OWL
Agenda

• Motivation
• Introduction Description Logics
• The Description Logic $ALC$
• Extensions of $ALC$
• Inference Problems
Agenda

- Motivation
- Introduction Description Logics
- The Description Logic $\mathcal{ALC}$
- Extensions of $\mathcal{ALC}$
- Inference Problems
Description Logics

- description logics (DLs) are one of the current KR paradigms
- have significantly influenced the standardization of Semantic Web languages
  - OWL is essentially based on DLs
- numerous reasoners
  
  Quonto     JFact     FaCT++     RacerPro
  Owlgres    Pellet    SHER      snorocket
  OWLIM      Jena      Oracle Prime QuOnto
  Trowl      HermiT    condor    CB
  ELK        konclude  RSacle
OWL Tools

good support by editors

- Protégé, http://protege.stanford.edu
- SWOOP, http://code.google.com/p/swoop/
Description Logics

- origin of DLs: semantic networks and frame-based systems
- downside of the former: only intuitive semantics - diverging interpretations
- DLs provide a formal semantics on logical grounds
- can be seen as decidable fragments of first-order logic (FOL), closely related to modal logics
- significant portion of DL-related research devoted to clarifying the computational effort of reasoning tasks in terms of their worst-case complexity
- despite high complexities, even for expressive DLs exist optimized reasoning algorithms with good average case behavior
Agenda

- Motivation
- Introduction Description Logics
- The Description Logic $\mathcal{ALC}$
- Extensions of $\mathcal{ALC}$
- Inference Problems
DL building blocks

- **individuals:** birte, cs63.800, sebastian, etc.
  - constants in FOL, resources in RDF
- **concept names:** Person, Course, Student, etc.
  - unary predicates in FOL, classes in RDF
- **role names:** hasFather, attends, worksWith, etc.
  - binary predicates in FOL, properties in RDF
    - can be subdivided into abstract and concrete roles (object und data properties)

The set of all individual, concept and role names is called **signature** or **vocabulary**
Constituents of a DL Knowledge Base

- **TBox** $\mathcal{T}$: Information about concepts and their taxonomic dependencies

- **ABox** $\mathcal{A}$: Information about individuals, their concept and role memberships

In more expressive DLs also:

- **RBox** $\mathcal{R}$: Information about roles and their mutual dependencies
Agenda

- Motivation
- Introduction Description Logics
- The Description Logic $\mathcal{ALC}$
- Extensions of $\mathcal{ALC}$
- Inference Problems
Complex Concepts

\( \mathcal{ALC} \), Attribute Language with Complement, is the simplest DL that is Boolean closed.

We define (complex) \( \mathcal{ALC} \) concepts as follows:

- every concept name is a concept,
- \( \top \) and \( \bot \) are concepts,
- for concepts \( C \) and \( D \), \( \neg C \), \( C \sqcap D \), and \( C \sqcup D \) are concepts,
- for a role \( r \) and a concept \( C \), \( \exists r.C \) and \( \forall r.C \) are concepts.

Example: \( \text{Student} \sqcap \forall \text{attendsCourse}.\text{MasterCourse} \)

Intuitively: describes the concept comprising all students that attend only master courses.
Concept Constructors vs. OWL

- \(\top\) corresponds to \texttt{owl:Thing}
- \(\bot\) corresponds to \texttt{owl:Nothing}
- \(\square\) corresponds to \texttt{owl:intersectionOf}
- \(\bigcirc\) corresponds to \texttt{owl:unionOf}
- \(\neg\) corresponds to \texttt{owl:complementOf}
- \(\forall\) corresponds to \texttt{owl:allValuesFrom}
- \(\exists\) corresponds to \texttt{owl:someValuesFrom}
Concept Axioms

