Querying and Managing Probabilistic XML

ProApproX: A Lightweight Approximation Query Processor over Probabilistic Trees

Asma Souihli Pierre Senellart

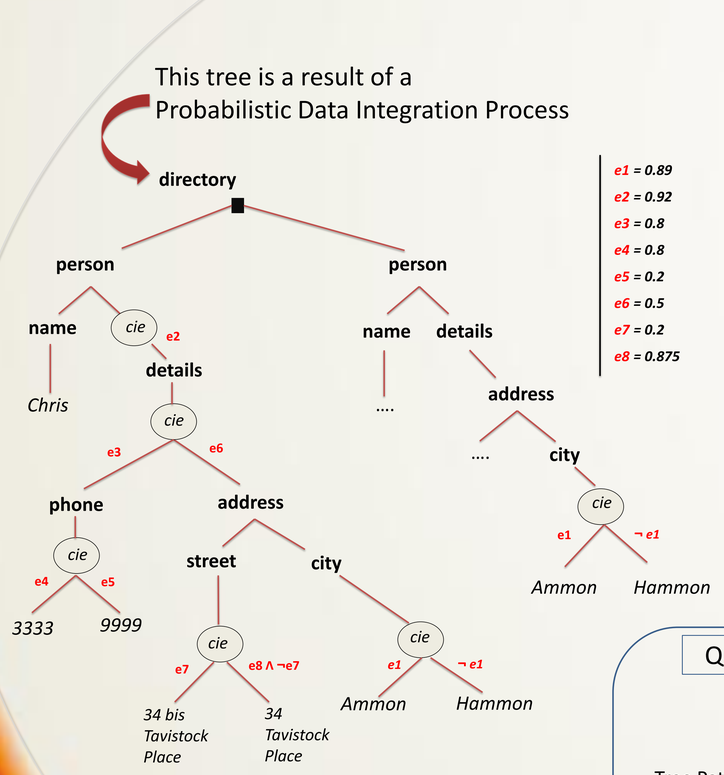
ProApproX is a query processor over probabilistic trees that represents a first step towards building a fully-fletched probabilistic semi-structured data management system

It relies on:

- A generalization of the different uncertain data models in XML
- Allows for efficient data querying with a subset of the Xpath query language
- Through techniques of exact calculations or efficient approximations of the result

Probabilistic Data

This tree is a result of a Probabilistic Data Integration Process

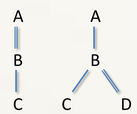


XML

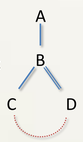
```
<directory>
<person >
  <name> chris </name>
     <details>
        <phone prob="e4">3333</phone>
        <phone prob="e5"> 9999</phone>
        <address>
            <street prob="e7">3333</street>
            <street prob="e8  -e7">9999</street>
            <city prob="e1">Ammon</city>
            <city prob="-e1">Hammon</city>
        </address>
     </details>
</person>

<person>
.....
</directory>
```

Querying P-Documents

Tree Pattern Queries (TPQ)



Tree Pattern Queries with joins (TPQJ)



Probability of a Query

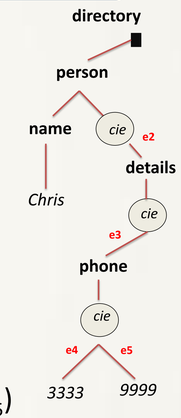
Q1: /person[name="Chris"]//phone/text()

- How to compute the **probability of the YES answer** to this query**?**
- Lineage to the query (probabilistic path of each answer)**: a DNF**

Phone 1 : $e_2 \wedge e_3 \wedge e_4$ ➡ Clause C_1

Phone 2 : $e_2 \wedge e_3 \wedge e_5$ ➡ Clause C_2

$\mathbf{F} = (e_2 \wedge e_3 \wedge e_4)$ **V** $(e_2 \wedge e_3 \wedge e_5)$

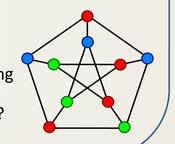


Complexity

Find and sum the probabilities of the satisfying assignments for the **DNF** (lineage formula) : #P-Hard problem

- No polynomial time algorithm for the exact solution **if P≠NP**
- #P problems ask "how many" rather than "are there any"

How many graph coloring using k colors are there for a particular graph G?



Approximations

When a Linear Computation is not possible, we run an appropriate approximation:

- Naïve Monte Carlo sampling for additive app. :
  Linear but could take exponentially many samples to converge to a good approximation for low probabilities
- Biased Monte Carlo sampling for multiplicative app. :
  Running time grows in $O(n^3 \ln n)$ in the number of clauses
- Self-Adjusting Coverage Algorithm for the DNF probability problem:
  For a fixed error ε and a fixed reliability factor δ, the algorithm is Linear in the length of *F* times $\frac{\ln(1/\delta)}{\varepsilon^2}$