Relational databases assume “exact” data
Increasingly breaking down
  “Sensor” data
    Physical/hardware sensors
    Also data observed by people (called “persono-scopes”)
  Data integration type of applications
    Confidence/trust/“quality” issues
    Machine generated schema mappings
    Automatically extracted information/knowledge
  Reasoning using Machine Learning techniques
    Many classifiers provide probabilities of belonging to different classes
  Data may be exact, but queries may not be
    inst-name like “UMD” (does this match “Univ of MD” ?)
    Common in Information Retrieval
Types of uncertainty

- “Nulls” (very interesting early work on this)
- More generally, missing values/incomplete databases
- Existence uncertainty: Tuples may or may not exist
  - Can be modeled logically, probabilities, fuzzy logic etc. . .
- Attribute values are uncertain
  - I saw a car, but not sure of the color
  - Sensor values have “noise” in them
- Approximations/Summarization
- Confidence/Trust/Quality
- “Correlations”
  - Either this tuple exists or that
  - If this attribute is 1, then that attribute is 2

Sensor Web: Types of Uncertainty

**Caption:** Even if the sensor web data sources were to publish data using intuitive well-defined interfaces, the complex and semantically disparate measures of data quality and uncertainty typically associated with it make sensor data fusion and aggregation a challenging task.
Lineage/Data Provenance

- Another thing databases don’t do well
- Goal: Trace back a query result to original data
- Provenance/Lineage
  - Carry around enough information during query processing to trace back
- Very important in sensor applications
  - Vision: Highly automated systems that process vast amounts of data to decide what to do next
  - Large amounts of aggregation, distributed processing
  - A small mistake/noise can cause large errors later on
  - Custom-build sensor processing code makes it harder

Probabilistic Databases

- Use probabilities to model the uncertainty
- Use laws of probability to process the data
- Tuple-level uncertainty
  - All attributes in a tuple are known precisely
  - Existence of the tuple is uncertain
- Attribute-level uncertainty
  - Tuples (identified by “keys”) exist for certain
  - Exact value of an attribute is uncertain
Probabilistic Databases

- Tuple-level vs Attribute-level
  - Is this a fundamental distinction?
  - Everything can be modeled using random variables
  - Normalization, probability-intervals etc can be used to combine the two
- But trying to create a model that can represent *everything* is probably doomed to fail
  - Intractability issues come up very soon
  - KISS
- Still no consensus on a data model/query language etc
  - People end up making different types of simplifications

A Tuple-level Uncertainty Model

- Proposed by Fuhr et al. in Information Retrieval Context
  - Later work by Dalvi and Suciu, by us here at UMD (Prithviraj Sen)
  - Examples from Dalvi and Suciu, VLDB 2004
- A simple probabilistic database

\[
S^p =
\begin{array}{cc}
A & B \\
\text{m} & 1 \\
\text{n} & 1
\end{array}
\quad
T^p =
\begin{array}{cc}
C & D \\
1 & \text{p'}
\end{array}
\]

\[
\begin{align*}
S^p &=
\begin{array}{cc}
A & B \\
\text{m} & 1 \\
\text{n} & 1
\end{array}
\quad
T^p &=
\begin{array}{cc}
C & D \\
1 & \text{p'}
\end{array}
\end{align*}
\]
Possible Worlds

- The best way to think of probabilistic databases
- A world is a fixed set of tuples – no uncertainty
- Assuming “independence”:

\[
S^p = \begin{array}{c|c}
A & B \\
\hline
s_1 & 'm' \\
s_2 & 'n'
\end{array}
\quad T^p = \begin{array}{c|c}
C & D \\
\hline
t_1 & 'p'
\end{array}
\]

\[
pwd(D^p) = \begin{array}{|c|c|}
\hline
\text{database instance} & \text{probability} \\
\hline
D_1 = \{s_1, s_2, t_1\} & 0.24 \\
D_2 = \{s_1, t_1\} & 0.24 \\
D_3 = \{s_2, t_1\} & 0.06 \\
D_4 = \{t_1\} & 0.06 \\
D_5 = \{s_1, s_2\} & 0.16 \\
D_6 = \{s_1\} & 0.16 \\
D_7 = \{s_2\} & 0.04 \\
D_8 = \emptyset & 0.04 \\
\hline
\end{array}
\]