For concepts $C, D$, a general concept inclusion (GCI) axiom has the form

$$C \sqsubseteq D$$

- $C \equiv D$ is an abbreviation for $C \sqsubseteq D$ and $D \sqsubseteq C$
- a TBox (terminological Box) consists of a set of GCIs
ABox

an $\mathcal{ALC}$ ABox assertion can be of one of the following forms

- $C(a)$, called concept assertion
- $r(a, b)$, called role assertion

an ABox consists of a set of ABox assertions
The Description Logic $\text{ALC}$

- $\text{ALC}$ is a syntactic variant of the modal logic $K$
- semantics defined in a model-theoretic way, that is, via interpretations
- can be expressed in first-order predicate logic
- a DL interpretation $\mathcal{I}$ consists of a domain $\Delta^\mathcal{I}$ and a function $\cdot^\mathcal{I}$, that maps
  - individual names $a$ to domain elements $a^\mathcal{I} \in \Delta^\mathcal{I}$
  - concept names $C$ to sets of domain elements $C^\mathcal{I} \subseteq \Delta^\mathcal{I}$
  - role names $r$ to sets of pairs of domain elements $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$
Schematic Representation of an Interpretation

TU Dresden Foundations of Semantic Web Technologies
Interpretation of Complex Concepts

The interpretation of complex concepts is defined inductively:

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>top</td>
<td>⊤</td>
<td>$\Delta^I$</td>
</tr>
<tr>
<td>bottom</td>
<td>⊥</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>negation</td>
<td>$\neg C$</td>
<td>$\Delta^I \setminus C^I$</td>
</tr>
<tr>
<td>conjunction</td>
<td>$C \sqcap D$</td>
<td>$C^I \cap D^I$</td>
</tr>
<tr>
<td>disjunction</td>
<td>$C \sqcup D$</td>
<td>$C^I \cup D^I$</td>
</tr>
<tr>
<td>universal quantifier</td>
<td>$\forall r. C$</td>
<td>${ x \in \Delta^I \mid (x, y) \in r^I \text{ implies } y \in C^I }$</td>
</tr>
<tr>
<td>existential quantifier</td>
<td>$\exists r. C$</td>
<td>${ x \in \Delta^I \mid \text{there is some } y \in \Delta^I, \text{ such that } (x, y) \in r^I \text{ and } y \in C^I }$</td>
</tr>
</tbody>
</table>
## Interpretation of Axioms

Interpretation can be extended to axioms:

<table>
<thead>
<tr>
<th>name</th>
<th>syntax</th>
<th>semantic</th>
<th>notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>inclusion</td>
<td>$C \sqsubseteq D$</td>
<td>holds if $C^\mathcal{I} \subseteq D^\mathcal{I}$</td>
<td>$\mathcal{I} \models C \sqsubseteq D$</td>
</tr>
<tr>
<td>equivalence</td>
<td>$C \equiv D$</td>
<td>holds if $C^\mathcal{I} = D^\mathcal{I}$</td>
<td>$\mathcal{I} \models C \equiv D$</td>
</tr>
<tr>
<td>concept assertion</td>
<td>$C(a)$</td>
<td>holds if $a^\mathcal{I} \in C^\mathcal{I}$</td>
<td>$\mathcal{I} \models C(a)$</td>
</tr>
<tr>
<td>role assertion</td>
<td>$r(a, b)$</td>
<td>holds if $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$</td>
<td>$\mathcal{I} \models r(a, b)$</td>
</tr>
</tbody>
</table>
Logical Entailment in Knowledge Bases

- Let $\mathcal{I}$ be an interpretation, $\mathcal{T}$ a TBox, $\mathcal{A}$ an Abox and $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ a knowledge base.
- $\mathcal{I}$ is a model for $\mathcal{T}$, if $\mathcal{I} \models ax$ for every axiom $ax$ in $\mathcal{T}$, written $\mathcal{I} \models \mathcal{T}$.
- $\mathcal{I}$ is a model for $\mathcal{A}$, if $\mathcal{I} \models ax$ for every assertion $ax$ in $\mathcal{A}$, written $\mathcal{I} \models \mathcal{A}$.
- $\mathcal{I}$ is a model for $\mathcal{K}$, if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$.
- An axiom $ax$ follows from $\mathcal{K}$, written $\mathcal{K} \models ax$, if every model $\mathcal{I}$ of $\mathcal{K}$ is also a model of $ax$. 
Semantics via Translation into FOL

