link*DB datasets. The first one is a collection of publications from the popular CiteSeer website, with linkages across authors generated using an entity linkage algorithm [6]. The second and the third contain $13,435$ movies coming from *IMDb* and *DBpedia*, with the entity linkages generated using the *Jaccard* and *Jaro* similarity techniques, respectively. We will show examples that no matter what threshold we choose, undesired merges occur.

The second part of the demonstration focuses on the illustration of the functionality and performance provided by *Link*DB through its efficient entity-aware query processing. It involves answering queries over the three datasets mentioned above. We will explain how having different type of correlations between attributes modifies the resulted entities, and also explain the factors that affect the time required for processing queries. We will explain the basic principles of our solution and how we did overcome the technical challenges for an efficient query processing. In addition, we will illustrate how query processing is affected by the different characteristics of the databases, such as the number of entities, the number of attributes, the probabilities on the attributes, the existing linkages, the structure that these linkages have, etc. Users will be able to pose their own queries and explanations will be provided for the returned results when needed.

5. REFERENCES

[1] P. Agrawal, O. Benjelloun, A. D. Sarma, C. Hayworth, S. U. Nabar, T. Sugihara, and J. Widom. Trio: A system for data, uncertainty, and lineage. In *VLDB*, pages 1151–1154, 2006.

[2] P. Andritsos, A. Fuxman, and R. J. Miller. Clean answers over dirty databases: A probabilistic approach. In *ICDE*, 2006.

[3] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate Record Detection: A Survey. *TKDE*, 19(1):1–16, 2007.

[4] A. Y. Halevy, M. J. Franklin, and D. Maier. Principles of dataspace systems. In *PODS*, pages 1–9, 2006.

[5] E. Ioannou, W. Nejdl, C. Niederée, and Y. Velegrakis. On-the-fly entity-aware query processing in the presence of linkage. *PVLDB*, 3(1):429–438, 2010.

[6] E. Ioannou, C. Niederée, and W. Nejdl. Probabilistic entity linkage for heterogeneous information spaces. In *CAiSE*, pages 556-570, 2008.