pDAML+OIL: A probabilistic extension to DAML+OIL based on probabilistic Datalog

Henrik Nottelmann, Norbert Fuhr

IPMU 2004

Perugia, Italy
July 5, 2004
Motivation (1)

**DAML+OIL**: description logics ($SHOIQ(D)$)

**Limitations**: important features missing from a practical perspective

1. no support for modelling uncertainties
2. no support for general rules (besides implicit rules, e.g. subClassOf)

**Our approach**: extend the language

- partial mapping DAML+OIL onto four-valued probabilistic Datalog (probabilistic variant of Horn logics, inference engine available)
- mapping can be used for defining uncertain knowledge + rules
Motivation (2)

Why DAML+OIL, why Datalog? want to solve practical problems

- distributed information retrieval, e.g. in peer-to-peer networks
- resource selection, retrieval, schema mapping can be described in probabilistic Datalog
- services are described using DAML-S
- task: dynamically compute execution plans
  idea: convert DAML-S into pDatalog

Known limitation: no complete mapping, e.g. disjunction
proposed mapping currently sufficient for us
Outline

1. Motivation
2. Probabilistic Datalog
3. Mapping DAML+OIL onto Datalog
4. Probabilistic DAML+OIL
5. Conclusion and outlook
Datalog: variant of predicate logic based on function-free Horn clauses

\[ h \leftarrow b_1, \ldots, b_n, \quad \text{where } h \text{ atom ("head"), } b_i \text{ literals ("body")} \]

Negation: only modularly stratified programs (w.r.t. instantiated program):

- No ground atom depends negatively upon itself.
- Semantics: exactly one (total) stable Herbrand model exists.

\( \text{father}(X,Y) \leftarrow \text{parent}(X,Y) \land \text{man}(X). \)
\( \text{parent}(jo,mary). \)
Probabilistic Datalog (1)

Probabilistic extension: every fact or rule has a probabilistic weight attached

person(mary).
0.8 man(jo).
0.5 man(X) :- person(X).

probability that a person is a man is 0.5

Computation of probabilities: based on event keys and event expressions (for recognising duplicate or disjoint events, “intensional semantics”)

- facts, instantiated rules as basic events (unique key, probability)
- derived facts as Boolean combination of the event keys of the underlying basic events
- independence assumption, disjoint events
Probabilistic Datalog (2)

Possible world: well-founded model of union of deterministic part and subset of the indeterministic part (without the probabilities)

Semantics: probability distribution $\mu$ over set of all “possible worlds”

\[ \begin{align*}
\text{person}(&\text{mary}). \\
0.8 & \text{ man}(&\text{jo}). \\
0.5 & \text{ man}(X) :- \text{ person}(X).
\end{align*} \]

\[ \begin{align*}
\mu(\{&\text{person}(&\text{mary})\}) & = 0.1 \\
\mu(\{&\text{person}(&\text{mary}), &\text{man}(&\text{jo})\}) & = 0.4 \\
\mu(\{&\text{person}(&\text{mary}), &\text{man}(&\text{mary})\}) & = 0.1 \\
\mu(\{&\text{person}(&\text{mary}), &\text{man}(&\text{mary}), &\text{man}(&\text{jo})\}) & = 0.4
\end{align*} \]
Four-valued Probabilistic Datalog

Four valued logics: also “unknown” and “inconsistent” (OWA instead of CWA)

0.6/0.2/0.2 man(jo).
0.9 woman(maria).
!man(laura).
!man(X) :- woman(X).

Models: each possible world can contain atom $a$, negation $\neg a$, none or both

Conversion into two-valued pDatalog:

- two distinct predicates for positive and negative knowledge
- three facts for “true”, “false” and “inconsistent”
Outline

1. Motivation
2. Probabilistic Datalog
3. Mapping DAML+OIL onto Datalog
4. Probabilistic DAML+OIL
5. Conclusion and outlook
Basic idea

**Goal:** model DAML+OIL axiomatic semantics by mapping them onto Datalog

**Obvious approach:** unary and binary predicates

```xml
<dam1:Class rdf:id="flight:Airport"/>
<flight:Airport rdf:ID="#DUS">
    <flight:close rdf:resource="#Dusseldorf"/>
</flight:Airport>

dam1:Class(flight:Airport).
flight:Airport(#DUS).
flight:close(#DUS,#Dusseldorf).
```
DAML+OIL modelling primitives (1)

**Modelling primitives:** transformed into sets of Datalog rules

```daml
<daml:Class rdf:id="flight:Airport">
  <rdfs:subClassOf rdf:resource="travel:Location"/>
</daml:Class>
```

```datalog
travel:Location(O) :- flight:Airport(O).
```
Problem: some primitives require four-valued logics

```
<daml:Class rdf:id="travel:City">
  <daml:disjointWith rdf:resource="flight:Airport"/>
</daml:Class>
```

```
!flight:Airport(O) :- travel:City(O).
!travel:City(O) :- flight:Airport(O).
```