translation of TBox axioms into first-order predicate logics through the mapping $\pi$ with $C, D$ complex classes, $r$ a role and $A$ an atomic class:

$$\pi(C \sqsubseteq D) = \forall x. (\pi(x)(C) \rightarrow \pi(x)(D)) \quad \pi(C \equiv D) = \forall x. (\pi(x)(C) \leftrightarrow \pi(x)(D))$$
Semantics via Translation into FOL

translation of TBox axioms into first-order predicate logics through the mapping \( \pi \) with \( C, D \) complex classes, \( r \) a role and \( A \) an atomic class:

\[
\begin{align*}
\pi(C \sqsubseteq D) &= \forall x. (\pi_x(C) \rightarrow \pi_x(D)) \\
\pi(C \equiv D) &= \forall x. (\pi_x(C) \leftrightarrow \pi_x(D)) \\
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg \pi_x(C) \\
\pi_x(C \cap D) &= \pi_x(C) \land \pi_x(D) \\
\pi_x(C \cup D) &= \pi_x(C) \lor \pi_x(D) \\
\pi_x(\forall r.C) &= \forall y. (r(x,y) \rightarrow \pi_y(C)) \\
\pi_x(\exists r.C) &= \exists y. (r(x,y) \land \pi_y(C))
\end{align*}
\]
Semantics via Translation into FOL

translation of TBox axioms into first-order predicate logics through the mapping $\pi$ with $C, D$ complex classes, $r$ a role and $A$ an atomic class:

\[
\begin{align*}
\pi(C \sqsubseteq D) &= \forall x. (\pi_x(C) \rightarrow \pi_x(D)) \\
\pi(C \equiv D) &= \forall x. (\pi_x(C) \leftrightarrow \pi_x(D)) \\
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg \pi_x(C) \\
\pi_x(C \sqcap D) &= \pi_x(C) \land \pi_x(D) \\
\pi_x(C \sqcup D) &= \pi_x(C) \lor \pi_x(D) \\
\pi_x(\forall r.C) &= \forall y. (r(x, y) \rightarrow \pi_y(C)) \\
\pi_x(\exists r.C) &= \exists y. (r(x, y) \land \pi_y(C))
\end{align*}
\]
Semantics via Translation into FOL

- translation only requires two variables

\[ \mathcal{ALC} \text{ is a fragment of FOL with two variables } \mathcal{L}_2 \]

\[ \text{satisfiability checking of sets of } \mathcal{ALC} \text{ axioms is decidable} \]
Agenda

- Motivation
- Introduction Description Logics
- The Description Logic $\mathcal{ALC}$
- Extensions of $\mathcal{ALC}$
- Inference Problems
Inverse Roles

- a role can be
  - a role name $r$ or
  - an inverse role $r^{-}$
- the semantics of inverse roles is defined as follows:
  \[(r^{-})^\mathcal{I} = \{(y, x) \mid (x, y) \in r^\mathcal{I}\}\]
- the extension of $\mathcal{ALC}$ by inverse roles is denoted as $\mathcal{ALCI}$
- corresponds to $\text{owl:inverseOf}$
Parts of a Knowledge Base

TBox $\mathcal{T}$: information about concepts and their taxonomic dependencies

ABox $\mathcal{A}$: information about individuals, their concepts and role connections

in more expressive DLs also:

RBox $\mathcal{R}$: information about roles and their mutual dependencies
Role Axioms

