TECHNISCHE UNIVERSITÄT DRESDEN

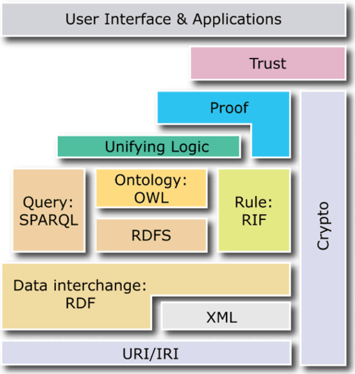
FOUNDATIONS OF SEMANTIC WEB TECHNOLOGIES

OWL & Description Logics

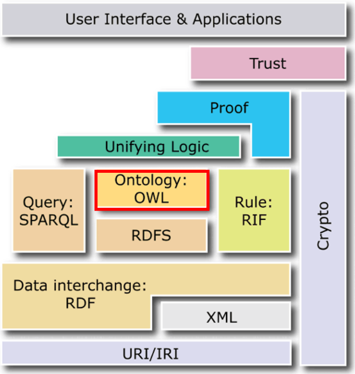
Sebastian Rudolph

DRESDEN concept
Exzellenz aus Wissenschaft und Kultur

OWL


User Interface & Applications
Trust
Proof
Unifying Logic
Query: SPARQL
Ontology: OWL
RDFS
Rule: RIF
Crypto
Data interchange: RDF
XML
URI/IRI

OWL


User Interface & Applications
Trust
Proof
Unifying Logic
Query: SPARQL
Ontology: OWL
RDFS
Rule: RIF
Crypto
Data interchange: RDF
XML
URI/IRI

Agenda

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

OWL Tools

good support by editors

- Protégé, `http://protege.stanford.edu`
- SWOOP, `http://code.google.com/p/swoop/`
- OWL Tools, `http://owltools.ontoware.org/`
- Neon Toolkit, `http://neon-toolkit.org/`

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.
  - $\rightsquigarrow$ constants in FOL, resources in RDF
- concept names: `Person`, `Course`, `Student`, etc.
  - $\rightsquigarrow$ unary predicates in FOL, classes in RDF
- role names: `hasFather`, `attends`, `worksWith`, etc.
  - $\rightsquigarrow$ 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 $\mathcal{A}$	informationen 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: $\mathtt{Student} \sqcap \forall \mathtt{attendsCourse}.\mathtt{MasterCourse}$
Intuitively: describes the concept comprising all students that attend only master courses

Concept Constructors vs. OWL

- $\top$ corresponds to `owl:Thing`
- $\bot$ corresponds to `owl:Nothing`
- $\sqcap$ corresponds to `owl:intersectionOf`
- $\sqcup$ corresponds to `owl:unionOf`
- $\neg$ corresponds to `owl:complementOf`
- $\forall$ corresponds to `owl:allValuesFrom`
- $\exists$ corresponds to `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

TBox $\mathcal{T}$

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

ABox $\mathcal{A}$

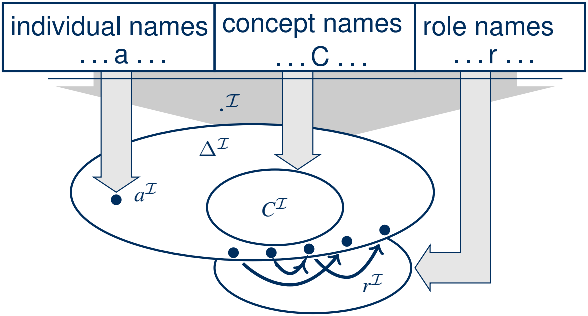
TECHNISCHE UNIVERSITÄT DRESDEN

The Description Logic $\mathcal{ALC}$

- $\mathcal{ALC}$ 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 $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

TECHNISCHE UNIVERSITÄT DRESDEN

Schematic Representation of an Interpretation



Interpretation of Complex Concepts

the interpretation of complex concepts is defined inductively:

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
universal quantifier	$\forall r.C$	$\{x \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}$ implies $y \in C^{\mathcal{I}}\}$
existential quantifier	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid$ there is some $y \in \Delta^{\mathcal{I}}$, such that $(x, y) \in r^{\mathcal{I}}$ and $y \in C^{\mathcal{I}}\}$

Interpretation of Axioms

interpretation can be extended to axioms:

name	syntax	semantic	notation
inclusion	$C \sqsubseteq D$	holds if $C^\mathcal{I} \subseteq D^\mathcal{I}$	$\mathcal{I} \models C \sqsubseteq D$
equivalence	$C \equiv D$	holds if $C^\mathcal{I} = D^\mathcal{I}$	$\mathcal{I} \models C \equiv D$
concept assertion	$C(a)$	holds if $a^\mathcal{I} \in C^\mathcal{I}$	$\mathcal{I} \models C(a)$
role assertion	$r(a, b)$	holds if $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$	$\mathcal{I} \models r(a, b)$

