The logical view on IR
IR as inference

q - query
d – document

retrieval:
search for documents which imply the query: \( d \rightarrow q \)

Example:  classical IR:  logical view:
\[
\begin{align*}
d &= \{t_1, t_2, t_3\} \\
q &= \{t_1, t_3\} \\
\text{retrieval: } q &\subseteq d \\
\text{retrieval: } d &\rightarrow q
\end{align*}
\]

\[
\begin{align*}
d &= t_1 \land t_2 \land t_3 \\
q &= t_1 \land t_3 \\
\text{retrieval: } d &\rightarrow q
\end{align*}
\]

advantage of inference-based approach:
step from term-based to knowledge-based retrieval
e.g. easy incorporation of additional knowledge

e.g.

example:
\[
\begin{align*}
d &: 'squares' \\
qu &: 'rectangles' \\
\text{thesaurus: } 'squares' &\rightarrow 'rectangles' \\
\Rightarrow: & d \rightarrow q
\end{align*}
\]
IR as uncertain inference

d: 'quadrangles'
q: 'rectangles'
⇒ uncertain knowledge required
'quadrangles' 0.3 'rectangles'

[Rijsbergen 86]:
IR as uncertain inference
Retrieval ⇔
estimate probability \( P(d \rightarrow q) = P(q|d) \)

Limitations of propositional logic:

conventional indexing (based on propositional logic): 

query: Is there a picture with a tree on the left of the house?
⇒ query cannot be expressed in propositional logic

predicate logic:

d: tree(t1). house(h1). left(h1,t1).
?- tree(X) & house(Y) & left(X,Y).

Relational Structures: Datalog

Datalog program: finite set of rules each expressing a conjunctive query

\[
\begin{align*}
& t(X_1, \ldots, X_k) : \neg r_1(U_{11}, \ldots, U_{1m_1}), \ldots, \neg r_n(U_{n1}, \ldots, U_{nm_n}) \\
& \text{where each variable } X_j \text{ occurs in the body of the rule (this way, every rule is safe). This corresponds to the logical formula} \\
& \forall X_1 \ldots \forall X_k \forall U_{11} \ldots \forall U_{nm_n} \quad t(X_1, \ldots, X_k) \land \neg r_1(U_{11}, \ldots, U_{1m_1}) \land \ldots \land \neg r_n(U_{n1}, \ldots, U_{nm_n}) \\
& \text{t}(X_1, \ldots, X_k) \text{ is called the head and} \\
& \text{r}_1(U_{11}, \ldots, U_{1m_1}), \ldots, \text{r}_n(U_{n1}, \ldots, U_{nm_n}) \text{ the body.} \\
& \text{A formula without a head is also called a \text{fact}.}
\end{align*}
\]

Datalog Properties

- horn predicate logic
- no functions
- restricted forms of negation allowed

\[
\begin{align*}
& t(X_1, \ldots, X_k) : \neg r_1(U_{11}, \ldots, U_{1m_1}), \ldots, \neg r_n(U_{n1}, \ldots, U_{nm_n}) \\
& \text{rules may be recursive (head predicate may occur in the body)} \\
& r(X, Y) : \neg l(X, Z), r(Z, Y) \\
& \text{sound and complete evaluation algorithms}
\end{align*}
\]
IR and Databases
The Logic View

Retrieval

- DB: given query \( q \), find objects \( o \) with \( o \rightarrow q \)
- IR: given query \( q \), find documents \( d \) with high values of \( P(d \rightarrow q) \)
- DB is a special case of IR!
  (in a certain sense)

This section: Focusing on the logic view

- Inference
- Vague predicates
- Query language expressiveness

Relational Model
Projection

<table>
<thead>
<tr>
<th>DOCNO</th>
<th>TERM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ir</td>
</tr>
<tr>
<td>1</td>
<td>db</td>
</tr>
<tr>
<td>2</td>
<td>ir</td>
</tr>
<tr>
<td>3</td>
<td>db</td>
</tr>
<tr>
<td>3</td>
<td>oop</td>
</tr>
<tr>
<td>4</td>
<td>ir</td>
</tr>
<tr>
<td>4</td>
<td>ai</td>
</tr>
<tr>
<td>5</td>
<td>db</td>
</tr>
<tr>
<td>5</td>
<td>oop</td>
</tr>
</tbody>
</table>

Projection: what is the collection about?
\( \text{topic}(T) := \text{index}(D,T) \).