- for $r, s$ roles, a role inclusion axiom – RIA has the form $r \sqsubseteq s$
- $r \equiv s$ is the abbreviation for $r \sqsubseteq s$ and $s \sqsubseteq r$
- an RBox (role box) or role hierarchy consists of a set of role axioms
- $r \sqsubseteq s$ holds in an interpretation $\mathcal{I}$ if $r^\mathcal{I} \subseteq s^\mathcal{I}$, written $\mathcal{I} \models r \sqsubseteq s$
- the extension of $ALC$ by role hierarchies is denoted with $ALCH$, if we also have inverse roles: $ALCHI$
- corresponds to owl:subPropertyOf
An Example Knowledge Base

**RBox** $\mathcal{R}$

own $\sqsubseteq$ careFor

**TBox** $\mathcal{T}$

Healthy $\sqsubseteq$ $\neg$ Dead

Cat $\sqsubseteq$ Dead $\sqcup$ Alive

HappyCatOwner $\sqsubseteq$ $\exists$owns.Cat $\sqcap$ $\forall$caresFor.Healthy

**ABox** $\mathcal{A}$

HappyCatOwner (schrödinger)
# An Example Knowledge Base

<table>
<thead>
<tr>
<th>RBox $\mathcal{R}$</th>
<th>own $\sqsubseteq$ careFor</th>
<th>“If somebody owns something, they care for it.”</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBox $\mathcal{T}$</td>
<td>Healthy $\sqsubseteq$ $\neg$ Dead</td>
<td>“Healthy beings are not dead.”</td>
</tr>
<tr>
<td></td>
<td>Cat $\sqsubseteq$ Dead $\sqcup$ Alive</td>
<td>“Every cat is dead or alive.”</td>
</tr>
<tr>
<td></td>
<td>HappyCatOwner $\sqsubseteq$ $\exists$ owns.Cat $\sqcap$ $\forall$ caresFor.Healthy</td>
<td>“A happy cat owner owns a cat and everything he cares for is healthy.”</td>
</tr>
<tr>
<td>ABox $\mathcal{A}$</td>
<td>HappyCatOwner (schrödinger)</td>
<td>“Schrödinger is a happy cat owner.”</td>
</tr>
</tbody>
</table>
Role Transitivity

- for $r$ a role, a transitivity axiom has the form $\text{Trans}(r)$
- $\text{Trans}(r)$ holds in an interpretation $\mathcal{I}$ if $r^\mathcal{I}$ is a transitive relation, i.e., $(x, y) \in r^\mathcal{I}$ and $(y, z) \in r^\mathcal{I}$ imply $(x, z) \in r^\mathcal{I}$, written $\mathcal{I} \models \text{Trans}(r)$
- the extension of $\mathcal{ALC}$ by transitivity axioms is denoted by $S$ (after the modal logic $S_5$)
- corresponds to $\text{owl:TransitiveProperty}$
Role Functionality

- for $r$ a role, a **functionality axiom** has the form $\text{Func}(r)$
- $\text{Func}(r)$ holds in an interpretation $\mathcal{I}$ if $(x, y_1) \in r^\mathcal{I}$ and $(x, y_2) \in r^\mathcal{I}$ imply $y_1 = y_2$, written $\mathcal{I} \models \text{Func}(r)$
- translation into FOL requires equality ($=$)
- the extension of $\mathcal{ALC}$ by functionality axioms is denoted by $\mathcal{ALCF}$
- corresponds to $\text{owl:FunctionalProperty}$
Simple and Non-Simple Roles

• given a role hierarchy \( R \), we let \( \sqsubseteq^* \) denote the reflexive and transitive closure w.r.t. \( \sqsubseteq \)

• for a role hierarchy \( R \), we can distinguish the roles in \( R \) into simple and non-simple roles

• a role \( r \) is non-simple w.r.t. \( R \), if there is a role \( t \) such that Trans(\( t \)) \( \in R \) and \( t \sqsubseteq^* R r \) holds

