IR Models based on Predicate Logic

Norbert Fuhr

The logical view on IR

IR as Inference
IR as uncertain inference
Propositional vs. Predicate Logic

advantage of inference-based approach:
step from term-based to knowledge-based retrieval
e.g. easy incorporation of additional knowledge
example:
d: 'squares'
q: 'rectangles'
thesaurus: 'squares' → 'rectangles'
⇒: \( d \rightarrow q \)

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

Example: classical IR: logical view:
\[ d = \{ t_1, t_2, t_3 \} \]
\[ q = \{ t_1, t_3 \} \]
retrieval: \( q \subset d \) ? retrieval: \( d \rightarrow q \) ?
IR as uncertain inference

\[ d: \text{'quadrangles'} \]

\[ q: \text{'rectangles'} \]

⇒ uncertain knowledge required

\[ \text{'quadrangles'} \rightarrow \text{'rectangles'} \]

[Rijsbergen 86]: IR as uncertain inference

 Retrieval \( \hat{=} \) estimate probability

\[ P(d \rightarrow q) = P(q|d) \]

Limitations of propositional logic:

conventional indexing (based on propositional logic): \( d = \{ \text{tree, house} \} \)

query: Is there a picture with a tree on the left of the house?

⇒ query cannot be expressed in propositional logic

predicate logic:

\[ d: \text{tree}(t_1). \text{house}(h_1). \text{left}(h_1,t_1). \]

\[ ?- \text{tree}(X) & \text{house}(Y) & \text{left}(X,Y). \]

Relational Structures: Datalog

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

\[ t(X_1, ..., X_k) : \neg r_1(U_{11}, ..., U_{1m_1}), ..., \neg r_n(U_{n1}, ..., U_{nm_n}) \]

where each variable \( X_i \) occurs in the body of the rule (this way, every rule is safe). This corresponds to the logical formula

\[ \forall X_1 \ldots \forall X_k \forall U_{11} \ldots \forall U_{nm_n} \]

\[ t(X_1, ..., X_k) \land \neg r_1(U_{11}, ..., U_{1m_1}) \land \ldots \land \neg r_n(U_{n1}, ..., U_{nm_n}) \]

\( t(X_1, ..., X_k) \) is called the head and

\( r_1(U_{11}, ..., U_{1m_1}), ..., r_n(U_{n1}, ..., U_{nm_n}) \) the body.

A formula without a head is also called a fact

Datalog Properties

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

\[ t(X_1, ..., X_k) : \neg r_1(U_{11}, ..., U_{1m_1}), ..., \neg r_n(U_{n1}, ..., U_{nm_n}) \]

- rules may be recursive (head predicate may occur in the body)

\[ r(X, Y) : \neg l(X, Z), r(Z, Y) \]

- sound and complete evaluation algorithms
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

Inference

IR with the Relational Model
The Probabilistic Relational Model
Interpretation of probabilistic weights
Extensions
  - Disjoint events
  - Relational Bayes
  - Probabilistic rules

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?
$\text{topic}(T) := \text{index}(D,T)$.  

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

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

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

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

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>DOC</th>
<th>TERM</th>
<th>( \beta )</th>
<th>S</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>d1</td>
<td>ir</td>
<td>0.7</td>
<td>d2</td>
<td>d1</td>
</tr>
<tr>
<td>0.5</td>
<td>d1</td>
<td>db</td>
<td>( \beta )</td>
<td>0.7</td>
<td>d2</td>
</tr>
</tbody>
</table>

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" [Pearl 88] → extensional semantics only correct for tree-shaped inference structures

Event keys and event expressions

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>( \kappa )</th>
<th>DOC</th>
<th>TERM</th>
<th>( \beta )</th>
<th>( \kappa )</th>
<th>S</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>dT(d1,ir)</td>
<td>d1</td>
<td>ir</td>
<td>0.7</td>
<td>l(d2,d1)</td>
<td>d2</td>
<td>d1</td>
</tr>
<tr>
<td>0.5</td>
<td>dT(d1,db)</td>
<td>d1</td>
<td>db</td>
<td>( \beta )</td>
<td>( \kappa )</td>
<td>0.7</td>
<td>l(d2,d1)</td>
</tr>
</tbody>
</table>

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

Intensional semantics

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

Method: Event keys and event expressions

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>( \kappa )</th>
<th>DOC</th>
<th>TERM</th>
<th>( \beta )</th>
<th>( \kappa )</th>
<th>S</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>dT(d1,ir)</td>
<td>d1</td>
<td>ir</td>
<td>0.7</td>
<td>l(d2,d1)</td>
<td>d2</td>
<td>d1</td>
</tr>
</tbody>
</table>

Recursion

\?- 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
     \[
     P(c_1 \lor \ldots \lor c_n) = \sum_{i=1}^{n} (-1)^{i-1} \sum_{1 \leq h < \ldots < j \leq n} P(c_h \land \ldots \land c_j).
     \]
   
   ▶ exponential complexity
   ▶ use only when necessary for correctness
   ▶ see [Dalvi & Suciu 07]

Possible worlds semantics

0.9 \( \text{docTerm}(d1, \text{ir}) \).