Relational Model
Selection

<table>
<thead>
<tr>
<th>DOCNO</th>
<th>TERM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ir</td>
</tr>
<tr>
<td>1</td>
<td>db</td>
</tr>
<tr>
<td>2</td>
<td>ir</td>
</tr>
<tr>
<td>3</td>
<td>db</td>
</tr>
<tr>
<td>3</td>
<td>oop</td>
</tr>
<tr>
<td>4</td>
<td>ir</td>
</tr>
<tr>
<td>4</td>
<td>ai</td>
</tr>
<tr>
<td>5</td>
<td>db</td>
</tr>
<tr>
<td>5</td>
<td>oop</td>
</tr>
</tbody>
</table>

Selection: which documents are about IR?
\( \text{aboutir}(D) := \text{index}(D,\text{ir}) \).
Relational Model

Join

<table>
<thead>
<tr>
<th>DOCNO</th>
<th>TERM</th>
<th>author</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ir</td>
<td>smith</td>
</tr>
<tr>
<td>1</td>
<td>db</td>
<td>smith</td>
</tr>
<tr>
<td>2</td>
<td>ir</td>
<td>miller</td>
</tr>
<tr>
<td>3</td>
<td>db</td>
<td>johnson</td>
</tr>
<tr>
<td>3</td>
<td>oop</td>
<td>firefly</td>
</tr>
<tr>
<td>4</td>
<td>ir</td>
<td>bradford</td>
</tr>
<tr>
<td>4</td>
<td>ai</td>
<td>bates</td>
</tr>
<tr>
<td>5</td>
<td>db</td>
<td>firefly</td>
</tr>
<tr>
<td>5</td>
<td>oop</td>
<td>firefly</td>
</tr>
</tbody>
</table>

Join: who writes about IR?
irauthor(A) :- index(D,ir) & author(D,A).

Relational Model

Union

<table>
<thead>
<tr>
<th>DOCNO</th>
<th>TERM</th>
<th>irordb</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ir</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>ir</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>db</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>oop</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>ir</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>ai</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>db</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>oop</td>
<td></td>
</tr>
</tbody>
</table>

Union: which documents are about IR or DB?
irordb(D) :- index(D,ir).
irordb(D) :- index(D,db).

Relational Model

Difference

<table>
<thead>
<tr>
<th>DOCNO</th>
<th>TERM</th>
<th>irnotdb</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ir</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>db</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ir</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>db</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>oop</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>ir</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ai</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>db</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>oop</td>
<td></td>
</tr>
</tbody>
</table>

Difference: which documents are about IR, but not DB?
irnotdb(D) :- index(D,ir) & not(index(D,db)).

The Probabilistic Relational Model

<table>
<thead>
<tr>
<th>index</th>
<th>DOCNO</th>
<th>TERM</th>
<th>aboutdb</th>
<th>aboutirdb</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>1</td>
<td>IR</td>
<td>0.7</td>
<td>1</td>
</tr>
<tr>
<td>0.7</td>
<td>1</td>
<td>DB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td>2</td>
<td>IR</td>
<td>0.7</td>
<td>1</td>
</tr>
<tr>
<td>0.5</td>
<td>3</td>
<td>DB</td>
<td>0.5</td>
<td>3</td>
</tr>
<tr>
<td>0.8</td>
<td>3</td>
<td>OOP</td>
<td>0.8</td>
<td>5</td>
</tr>
<tr>
<td>0.9</td>
<td>4</td>
<td>IR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td>4</td>
<td>AI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>5</td>
<td>DB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.3</td>
<td>5</td>
<td>OOP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Which documents are about DB?
aboutdb(D) :- index(D,db).