• all other roles are are simple

• Example: \( R = \{ u \sqsubseteq t, \ t \sqsubseteq s, \ s \sqsubseteq r, \ q \sqsubseteq r, \ \text{Trans}(t) \} \)

\[ q \]
\[ u \rightarrow t \rightarrow s \rightarrow r \]

non-simple:
Simple and Non-Simple Roles

- for a role hierarchy $\mathcal{R}$, we let $\sqsubseteq_\mathcal{R}$ denote the reflexive and transitive closure w.r.t. $\sqsubseteq$
- for a role hierarchy $\mathcal{R}$, we can distinguish the roles in $\mathcal{R}$ into simple and non-simple roles
- a role $r$ is non-simple w.r.t. $\mathcal{R}$, if there is a role $t$ such that Trans$(t) \in \mathcal{R}$ and $t \sqsubseteq_\mathcal{R} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \ t \sqsubseteq s, \ s \sqsubseteq r, \ q \sqsubseteq r, \ \text{Trans}(t)\}$

### Diagram

- $q$ is the non-simple role.

**TU Dresden**

**Foundations of Semantic Web Technologies**
Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\subseteq_{\mathcal{R}}$ denote the reflexive and transitive closure w.r.t. $\subseteq$
- for a role hierarchy $\mathcal{R}$, we can distinguish the roles in $\mathcal{R}$ into simple and non-simple roles
- a role $r$ is non-simple w.r.t. $\mathcal{R}$, if there is a role $t$ such that $\text{Trans}(t) \in \mathcal{R}$ and $t \subseteq_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \subseteq t, \quad t \subseteq s, \quad s \subseteq r, \quad q \subseteq r, \quad \text{Trans}(t)\}$

non-simple: $t, s$
Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\subseteq^*_{\mathcal{R}}$ denote the reflexive and transitive closure w.r.t. $\subseteq$
- for a role hierarchy $\mathcal{R}$, we can distinguish the roles in $\mathcal{R}$ into simple and non-simple roles
- a role $r$ is non-simple w.r.t. $\mathcal{R}$, if there is a role $t$ such that $\text{Trans}(t) \in \mathcal{R}$ and $t \subseteq^*_{\mathcal{R}} r$ holds
- all other roles are simple
- Example: $\mathcal{R} = \{ u \sqsubseteq t, \ t \sqsubseteq s, \ s \sqsubseteq r, \ q \sqsubseteq r, \ \text{Trans}(t) \}$

\[
\begin{aligned}
q \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
u \rightarrow t \rightarrow s \rightarrow r
\end{aligned}
\]

non-simple: $t, s, r$
Simple and Non-Simple Roles

- Given a role hierarchy $\mathcal{R}$, we let $\subseteq^\mathcal{R}$ denote the reflexive and transitive closure w.r.t. $\subseteq$

- For a role hierarchy $\mathcal{R}$, we can distinguish the roles in $\mathcal{R}$ into simple and non-simple roles

- A role $r$ is non-simple w.r.t. $\mathcal{R}$, if there is a role $t$ such that $\text{Trans}(t) \in \mathcal{R}$ and $t \subseteq^\mathcal{R} r$ holds

- All other roles are simple

- Example: $\mathcal{R} = \{u \subseteq t, \ t \subseteq s, \ s \subseteq r, \ q \subseteq r, \ \text{Trans}(t)\}$

\[
\begin{align*}
    & q \\
    & \downarrow \\
    & u \rightarrow t \rightarrow s \rightarrow r
\end{align*}
\]

Non-simple: $t, s, r$  Simple: $q, u$
(Unqualified) Number Restrictions

- for a simple role $s$ and a natural number $n$, $\leq n_s$, $\geq n_s$ and $= n_s$ are concepts
- the semantics is defined by:

\[
(\leq n_s)^\mathcal{I} = \{ x \in \Delta^\mathcal{I} | \#\{ y \in \Delta^\mathcal{I} | (x, y) \in s^\mathcal{I} \} \leq n \}
\]

\[
(\geq n_s)^\mathcal{I} = \{ x \in \Delta^\mathcal{I} | \#\{ y \in \Delta^\mathcal{I} | (x, y) \in s^\mathcal{I} \} \geq n \}
\]

\[
(= n_s)^\mathcal{I} = \{ x \in \Delta^\mathcal{I} | \#\{ y \in \Delta^\mathcal{I} | (x, y) \in s^\mathcal{I} \} = n \}
\]

