Practical Reasoning with OWL and DL-Safe Rules

Peter Haase¹  Pascal Hitzler¹  Markus Krötzsch¹
Jürgen Angele²  Rudi Studer¹²³

¹ Institute AIFB, Universität Karlsruhe
² ontoprise GmbH, Karlsruhe
³ FZI Karlsruhe
Web Ontology Language OWL

- W3C recommendation since April 2004
- conceived as extension of RDFS
- three different flavours:
  - OWL Lite: based on a fragment of first-order logic (FOL)
  - OWL DL: larger fragment of FOL
  - OWL Full: “combination” of OWL DL with all features of RDFS (reification, ...)

⇝ Here: focus on OWL DL
OWL DL

- OWL DL is based on the description logic $\mathcal{SHOIN}(D)$
- typical reasoning tasks for OWL DL are decidable
  $\Rightarrow$ relevant for many applications
- implementing OWL DL reasoning is difficult
- largely compatible with RDFS
  (but not containing all of RDFS)
Overview

1. Description logics
2. The KAON2 reasoner
3. Conjunctive queries
4. Semantic Web rules
5. Outlook
Description logics (DLs)

DLs are logical formalisms that correspond to certain fragments of first-order logic

- used for describing knowledge in a precise and well-defined way
  - suitable as ontology languages
- formal semantics allows for unambiguous interpretation
- reasoning with DLs typically decidable
  - fully automatic processing of knowledge
- there are many DLs, defined by their expressive features: expressivity vs. complexity
Simple description logics

Three types of modelling primitives:

1. **Individuals**, describing single elements of the domain.
   E.g. *eswc2006, markus, . . .*

2. **Concepts**, describing sets of individuals.
   E.g. *Conference, Human, . . .*

3. **Roles**, describing binary relations between individuals.
   E.g. *loves, participates_in, . . .*

Simple assertions about individuals, e.g.

\[
eswc2004 : \text{Conference} \\
(markus, eswc2004) : \text{participates_in}
\]

\(\leadsto\) **ABox** of assertional axioms

Applications involve **complex classes** and relationships between them

\(\leadsto\) combine roles and classes with logical operators
A simple DL: $\mathcal{ALC}$

- **$\mathcal{ALC}$**: “Attribute Language with Complement”

Building concepts:

| $C \sqcap D$ | individuals in $C$ and $D$ |
| $C \sqcup D$ | individuals in $C$ or $D$ |
| $\neg C$    | individuals not in $C$    |
| $\exists R.C$ | individuals with some relation $R$ to $C$ |
| $\forall R.C$ | individuals with all relations $R$ to $C$ |

Stating relationships between concepts

| $C \sqsubseteq D$ | all individuals of $C$ are also in $D$ |
| $C \equiv D$ | the individuals of $C$ and $D$ are the same |

$\hookrightarrow$ **TBox** of terminological axioms

Knowledge base = ABox + TBox
Conference ⊆ Event
Conference ⊆ ∀ participant.Person

Person ⊆ Female ⊔ Male

eswc2006 : Conference
(eswc2006, markus) : participant

We would like to **conclude** also that

markus : Person

Every conference is an event.
Everybody who participates in a conference is a person.
Persons are female or male.
ESWC2006 is a conference.
Markus participates in ESWC.

Markus is a person.
Reasoning tasks

Classical tasks for reasoning with description logics:

- **Instance checking.** Is individual $a$ in concept $C$?
- **Concept subsumption.** Is concept $C$ more general than $D$?
- **Concept satisfiability.** Does the definition of concept $C$ allow any instances of this concept?
- **Knowledge base satisfiability.** Is the combined knowledge of TBox and ABox free of contradictions?

Checking knowledge base satisfiability suffices

- Individual $a$ in concept $C$, if $a : \neg C$ leads to a contradiction
- $C$ is more general than $D$, if $x : C \sqcap \neg D$ leads to a contradiction
- $C$ is unsatisfiable if $x : C$ leads to a contradiction (with $x$ is some hitherto unused unused individual)
Example – drawing conclusions

\[\text{Conference} \subseteq \text{Event}\]
\[\text{Conference} \subseteq \forall \text{ participant}. \text{Person}\]

\[
\begin{align*}
\text{markus} : \text{Person} & \quad \text{Markus is a person.} \\
(\text{markus, escw2006}) : \text{participant} & \quad \text{Oops! ESWC participates in Markus?}
\end{align*}
\]

Isn’t this a contradiction since ESWC is no person?

\[\text{eswc2006} : \text{Person} \quad \text{Uh oh . . . ESWC is a person.}\]

What is missing?

\[\text{Person} \subseteq \neg \text{Events}\]
\[\text{eswc2006} : \text{Conference} \quad \text{ESWC is a conference.}\]

Every conference is an event.
Everybody who participates in a conference is a person.
Persons are not events.
Reasoning