Which documents are about IR and DB?
aboutirdb(D) :- index(D,ir) & index(D,db).
Extensional vs. intensional semantics

about(D,T) :- docTerm(D,T).
about(D,T) :- link(D,D1) & about(D1,T)
q(D) :- about(D,ir) & about(D,db).

Extensional semantics:
weight of derived fact as function of weights of subgoals
\[ P(q(d2)) = P(about(d2,ir)) \cdot P(about(d2,db)) = (0.7 \cdot 0.9) \cdot (0.7 \cdot 0.5) \]

Problem
"improper treatment of correlated sources of evidence" \[\text{[Pearl 88]}\]
→ extensional semantics only correct for tree-shaped inference structures

Intensional semantics

weight of derived fact as function of weights of underlying ground facts

Method: Event keys and event expressions

？- docTerm(D,ir) & docTerm(D,db).
gives
d1 [dT(d1,ir) & dT(d1,db)] 0.9 \cdot 0.5 = 0.45

Event keys and event expressions

about(D,T) :- docTerm(D,T).
about(D,T) :- link(D,D1) & about(D1,T)

？- about(D,ir) & about(D,db).
gives
d1 [dT(d1,ir) & dT(d1,db)] 0.9 \cdot 0.5 = 0.45
d2 [l(d2,d1) & l(d1,ir) & l(d2,d1) & dT(d1,db)] 0.7 \cdot 0.9 \cdot 0.5 = 0.315

Recursion

about(D,T) :- docTerm(D,T).
about(D,T) :- link(D,D1) & about(D1,T).

？- about(D,ir)
d1 [dT(d1,ir) | l(d1,d2) & l(d2,d3) & l(d3,d1) & dT(d1,ir) | ...] 0.900
d3 [l(d3,d1) & dT(d1,ir)] 0.720
d2 [l(d2,d3) & l(d3,d1) & dT(d1,ir)] 0.288

？- about(D,ir) & about(D,db)
d1 [dT(d1,ir) & dT(d1,db)] 0.450
d3 [l(d3,d1) & dT(d1,ir) & l(d3,d1) & dT(d1,db)] 0.360
d2 [l(d2,d3) & l(d3,d1) & dT(d1,ir) & dT(d1,db)] 0.144
Computation of probabilities for event expressions

1. transformation of expression into disjunctive normal form
2. application of sieve formula:
   - simple case of 2 conjuncts: \( P(a \lor b) = P(a) + P(b) - P(a \land b) \)
   - general case:
     \( c_i \) – conjunct of event keys
     \[
     P(c_1 \lor \ldots \lor c_n) = \sum_{j=1}^{n} (-1)^{i-1} \sum_{1 \leq i_1 < \ldots < i_j \leq n} P(c_{i_1} \land \ldots \land c_{i_j}).
     \]
   - \( \sim \) exponential complexity
   - \( \sim \) use only when necessary for correctness
   - see [Dalvi & Suciu 07]

Possible worlds semantics

\[
\begin{align*}
0.9 & \text{ docTerm(d1,ir).} \\
P(W_1) & = 0.9: \{\text{docTerm(d1,ir)}\} \\
P(W_2) & = 0.1: \{\} \\
\end{align*}
\]

Disjoint events

\[
\begin{align*}
0.6 & \text{ docTerm(d1,ir).} \quad 0.5 \text{ docTerm(d1,db).} \\
\end{align*}
\]