Example:

```
travel:City(c).  ->  !flight:Airport(c).
flight:Airport(c).  ->  !travel:City(c).
```

inconsistent knowledge
Cardinality restrictions

Datalog: no support for counting

Approach: introduce specific auxiliary rules (with $O(n^2)$ literals)

```xml
<daml:Class rdf:ID="Person">
  <rdfs:subClassOf>
    <daml:Restriction>
      <daml:onProperty rdf:resource="#hasParent"/>
      <daml:cardinality>2</daml:cardinality>
    </daml:Restriction>
  </rdfs:subClassOf>
</daml:Class>

hasParent_diff3(O) :- hasParent(O, Y1) & hasParent(O, Y2) & hasParent(O, Y3) & 
  <> (Y1, Y2) & <> (Y1, Y3) & <> (Y2, Y3).

!Person(O) :- Person(O) & hasParent_diff3(O).
!Person(O) :- Person(O) & !hasParent_diff2(O).
```
Limitation: Disjunction (1)

**Datalog:** no disjunction in head

```xml
<daml:Class rdf:ID="Person">
    <daml:unionOf rdf:parseType="daml:collection">
        <daml:Class rdf:about="#Man"/>
        <daml:Class rdf:about="#Woman"/>
    </daml:unionOf>
</daml:Class>
```

Person(X) :- Man(X).
Person(X) :- Woman(X).
!Person(X) :- !Man(X) & !Woman(X).

**missing: positive knowledge** \(\text{Man}(X) \lor \text{Woman}(X) \leftarrow \text{Person}(X)\)
Limitation: Disjunction (2)

**Datalog:** no disjunction in head

```daml
<daml:Class rdf:ID="Person">
  <daml:unionOf rdf:parseType="daml:collection">
    <daml:Class rdf:about="#Man"/>
    <daml:Class rdf:about="#Woman"/>
  </daml:unionOf>
</daml:Class>
```

Person(X) :- Man(X).
Person(X) :- Woman(X).
Woman(X) :- Person(X) & !Man(X).
Man(X) :- Person(X) & !Woman(X).

**program not modularly stratified, no unique model**
Outline

1. Motivation
2. Probabilistic Datalog
3. Mapping DAML+OIL onto Datalog
4. Probabilistic DAML+OIL
5. Conclusion and outlook
Probabilistic DAML+OIL statements

**Goal:** weighted DAML+OIL statements, employing four-valued probabilistic Datalog

```xml
<flight:Airport rdf:ID="#FLR">
  <travel:close rdf:resource="#Siena" pdaml:prob="0.7"/>
</flight:Airport>

<Person rdf:ID="#Bob" pdaml:prob="0.7/0.1/0.1"/>

flight:Airport(#FLR).
0.7 travel:close(#FLR,#Siena).
0.7/0.1/0.1 Person(#Bob).
```
Probabilistic modelling primitives

Simple case: 50% of all persons are men

```xml
<daml:Class rdf:id="#Man">
  <rdfs:subClassOf rdf:resource="Person" pdaml:prob="0.5"/>
</daml:Class>
```

0.5 Man(X) :- Person(X).

Not suitable for all primitives:

```xml
<daml:Class rdf:id="Man" pdaml:prob="0.5"/>
```

not helpful (should know if something is a class, a property or an instance)
Probabilistic rules in pDAML+OIL

Rules: for advanced DAML+OIL applications, model of pDatalog rules

```xml
<pdaml:Rule>
  <pdaml:head pdaml:prob="0.6">
    <pdaml:PositiveLiteral>
      <rdf:subject rdf:resource="#T"/>
      <rdf:predicate rdf:resource="travel:candflight"/>
      <rdf:object rdf:resource="#F"/>
    </pdaml:PositiveLiteral>
  </pdaml:head>
  <pdaml:body>
    ...</pdaml:PositiveLiteral>
  </pdaml:body>
</pdaml:Rule>
```
Outline

1. Motivation
2. Probabilistic Datalog
3. Mapping DAML+OIL onto Datalog
4. Probabilistic DAML+OIL
5. Conclusion and outlook
Conclusion and outlook

Conclusion:

- first step to extending DAML+OIL towards uncertainty and rules by mapping DAML+OIL onto Datalog variant
- useful in practical environments
- demonstrated need for four-valued pDatalog (e.g. \texttt{disjointWith})

Outlook:

- closer look at features, computational complexity, limitations
- weights for modelling primitives
- prototype
- switch to OWL