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 π with C, D complex classes, r a role and A an atomic class:

$$\pi(C \sqsubseteq D) = \forall x.(\pi_x(C) \rightarrow \pi_x(\mathrm{D})) \qquad \pi(C \equiv D) = \forall x.(\pi_x(C) \leftrightarrow \pi_x(\mathrm{D}))$$

Semantics via Translation into FOL

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

$$\pi(C \sqsubseteq D) = \forall x.(\pi_x(C) \rightarrow \pi_x(\mathrm{D})) \qquad \pi(C \equiv D) = \forall x.(\pi_x(C) \leftrightarrow \pi_x(\mathrm{D}))$$

$$\begin{aligned}
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg\pi_x(C) \\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(\mathrm{D}) \\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(\mathrm{D}) \\
\pi_x(\forall r.C) &= \forall y.(r(x,y) \rightarrow \pi_y(C)) \\
\pi_x(\exists r.C) &= \exists y.(r(x,y) \wedge \pi_y(C))
\end{aligned}$$

Semantics via Translation into FOL

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

$$\pi(C \sqsubseteq D) = \forall x.(\pi_x(C) \rightarrow \pi_x(\mathrm{D})) \qquad \pi(C \equiv D) = \forall x.(\pi_x(C) \leftrightarrow \pi_x(\mathrm{D}))$$

$$\begin{aligned}
\pi_x(A) &= A(x) & \pi_y(A) &= A(y) \\
\pi_x(\neg C) &= \neg\pi_x(C) & \pi_y(\neg C) &= \neg\pi_y(C) \\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(\mathrm{D}) & \pi_y(C \sqcap D) &= \pi_y(C) \wedge \pi_y(\mathrm{D}) \\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(\mathrm{D}) & \pi_y(C \sqcup D) &= \pi_y(C) \vee \pi_y(\mathrm{D}) \\
\pi_x(\forall r.C) &= \forall y.(r(x,y) \rightarrow \pi_y(C)) & \pi_y(\forall r.C) &= \forall x.(r(y,x) \rightarrow \pi_x(C)) \\
\pi_x(\exists r.C) &= \exists y.(r(x,y) \wedge \pi_y(C)) & \pi_y(\exists r.C) &= \exists x.(r(y,x) \wedge \pi_x(C))
\end{aligned}$$

Semantics via Translation into FOL

- translation only requires two variables

$\rightsquigarrow$ $\mathcal{ALC}$ is a fragment of FOL with two variables $\mathcal{L}_2$

$\rightsquigarrow$ satisfiability checking of sets of $\mathcal{ALC}$ 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 `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 $\mathcal{ALC}$ by role hierarchies is denoted with $\mathcal{ALCH}$, if we also have inverse roles: $\mathcal{ALCHI}$
- corresponds to `owl:subPropertyOf`

RBox $\mathcal{R}$

An Example Knowledge Base

RBox $\mathcal{R}$

$$\mathtt{own} \sqsubseteq \mathtt{careFor}$$

TBox $\mathcal{T}$

$$\begin{aligned} \mathtt{Healthy} &\sqsubseteq \neg\,\mathtt{Dead} \\ \mathtt{Cat} &\sqsubseteq \mathtt{Dead} \sqcup \mathtt{Alive} \\ \mathtt{HappyCatOwner} &\sqsubseteq \exists\mathtt{owns.Cat} \sqcap \forall\mathtt{caresFor.Healthy} \end{aligned}$$

ABox $\mathcal{A}$

$$\mathtt{HappyCatOwner}\ (\mathtt{schrödinger})$$

An Example Knowledge Base

RBox $\mathcal{R}$

$$\texttt{own} \sqsubseteq \texttt{careFor}$$

"If somebody owns something, they care for it."

TBox $\mathcal{T}$

$$\texttt{Healthy} \sqsubseteq \neg\texttt{Dead}$$

"Healthy beings are not dead."

$$\texttt{Cat} \sqsubseteq \texttt{Dead} \sqcup \texttt{Alive}$$

"Every cat is dead or alive."

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

"A happy cat owner owns a cat and everything he cares for is healthy."