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

0.6 \( \text{docTerm}(d1, \text{ir}) \). 0.5 \( \text{docTerm}(d1, \text{db}) \).

Possible interpretations:

- \( I_1 \):
  \[
  \begin{align*}
P(W_1) &= 0.3: \{\text{docTerm}(d1, \text{ir})\} \\
P(W_2) &= 0.3: \{\text{docTerm}(d1, \text{ir}), \text{docTerm}(d1, \text{db})\} \\
P(W_3) &= 0.2: \{\text{docTerm}(d1, \text{db})\} \\
P(W_4) &= 0.2: \{\}
\end{align*}
\]

- \( I_2 \):
  \[
  \begin{align*}
P(W_1) &= 0.5: \{\text{docTerm}(d1, \text{ir})\} \\
P(W_2) &= 0.1: \{\text{docTerm}(d1, \text{ir}), \text{docTerm}(d1, \text{db})\} \\
P(W_3) &= 0.4: \{\text{docTerm}(d1, \text{db})\} \\
P(W_4) &= 0.4: \{\}
\end{align*}
\]

- \( I_3 \):
  \[
  \begin{align*}
P(W_1) &= 0.1: \{\text{docTerm}(d1, \text{ir})\} \\
P(W_2) &= 0.5: \{\text{docTerm}(d1, \text{ir}), \text{docTerm}(d1, \text{db})\} \\
P(W_3) &= 0.4: \{\text{docTerm}(d1, \text{db})\} \\
P(W_4) &= 0.4: \{\}
\end{align*}
\]

Disjoint events

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

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

probabilistic logic:

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

probabilistic Datalog with independence assumptions:

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

[Roelleke et al. 07]

Role of the relational Bayes: Generation of a probabilistic database

![Diagram of non-probabilistic database and Bayes leading to probabilistic database](image)

Relational Bayes

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

<table>
<thead>
<tr>
<th>Nationality</th>
<th>City</th>
<th>$P(\text{Nationality} \mid \text{City})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;British&quot;</td>
<td>&quot;London&quot;</td>
<td>0.600</td>
</tr>
<tr>
<td>&quot;British&quot;</td>
<td>&quot;London&quot;</td>
<td>0.200</td>
</tr>
<tr>
<td>&quot;Scottish&quot;</td>
<td>&quot;London&quot;</td>
<td>0.200</td>
</tr>
<tr>
<td>&quot;German&quot;</td>
<td>&quot;Hamburg&quot;</td>
<td>0.500</td>
</tr>
<tr>
<td>&quot;German&quot;</td>
<td>&quot;Hamburg&quot;</td>
<td>0.250</td>
</tr>
<tr>
<td>&quot;British&quot;</td>
<td>&quot;Hamburg&quot;</td>
<td>0.667</td>
</tr>
<tr>
<td>&quot;German&quot;</td>
<td>&quot;Dortmund&quot;</td>
<td>0.333</td>
</tr>
<tr>
<td>&quot;Turkish&quot;</td>
<td>&quot;Dortmund&quot;</td>
<td>1.000</td>
</tr>
<tr>
<td>&quot;Scottish&quot;</td>
<td>&quot;Glasgow&quot;</td>
<td></td>
</tr>
</tbody>
</table>

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:

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

$P(W_2) = 0.3: \{\text{man(peter)}\}$
Probabilistic rules
Rules for uncertain facts:

# gender is disjoint on the first attribute
0.7 \text{1-sports}(X) \quad :- \text{gender}(X,\text{male}).
0.4 \text{1-sports}(X) \quad :- \text{gender}(X,\text{female}).
0.5 \text{gender}(X,\text{male}) \quad :- \text{human}(X).
0.5 \text{gender}(X,\text{female}) \quad :- \text{human}(X).
\text{human}(\text{jo}).