Query Evaluation

- Evaluate on each possible world
- Combine the results together somehow
  - Semantics of this can be tricky
- Example: \( \pi_D(S \bowtie_{B=C} T) \)

\[
S^p = \begin{array}{c|c}
A & B \\
\hline
s_1 & 'm' \\
s_2 & 'n'
\end{array}
\quad T^p = \begin{array}{c|c}
C & D \\
\hline
t_1 & 'p'
\end{array}
\]

\[
pwd(D^p) = \begin{array}{|c|c|}
\hline
\text{database instance} & \text{probability} \\
\hline
D_1 = \{s_1, s_2, t_1\} & 0.24 \\
D_2 = \{s_1, t_1\} & 0.24 \\
D_3 = \{s_2, t_1\} & 0.06 \\
D_4 = \{t_1\} & 0.06 \\
D_5 = \{s_1, s_2\} & 0.16 \\
D_6 = \{s_1\} & 0.16 \\
D_7 = \{s_2\} & 0.04 \\
D_8 = \emptyset & 0.04 \\
\hline
\end{array}
\]

Result

<table>
<thead>
<tr>
<th>\text{answer}</th>
<th>\text{probability}</th>
</tr>
</thead>
<tbody>
<tr>
<td>{p}</td>
<td>0.54</td>
</tr>
<tr>
<td>{p}</td>
<td>0.46</td>
</tr>
<tr>
<td>\emptyset</td>
<td></td>
</tr>
<tr>
<td>\emptyset</td>
<td></td>
</tr>
<tr>
<td>\emptyset</td>
<td></td>
</tr>
<tr>
<td>\emptyset</td>
<td></td>
</tr>
</tbody>
</table>
Another Example: $\pi_A(S \Join_{B=C} T)$

$$S^p = \begin{array}{c|c|c}
A & B & \text{probability} \\
\hline
s_1 & m & 1 & 0.8 \\
s_2 & n & 1 & 0.5 \\
\end{array}$$

$$T^p = \begin{array}{c|c}
C & D & \text{probability} \\
\hline
t_1 & p & 0.6 \\
\end{array}$$

$$p_{\text{wul}}(D^p) = \begin{array}{c|c|c|c}
\text{database instance} & \text{probability} & \text{Result} & \text{probability} \\
\hline
D_1 = \{s_1, s_2, t_1\} & 0.24 & \{m', n\} & \{m\} & 0.24 \\
D_2 = \{s_1, t_1\} & 0.24 & \{m\} & \emptyset & 0.06 \\
D_3 = \{s_2, t_1\} & 0.06 & \{m\} & \emptyset & 0.06 \\
D_4 = \{t_1\} & 0.06 & \emptyset & \emptyset & 0.46 \\
D_5 = \{s_1, s_2\} & 0.16 & \emptyset & \emptyset & 0.06 \\
D_6 = \{s_1\} & 0.16 & \emptyset & \emptyset & 0.06 \\
D_7 = \{s_2\} & 0.04 & \emptyset & \emptyset & 0.04 \\
D_8 = \emptyset & 0.04 & \emptyset & \emptyset & 0.04 \\
\end{array}$$

Query Evaluation

Can’t convert to possible worlds and evaluate the query
- Too many possible worlds
- With just 20 uncertain tuples, number of worlds is at least $2^{20} = 1$ million.