Possible interpretations:

\[
\begin{align*}
l_1: & \quad P(W_1) = 0.3: \{\text{docTerm(d1,ir)}\} \\
& \quad P(W_2) = 0.3: \{\text{docTerm(d1,ir), docTerm(d1,db)}\} \\
& \quad P(W_3) = 0.2: \{\text{docTerm(d1,db)}\} \\
& \quad P(W_4) = 0.2: \{\} \\
l_2: & \quad P(W_1) = 0.5: \{\text{docTerm(d1,ir)}\} \\
& \quad P(W_2) = 0.1: \{\text{docTerm(d1,ir), docTerm(d1,db)}\} \\
& \quad P(W_3) = 0.4: \{\text{docTerm(d1,db)}\} \\
l_3: & \quad P(W_1) = 0.1: \{\text{docTerm(d1,ir)}\} \\
& \quad P(W_2) = 0.5: \{\text{docTerm(d1,ir), docTerm(d1,db)}\} \\
& \quad P(W_3) = 0.4: \{\} \\
\end{align*}
\]

\[
\begin{array}{|c|c|c|}
\hline
\beta & \text{City} & \text{State} \\
\hline
0.7 & \text{Paris} & \text{France} \\
0.2 & \text{Paris} & \text{Texas} \\
0.1 & \text{Paris} & \text{Idaho} \\
\hline
\end{array}
\]

\[
\begin{align*}
\text{Interpretation:} \\
P(W_1) & = 0.7: \{\text{cityState(paris, france)}\} \\
P(W_2) & = 0.2: \{\text{cityState(paris, texas)}\} \\
P(W_3) & = 0.1: \{\text{cityState(paris, idaho)}\} \\
\end{align*}
\]

probabilistic logic:

\[
0.1 \leq P(\text{docTerm(d1,ir)} \& \text{docTerm(d1,db)}) \leq 0.5
\]

probabilistic Datalog with independence assumptions:

\[
P(\text{docTerm(d1,ir)} \& \text{docTerm(d1,db)}) = 0.3
\]
Relational Bayes

[Roelleke et al. 07]

Role of the relational Bayes: Generation of a probabilistic database

Non-probabilistic database

Bayes

Probabilistic database

Example: \( P(\text{Nationality} \mid \text{City}) \)

<table>
<thead>
<tr>
<th>Nationality</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>British</td>
<td>London</td>
</tr>
<tr>
<td>British</td>
<td>London</td>
</tr>
<tr>
<td>Scottish</td>
<td>London</td>
</tr>
<tr>
<td>French</td>
<td>London</td>
</tr>
<tr>
<td>German</td>
<td>Hamburg</td>
</tr>
<tr>
<td>German</td>
<td>Hamburg</td>
</tr>
<tr>
<td>Danish</td>
<td>Hamburg</td>
</tr>
<tr>
<td>British</td>
<td>Hamburg</td>
</tr>
<tr>
<td>German</td>
<td>Dortmund</td>
</tr>
<tr>
<td>German</td>
<td>Dortmund</td>
</tr>
<tr>
<td>Turkish</td>
<td>Dortmund</td>
</tr>
<tr>
<td>Scottish</td>
<td>Glasgow</td>
</tr>
</tbody>
</table>

\[
P(\text{Nationality} \mid \text{City}) = \frac{\text{Support}(\text{Nationality} \land \text{City})}{\text{Support}(\text{City})}
\]

Example: \( P(t \mid d) \)

<table>
<thead>
<tr>
<th>Term</th>
<th>DocId</th>
</tr>
</thead>
<tbody>
<tr>
<td>sailing</td>
<td>doc1</td>
</tr>
<tr>
<td>sailing</td>
<td>doc2</td>
</tr>
<tr>
<td>sailing</td>
<td>doc3</td>
</tr>
<tr>
<td>sailing</td>
<td>doc4</td>
</tr>
<tr>
<td>sailing</td>
<td>doc5</td>
</tr>
<tr>
<td>boats</td>
<td>doc1</td>
</tr>
<tr>
<td>boats</td>
<td>doc2</td>
</tr>
<tr>
<td>boats</td>
<td>doc3</td>
</tr>
<tr>
<td>boats</td>
<td>doc4</td>
</tr>
<tr>
<td>boats</td>
<td>doc5</td>
</tr>
<tr>
<td>east</td>
<td>doc3</td>
</tr>
<tr>
<td>coast</td>
<td>doc3</td>
</tr>
<tr>
<td>coast</td>
<td>doc4</td>
</tr>
<tr>
<td>coast</td>
<td>doc5</td>
</tr>
<tr>
<td>sailing</td>
<td>doc4</td>
</tr>
<tr>
<td>sailing</td>
<td>doc5</td>
</tr>
</tbody>
</table>

\[
P(t \mid d) = \frac{\text{Support}(t \land \text{DocId})}{\text{Support}(\text{DocId})}
\]