Most common reasoning method: **tableau calculus**

- closely related to tableaux in FOL and modal logics
- approach: try to construct a valid model for a knowledge base
  ⇝ determine whether knowledge base is satisfiable
- possible results:
  1. model constructed successfully ⇝ satisfiable
  2. *all* attempts of model construction fail ⇝ unsatisfiable
  3. the algorithms fails to halt
The tableau calculus

eswc : Conference
Conference ⊑ ∃ progChair . Person
∃ progChair . ⊤ ⊑ Event

Can we derive eswc : Event?

eswc : ¬ Event
eswc : Conference

eswc : ¬ conference

eswc : ¬ ∃ progChair . Person
(eswc, x) : progChair
x : Person

eswc : ∃ progChair . ⊤

eswc : ∀ progChair . ⊥
x : ⊥
Termination

\[ \text{markus} : \text{Person} \]

\[ \text{markus} : \neg \text{Person} \]

\[ \text{x} : \neg \text{Person} \]

\[ \text{x} : \exists \text{parent} . \text{Person} \]

\[ \text{x} : \text{Person} \]

\[ \text{x} : \exists \text{parent} . \text{Person} \]

\[ \text{x} : \text{Person} \]

\[ \text{x} : \exists \text{parent} . \text{Person} \]

\[ \text{x} : \text{Person} \]

\[ \leadsto \text{sophisticated blocking techniques needed to ensure termination} \]
More expressive DLs

### Concepts

<table>
<thead>
<tr>
<th><strong>ALC</strong></th>
<th>Boolean operators, modalities: ⊓, ⊔, ¬, ∀R, ∃R</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N</strong></td>
<td>Number restrictions</td>
</tr>
<tr>
<td></td>
<td>≥2 has_child</td>
</tr>
<tr>
<td></td>
<td>≤1 has_mother</td>
</tr>
<tr>
<td><strong>Q</strong></td>
<td>Qualified number restr.</td>
</tr>
<tr>
<td></td>
<td>≥2 has_child. Doctor</td>
</tr>
<tr>
<td><strong>O</strong></td>
<td>Nominals</td>
</tr>
<tr>
<td></td>
<td>{peter, pascal, markus}</td>
</tr>
</tbody>
</table>

### Individuals

| ≈  | Same             | markus ≈ m_krötzsch                      |
| ≱ | Different         | peter ≱ markus                           |

### Roles

<table>
<thead>
<tr>
<th><strong>H</strong></th>
<th>Subrole hierarchy</th>
<th>has_mother ⊆ has_ancestor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I</strong></td>
<td>Inverse roles</td>
<td>has_ancestor⁻ ⊆ has_offspring</td>
</tr>
<tr>
<td><strong>S</strong></td>
<td>(<strong>ALC</strong> +) role transitivity</td>
<td>Trans(has_ancestor)</td>
</tr>
</tbody>
</table>

**OWL DL: SHOIN + concrete domains (datatypes)**
Recall: $P \subseteq NP \subseteq \text{PSPACE} \subseteq \text{EXPTIME} \subseteq \text{NEXPTIME}$

Reasoning with (many) DLs is computationally hard:

- $\mathcal{ALC}$ with empty TBox: PSPACE
- $\mathcal{ALC}$: EXPTIME
- $\text{SHOIN} (\text{OWL DL})$: NEXPTIME

However, worst case $\neq$ average case!

- Highly optimized (practically efficient) reasoners exist.
- Some more restricted DLs are tractable (= decidable in P)
Back to OWL DL

Conference \subseteq \forall participant.(Female \sqcup Male)