Extensional semantics
- Combine probabilities when tuples are put together
- Issues with correlations

Intensional semantics
- Represent each result as a boolean formula
- Evaluate it at end
- #P-Complete
Extensional: Example 1

\( \pi_D(S \bowtie_{B=C} T) \to \) Do Join first, then Projection

\[\begin{array}{c|c}
S^p & T^p \\
\hline
s_1 & 0.8 \\
s_2 & 0.5 \\
\end{array}\]

\[\begin{array}{c|c|c}
A & B & C & D \\
\hline
\text{\`m} & 1 & \text{\`p} & 0.6 \\
\text{\`n} & 1 & \text{\`p} & 0.6 \\
\end{array}\]

Joins: assume independence

\[\begin{array}{c|c|c|c|c}
A & B & C & D & \text{prob} \\
\hline
\text{\`m} & 1 & 0.8 & \text{\`p} & 0.48 \\
\text{\`n} & 1 & 0.5 & \text{\`p} & 0.30 \\
\end{array}\]

Projection: union probability; assume independence

\[\begin{array}{c|c|c|c|c}
D & \text{prob} \\
\hline
\text{\`p} & (1 - (1 - 0.48)(1 - 0.3)) = 0.636 \\
\end{array}\]

Umm.. This is wrong !!
Why ? The two tuples above are not independent.

Extensional: Example 2

\( \pi_D(S \bowtie_{B=C} T) \to \) Do Projection first, then Join

\[\begin{array}{c|c}
S^p & T^p \\
\hline
s_1 & 0.8 \\
s_2 & 0.5 \\
\end{array}\]

\[\begin{array}{c|c|c}
A & B & C & D \\
\hline
\text{\`m} & 1 & \text{\`p} & 0.6 \\
\text{\`n} & 1 & \text{\`p} & 0.6 \\
\end{array}\]

Projection: union probability; assume independence

\[\begin{array}{c|c|c|c|c}
B & \text{prob} \\
\hline
1 & (1 - (1 - 0.8)(1 - 0.5)) = 0.9 \\
\end{array}\]

Join: assume independence

\[\begin{array}{c|c|c|c|c}
B & C & D & \text{prob} \\
\hline
1 & 1 & \text{\`p} & 0.9 \times 0.6 = 0.54 \\
\end{array}\]

This is correct.
The correctness unfortunately depends on the plan used.
Called “safe plans” [Dalvi, Suciu 2004]
Intensional

- \( \pi_D(S \bowtie_{B=C} T) \rightarrow \) Do Join first, then Projection

\[
\begin{array}{|c|c|c|}
\hline
 & A & B \\
\hline
s_1 & 'm' & 1 \\
s_2 & 'n' & 1 \\
\hline
0.8 & 0.5 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
 & C & D \\
\hline
t_1 & 1 & 'p' \\
0.6 & \\
\hline
\end{array}
\]

Join (intersection):

- \( \bowtie \):

Join (intersection):

- \( \bowtie \):

Projection (union):

- \( \cup \):

Evaluate the expression at the end (this can be #P-Hard)

- Can handles correlations (between tuples)

Query Evaluation

- Intensional vs Extensional
  - Extensional is fast, but not always correct/applicable
  - Intensional is too expensive

- Sen and Deshpande, ICDE 2007
  - Using Graphical Models to represent the correlations between tuples
  - “Closed”
  - Always correct (which means still #P-Hard in general)
  - Can achieve the best of both worlds
    - If data doesn’t have correlations, as fast as extensional

- Still many unanswered questions
  - Can we do better if approximations are allowed?
    - Probabilities are rarely required exactly
    - “Ranking” is all thats needed
Attribute-level Uncertainty

- In some cases easier, in some cases harder to deal with
- Continuous distributions common (e.g. in sensor applications)
- Hard to do exact querying over continuous attributes
- What about correlations?
  - Temperatures are nearby locations are usually correlated
- Much work on this topic; some we are doing here

Lineage

- Given a tuple $t$, we would like to know:
  - When $t$ was derived?
  - How $t$ was derived?
  - What data was used to derive $t$?
  - Who (which user) cause $t$ to be derived?
- Can keep the information along with the $t$
  - Propagate it as the new result tuples are generated
  - Clearly not feasible
  - Try storing with relations instead of each tuple
  - Annotate the tuples somehow
  - ...  
- A lot of open questions, few answers
Very interesting “hot” area
Lot of work by many groups
  - On defining models, efficient query processing, building systems etc.
  - We are doing a bunch of work here
A new conference, called SUM, organized by VS (Oct, in DC)
No real consensus yet emerging
“Killer Application” ??
Key is keeping it simple, otherwise no one will use it