- the extension of $\mathcal{ALC}$ by (unqualified) number restrictions is denoted by $\mathcal{ALCN}$
- correspond to $\text{owl:maxCardinality}$, $\text{owl:minCardinality}$, and $\text{owl:cardinality}$
- restriction to simple roles ensures decidability e.g. for checking knowledge base satisfiability
- definition of TBox requires an RBox being already defined
(Unqualified) Number Restrictions in FOL

- translation into FOL requires equality or counting quantifiers
- translation defined as follows (likewise for \( \pi_y \)):

  \[
  \pi_x(\leq n s) = \exists^{\leq n} y. (s(x, y)) \\
  \pi_x(\geq n s) = \exists^{\geq n} y. (s(x, y)) \\
  \pi_x(= n s) = \exists^{\leq n} y. (s(x, y)) \land \exists^{\geq n} y. (s(x, y))
  \]

- the following equivalences hold:

  \[
  \neg(\leq n s) = \geq n + 1 s \\
  \neg(\geq 0 s) = \bot \\
  \leq 0 s = \forall s. \bot \\
  \bot \sqsubseteq \leq 1 s = \text{Func}(s)
  \]
  \[
  \neg(\geq n s) = \leq n - 1 s, \quad n \geq 1 \\
  \geq 1 s = \exists s. \top
  \]
Nominals or Closed Classes

- defines a class by complete enumeration of its instances
- for $a_1, \ldots, a_n$ individuals, $\{a_1, \ldots, a_n\}$ is a concept
- semantics defined as follows:

  $$\begin{align*}
  \text{DL: } (\{a_1, \ldots, a_n\})^T &= \{a_1^T, \ldots, a_n^T\} \\
  \text{FOL: } \pi_x(\{a_1, \ldots, a_n\}) &= (x = a_1 \lor \ldots \lor x = a_n)
  \end{align*}$$

- extension of $\mathcal{ALC}$ by nominals denoted as $\mathcal{ALCO}$
- corresponds to $\text{owl:oneOf}$
Nominals for Encoding Further OWL Constructors

- `owl:hasValue` "forces" role to a certain individual

```xml
<owl:Class rdf:ID="Woman">
  <owl:equivalentClass>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasGender"/>
      <owl:hasValue rdf:resource="#female"/>
    </owl:Restriction>
  </owl:equivalentClass>
</owl:Class>
```

- In description logic:

  $$\text{Woman} \equiv \exists \text{hasGender}.\{\text{female}\}$$
Further Kinds of ABox Assertions

an ABox assertion can have one of the following forms

- $C(a)$ (concept assertion)
- $r(a, b)$ (role assertion)
- $\neg r(a, b)$ (negative role assertion)
- $a \approx b$ (equality assertion)
- $a \not\approx b$ (inequality assertion)
Further Kinds of ABox Assertions

an ABox assertion can have one of the following forms

- $C(a)$ (concept assertion)
- $r(a, b)$ (role assertion)
- $\neg r(a, b)$ (negative role assertion)
- $a \approx b$ (equality assertion)
- $a \not\approx b$ (inequality assertion)
Internalization of ABox Assertions

if nominals are supported, every knowledge base with an ABox can be transformed into an equivalent KB without ABox:

\[
\begin{align*}
C(a) &= \{a\} ⊑ C \\
r(a, b) &= \{a\} ⊑ ∃r.\{b\} \\
¬r(a, b) &= \{a\} ⊑ ∀r.(¬\{b\}) \\
a \approx b &= \{a\} ≡ \{b\} \\
a \not\approx b &= \{a\} ⊑ ¬\{b\}
\end{align*}
\]
Overview Nomenclature

\( \text{ALC} \) Attribute Language with Complement

\( S \quad \text{ALC} + \) role transitivity

\( H \quad \text{subroles} \)