Probabilistic rules

Rules for deterministic facts:

0.7 \( \text{likes-sports}(X) :- \text{man}(X) \).
0.4 \( \text{likes-sports}(X) :- \text{woman}(X) \).
\( \text{man}(\text{peter}) \).

Interpretation:

\[
P(W_1) = 0.7: \{\text{man}(\text{peter}), \text{likes-sports}(\text{peter})\}
\]

\[
P(W_2) = 0.3: \{\text{man}(\text{peter})\}
\]
# Probabilistic rules

## Rules for uncertain facts:

```prolog
# gender is disjoint on the first attribute
0.7 l-sports(X) :- gender(X,male).
0.4 l-sports(X) :- gender(X,female).
0.5 gender(X,male) :- human(X).
0.5 gender(X,female) :- human(X).
human(jo).
```

**Interpretation:**

$$P(W_1) = 0.35: \{\text{gender(jo,male)}, \text{l-sports(jo)}\}$$

$$P(W_2) = 0.15: \{\text{gender(jo,male)}\}$$

$$P(W_3) = 0.20: \{\text{gender(jo,female)}, \text{l-sports(jo)}\}$$

$$P(W_4) = 0.30: \{\text{gender(jo,female)}\}$$

?- l-sports(jo)  

$$P(W_1) + P(W_3) = 0.55$$

## Rules for independent events

### Modeling probabilistic inference networks

```prolog
sameauthor(D1,D2) :- author(D1,X) & author(D2,X).
0.5 link(D1,D2) :- refer(D1,D2).
0.2 link(D1,D2) :- sameauthor(D1,D2).

?? link(D1,D2) :- refer(D1,D2) & sameauthor(D1,D2).

P(l|r), P(l|s) \rightarrow P(l|r \land s)\
```

### Vague Predicates

**Probabilistic inference networks, rules define link matrix**

![Diagram of Probabilistic Inference Networks]
Vague Predicates
Motivating Example

Current IR systems are based on proposition logic (query term present/absent in document)

Similarity of values not considered

but multimedia IR deals with similarity already

⇝ transition from propositional to predicate logic necessary

Vague Predicates in Probabilistic Datalog

Example: Shopping 45 inch LCD TV

<table>
<thead>
<tr>
<th>X ≈ Y</th>
<th>( \beta )</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7</td>
<td>42</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>43</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>0.9</td>
<td>44</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>45</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>0.9</td>
<td>46</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>47</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Data types and vague predicates in IR

Data type: domain + (vague) predicates

- Language (multilingual documents) / (language-specific stemming)
- Person names / “his name sounds like Jones”
- Dates / “about a month ago”
- Amounts / “orders exceeding 1 Mio $”
- Technical measurements / “at room temperature”
- Chemical formulas
Vague Criteria in Fact Databases

"I am looking for a 45-inch LCD TV with
▶ wide viewing angle
▶ high contrast
▶ low price
▶ high user rating”

→ vague criteria are very frequent in end-user querying of fact databases
→ but no appropriate support in SQL

vague conditions → similar to fuzzy predicates

Probabilistic Modeling of Vague Predicates

[Fuhr 90]

▶ learn vague predicates from feedback data
▶ construct feature vector $\vec{x}(q_i,d_i)$ from query value $q_i$ and document value $d_i$ (e.g. relative difference)
▶ apply logistic regression