Interpretation:
\[ P(W_1) = 0.35: \{\text{gender}(\text{jo},\text{male}), \text{1-sports}(\text{jo})\} \]
\[ P(W_2) = 0.15: \{\text{gender}(\text{jo},\text{male})\} \]
\[ P(W_3) = 0.20: \{\text{gender}(\text{jo},\text{female}), \text{1-sports}(\text{jo})\} \]
\[ P(W_4) = 0.30: \{\text{gender}(\text{jo},\text{female})\} \]
\[ \text{?- 1-sports}(\text{jo}) \quad P(W_1) + P(W_3) = 0.55 \]

Rules for independent events
Modeling probabilistic inference networks

sameauthor(D1,D2) :- author(D1,X) & author(D2,X).
0.5 \text{link}(D1,D2) :- refer(D1,D2).
0.2 \text{link}(D1,D2) :- sameauthor(D1,D2).

\[ \text{?- link}(D1,D2) \quad \text{refer}(D1,D2) & \text{sameauthor}(D1,D2). \]

\[ P(l|r), P(l|s) \rightarrow P(l|r \land s)? \]

Vague Predicates
The Logical View on Vague Predicates
Vague Predicates in IR and Databases
Probabilistic Modeling of Vague Predicates
Propositional vs. Predicate Logic

- 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
- \(\Rightarrow\) transition from propositional to predicate logic necessary

Vague Predicates in Probabilistic Datalog

[Fuhr & Roelleke 97] [Fuhr 00]

\[ X \approx Y \]

<table>
<thead>
<tr>
<th>(\beta)</th>
<th>(X)</th>
<th>(Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7</td>
<td>42</td>
<td>45</td>
</tr>
<tr>
<td>0.8</td>
<td>43</td>
<td>45</td>
</tr>
<tr>
<td>0.9</td>
<td>44</td>
<td>45</td>
</tr>
<tr>
<td>1.0</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>0.9</td>
<td>46</td>
<td>45</td>
</tr>
<tr>
<td>0.8</td>
<td>47</td>
<td>45</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 ⃗x(q, d) from query value q and document value d (e.g. relative difference)
▶ apply logistic regression

Expressiveness

Expressiveness

Retrieve Rules, Joins, Aggregations and Restructuring
Expressiveness in XML Retrieval

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).
**Expressiveness**

Joins

*IR authors:*

irauthor(N):- about(D,ir) & author(D,N).

Smith's IR papers cited by Miller

?- author(D,smith) & about(D,ir) & author(D1,miller) & cites(D,D1).

---

**Expressiveness**

Aggregation (1)

Who are the major IR authors?

<table>
<thead>
<tr>
<th>index</th>
<th>DNO</th>
<th>TERM</th>
<th>author</th>
<th>DNO</th>
<th>NAME</th>
<th>irauthor</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>1</td>
<td>ir</td>
<td></td>
<td>1</td>
<td>smith</td>
<td>0.98</td>
</tr>
<tr>
<td>0.8</td>
<td>1</td>
<td>db</td>
<td></td>
<td>1</td>
<td>smith</td>
<td>0.6</td>
</tr>
<tr>
<td>0.6</td>
<td>2</td>
<td>ir</td>
<td></td>
<td>2</td>
<td>miller</td>
<td>0.8</td>
</tr>
<tr>
<td>0.8</td>
<td>3</td>
<td>ir</td>
<td></td>
<td>3</td>
<td>smith</td>
<td>0.7</td>
</tr>
<tr>
<td>0.7</td>
<td>3</td>
<td>ai</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

irauthor(A):- index(D,ir) & author(D,A).

Aggregation through projection!

---

**Expressiveness**

Aggregation (2)

Who are the major IR authors?

<table>
<thead>
<tr>
<th>index</th>
<th>DNO</th>
<th>TERM</th>
<th>author</th>
<th>DNO</th>
<th>NAME</th>
<th>irauthors</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>1</td>
<td>ir</td>
<td></td>
<td></td>
<td></td>
<td>0.98</td>
</tr>
<tr>
<td>0.9</td>
<td>1</td>
<td>db</td>
<td></td>
<td></td>
<td></td>
<td>0.6</td>
</tr>
<tr>
<td>0.8</td>
<td>2</td>
<td>ir</td>
<td></td>
<td></td>
<td></td>
<td>0.6</td>
</tr>
<tr>
<td>0.8</td>
<td>3</td>
<td>ir</td>
<td></td>
<td></td>
<td></td>
<td>0.7</td>
</tr>
<tr>
<td>0.7</td>
<td>3</td>
<td>ai</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

irauth(D,A):- index(D,ir) & author(D,A).

