Probabilistic/Uncertain Data Management -- III


Slides based on the Suciu/Dalvi SIGMOD’05 tutorial
What is a Probabilistic Database?

- “An item belongs to the database” is a probabilistic event
  - Tuple-existence uncertainty
  - Attribute-value uncertainty

- “A tuple is an answer to the query” is a probabilistic event

- Can be extended to all data models; we discuss only probabilistic *relational* data
Possible Worlds Semantics

The set of all possible database instances:

\[ \text{INST} = \{I_1, I_2, I_3, \ldots, I_N\} \]

**Definition** A *probabilistic database* \(I^p\) is a probability distribution on \(\text{INST}\)

\[ \text{Pr} : \text{INST} \rightarrow [0,1] \quad \text{s.t.} \quad \sum_{i=1,N} \text{Pr}(I_i) = 1 \]

**Definition** A *possible world* is \(I\) s.t. \(\text{Pr}(I) > 0\)
Query Semantics

Given a query Q and a probabilistic database Ip, what is the meaning of Q(Ip)?
Query Semantics

Semantics 1: Possible Answers
A probability distribution on *sets of tuples*

\[ \forall A. \Pr(Q = A) = \sum_{I \in \text{INST. } Q(I) = A} \Pr(I) \]

Semantics 2: Possible Tuples
A probability function on *tuples*

\[ \forall t. \Pr(t \in Q) = \sum_{I \in \text{INST. } t \in Q(I)} \Pr(I) \]
Possible Worlds Query Semantics

Possible answers semantics
• Precise
• Can be used to compose queries
• Difficult user interface

Possible tuples semantics
• Less precise, but simple; sufficient for most apps
• Cannot be used to compose queries
• Simple user interface
Possible Worlds Semantics: Summary

*Complete* model; Clean formal semantics for SQL queries

*Not* very useful as a representation or implementation tool
• HUGE number of possible worlds!

Need more effective representation formalisms
• Something that users can understand/explore
• Allow more efficient query execution
  – Avoid “possible worlds explosion”
• *Perhaps giving up completeness*
Representation Formalisms

Problem
Need a good representation formalism

- Will be interpreted as possible worlds
- Several formalisms exist, but no winner
Evaluation of Formalisms

Completeness?
• What possible worlds can it represent?
• What probability distributions on worlds?

Closure?
• Is it closed under evaluation of query operators?
A Complete Formalism: Intensional Databases

Atomic event ids

\[ e_1, e_2, e_3, \ldots \]

Probabilities:

\[ p_1, p_2, p_3, \ldots \in [0,1] \]

Event expressions: \( \land, \lor, \neg \)

\[ e_3 \land (e_5 \lor \neg e_2) \]

Intensional probabilistic database J: each tuple \( t \) has an event attribute \( t.E \)
Intensional DB $\Rightarrow$ Possible Worlds

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>$e_1 \land (e_2 \lor e_3)$</td>
</tr>
<tr>
<td>Sue</td>
<td>Denver</td>
<td>$(e_1 \land e_2) \lor (e_2 \land e_3)$</td>
</tr>
</tbody>
</table>

\[
J = \new{1-p_1}{0} \times \new{1-p_2}{0} \times \new{1-p_3}{0}
\]

\[
\new{1-p_1}{1-p_2}(1-p_3) + (1-p_1)(1-p_2)p_3 + (1-p_1)p_2(1-p_3) + p_1(1-p_2)(1-p_3)
\]

\[
\new{p_1}{1-p_2} p_3 + (1-p_1)p_2 p_3
\]

\[
p_1 p_2(1-p_3) + p_1 p_2 p_3
\]

\[
\new{1}{1}
\]

\[
e_1 e_2 e_3 = 000 \quad 001 \quad 010 \quad 011 \quad 100 \quad 101 \quad 110 \quad 111
\]
Possible Worlds $\Rightarrow$ Intensional DB

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
</tr>
<tr>
<td>John</td>
<td>Boston</td>
</tr>
<tr>
<td>Sue</td>
<td>Seattle</td>
</tr>
</tbody>
</table>

$E_1 = e_1$

$E_2 = \neg e_1 \wedge e_2$

$E_3 = \neg e_1 \wedge \neg e_2 \wedge e_3$

$E_4 = \neg e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge e_4$

$\Pr(e_1) = p_1$

$\Pr(e_2) = p_2/(1-p_1)$

$\Pr(e_3) = p_3/(1-p_1-p_2)$

$\Pr(e_4) = p_4/(1-p_1-p_2-p_3)$

"Prefix code"