Expressiveness
Formulating Retrieval Rules

about(D,T) :- docTerm(D,T).
consider document linking / anchor text
about(D,T) :- link(D1,D),about(D1,T).
consider term hierarchy
about(D,T) :- subconcept(T,T1) & about(D,T1).
field-specific term weighting
0.9 docTerm(D,T) :- occurs(D,T,title).
0.5 docTerm(D,T) :- occurs(D,T,body).
IR authors:

\[ \text{irauthor}(N) :- \text{about}(D, \text{ir}) \land \text{author}(D, N). \]

Smith’s IR papers cited by Miller

?- \text{author}(D, \text{smith}) \land \text{about}(D, \text{ir}) \land \\
\text{author}(D1, \text{miller}) \land \text{cites}(D, D1).

### Expressiveness

#### Aggregation (1)

Who are the major IR authors?

<table>
<thead>
<tr>
<th>DNO</th>
<th>TERM</th>
<th>β</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>ir</td>
<td>1</td>
<td>smith</td>
</tr>
<tr>
<td>0.8</td>
<td>db</td>
<td>1</td>
<td>smith</td>
</tr>
<tr>
<td>0.6</td>
<td>ir</td>
<td>2</td>
<td>miller</td>
</tr>
<tr>
<td>0.8</td>
<td>ir</td>
<td>3</td>
<td>smith</td>
</tr>
<tr>
<td>0.7</td>
<td>ai</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Aggregation through projection!

### Expressiveness

#### Aggregation (2)

Who are the major IR authors?

\[
\begin{array}{|c|c|c|c|c|}
\hline
\beta & DNO & TERM & DNO & NAME \\
\hline
0.9 & 1 & ir & 1 & smith \\
0.8 & 1 & db & 1 & smith \\
0.6 & 2 & ir & 2 & miller \\
0.8 & 3 & ir & 3 & smith \\
0.7 & 3 & ai &  & \\
\hline
\end{array}
\]

Aggregation through summing:

\[ \text{irauthor}(A) :- \text{index}(D, \text{ir}) \land \text{author}(D, A). \]

\[ \text{ira} \text{auths} \text{ SUM}(\text{Name}) :- \text{ir} \text{dbauth}(\text{Doc}, \text{Name}) \mid (\text{Name}) \]

### Expressiveness in XML Retrieval

[Fuhr & Lalmas 07]

Content Typing

Object Types

Data Types

Text only

- Nested structure
- Named fields
- XPath
- XQuery
- Structure
XML structure: 1. Nested Structure

- XML document as hierarchical structure
- Retrieval of elements (subtrees)
- Typical query language does not allow for specification of structural constraints
- Relevance-oriented selection of answer elements: return the most specific relevant elements

XML structure: 2. Named Fields

- Reference to elements through field names only
- Context of elements is ignored (e.g. author of article vs. author of referenced paper)
- Post-Coordination may lead to false hits (e.g. author name – author affiliation)

Example: Dublin Core
  <dc:title>Generic Algebras</dc:title>
  <dc:creator>A. Smith (ESI), B. Miller (CMU)</dc:creator>
  <dc:subject>Orthogonal group, Symplectic group</dc:subject>
  <dc:date>2001-02-27</dc:date>
  <dc:format>application/postscript</dc:format>
  <dc:source>ESI preprints</dc:source>
  <dc:language>en</dc:language>
</oai_dc:dc>

XML structure: 3. XPath

/document/chapter[about(./heading, XML) AND about(./section//*,syntax)]

XML structure: 3. XPath (cont’d)

- Full expressiveness for navigation through document tree (+links)
  - Parent/child, ancestor/descendant
  - Following/preceding, following-sibling, preceding-sibling
  - Attribute, namespace
- Selection of arbitrary elements/subtrees (but answer can be only a single element of the originating document)
XML structure: 4. XQuery

Higher expressiveness, especially for database-like applications:
- Joins (trees → graphs)
- Aggregations
- Constructors for restructuring results

