Probabilistic/Uncertain Data Management -- IV

A Restricted Formalism: Explicit Independent Tuples

**Tuple independent probabilistic database**

\[
\text{TUP} = \{t_1, t_2, \ldots, t_M\} = \text{all tuples}
\]

\[
\text{pr : TUP} \rightarrow [0,1]
\]

\[
\Pr(I) = \prod_{t \in I} \text{pr}(t) \times \prod_{t \notin I} (1 - \text{pr}(t))
\]

\[
\text{INST} = \mathcal{P}(\text{TUP})
\]

\[
N = 2^M
\]

No restrictions
Tuple Prob. ⇒ Possible Worlds

\[
\begin{align*}
J &= \{ \text{possible worlds} \} \\
\|\mathcal{P}\| &= 1 \\
\mathbb{E}[\text{size}(\mathcal{P})] &= 2.3 \text{ tuples}
\end{align*}
\]

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>(\mathbb{P}(\cdot))</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>0.8</td>
</tr>
<tr>
<td>Sue</td>
<td>Boston</td>
<td>0.6</td>
</tr>
<tr>
<td>Fred</td>
<td>Boston</td>
<td>0.9</td>
</tr>
</tbody>
</table>

\[
\sum = 1
\]
Tuple-Independent DBs are Incomplete

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>$pr$</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>$p_1$</td>
</tr>
<tr>
<td>Sue</td>
<td>Seattle</td>
<td>$p_2$</td>
</tr>
</tbody>
</table>

$p_1 p_2 = I_p$

Very limited – cannot capture correlations across tuples

**Not Closed**

- Query operators can introduce complex correlations!
Query Evaluation on Probabilistic DBs

• Focus on possible tuple semantics
  – Compute likelihood of individual answer tuples
• Probability of Boolean expressions
  – Key operation for Intensional Query Evaluation
• Complexity of query evaluation
Complexity of Boolean Expression Probability

**Theorem** [Valiant:1979]
For a boolean expression E, computing Pr(E) is \#P-complete

NP = class of problems of the form “is there a witness?” SAT
\#P = class of problems of the form “how many witnesses?” \#SAT

The decision problem for 2CNF is in PTIME
The counting problem for 2CNF is \#P-complete
Query Complexity

Data complexity of a query $Q$:
- Compute $Q(I^p)$, for probabilistic database $I^p$

Simplest scenario only:
- Possible tuples semantics for $Q$
- Independent tuples for $I^p$
Extensional Query Evaluation
Relational ops compute probabilities

Unlike intensional evaluation, data complexity: PTIME

SELECT DISTINCT x.City
FROM Person^p x, Purchase^p y
WHERE x.Name = y.Cust
and y.Product = 'Gadget'

\[\Pi \left( \begin{array}{ccc} Sea & 1-(1-p_1q_1)(1-p_1q_2)(1-p_1q_3) \\
Jon & Sea & p_1q_1 \\
Jon & Sea & p_1q_2 \\
Jon & Sea & p_1q_3 \\
\end{array} \right) \times \left( \begin{array}{ccc} Jon & 1-(1-q_1)(1-q_2)(1-q_3) \\
Jon & q_1 \\
Jon & q_2 \\
Jon & q_3 \\
\end{array} \right) \]

Wrong!

Correct

 Depends on plan !!!
Query Complexity

Sometimes \( \not\exists \) correct ("safe") extensional plan

\[
Q_{\text{bad}} \leftarrow R(x), S(x,y), T(y)
\]

Data complexity is \#P complete

**Theorem** The following are equivalent

- \( Q \) has PTIME data complexity
- \( Q \) admits an extensional plan (and one finds it in PTIME)
- \( Q \) does not have \( Q_{\text{bad}} \) as a subquery

[Dalvi&Suciu:2004]
Computing a Safe SPJ Extensional Plan

Problem is due to projection operations
• An “unsafe” extensional projection combines tuples that are correlated assuming independence

