Semistructured Data: The Next Step

Dan Suciu
Thesis

Back then:
semistructured data addressed imprecisions in structure: e.g. data integration, “new data”

Novel paradigm based on semi-structure

Today:
many more variety of imprecisions found in data and schema: e.g. data integration, “new data”

Novel paradigm based on probabilities
Outline

- Semistructured data: past and present
- Probabilistic databases: the new paradigm
- Supporting technologies and theoretical roots
- Towards a theory of probabilistic queries
- Conclusions
Origins 1/2

Motivation: imprecisions in the structure

data integration [Tsimmis’94–98]
handling “new data” [Lorel’97], [UnQL’95]

Solution:
relax the structure
Origins 2/2

Enabling technologies:
- OO databases
- non first normal form data
- querying SGML files
- text databases

Theoretical roots:
- trees, regular expressions
Today

Semistructured data in industry:
XML support in RDBMS
XML engines
XML for data exchange
stream XML processing

Semistructured data in research:
often largest session in SIGMOD/VLDB/PODS
dedicated symposiums, conferences, workshops
2nd Dagstuhl workshop !
Today:
Imprecisions Everywhere

- misspellings
- different terminologies
- object matching
- schema matching
- measurement errors
- constraint violations

None is addressed by semistructured data
Handling Imprecisions Today

- several tools:
  - edit distance, q-gram distance, data cleaning, ontologies (WordNet), record linkage, schema matchings, repairs, ....
- always outside the database system

Should be done inside DBMS
Summary of Past and Present

- imprecisions in data structure:
  lead to semistructured data model
  
- many more imprecisions remain today
  require new data model
Outline

- Semistructured data: past and present
- Probabilistic databases: the new paradigm
- Supporting technologies and theoretical roots
- Towards a theory of probabilistic
- Conclusions
Probabilistic Database

**Definition**
A prob DB = a database where each tuple has a probability

will make this more precise later

first, let’s see examples
**Uncertain Predicates 1/2**

misspellings, data translation, ontologies

```sql
SELECT DISTINCT A.name
FROM Actor A, Casts C, Film F
WHERE C.filmID = F.filmID and C.actorID = A.actorID
and A.name ≈ 'Kevin'
and F.year ≈ 1995
and F.rating ≈ 'high'
```

Q(x) :- Actor(x,y), Casts(y,z), Film(z,u,v),
    x ≈ 'Kevin', u ≈ 1995, v ≈ 'high'
Uncertain Predicates 2/2

Assign a probabilistic score to each tuple

\[ \downarrow \]

probabilistic database

Actor.name \approx \text{‘Kevin’}

Film.rating \approx \text{high}

<table>
<thead>
<tr>
<th>actorID</th>
<th>name</th>
<th>prob. score</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>Kevin B.</td>
<td>1.0</td>
</tr>
<tr>
<td>a2</td>
<td>R. Keven</td>
<td>0.9</td>
</tr>
<tr>
<td>a3</td>
<td>Calvin C.</td>
<td>0.8</td>
</tr>
<tr>
<td>a4</td>
<td>Collins</td>
<td>0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filmID</th>
<th>name</th>
<th>rating</th>
<th>prob. score</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>...</td>
<td>9.2</td>
<td>0.95</td>
</tr>
<tr>
<td>f2</td>
<td>...</td>
<td>7.7</td>
<td>0.65</td>
</tr>
<tr>
<td>m3</td>
<td>...</td>
<td>5.3</td>
<td>0.1</td>
</tr>
<tr>
<td>m4</td>
<td>...</td>
<td>8.1</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Now evaluate \( Q(x) :- \text{Actor}(x,y), \text{Casts}(y,z), \text{Film}(z,u,v) \)
on probabilistic database
Inexact Schema Matches 1/2

S1(customerName, officeAddress, homeAddress)

prob=0.8

?  prob=0.2

S2(name, address)
Inexact Schema Matches 2/2

Query translation:

```sql
select *
from S2
where S2.name ≈ 'Kevin'
    and S2.address = 'Seattle'
```

“Run” this on S1

<table>
<thead>
<tr>
<th>customerName</th>
<th>officeAddress</th>
<th>homeAddress</th>
<th>prob. score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kevin</td>
<td>Spokane</td>
<td>Seattle</td>
<td>0.2 * 1.00</td>
</tr>
<tr>
<td>Keven</td>
<td>Portland</td>
<td>Bellvue</td>
<td>0.0 * 0.95</td>
</tr>
<tr>
<td>Calvin</td>
<td>Seattle</td>
<td>Redmond</td>
<td>0.8 * 0.37</td>
</tr>
<tr>
<td>Collins</td>
<td>Ballard</td>
<td>Seattle</td>
<td>0.2 * 0.03</td>
</tr>
</tbody>
</table>

