PEPX: A Query-Friendly Probabilistic XML Database

Te Li, Qihong Shao, and Yi Chen
Dept. of Computer Science and Engineering, Arizona State University
Tempe, AZ, USA
{te.li, qihong.shao, yi}@asu.edu

Categories and Subject Descriptors
H.2.4 [**Systems**]: Query processing

General Terms
Design, Performance

Keywords
XML, probabilistic data models, query processing

1. INTRODUCTION

Uncertain data widely exists in various applications due to several reasons. First, data generated by automated information extraction and data analysis tools is error-prone [5]. Second, data integrated from various sources is often uncertain or even conflicting, depending on the trustworthiness of its sources and the quality of the data mapping procedures [9]. Furthermore, experimental results and sensor data are observed in conditions that may be subject to a range of variability in natural environment, mechanical defects, contaminated samples, etc. [3].

The abundance of uncertain data demands an effective database management system that records our confidence about the data as probabilities, supports efficient query evaluation, and provides confidence indicators for query results. Toward this goal, a lot of work has been done to manage uncertain data in relational databases. Recently, there is a growing interest in designing XML-based models to represent uncertain data due to the following reasons [8, 9]. First, XML can represent data uncertainty of different levels more naturally and succinctly compared with the relational data models. Second, the semi-structure and flexibility of XML data model fits well in applications such as information extraction and data integration where data is often uncertain.

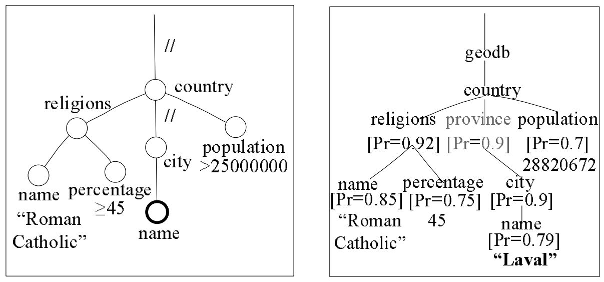
Several XML-based probabilistic data models of different expressiveness have been proposed [8, 4, 9]. One common feature of those models is that the probability of a node is assigned conditioned on the existence of its parent node. Naturally, a node may exist only if its parent node exists, therefore these models present a good conceptual representation for uncertain data. However, as we will see, this intuitive model may incur big overhead on query processing performance.

Existing XML query processing techniques [2] only need to retrieve and process related data nodes to find query results. Nevertheless, using current probabilistic XML data models, it is inevitable to access all the ancestors of the XML nodes in a pattern match to the query in order to compute the probability of the query result. Therefore, the benefits of the existing XML query evaluation techniques can not be fully leveraged.

For example, suppose that a geographer would like to find out city names in a geographic XML database using an XPath query `/geodb/country/province/city/name`. To process this query on regular XML data, we can apply existing XML query processing techniques [2], such that query results can be retrieved without accessing the intermediate nodes with tag `geodb`, `country`, `province` and `city`. However, in order to calculate the probability of a query result, existing probabilistic XML data models need to retrieve all these intermediate nodes.

The problem becomes more severe if we extend the existing models for probabilistic XML data to support queries containing descendant axes. Let us consider another XPath query `/geodb//city/name`. To calculate the probability of a query result using the existing data models, we need to walk the data tree from the node matching `name`, retrieve the probabilities of all its ancestors: `city`, `province`, `country`, till the root `geodb`. A widely used technique [6, 2, 1], which efficiently handles descendant axes using an interval-based labeling scheme without accessing intermediate data nodes with tag `province` and `country`, can not be fully exploited to achieve efficiency.

In this paper, we argue that physical models are needed in addition to the existing conceptual models for probabilistic XML data in order to achieve query evaluation efficiency. We propose a physical model for probabilistic XML data which proactively materializes part of probability calculation to speed up query evaluation. This model is named as *PEPX*, Probability Encoded Probabilistic XML. We design an efficient query evaluation algorithm for this data model that does not require extra node accesses to compute the probabilities of query results by leveraging the encoding. An algorithm is proposed to maintain the probability encoding upon updates. Experimental results demonstrate the efficiency of our approach. The proposed physical model for probabilistic XML data can be adapted to existing conceptual models [8, 4, 9] to improve query evaluation performance without sacrificing expressiveness.



(a) Q (b) M

Figure 1: Query Q and its Pattern Match M

2. THE PEPX DATA MODEL AND QUERY EVALUATION

To model uncertain XML data, we associate a probability $Prob(n)$ with each node n in an XML tree, where $0 \le Prob(n) \le 1$. $Prob(n)$ denotes our confidence about the existence of node n, as well as the correctness of n's value if n is a leaf node.