\( O \quad \text{closed classes} \)

\( I \quad \text{inverse roles} \)

\( N \quad \text{(unqualified) number restrictions} \)

\( (D) \quad \text{datatypes} \)

\( F \quad \text{functional roles} \)

OWL DL is \( SHOIN(D) \) and OWL Lite is \( SHIF(D) \)
## Different Terms in DLs and in OWL

<table>
<thead>
<tr>
<th>OWL</th>
<th>DL</th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
<td>concept</td>
</tr>
<tr>
<td>property</td>
<td>role</td>
</tr>
<tr>
<td>object property</td>
<td>abstract role</td>
</tr>
<tr>
<td>data property</td>
<td>concrete role</td>
</tr>
<tr>
<td>oneOf</td>
<td>nominal</td>
</tr>
<tr>
<td>ontology</td>
<td>knowledge base</td>
</tr>
<tr>
<td></td>
<td>TBox, RBox, ABox</td>
</tr>
</tbody>
</table>
Example: A More Complex Knowledge Base

Human ⊑ Animal ⊓ Biped
Man ⊑≡ Human ⊓ Male
Male ⊑¬ Female
{President Obama} ⊑≡ {Barack Obama}
{john} ⊑≡¬{peter}
hasDaughter ⊑ hasChild
hasChild ⊑≡ hasParent
cost ⊑≡ price
Trans(ancestor)
Func(hasMother)
Func(hasSSN)
Open versus Closed World Assumption

OWA  Open World Assumption
– the existence of further individuals is possible, if they are not explicitly excluded
– OWL uses the OWA

CWA  Closed World Assumption
– it is assumed that the knowledge base contains all individuals and facts
Open versus Closed World Assumption

**OWA** Open World Assumption
- the existence of further individuals is possible, if they are not explicitly excluded
- OWL uses the OWA

**CWA** Closed World Assumption
- it is assumed that the knowledge base contains all individuals and facts

<table>
<thead>
<tr>
<th>Are all of Bill’s children male?</th>
<th>no idea, if we assume not to know everything about Bill</th>
<th>if we know everything then all of Bill’s children are male</th>
</tr>
</thead>
<tbody>
<tr>
<td>child(bill, bob)</td>
<td>⊨? (∀ child.Man)(bill)</td>
<td>DL answers</td>
</tr>
<tr>
<td>Man(bob)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(≤ 1 child)(bill)</td>
<td>⊨? (∀ child.Man)(bill)</td>
<td>Prolog</td>
</tr>
</tbody>
</table>

TU Dresden Foundations of Semantic Web Technologies
Open versus Closed World Assumption

**OWA**  Open World Assumption
- the existence of further individuals is possible, if they are not explicitly excluded
- OWL uses the OWA

**CWA**  Closed World Assumption
- it is assumed that the knowledge base contains all individuals and facts

<table>
<thead>
<tr>
<th>Are all of Bill’s children male?</th>
<th>no idea, if we assume not to know everything about Bill</th>
<th>if we know everything then all of Bill’s children are male</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{child}(\text{bill}, \text{bob})$</td>
<td>$\models \ ? \ (\forall \text{child}.\text{Man})(\text{bill})$</td>
<td>DL answers don’t know</td>
</tr>
<tr>
<td>$\leq 1 \text{ child}(\text{bill})$</td>
<td>$\models \ ? \ (\forall \text{child}.\text{Man})(\text{bill})$</td>
<td>Prolog</td>
</tr>
</tbody>
</table>

TU Dresden  Foundations of Semantic Web Technologies
Open versus Closed World Assumption

**OWA**  Open World Assumption
- the existence of further individuals is possible, if they are not explicitly excluded
- OWL uses the OWA