$p_1 = \text{John Seattle}

p_2 = \text{John Seattle}

p_3 = \text{Sue Seattle}

p_4 = \text{John Boston}

$J = E_1 \lor E_2$

$J = E_1 \lor E_4$

$J = E_1 \lor E_2 \lor E_3$

Intensional DBs are complete
Closure Under Operators

One still needs to compute probability of event expression
Summary on Intensional Databases

Event expression for each tuple
• Possible worlds: any subset
• Probability distribution: any
Complete… but impractical
• Evaluate the probability of long event expressions

Important abstraction: consider restrictions

Related to c-tables [Imilelinski&Lipski:1984]
A Restricted Formalism: Explicit Independent Tuples

**Tuple independent** probabilistic database

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

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

No restrictions

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

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

\[
N = 2^M
\]
**Tuple Prob. ⇒ Possible Worlds**

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>( p )</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>

\[
I^p = \begin{array}{ccccccccc}
\emptyset & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto} & \text{John} & \text{Seattl} & \text{Sue} & \text{Bosto} & \text{Fred} & \text{Bosto}
\end{array}
\]

\[
J = \begin{array}{cc}
I_1 & I_2 \\
(1-p_1) & (1-p_1)(1-p_2)(1-p_3)
\end{array}
\begin{array}{cc}
I_3 & I_4 \\
(1-p_1)p_2(1-p_3) & (1-p_1)(1-p_2)p_3
\end{array}
\begin{array}{cc}
I_5 & I_6 \\
p_1p_2(1-p_3) & p_1(1-p_2)p_3
\end{array}
\begin{array}{cc}
I_7 & I_8 \\
(1-p_1)p_2p_3 & p_1p_2p_3
\end{array}
\]

\[
\sum = 1
\]

\[
E[ \text{size}(I^p) ] = 2.3 \text{ tuples}
\]
**Tuple-Independent DBs are Incomplete**

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>(p_1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td></td>
</tr>
<tr>
<td>Sue</td>
<td>Seattle</td>
<td></td>
</tr>
</tbody>
</table>

\[p_1p_2 \leq \text{IP}\]

\[\emptyset \leq 1-p_1 - p_1p_2\]

Very limited – cannot capture correlations across tuples

*Not Closed*

- Query operators can introduce complex correlations!
**Tuple Prob. ⇒ Query Evaluation**

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>p₁</td>
</tr>
<tr>
<td>Sue</td>
<td>Boston</td>
<td>p₂</td>
</tr>
<tr>
<td>Fred</td>
<td>Boston</td>
<td>p₃</td>
</tr>
</tbody>
</table>

**Query Evaluation**

SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Customer
and y.Product = ‘Gadget’

<table>
<thead>
<tr>
<th>Customer</th>
<th>Product</th>
<th>Date</th>
<th>pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Gizmo</td>
<td>...</td>
<td>q₁</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>...</td>
<td>q₂</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>...</td>
<td>q₃</td>
</tr>
<tr>
<td>Sue</td>
<td>Camera</td>
<td>...</td>
<td>q₄</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>...</td>
<td>q₅</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>...</td>
<td>q₆</td>
</tr>
<tr>
<td>Fred</td>
<td>Gadget</td>
<td>...</td>
<td>q₇</td>
</tr>
</tbody>
</table>

**Tuple Probability**

<table>
<thead>
<tr>
<th>Tuple</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seattle</td>
<td>( p₁(1-(1-q₂)(1-q₃)) )</td>
</tr>
<tr>
<td>Boston</td>
<td>( 1-(1-p₂(1-(1-q₅)(1-q₆))) \times (1-p₃q₇) )</td>
</tr>
</tbody>
</table>
## Application: Similarity Predicates

### Table: Person x, Purchase y

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>Profession</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>statistician</td>
</tr>
<tr>
<td>Sue</td>
<td>Boston</td>
<td>musician</td>
</tr>
<tr>
<td>Fred</td>
<td>Boston</td>
<td>physicist</td>
</tr>
</tbody>
</table>

### Table: Product and Category

<table>
<thead>
<tr>
<th>Cust</th>
<th>Product</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Gizmo</td>
<td>dishware</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>instrument</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>instrument</td>
</tr>
<tr>
<td>Sue</td>
<td>Camera</td>
<td>musicware</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>microphone</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>instrument</td>
</tr>
<tr>
<td>Fred</td>
<td>Gadget</td>
<td>microphone</td>
</tr>
</tbody>
</table>