Example: List each publisher and the average price of its books
FOR $p$ IN distinct(document("bib.xml")//publisher)
LET $a := \text{avg}(\text{document("bib.xml")//book[publisher = $p]/price})$
RETURN
<publisher>
  <name> $p$/text() </name>
  <avgprice> $a$ </avgprice>
</publisher>

XML content typing: 1. Text

<book>
  <author>John Smith</author>
  <title>XML Retrieval</title>
  <chapter> <heading>Introduction</heading>
    This text explains all about XML and IR.
  </chapter>
  <chapter>
    <heading>XML Query Language XQL</heading>
    <section>
      <heading>Examples</heading>
    </section>
    <section>
      <heading>Syntax</heading>
      Now we describe the XQL syntax.
    </section>
  </chapter>
</book>

XML content typing: 2. Data Types

- Data type: domain + (vague) predicates (see above)
- Close relationship to XML Schema, but
  - XMLS supports syntactic type checking only
  - No support for vague predicates
Pablo Picasso (October 25, 1881 - April 8, 1973) was a Spanish painter and sculptor. In Paris, Picasso entertained a distinguished coterie of friends in the Montmartre and Montparnasse quarters, including André Breton, Guillaume Apollinaire, and writer Gertrude Stein.

To which other artists did Picasso have close relationships? Did he ever visit the USA?

Named entity recognition methods allow for automatic markup of object types

Object types support increased precision

Tag semantics modelled in OWL
Semantic Web (ontology) languages

**RDF:** “Resource description language”
- semantic markup language, only resources and their properties, serialisation in XML

**RDFS:** “RDF Schema”, schema definition language for RDF

**OWL:** extends RDF/RDFS by richer modelling primitives,
- OWL Lite/DL/Full
  - OWL Lite contains simple primitives
  - OWL DL corresponds to expressive description logic
  - OWL Full is OWL DL + RDF

knowledge base can be modelled as collection of RDF triples (RDF/XML serialisation)
- alternative encoding: abstract syntax (easier to read)

**Objects, classes, literals and datatypes**

- Two distinct domains:
  - **Classes:** for objects
  - **Data types:** for literals

**Classes (1)**

Class(Female partial Animal)

```xml
<owl:Class rdf:ID="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <owl:disjointWith rdf:resource="#Male"/>
</owl:Class>
```

```xml
<Person rdf:type owl:Class rdf:ID="Person"/>
<Person rdf:type rdfs:subClassOf owl:Class rdf:ID="Animal"/>
<Person rdf:type rdfs:subClassOf owl:Class rdf:ID="Man"/>
<Person rdf:type rdfs:subClassOf owl:Class rdf:ID="Human"/>
```
Classes (2)

Class(Male partial Animal)
DisjointClasses(Male Female)

```xml
<owl:Class rdf:ID="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <owl:disjointWith rdf:resource="#Female"/>
</owl:Class>
```

Object properties (2)

```xml
<owl:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
  <owl:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</owl:Class>
```

Datatype properties

```xml
<owl:DatatypeProperty rdf:ID="shoesize">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="xsd:decimal"/>
  <rdf:type rdf:resource="owl:FunctionalProperty"/>
</owl:Class>
```
Property restrictions

Class(Person partial Animal restriction(hasParent allValuesFrom(Person))
    restriction(hasParent cardinality(2)))

<owl:Class rdf:ID="Person">
    <rdfs:subClassOf rdf:resource="#Animal"/>
    <owl:Restriction>
        <owl:onProperty rdf:resource="#hasParent"/>
        <owl:allValuesFrom rdf:resource="#Person"/>
    </owl:Restriction>
</rdfs:subClassOf>
<owl:Restriction>
    <owl:onProperty rdf:resource="#hasParent"/>
    <owl:cardinality>2</owl:cardinality>
</owl:Restriction>
</owl:Class>

Instances

Individual(Kain type(Male) value(hasFather Adam)
    value(hasMother Eve)
    value(shoesize 10))