irauthors SUM(Name) :- irdbauth(Doc,Name) | (Name)

---

**Expressiveness in XML Retrieval**

[Fuhr & Lalmas 07]
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

Example: Dublin Core

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

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)

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 $a :=$ avg(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>
  //chapter[about(., XML query language]

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

XML content typing
Tag semantics modelled in OWL

Retrieval of structured documents: POOL

Structure of POOL programs
Contexts and augmentation
Augmentation and inconsistencies
Goals

- retrieval of structured documents  
  → hierarchical logical structure
- abstraction from node types  
  → contexts as untyped nodes
- multimedia retrieval  
  → expressiveness of restricted predicate logic

Structure of POOL programs

object: identifier + content
context: object with nonempty content
  a1[], s11[], s12[]
program: set of clauses
  clause: context / proposition / rule
  proposition:
    ▶ term
      (image, presentation)
    ▶ classification
      (article(a1), section(s11))
    ▶ attribute
      (s11.author(smith), a1.pubyear(1997))

POOL Example Program

a1[ s11[ image 0.6 retrieval presentation ]
    s12[ ss121[ audio indexing ]
        ss122[ video not presentation ] ] ]
s11.author(smith)
s121.author(miller) s122.author(jones)
a1.pubyear(1997)
article(a1) section(s11) section(s12)
subsection(ss121) subsection(ss122)

Rules in POOL

rule: head :- body
head: proposition / context containing a proposition
body: conjunction of subgoals (propositions or contexts)
docnode(D) :- article(D)
docnode(D) :- section(D)
docnode(D) :- subsection(D)
mm-ir-doc(D) :- docnode(D) &
  D[audio & retrieval]
german-paper(D) :- D.author.country(germany)

query: ?- body
?- D[audio & indexing]
Contexts and augmentation

clauses only hold for context where stated

**augmentation:**
propagation of propositions to surrounding contexts

\[ a_1[s_1[\text{image} \ 0.6 \text{retrieval} \ \text{presentation}]] \]
\[ s_{12}[s_{121}[\text{audio indexing}]] \]
\[ s_{122}[\text{video} \ \text{not presentation}]] \]

?- D[\text{audio & video}]
\[ \sim s_{12} \]
\[ \sim a_1 \]

Augmentation with Uncertainty:

?- audio
\[ \sim 1.00 \ s_{121} \]
\[ \sim 0.60 \ s_{12} \]
\[ \sim 0.36 \ a_1 \]

?- D[\text{audio & video}]
\[ \sim 0.36 \ s_{12} \]
\[ \sim 0.22 \ a_1 \]

augmentation with uncertainty prefers most specific context!

Augmentation and Inconsistencies

\[ d_1[s_1[\text{audio indexing}]] \]
\[ s_{21}[s_{211}[\text{image} \ \text{retrieval}]] \]
\[ s_{22}[\text{video} \ \text{not retrieval}]] \]

?- D[\text{audio & indexing}]
\[ \sim s_1 \]
\[ \sim d_1 \]

?- D[video & image]
\[ \sim s_2 \]
\[ \sim d_1 \]

?- D[video & retrieval]
\[ \sim \text{retrieval is inconsistent in } s_2 \]

Four-Valued Logic

<table>
<thead>
<tr>
<th>truth values: unknown, true, false, inconsistent</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \wedge )</td>
</tr>
<tr>
<td>( \vee )</td>
</tr>
<tr>
<td>( \neg )</td>
</tr>
</tbody>
</table>

\( T = \{ \text{true} \} \)
\( F = \{ \text{false} \} \)
\( U = \{ \text{true}, \text{false} \} \)
\( I = \{ \} \)
Applying Four-Valued Logic

d1[ s1[ audio indexing ]
   s2[ s21[ image retrieval]
   s22[ video not retrieval ] ] ]

s22:
video \rightarrow true
image \rightarrow unknown
retrieval \rightarrow false
s2:
image \rightarrow true
video \rightarrow true
retrieval \rightarrow inconsistent

Description logic

Thesaurus
Introduction into OWL
SPARQL

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)

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

Class (Male partial Animal)

DisjointClasses (Male Female)

(ObjectProperty)

Object properties (1)

ObjectProperty (hasParent domain(Animal) range(Animal))

<owl:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</owl:ObjectProperty>
Object properties (2)

ObjectProperty(hasFather super(hasParent) range(Male))

Datatype properties

DatatypeProperty(shoesize Functional domain(Animal) range(xsd:decimal))