Projection over a join that projects away at least one of the join attrs ➔ Unsafe projection!
• *Intuitive:* Joins create correlated output tuples
Computing a Safe SPJ Extensional Plan

Algorithm for Safe Extensional SPJ Evaluation

• Apply safe projections as late as possible in the plan

• If no more safe projections exist, look for joins where all attributes are included in the output
  – Recurse on the LHS, RHS of the join

Sound and complete safe SPJ evaluation algorithm

• *If a safe plan exists*, the algo finds it!
Summary on Query Complexity

Extensional query evaluation:
• Very popular
• Guarantees polynomial complexity
• However, result depends on query plan and correctness not always possible!

General query complexity
• #P complete (not surprising, given #SAT)
• Already #P hard for very simple query (Q_{bad})

Probabilistic databases have high query complexity
Efficient Approximate Evaluation: Monte-Carlo Simulation

Run evaluation with no projection/dup elimination till the very final step

• Intermediate tuples carry all attributes

• Each result tuple = \( \text{group } t_1, \ldots, t_n \) of tuples with the same projection attribute values
  – \( \text{Prob}(\text{group}) = \text{Prob}(C_1 \text{ OR } C_2 \text{ OR } \ldots \text{ OR } C_n) \), where each \( C_i = e_1 \text{ AND } e_2 \ldots \text{ AND } e_k \)
  – Evaluate the probability of a large DNF expression
  – Can be efficiently approximated through MC simulation (a.k.a. sampling)
Monte Carlo Simulation

Naïve:

\[ E = X_1X_2 \lor X_1X_3 \lor X_2X_3 \]

\[ \text{Cnt} \leftarrow 0 \]
\[ \text{repeat \ N times} \]
\[ \quad \text{randomly choose } X_1, X_2, X_3 \in \{0,1\} \]
\[ \quad \text{if } E(X_1, X_2, X_3) = 1 \]
\[ \quad \quad \text{then } \text{Cnt} = \text{Cnt} + 1 \]
\[ \text{P} = \frac{\text{Cnt}}{N} \]
\[ \text{return } P /\ast \sim \Pr(E) /\ast \]

\textbf{Theorem.} If \( N \geq \frac{1}{\Pr(E)} \times \left(4\ln(2/\delta)/\epsilon^2\right) \) then:
\[ \Pr[ | P/\Pr(E) - 1 | > \epsilon ] < \delta \]

\(0/1\)-estimator theorem

Works for any \( E \)

Not in PTIME

May be very big

\[ X_1X_2, X_1X_3, X_2X_3 \]
Monte Carlo Simulation

Improved:

\[ E = C_1 \lor C_2 \lor \ldots \lor C_m \]

\[
\text{Cnt} \leftarrow 0; \quad S \leftarrow \Pr(C_1) + \ldots + \Pr(C_m);
\]

\textbf{repeat} N times

- randomly choose \( i \in \{1,2,\ldots, m\} \), with prob. \( \Pr(C_i) / S \)
- randomly choose \( X_1, \ldots, X_n \in \{0,1\} \) s.t. \( C_i = 1 \)
- \textbf{if} \( C_1=0 \) and \( C_2=0 \) and \ldots and \( C_{i-1}=0 \)

- \textbf{then} \( \text{Cnt} = \text{Cnt}+1 \)

\[
P = \text{Cnt}/N \times 1/
\]

\textbf{return} \( P /\ast \approx \Pr(E) /\ast\)

\textbf{Theorem.} If \( N \geq (1/m) \times (4\ln(2/\delta)/\varepsilon^2) \) then:

\[
\Pr[|P/\Pr(E) - 1| > \varepsilon] < \delta
\]

Only for \( E \) in DNF
In \text{PTIME}
Summary on Monte Carlo

Some form of simulation is needed in probabilistic databases, to cope with the \#P-hardness bottleneck