**CWA**  Closed World Assumption
- it is assumed that the knowledge base contains all individuals and facts

<table>
<thead>
<tr>
<th>Are all of Bill’s children male?</th>
<th>if we assume not to know everything about Bill</th>
<th>if we know everything then all of Bill’s children are male</th>
</tr>
</thead>
<tbody>
<tr>
<td>child(bill, bob) &amp; Man(bob)</td>
<td>$\models (?) (\forall$ child.Man)(bill)</td>
<td>DL answers don’t know</td>
</tr>
<tr>
<td>$(\leq 1$ child)(bill)</td>
<td>$\models (?) (\forall$ child.Man)(bill)</td>
<td>Prolog yes</td>
</tr>
</tbody>
</table>
Open versus Closed World Assumption

OWA  Open World Assumption
  – the existence of further individuals is possible, if they are not explicitly excluded
  – OWL uses the OWA

CWA  Closed World Assumption
  – it is assumed that the knowledge base contains all individuals and facts

<table>
<thead>
<tr>
<th>Are all of Bill’s children male?</th>
<th>no idea, if we assume not to know everything about Bill</th>
<th>if we know everything then all of Bill’s children are male</th>
</tr>
</thead>
<tbody>
<tr>
<td>child(bill, bob)</td>
<td></td>
<td>=? (∀ child.Man)(bill)</td>
</tr>
<tr>
<td>Man(bob)</td>
<td>Prolog yes</td>
<td></td>
</tr>
<tr>
<td>(≤ 1 child)(bill)</td>
<td></td>
<td>=? (∀ child.Man)(bill)</td>
</tr>
</tbody>
</table>
## Open versus Closed World Assumption

**OWA**  
Open World Assumption  
- the existence of further individuals is possible, if they are not explicitly excluded  
- OWL uses the OWA

**CWA**  
Closed World Assumption  
- it is assumed that the knowledge base contains all individuals and facts

<table>
<thead>
<tr>
<th>Are all of Bill’s children male?</th>
<th>no idea, if we assume not to know everything about Bill</th>
<th>if we know everything then all of Bill’s children are male</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>child(bill, bob)</code></td>
<td>$\vdash ? (\forall \text{child.Man})(bill)$</td>
<td>DL answers don’t know</td>
</tr>
<tr>
<td><code>Man(bob)</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>\leq 1 \text{child})(bill)</code></td>
<td>$\vdash ? (\forall \text{child.Man})(bill)$</td>
<td>yes</td>
</tr>
</tbody>
</table>
Agenda

• Motivation
• Introduction Description Logics
• The Description Logic $\mathcal{ALC}$
• Extensions of $\mathcal{ALC}$
• Inference Problems
Important Inference Problems for a Knowledge Base $\mathcal{K}$

- global consistency of the knowledge base: $\mathcal{K} \models \neg \text{false}$? $\mathcal{K} \models \top \sqsubseteq \bot$?
  - Is the knowledge base “plausible”?
- class consistency: $\mathcal{K} \models C \sqsubseteq \bot$?
  - Is the class $C$ necessarily empty?
- class inclusion (subsumption): $\mathcal{K} \models C \sqsubseteq D$?
  - taxonomic structure of the knowledge base
- class equivalence: $\mathcal{K} \models C \equiv D$?
  - Do two classes comprise the same individual sets?
- class disjointness: $\mathcal{K} \models C \cap D \sqsubseteq \bot$?
  - Are two classes disjoint?
- class membership: $\mathcal{K} \models C(a)$?
  - Is the individual $a$ contained in class $C$?
- instance retrieval: find all $x$ with $\mathcal{K} \models C(x)$
  - Find all (known!) members of the class $C$. 
Decidability of OWL DL

- decidability means that there is a terminating algorithm for all the aforementioned inference problems
- OWL DL is a fragment of FOL, thus FOL inference procedures could be used in principle (Resolution, Tableaux)
  - but these are not guaranteed to terminate!
- problem: find algorithms that are guaranteed to terminate
- no “naive” solutions for this
OWL 2: Outlook

- OWL 2 extends the fragments introduced here by further constructors
- OWL 2 also defines simpler fragments (PTime for standard inferencing problems)
- diverse tools for automated inferencing
- editors support creation of ontologies / knowledge bases