Property restrictions

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

Instances

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

OWL Class Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
<th>Modal Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>( C_1 \cap \ldots \cap C_n )</td>
<td>Human ( \cap ) Male</td>
<td>( C_1 \wedge \ldots \wedge C_n )</td>
</tr>
<tr>
<td>unionOf</td>
<td>( C_1 \cup \ldots \cup C_n )</td>
<td>Doctor ( \cup ) Lawyer</td>
<td>( C_1 \vee \ldots \vee C_n )</td>
</tr>
<tr>
<td>complementOf</td>
<td>( \neg C )</td>
<td>( \neg )Male</td>
<td>( \neg )C</td>
</tr>
<tr>
<td>oneOf</td>
<td>( {x_1 } \cup \ldots \cup {x_n } )</td>
<td>( \forall P.C ) ( \exists )P.C</td>
<td>( \forall [P]C ) ( \exists [P]C )</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>( \forall P.C )</td>
<td>( \exists )hasChild,Doctor</td>
<td>( \exists )hasChild,Lawyer</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>( \exists P.C )</td>
<td>( \exists )hasChild</td>
<td>( \exists )hasChild</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>( \leq n P )</td>
<td>( \leq 1 )hasChild</td>
<td>( \leq 2 )hasChild</td>
</tr>
<tr>
<td>minCardinality</td>
<td>( \geq n P )</td>
<td>( \geq 2 )hasChild</td>
<td>( \geq 2 )hasChild</td>
</tr>
</tbody>
</table>

OWL Axioms

<table>
<thead>
<tr>
<th>Axiom</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>superClassOf</td>
<td>( C_1 \subseteq C_2 )</td>
<td>Human ( \subseteq ) Animal ( \cap ) Biped</td>
</tr>
<tr>
<td>equivalentClass</td>
<td>( C_1 \equiv C_2 )</td>
<td>Man ( \equiv ) Human ( \cap ) Male</td>
</tr>
<tr>
<td>disjointWith</td>
<td>( C_1 \subseteq \neg C_2 )</td>
<td>Male ( \subseteq \neg )Female</td>
</tr>
<tr>
<td>sameIndividualAs</td>
<td>( {x_1} \equiv {x_2} )</td>
<td>{President.Bush} \equiv {G.W_Bush}</td>
</tr>
<tr>
<td>differentFrom</td>
<td>( {x_1} \not\subseteq {x_2} )</td>
<td>{john} \not\subseteq {peter}</td>
</tr>
<tr>
<td>subPropertyOf</td>
<td>( P_1 \subseteq P_2 )</td>
<td>hasDaughter ( \subseteq ) hasChild</td>
</tr>
<tr>
<td>equivalentProperty</td>
<td>( P_1 \equiv P_2 )</td>
<td>cost ( \equiv ) price</td>
</tr>
<tr>
<td>inverseOf</td>
<td>( P_1 \equiv P_2 )</td>
<td>hasChild ( \equiv ) hasParent(^{-})</td>
</tr>
<tr>
<td>transitiveProperty</td>
<td>( P^+ \subseteq P )</td>
<td>ancestor(^{+}) ( \subseteq ) ancestor</td>
</tr>
<tr>
<td>functionalProperty</td>
<td>( T \subseteq \leq 1 P )</td>
<td>T ( \subseteq \leq 1 )hasMother</td>
</tr>
<tr>
<td>inverseFunctionalProperty</td>
<td>( T \subseteq \leq 1 P^{-} )</td>
<td>T ( \subseteq \leq 1 )hasSSN(^{-})</td>
</tr>
</tbody>
</table>

OWL DL Semantics

\[
(C \cap D)^I = C^I \cap D^I \\
(C \cup D)^I = C^I \cup D^I \\
(\neg C)^I = \Delta^I \setminus C^I \\
\{x\}^I = \{x^I\} \\
(\exists R.C)^I = \{x \mid \exists y. \langle x, y \rangle \in R^I \land y \in C^I\} \\
(\forall R.C)^I = \{x \mid \forall y. \langle x, y \rangle \in R^I \Rightarrow y \in C^I\} \\
(\leq n R)^I = \{x \mid \# \{y \mid \langle x, y \rangle \in R^I\} \leq n\} \\
(\geq n R)^I = \{x \mid \# \{y \mid \langle x, y \rangle \in R^I\} \geq n\}
\]
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

Semantic 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
    ```sparql
    ```
  - 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.
    ```sparql
    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

http://www.eecs.qmul.ac.uk/~thor/

Outlook

IR Systems vs. DBMS

Don't program search engines, design them

http://www.spinque.com/
Towards an IRMS

References I


References II


References III