ABox $\mathcal{A}$

$$\texttt{HappyCatOwner}(\texttt{schrödinger})$$

"Schrödinger is a happy cat owner. "

TECHNISCHE UNIVERSITÄT DRESDEN

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 $\mathcal{S}$ (after the modal logic S_5)
- corresponds to `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 `owl:FunctionalProperty`

TECHNISCHE UNIVERSITÄT DRESDEN

Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\sqsubseteq^\star_{\mathcal{R}}$ 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 $t \sqsubseteq^\star_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \quad t \sqsubseteq s, \quad s \sqsubseteq r, \quad q \sqsubseteq r, \quad \text{Trans}(t)\}$

$$\begin{tikzcd} & & q \arrow[dr] & \\ u \arrow[r] & t \arrow[r] & s \arrow[r] & r \end{tikzcd}$$

non-simple:

Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\sqsubseteq^{\star}_{\mathcal{R}}$ 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 $\mathsf{Trans}(t) \in \mathcal{R}$ and $t \sqsubseteq^{\star}_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \quad t \sqsubseteq s, \quad s \sqsubseteq r, \quad q \sqsubseteq r, \quad \mathsf{Trans}(t)\}$

$$\begin{tikzcd}
 & & q \arrow[dr] & \\
u \arrow[r] & t \arrow[r] & s \arrow[r] & r
\end{tikzcd}$$

non-simple: t

Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\sqsubseteq^{\star}_{\mathcal{R}}$ 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 $\text{Trans}(t) \in \mathcal{R}$ and $t \sqsubseteq^{\star}_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \quad t \sqsubseteq s, \quad s \sqsubseteq r, \quad q \sqsubseteq r, \quad \text{Trans}(t)\}$

$$
\begin{tikzcd}
 & & q \arrow[dr] & \\
u \arrow[r] & t \arrow[r] & s \arrow[r] & r
\end{tikzcd}
$$

non-simple: t, s

Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\sqsubseteq^\star_{\mathcal{R}}$ 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 $\mathsf{Trans}(t) \in \mathcal{R}$ and $t \sqsubseteq^\star_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \quad t \sqsubseteq s, \quad s \sqsubseteq r, \quad q \sqsubseteq r, \quad \mathsf{Trans}(t)\}$

$$
\begin{tikzcd}
 & & q \arrow[dr] & \\
u \arrow[r] & t \arrow[r] & s \arrow[r] & r
\end{tikzcd}
$$

non-simple: t, s, r

Simple and Non-Simple Roles

- given a role hierarchy $\mathcal{R}$, we let $\sqsubseteq^{\star}_{\mathcal{R}}$ 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 $\text{Trans}(t) \in \mathcal{R}$ and $t \sqsubseteq^{\star}_{\mathcal{R}} r$ holds
- all other roles are are simple
- Example: $\mathcal{R} = \{u \sqsubseteq t, \quad t \sqsubseteq s, \quad s \sqsubseteq r, \quad q \sqsubseteq r, \quad \text{Trans}(t)\}$

$$\begin{tikzcd}
 & & q \arrow[dr] & \\
u \arrow[r] & t \arrow[r] & s \arrow[r] & r
\end{tikzcd}$$

non-simple: t, s, r simple: q, u

(Unqualified) Number Restrictions

- for a simple roe s and a natural number n, $\leqslant n\, s$, $\geqslant n\, s$ and $= n\, s$ are concepts
- the semantics is defined by:

$$(\leqslant n\, s)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \in \Delta^{\mathcal{I}} \mid (x,y) \in s^{\mathcal{I}}\} \leq n\}$$
$$(\geqslant n\, s)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \in \Delta^{\mathcal{I}} \mid (x,y) \in s^{\mathcal{I}}\} \geq n\}$$
$$(= n\, s)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \in \Delta^{\mathcal{I}} \mid (x,y) \in s^{\mathcal{I}}\} = n\}$$

- the extension of $\mathcal{ALC}$ by (unqualified) number restrictions is denoted by $\mathcal{ALCN}$
- correspond to `owl:maxCardinality`, `owl:minCardinality`, and `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 π_y):

$$\pi_x(\leqslant n\, s) = \exists^{\leqslant n} y.(s(x,y))$$
$$\pi_x(\geqslant n\, s) = \exists^{\geqslant n} y.(s(x,y))$$
$$\pi_x(= n\, s) = \exists^{\leqslant n} y.(s(x,y)) \wedge \exists^{\geqslant n} y.(s(x,y))$$

- the following equivalences hold:

$$\neg(\leqslant n\, s) = \geqslant n+1\, s \qquad \neg(\geqslant n\, s) = \leqslant n-1\, s, \quad n \geq 1$$
$$\neg(\geqslant 0\, s) = \bot \qquad \geqslant 1\, s = \exists s.\top$$
$$\leqslant 0\, s = \forall s.\bot \qquad \top \sqsubseteq \leqslant 1 s = \mathsf{Func}(s)$$