is expressed in OWL/RDF as:

```xml
<owl:Class rdf:ID="Conference">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="participant"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="Female"/>
          <owl:Class rdf:about="Male"/>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```
KAON2 ontology management suite

KAON2: OWL reasoner and ontology management API

KAON2 reasoner:
- not based on tableau methods
- based on first-order resolution calculus
- goal: efficient reasoning for large ABoxes
- binaries available from http://kaon2.semanticweb.org
KAON2 answers queries in two processing steps:

1. **OWL DL ABox**
2. **Transformation to disjunctive datalog program**
3. **Disjunctive datalog engine**
4. **Datalog can be cached and reused for various queries**

OWL DL TBox (w/o nominals)

OWL DL query
OWL as a fragment of FOL

OWL can be translated into first-order logic:

\[
C \sqsubseteq D \quad \mapsto \quad \forall x. C(x) \rightarrow D(x)
\]

\[
a : \exists R. C \quad \mapsto \quad \exists x. (R(a, x) \land C(x))
\]

\[
a : \leq 1 R \quad \mapsto \quad \forall x. \forall y. ((R(a, x) \land R(a, y)) \rightarrow x \equiv y)
\]

... \quad \mapsto \quad ...

\[\leadsto\] application of first-order reasoning techniques possible
KAON2 transformation in detail

**OWL DL**

- *SHIQ TBox* (qualified number restr., no nominals)

**Elimination of Transitivity Axioms**

- Transitivity Axiom: $\forall R.C \subseteq \forall R.\forall R.C$

**Translation into clauses**

- $\exists R.C(a) \Rightarrow R(a, f(a)) \land C(f(a))$

**Disjunctive datalog**

- Negation- & function-free logic program

**Eliminate function symbols**

- Add constants: $f(x) \Rightarrow x_f$

**Saturation**

- Use resolution to add inferred clauses, EXPTIME!
Example

**Person** $\sqsubseteq \exists \text{childOf}. \text{Person}$

**NoSiblings** $\sqsubseteq \text{Person} \sqcap \forall \text{childOf}. \leq 1 \text{hasChild}. \top$

**Parent** $\equiv \exists \text{hasChild}. \top$

**childOf** $\equiv \text{hasChild}^{-1}$

\[
\begin{align*}
\text{childof}(X, X_{f0}) & : - \text{person}(X), \text{kaon2s}_{f0}(X, X_{f0}). \\
\text{person}(X_{f0}) & : - \text{person}(X), \text{kaon2s}_{f0}(X, X_{f0}). \\
\text{person}(X) & : - \text{nosiblings}(X). \\
\text{kaon2equal}(Y_1, Y_2) & : - \text{nosiblings}(X), \text{childof}(X, Z), \\
& \hspace{1cm} \text{haschild}(Z, Y_1), \text{haschild}(Z, Y_2). \\
\text{haschild}(X, X_{f1}) & : - \text{parent}(X), \text{kaon2s}_{f1}(X, X_{f1}). \\
\text{parent}(X) & : - \text{haschild}(X, Y). \\
\text{haschild}(Y, X) & : - \text{childof}(X, Y). \\
\text{childof}(Y, X) & : - \text{haschild}(X, Y).
\end{align*}
\]
Complexity and efficiency

1. Process query and TBox to obtain disjunctive datalog $\leadsto \text{ExpTime}$
2. Add ABox
3. Use Datalog reasoner for query answering $\leadsto \text{NP}$

Features

- TBox translation not necessary for every query
- Datalog-reasoning exploits well-known optimisation strategies (e.g. *magic sets*)
- Data complexity is NP
- Overall algorithm is worst-case optimal ($\text{ExpTime}$)
KAON2: strengths and limitations

Problem: comparison to other reasoners difficult
due to different architectures, caching mechanisms, pre-processing, ...

- best results for large ABoxes, medium complexity TBoxes
  \(\rightarrow\) generally superior to tableaux algorithms
- still able to solve non-trivial TBox problems
  \(\rightarrow\) generally inferior to tableaux algorithms
- no support for nominals
- additional reasoning features: see below
- powerful API and ontology management tools: see second half of tutorial
Conjunctive queries

The knowledge base can be queried for conjunctions of

- terms $A(x)$ and $\neg A(x)$ where $A$ is a concept name, and
- terms $R(x, y)$ where $R$ is a simple role (one without transitive subroles).

The conjunctive query asks for concrete individuals that are valid fillers for the distinguished variables.

Example

$\exists y, z : Conference(x) \land location(x, y) \land weather(y, z) \land \neg rainy(z)$:

“Find known conferences at some (possibly unknown) location where the weather is no rainy.”

$\leadsto$ additional query expressivity to extend DL reasoning.
Expressiveness of OWL is limited

Example: uncles in OWL

Given DL roles \textit{parent}, \textit{brother}, and \textit{uncle}, one cannot describe their exact relationship, i.e.

“Someones uncle is the brother of her parent”
cannot be expressed in OWL.

\[ \text{parent}(x, y) \land \text{brother}(y, z) \rightarrow \text{uncle}(x, z) \]

\textit{Rules might add additional expressiveness:}

\rightarrow \text{Semantic Web Rule Language (SWRL)}
Problem

OWL DL + SWRL is not decidable anymore.

\[ \Rightarrow \text{restriction of SWRL rules} \]

Safety condition

Every variable appears in a non-DL atom in the rule condition.

Example:

\[
O(x), O(y), O(z), \text{parent}(x, y) \land \text{brother}(y, z) \rightarrow \text{uncle}(x, z)
\]

where \( O \) is not a concept from the DL knowledge base.
DL-safe rules in KAON2

\[ O(x), O(y), O(z), \text{parent}(x, y) \land \text{brother}(y, z) \rightarrow \text{uncle}(x, z) \]

What means \( O(x) \)?

\[ \sim \text{Add fact } O(a) \text{ for every known individual } a. \]

Intuition: DL-safe rules are SWRL rules that are restricted to known individuals.

In KAON2:

- DL-safe rules can be added to the disjunctive datalog output.
- No additional pre-processing required.
- Complexity of Datalog reasoning still NP.
OWL: future development

- **Further extension of OWL DL**: OWL 1.1
  - Additional expressivity with similar complexity

  [http://www-db.research.bell-labs.com/user/pfps/owl/overview.html](http://www-db.research.bell-labs.com/user/pfps/owl/overview.html)

- **Rule languages**: W3C working group “RIF”

  [http://owl-workshop.man.ac.uk/Tractable.html](http://owl-workshop.man.ac.uk/Tractable.html)

- **Tractable fragments of OWL**: Interesting DLs with polynomial decision problems
  - e.g. Horn-$SHIQ$, EL++, DL-Lite, ...