To evaluate a tree pattern query on probabilistic XML data, we not only need to search query results, but also compute the probability of each result. Suppose that a geographer would like to find out the names of the cities in a country which has a population larger than 25,000,000 and at least 45% of the population are Roman Catholic believers. This can be expressed as an XPath query Q: `//country[religions[name="Roman Catholic"][percentage>=45]][population>25000000]//city/name`. Q can be presented as the query tree in Figure 1(a).

Consider a pattern match M to Q as shown in Figure 1(b). Each node in dark color has a corresponding node in the query tree, while the nodes in grey color are obtained from the XML data such that all the nodes in a pattern match are connected through parent-child relationship up to the root of the data tree. The Pr associated with a node n in M records the probability of the existence of n: $Prob(n)$. A node with a missing probability can be assumed to have a default probability value of 1.

The probability of pattern match M is computed as follows [1].

$$\begin{aligned} Prob(M) &= Prob(name)Prob(percentage)Prob(population) \\ &\quad Prob(name)/(Prob^2(country)Prob(religions)) \\ &= \frac{0.85 \times 0.75 \times 0.7 \times 0.79}{1^2 \times 0.92} = 0.38 \end{aligned}$$

In general, the probability of a pattern match can be computed according to the following theorem.

Theorem 1. *Given a pattern match M, let B be the set of branch nodes and L be the set of leaf nodes in M. The probability of M can be computed from the probabilities of nodes in M as follows:*

$$Prob(M) = \frac{\prod_{l \in L} Prob(l)}{\prod_{b \in B} Prob^{C(b)-1}(b)},$$

where $C(b)$ returns the number of children of node b.

[1]We use the tag of a node to refer that node for illustration purpose.

In contrast, previous work on probabilistic XML data models [8, 4, 9] record the *conditional probability* of the existence of an XML node n (n is not the root) given the existence of its parent node $\rho(n)$, i.e., $Prob(n|\rho(n))$. To compute the probability of pattern match M in Figure 1(b), we need to access all the data nodes and multiply their probabilities.

As we can see, the efficiency of PEPX compared with previous work are two-fold. First, PEPX requires fewer disk accesses. According to Theorem 1, to compute the probability of a pattern match, PEPX needs to access only the data nodes that match the branch or leaf nodes in the query tree. This gives us an opportunity to exploit existing XML query optimization techniques [6, 2, 1]. In contrast, previous work on probabilistic XML data requires accessing all the nodes in a pattern match. Second, PEPX requires fewer unit operations (multiplication, power, division) in probability computation. The number of unit operations that PEPX performs for each pattern match is equal to the number of leaf and branch nodes in the query tree; while the number of unit operations in previous approaches is equal to the number of nodes in the pattern match.

3. IMPLEMENTATION AND CONCLUSION

We have implemented a prototype of PEPX, which supports tree pattern query processing and update maintenance. To evaluate the effectiveness of the PEPX system, we compared its performance with ProTDB [8]. Experimental results show substantial performance speedup of PEPX, especially for queries containing descendant axes. More details of the PEPX system and experimental evaluation can be found in [7].

PEPX represents a physical model, which leverages an encoding of node probabilities to achieve efficient probability computation for query results. This model can be adapted to existing conceptual models for probabilistic XML data to improve query evaluation performance without affecting the expressiveness.

4. REFERENCES

[1] N. Bruno, N. Koudas, and D. Srivastava. Holistic twig joins: Optimal XML pattern matching. In *SIGMOD*, 2002.

[2] Y. Chen, S. Davidson, and Y. Zheng. BLAS: An Efficient XPath Processing System. In *SIGMOD*, 2004.

[3] M. Garofalakis, K. P. Brown, M. J. Franklin, J. M. Hellerstein, D. Z. Wang, E. Michelakis, L. Tancau, E. Wu, S. R. Jeffery, and R. Aipperspach. Probabilistic data management for pervasive computing: The data furnace project. *Data Engineering Bulletin*, 29(1), March 2006.

[4] E. Hung, L. Getoor, and V. S. Subrahmanian. PXML: A Probabilistic Semistructured Data Model and Algebra. In *ICDE*, 2003.

[5] T. Jayram, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, and H. Zhu. Avatar information extraction system. *Data Engineering Bulletin*, 29(1), March 2006.

[6] Q. Li and B. Moon. Indexing and Querying XML Data for Regular Path Expressions. In *VLDB*, 2001.

[7] T. Li, Q. Shao, and Y. Chen. PEPX: A Query-Friendly Probabilistic XML Database. Technical Report TR-06-016, Arizona State University, 2006.

[8] A. Nierman and H. V. Jagadish. ProTDB: Probabilistic Data in XML. In *VLDB*, 2002.

[9] M. van Keulen, A. de Keijzer, and W. Alink. Probabilistic XML Approach to Data Integration. In ICDE, 2005.