– Naïve MC: works well when Prob is big
– Improved MC: needed when Prob is small

Recent work [Re,Dalvi,Suciu, ICDE’07] describes optimized MC for top-k tuple evaluation
Handling Tuple Correlations

Tuple correlations/dependencies arise naturally
– *Sensor networks*: Temporal/spatial correlations
– During query evaluation (even starting with independent tuples)

Need representation formalism that can capture and evaluate queries over such correlated tuples
Capturing Tuple Correlations: Basic Ideas

Use key ideas of Probabilistic Graphical Models (PGMs)
  – Bayes and Markov networks are special cases

Tuple-based random variables
  – Each tuple \( t \) corresponds to a Boolean RV \( X_t \)

Factors capturing correlations across subsets of RVs
  – \( f(X) \) is a function of a (small) subset \( X \) of the \( X_t \) RVs

[Sen, Deshpande:2007]
Capturing Tuple Correlations: Basic Ideas

Associate each probabilistic tuple with a binomial RV

Define PGM factors capturing correlations across subsets of tuple RVs

Probability of a possible world = product of all PGM factors

- PGM = factored, economical representation of possible worlds distribution

- **Closed & complete** representation formalism
Example: Mutual Exclusion

Want to capture mutual exclusion (XOR) between tuples \( s_1 \) and \( t_1 \)
Example: Positive Correlation

Want to capture positive correlation between tuples $s_1$ and $t_1$
**PGM Representation**

*Definition:* A PGM is a graph whose nodes represent RVs and edges represent correlations

**Factors** correspond to the cliques of the PGM graph

- Graph structure encodes conditional independencies
- Joint pdf = $\prod$ clique factors
- Economical representation ($O(2^k)$, $k=|\text{max clique}|$)
Query Evaluation: Basic Ideas

Carefully represent correlations between base, intermediate, and result tuples to generate a PGM for the query result distribution

– Each relational op generates *Boolean factors* capturing the dependencies of its input/output tuples

Final model = product of all generated factors

Cast probabilistic computations in query evaluation as a *probabilistic inference* problem over the resulting (factored) PGM

– Can import ML techniques and optimizations
Query Evaluation: Example

\[ \Pi_D(S \bowtie_{B=c} T) \]

\[
\begin{array}{|c|c|}
\hline
S: & \quad A & B \\
\hline
s_1 & m & 1 \\
\hline
s_2 & n & 1 \\
\hline
\end{array}
\]

\[ f_{s_1}, f_{s_2} \]

\[
\begin{array}{|c|c|c|c|}
\hline
T: & A & B & C & D \\
\hline
\hline
& m & 1 & 1 & p \\
\hline
& n & 1 & 1 & p \\
\hline
\end{array}
\]

\[ f_{i_1, i_2, s_1, s_2, r_1, r_2} \]

\[ f_{\text{AND}} \]

\[ f_{\text{OR}} \]

\[ \Pi_D(S \bowtie_{B=c} T) \rightarrow S \bowtie_{B=c} T \rightarrow \Pi_D(S \bowtie_{B=c} T) \]

\[
\begin{array}{|c|c|}
\hline
r_1 & D \\
\hline
\end{array}
\]

\[ f_{r_1, i_1, i_2} \]
Probabilistic DBs: Summary

• Principled framework for managing uncertainties
  – Uncertainty management: ML, AI, Stats
  – Benefits of DB world: declarative QL, optimization, scale to large data, physical access structs, …

• Prob DBs = “Marriage” of DBs and ML/AI/Stats
  – ML folks have also been moving our way: Relational extensions to ML models (PRMs, FO models, inductive logic programming, …)
Probabilistic DBs: Future

- Importing more sophisticated ML techniques and tools inside the DBMS
  - Inference as queries, FO models and optimizations, access structs for relational queries + inference, …

- More on the algorithmic front: Probabilistic DBs and possible worlds semantics brings new challenges
  - E.g., approximate query processing, probabilistic data streams (e.g., sketching), …