Probabilities allow us to consider both mappings
### Handling Inconsistencies 1/2

<table>
<thead>
<tr>
<th>Data Source V1</th>
<th>Data Source V2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>Larry Big</td>
<td>Larry Big</td>
</tr>
<tr>
<td>City</td>
<td>City</td>
</tr>
<tr>
<td>Seattle</td>
<td>Bellvue</td>
</tr>
<tr>
<td>Email</td>
<td>Phone</td>
</tr>
<tr>
<td>lb@com</td>
<td>...</td>
</tr>
</tbody>
</table>

Global constraint: each person, only one city

**Q:** Larry Big’s email?

**A:** no answers, due to ‘repairs’

[Chomicki]
## Handling Inconsistencies 2/2

### Data Source V1

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>Email</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Larry Big</td>
<td>Seattle</td>
<td>lb@com</td>
<td>p1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

### Data Source V2

<table>
<thead>
<tr>
<th>Name</th>
<th>City</th>
<th>Phone</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Larry Big</td>
<td>Bellvue</td>
<td>...</td>
<td>p2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

**Global constraint:** $p1 + p2 \leq 1$

No other info? $p1 = p2 = 0.5$

**Q:** Larry Big's email?

**A:** lb@com, with probability $p1 = 0.5$

**Probabilities allow us to use all the data**
**Formal Definition**

**Definition**

A probabilistic database = a probability distribution on all instances

\[
\text{Pr: } \text{Inst} \rightarrow [0,1], \quad \sum_{I \in \text{Inst}} \text{Pr}[I] = 1
\]

**Notation:** probabilistic database \( I^p \)

**Very powerful:** any correlations between tuples

"Possible Worlds"
**Query Semantics**

**Definition**

Given $I^P$, query $Q$, tuple $t$:

$$\Pr[t \in Q] = \sum_{I \in \text{Inst}: \ t \in Q(I)} \Pr[I]$$

**Notation:** $Q(I^P) = \text{prob distribution on tuples}$

Return to user: ranked tuples, top $k$

**Note:** EVERY query has well defined semantics!
Examp

Film

<table>
<thead>
<tr>
<th>filmID</th>
<th>title</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>t1</td>
<td>1995</td>
</tr>
<tr>
<td>f2</td>
<td>t2</td>
<td>1982</td>
</tr>
<tr>
<td>m3</td>
<td>t3</td>
<td>1982</td>
</tr>
<tr>
<td>m4</td>
<td>t4</td>
<td>1971</td>
</tr>
</tbody>
</table>

prob

P1
P2
P3
P4

Actor

<table>
<thead>
<tr>
<th>filmID</th>
<th>actorName</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>Kevin B.</td>
</tr>
<tr>
<td>f1</td>
<td>R. Keven</td>
</tr>
<tr>
<td>f1</td>
<td>Calvin C.</td>
</tr>
<tr>
<td>f2</td>
<td>Collins</td>
</tr>
<tr>
<td>f2</td>
<td>...</td>
</tr>
</tbody>
</table>

prob

q1
q2
q3
q4
...

What are the “possible worlds”? How do we “evaluate” a query?
Example: Possible Worlds

<table>
<thead>
<tr>
<th>filmID</th>
<th>title</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>T1</td>
<td>1995</td>
</tr>
<tr>
<td>f2</td>
<td>T2</td>
<td>1982</td>
</tr>
<tr>
<td>m3</td>
<td>T3</td>
<td>1982</td>
</tr>
<tr>
<td>m4</td>
<td>T4</td>
<td>1971</td>
</tr>
</tbody>
</table>

\[(1-p_1)(1-p_2)p_3(1-p_4)\]

\[p_1p_2(1-p_3)p_4\]

\[p_1p_2p_3p_4\]

\[\mathbb{I}_p\]
**Example: Query Evaluation**

<table>
<thead>
<tr>
<th>Film^P</th>
<th>Cast^P</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>filmID</td>
</tr>
<tr>
<td>1995</td>
<td>f1</td>
</tr>
<tr>
<td>1982</td>
<td>f2</td>
</tr>
<tr>
<td>1982</td>
<td>m3</td>
</tr>
</tbody>
</table>

Q(x) :- Film(x,y), Cast(y,z)

\[
Prob(1995) = P₁ \times (1 - (1 - q₁)(1 - q₂)(1 - q₃))
\]

\[
Prob(1982) = 1 - (1 - P₂ \times (1 - (1 - q₄)(1 - q₅))) \times \\
(1 - P₃ \times (1 - (1 - q₆)))
\]
Summary of the new Paradigm