### SQL Query

```sql
SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Cust
  AND y.Product = 'Gadget'
  AND x.profession ~ 'scientist'
  AND y.category ~ 'music'
```
**Application: Similarity Predicates**

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>Profession</th>
<th>pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Seattle</td>
<td>statistician</td>
<td>p₁=0.8</td>
</tr>
<tr>
<td>Sue</td>
<td>Boston</td>
<td>musician</td>
<td>p₂=0.2</td>
</tr>
<tr>
<td>Fred</td>
<td>Boston</td>
<td>physicist</td>
<td>p₃=0.9</td>
</tr>
</tbody>
</table>

**Step 1:** evaluate ~ predicates

**Step 2:** evaluate rest of query

```
SELECT DISTINCT x.city
FROM Personᵖ x, Purchaseᵖ y
WHERE x.Name = y.Cust
  and y.Product = ‘Gadget’
  and x.profession ~ ‘scientist’
  and y.category ~ ‘music’
```

<table>
<thead>
<tr>
<th>Cust</th>
<th>Product</th>
<th>Category</th>
<th>pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Gizmo</td>
<td>dishware</td>
<td>q₁=0.2</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>instrument</td>
<td>q₂=0.6</td>
</tr>
<tr>
<td>John</td>
<td>Gadget</td>
<td>instrument</td>
<td>q₃=0.6</td>
</tr>
<tr>
<td>Sue</td>
<td>Camera</td>
<td>musicware</td>
<td>q₄=0.9</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>microphone</td>
<td>q₅=0.7</td>
</tr>
<tr>
<td>Sue</td>
<td>Gadget</td>
<td>instrument</td>
<td>q₆=0.6</td>
</tr>
<tr>
<td>Fred</td>
<td>Gadget</td>
<td>microphone</td>
<td>q₇=0.7</td>
</tr>
</tbody>
</table>

**Probability**

- **Seattle**
  \[ p₁(1-(1-q₂)(1-q₃)) \]

- **Boston**
  \[ 1-(1-p₂(1-(1-q₅)(1-q₆)))(1-p₃q₇) \]
Summary on Explicit Independent Tuples

Independent tuples
• Possible worlds: subsets
• Probability distribution: restricted
• Closure: no
Query Evaluation on Probabilistic DBs

- Focus on possible tuple semantics
  - Compute likelihood of individual answer tuples
- Probability of Boolean expressions
- Complexity of query evaluation
Probability of Boolean Expressions

\[ E = X_1 X_3 \lor X_1 X_4 \lor X_2 X_5 \lor X_2 X_6 \]

Randomly make each variable \textbf{true} with the following probabilities

\[
\Pr(X_1) = p_1, \quad \Pr(X_2) = p_2, \quad \ldots, \quad \Pr(X_6) = p_6
\]

What is \( \Pr(E) \) ???

Answer: re-group cleverly

\[ E = X_1 (X_3 \lor X_4) \lor X_2 (X_5 \lor X_6) \]

\[
\Pr(E) = 1 - (1 - p_1 (1 - (1 - p_3)(1 - p_4))) \\
(1 - p_2 (1 - (1 - p_5)(1 - p_6)))
\]
Now let’s try this:

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

No clever grouping seems possible. Brute force:

\[
\Pr(E) = (1-p_1)p_2p_3 + p_1(1-p_2)p_3 + p_1p_2(1-p_3) + p_1p_2p_3
\]

<table>
<thead>
<tr>
<th>X_1</th>
<th>X_2</th>
<th>X_3</th>
<th>E</th>
<th>Pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>(1-p_1)p_2p_3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>p_1(1-p_2)p_3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>p_1p_2(1-p_3)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>p_1p_2p_3</td>
</tr>
</tbody>
</table>

Seems inefficient in general…
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
Summary on Boolean Expression Probability

• #P-complete

• It’s hard even in simple cases: 2DNF

• Can approximate through Monte Carlo (MC) simulation
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’

\[ \Pi \]

Sea 1-(1-p_1 q_1)(1-p_1 q_2)(1-p_1 q_3) ×

Jon 1-(1-q_1)(1-q_2)(1-q_3)

Jon Sea p_1 q_1
Jon Sea p_1 q_2
Jon Sea p_1 q_3

Jon Sea p_1

Jon q_1
Jon q_2
Jon q_3

Correct

Wrong!

[Dalvi & Suciu: 2004]

Depending on plan!!!
Query Complexity

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

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

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