<Kain rdf:ID="Kain">
    <hasFather rdf:resource="#Adam"/>
    <hasMother rdf:resource="#Eve"/>
    <shoesize>10</shoesize>
</Kain>

Further modelling primitives

owl:inverseOf: inverse property: \( p(a, b) \leftrightarrow r(b, a) \)
owl:TransitiveProperty: \( p(a, b), p(b, c) \rightarrow p(a, c) \)
owl:SymmetricProperty: \( p(a, b) \rightarrow p(b, a) \)
owl:InverseFunctionalProperty: inverse property is functional
owl:hasValue at least one property value equals object or datatype value
owl:someValuesFrom at least one property value is instance of class, expression or datatype
owl:interSectionOf, owl:unionOf, owl:complementOf: boolean combinations of class expressions
owl:oneOf: define class by enumerating its instances

Limitations of OWL

OWL lacks support for

- **uncertainty**: only deterministic relationships possible, no weighting or probabilistic facts
  ⇒ “Pr(hasFather(lisa,thomas))=0.9” cannot be expressed

- **rules**: no general rules, only specific rules like subClassOf, TransitiveProperty . . .
  ⇒ “if hasParent(A,B) and hasParent(C,D) and hasSibling(B,D), then hasCousion(A,C)” cannot be expressed

- **n-ary datatype predicates**: OWL datatypes are based on XML Schema datatypes, thus providing only unary datatype predicates
  ⇒ “sameDomain(foo@bar.de,baz@bar.de)” cannot be expressed
  ⇒ IR queries cannot be expressed directly in OWL
OWL: Conclusion

- OWL extends RDF(S) by additional modelling primitives
- well-defined semantics, based on description logics
- does not support all RDF features (no reification, only three levels owl:Class, classes and objects)
- lacks important features:
  - only deterministic features, no probabilistic relationships
  - no rules (but in SWRL)
  - restricted datatype predicates (due to XML Schema)
- OWL and associated languages become standard in the Semantic Web

Semantics Web Layers

SPARQL

query language for getting information from RDF (OWL) graphs

Facilities for

- extract information in the form of URIs, blank nodes, plain and typed literals
- extract RDF subgraphs
- construct new RDF graphs based on information in the queried graphs

Features:

- matching graph patterns
- variables – global scope; indicated by '?' or '$'

SPARQL: Basic Graph Pattern

- Set of Triple Patterns
  - Triple Pattern – similar to an RDF Triple (subject, predicate, object), but any component can be a query variable; literal subjects are allowed
- Matching a triple pattern to a graph: bindings between variables and RDF Terms
- Matching of Basic Graph Patterns
  - A Pattern Solution of Graph Pattern GP on graph G is any substitution S such that S(GP) is a subgraph of G.
  - SELECT ?x ?v WHERE ?x ?v ?x
SPARQL: Group Patterns + Value Constraints

Group Pattern: A set of graph patterns which must all match
Value Constraints: restrict RDF terms in a solution
SELECT ?n WHERE
  ?n profession "Physicist" . ?n isa "Politician"

SPARQL: Query forms

SELECT returns all, or a subset of the variables bound in a
query pattern match
formats: XML or RDF/XML
CONSTRUCT returns an RDF graph constructed by substituting
variables in a set of triple templates
DESCRIBE returns an RDF graph that describes the resources
found.
ASK returns whether a query pattern matches or not.

Conclusion and Outlook

Conclusion
Inference
- Probabilistic relational model supports integration of IR+DB
- Probabilistic Datalog as powerful inference mechanism
- Allows for formulating retrieval strategies as logical rules

Vague predicates
- Natural extension of IR methods to attribute values
- Vague predicates can be learned from feedback data
- Transition from propositional to predicate logic

Expressive query language
- Joins
- Aggregations
- (Re)structuring of results
Outlook
IR Systems vs. DBMS

Separation between IRS and IR application?

Towards an IRMS

Don’t program search engines, design them

http://www.spinque.com/