- System converts all imprecisions into prob db $I^p$
- User asks standard query $Q$
- System computes $Q(I^p)$, ranks tuples, returns top $k$
- Excellent exploratory queries, not business process

Variations:
- Expose probabilities: Trio [Stanford]
- Hide probabilities: MystiQ [UW]
Outline

- Semistructured data: past and present
- Probabilistic databases: the new paradigm
- Supporting technologies and theoretical roots
- Towards a theory of probabilistic queries
- Conclusions
Technology: Top-k Ranking

Data is exact, but user is willing to accept inexact matches

SELECT *
FROM Houses
WHERE no_bdr = 4
    and no_garages = 3
    and price = 300k

Rich recent literature on evaluation techniques
Technology: Optimal Aggregation

Theorem [Fagin, Lotem, Naor’01,03]
There are optimal algorithms for computing top k
Technology:
Ranking Query Results

Data/queries are inexact

<table>
<thead>
<tr>
<th>Keyword searches</th>
<th>DBExplorer [Chaudhuri'02]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DISCOVER [Hristidis'02]</td>
</tr>
<tr>
<td></td>
<td>BANKS [Bhalotia'02]</td>
</tr>
<tr>
<td>Approximate SQL</td>
<td>[Agrawal'03]</td>
</tr>
<tr>
<td></td>
<td>[Dalvi'04]</td>
</tr>
<tr>
<td>Using ontologies</td>
<td>XXL [Theobald'02]</td>
</tr>
<tr>
<td></td>
<td>TOSS [Hung'04]</td>
</tr>
</tbody>
</table>
Technology: Object Matching

AKA: Object fusion, record linkage, merge/purge, de-duplication, ....

Problem: given n objects, find duplicates assuming typos, misspellings, ...

Theorem: AI-complete

Long history, rich literature
Technology: Information Extraction

KnoItAll [Etzioni et al.’04]

Collect large body of information by searching the Web

Extractors defined per class. E.g. scientist

Albert Einstein  0.9262188000
Niels Bohr       0.9207664000
Enrico Fermi     0.9199148000
Marie Curie      0.9156590000
Max Planck       0.9126473000
Ernest Rutherford 0.9108679000
Michael Faraday   0.9105023000
Louis Pasteur    0.8986677000
Stephen Hawking  0.8986677000
Henry Cavendish  0.8981776000

Lots of useful probabilistic databases!
Theory: Random Graphs (1/2)

Erdos and Reny

Independent tuple distribution: \( \Pr[t] = p(n) \);

Given property \( Q \), let \( \mu[Q] = \lim_{n \to \infty} \Pr[Q] \)

**Theorem** Every monotone \( Q \) has “threshold function” \( t(n) \):

- if \( p(n) \ll t(n) \) then \( \mu[Q] = 0 \)
- if \( p(n) \gg t(n) \) then \( \mu[Q] = 1 \)

\[ p(n) = \frac{c}{n^2} \]
Theory: Random Graphs (2/2)

Erdos and Reny

Program
find “threshold” functions $p(n)$ for various properties $Q$

Important example: subgraph property: $Q \subseteq G$

Threshold is: $t(n) = 1/n^h$ (note: $>> 1/n^2$)
where $h = \min_H H/e(H)$ (inv. density)
Theory: 0/1 Laws

Fagin’76

$\Pr[t] = 1/2$

for an $Q$ in $FO$, $\mu[Q] = 0$ or $\mu[Q] = 1$

Shelah and Spencer

$\Pr[t] = 1/n^\beta$

0/1 law holds if $\beta$ is irrational

Grove, Halpern, Koller

$\Pr[t] = 1/2$

$\mu[Q|V]$ does not always exist, and it’s undecidable

$p(n) = C/n^2$
Theory: Prob DB’s (1/3)

Formalism for specifying a prob db

Independent events: e1, e2, ...
with probabilities: p1, p2, ...

Intensional database:

\[ I^p = \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x</td>
<td>e1</td>
</tr>
<tr>
<td>a</td>
<td>y</td>
<td>e3 &amp; e4</td>
</tr>
<tr>
<td>b</td>
<td>z</td>
<td>e4</td>
</tr>
<tr>
<td>b</td>
<td>u</td>
<td>e2 \lor (-e4 &amp; e1)</td>
</tr>
</tbody>
</table>

Extensional database: all events are independent
Possible worlds semantics

given $E \subseteq \{e_1, e_2, \ldots\}$ define $I = \text{inst}(I^P, E)$

$$\Pr[I] = \sum \{\Pr[E] \mid I = \text{inst}(I^P, E)\}$$

**Theorem** [Dalvi&S] Every probability distribution $\Pr[I]$ comes from an intentional db

**Theorem** Computing $\Pr[t \in I^P]$ is $\#P$ complete
Theory: Prob DB’s (3/3)

Theorem [Fuhr & Roellke] Closed under FO evaluation

\[ \exists y. \text{actor}(x,y) \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x</td>
<td>e1</td>
</tr>
<tr>
<td>a</td>
<td>y</td>
<td>e3 &amp; e4</td>
</tr>
<tr>
<td>b</td>
<td>z</td>
<td>e4</td>
</tr>
<tr>
<td>b</td>
<td>u</td>
<td>e2 \lor (\neg e4 &amp; e1)</td>
</tr>
</tbody>
</table>

\[
\begin{array}{|c|c|}
\hline
A & E \\
\hline
a & e1 \lor (e3 \& e4) \\
b & e4 \lor (e2 \lor (\neg e4 \& e1)) \\
\hline
\end{array}
\]

Intensional databases: powerful but expensive
Summary of Supporting Work

- Supporting technology
  - a lot exists
  - need to understand it in order to make correct assumptions

- Theoretical roots
  - Old and solid
  - Do not apply out of the box
Outline

- Semistructured data: past and present
- Probabilistic databases: the new paradigm
- Supporting technologies and theoretical roots
- Towards a theory of probabilistic queries
- Conclusions
An Agenda for a Theory of Probabilistic Queries

- Specification languages
- Query evaluation: data complexity
- Query evaluation using views
- Randomized PTIME query approximation
- Probabilistic XML
Specification Languages

- Extensional databases
  [Fuhr&Roellke’97]
- Intensional databases
- Probabilistic Relational Model
  [Fefferman’99]
- Implicit probabilities
  [Erdos&Reny,Shelah&Spencer]

Agenda Item:
Expressibility/complexity/conciseness tradeoffs
Query Evaluation (1/3)

Given: $Q, I^P$
Compute: $Q(I^P)$  Variation: return top $k$

$Q(x) :- \text{Film}(x,y), \text{Cast}(y,z)$

\[
\text{Prob}(1982) = 1 - \left(1 - p_2 \ast (1 - (1 - q_4) \ast (1 - q_5)))\right) \ast (1 - p_3 \ast (1 - (1 - q_6)))
\]

Data complexity of $Q$ is in PTIME
Query Evaluation (2/3)

Q(x,u) :- Film(x,y), Cast(y,z), Actor(z,u)

Prob(1982, Kevin) = ???

Data complexity of Q is #P-complete
Query Evaluation (3/3)

**Theorem** [Dalvi&S.’04,Graedel’98] Data complexity is:

<table>
<thead>
<tr>
<th>Q</th>
<th>Q(I_p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>... R(x,...), S(x,y,...), T(y,...),...</td>
<td>#P-complete</td>
</tr>
<tr>
<td>otherwise</td>
<td>PTIME</td>
</tr>
</tbody>
</table>

But: restrictions on Q, I_p

**Agenda item:**
Data complexity = f(k, specification lang, query lang)
Query Answering Using Views

Given: \( Q, V, J^P \)

Possible world: \( I^P \) s.t. \( V(I^P) = J^P \)

Compute: \( Q(I^P) = \text{Answer}(Q, V, J^P), \text{ top k} \)

**Theorem**[Dalvi&S.’05] \( \text{Answer}(Q, V, J^P) \) computable in exptime

**Corollary** \( \mu[Q \mid V] \) computable in exptime

\( p(n) = C/n^2 \)

But: restrictions on \( Q, J^P \)

**Agenda item:**
Data complexity = \( f(k, \text{specification lang, query lang}) \)
Query Approximation

Given: \( Q, I^P \)

Compute: approximate \( Q(I^P) \) in PTIME, top \( k \)

**Theorem** [Karp’89] Given a DNF expression \( E \), \( \Pr[E] \) can be approximated efficiently using Monte Carlo simulation

**Agenda item:**
Algorithm complexity = \( f(\# \text{ calls to query engine}) \)
Probabilistic XML

œ PXML: introduced by [Hung,Getoor,Subr.’03]

Still open:
œ Possible world semantics
œ Query evaluation

Agenda item:
Theory of probabilistic XML data
Outline

- Semistructured data: past and present
- Probabilistic databases: the new paradigm
- Supporting technologies
- Theoretical roots
- Towards a theory of probabilistic queries
- Conclusions
Conclusions

- Imprecisions in data are ubiquitous
- Data management is ripe for a paradigm shift
- Pre-existing technology, theory
- Lots of interest from the industry
- Kinds of theory: exploratory v.s. explanatory
Questions ?