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:

$$\text{DL: } (\{a_1, \ldots, a_n\})^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$$
$$\text{FOL: } \pi_x(\{a_1, \ldots, a_n\}) = (x = a_1 \vee \ldots \vee x = a_n)$$

- extension of $\mathcal{ALC}$ by nominals denoted as $\mathcal{ALCO}$
- corresponds to `owl:oneOf`

Nominals for Encoding Further OWL Constructors

- `owl:hasValue` “forces” role to a certain individual

```
<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:

$$\text{Woman} \equiv \exists \text{hasGender}.\{\text{female}\}$$

TECHNISCHE UNIVERSITÄT DRESDEN

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:

$$
\begin{aligned}
C(a) &= \{a\} \sqsubseteq C \\
r(a,b) &= \{a\} \sqsubseteq \exists r.\{b\} \\
\neg r(a,b) &= \{a\} \sqsubseteq \forall r.(\neg\{b\}) \\
a \approx b &= \{a\} \equiv \{b\} \\
a \not\approx b &= \{a\} \sqsubseteq \neg\{b\}
\end{aligned}
$$

TECHNISCHE UNIVERSITÄT DRESDEN

Overview Nomenclature

$\mathcal{ALC}$	Attribute Language with Complement
$\mathcal{S}$	$\mathcal{ALC}$ + role transitivity
$\mathcal{H}$	subroles
$\mathcal{O}$	closed classes
$\mathcal{I}$	inverse roles
$\mathcal{N}$	(unqualified) number restrictions
(D)	datatypes
$\mathcal{F}$	functional roles

OWL DL is $\mathcal{SHOIN}(\mathrm{D})$ and OWL Lite is $\mathcal{SHIF}(\mathrm{D})$

Different Terms in DLs and in OWL

OWL	**DL**
class	concept
property	role
object property	abstract role
data property	concrete role
oneOf	nominal
ontology	knowledge base
–	TBox, RBox, ABox

Example: A More Complex Knowledge Base

$$\begin{aligned}
\mathsf{Human} &\sqsubseteq \mathsf{Animal} \sqcap \mathsf{Biped} \\
\mathsf{Man} &\equiv \mathsf{Human} \sqcap \mathsf{Male} \\
\mathsf{Male} &\sqsubseteq \neg\mathsf{Female} \\
\{\mathsf{President_Obama}\} &\equiv \{\mathsf{Barack_Obama}\} \\
\{\mathsf{john}\} &\sqsubseteq \neg\{\mathsf{peter}\} \\
\mathsf{hasDaughter} &\sqsubseteq \mathsf{hasChild} \\
\mathsf{hasChild} &\equiv \mathsf{hasParent}^{-} \\
\mathsf{cost} &\equiv \mathsf{price} \\
&\mathsf{Trans}(\mathsf{ancestor}) \\
&\mathsf{Func}(\mathsf{hasMother}) \\
&\mathsf{Func}(\mathsf{hasSSN}^{-})
\end{aligned}$$

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

	Are all of Bill's children male?	no idea, if we assume not to know everything about Bill	if we know everything then all of Bill's children are male
child(bill, bob) Man(bob)	$\models^?$ $(\forall$ child.Man)(bill)	DL answers	Prolog
$(\leqslant 1$ child)(bill)	$\models^?$ $(\forall$ child.Man)(bill)		

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

		Are all of Bill's children male?	no idea, if we assume not to know everything about Bill	if we know everything then all of Bill's children are male
child(bill, bob) Man(bob)	$\models^?$	$(\forall$ child.Man)(bill)	DL answers don't know	Prolog
$(\leqslant 1$ child)(bill)	$\models^?$	$(\forall$ child.Man)(bill)		

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

		Are all of Bill's children male?	no idea, if we assume not to know everything about Bill	if we know everything then all of Bill's children are male
child(bill, bob) Man(bob)	$\models^?$	$(\forall$ child.Man)(bill)	DL answers don't know	Prolog yes
$(\leqslant 1$ child)(bill)	$\models^?$	$(\forall$ child.Man)(bill)		

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

		Are all of Bill's children male?	no idea, if we assume not to know everything about Bill	if we know everything then all of Bill's children are male
child(bill, bob) Man(bob)	$\models^?$	$(\forall$ child.Man)(bill)	DL answers don't know	Prolog yes
$(\leqslant 1$ child)(bill)	$\models^?$	$(\forall$ child.Man)(bill)	yes	

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

		Are all of Bill's children male?	no idea, if we assume not to know everything about Bill	if we know everything then all of Bill's children are male
child(bill, bob) Man(bob)	$\models^?$	$(\forall$ child.Man)(bill)	DL answers don't know	Prolog yes
$(\leqslant 1$ child)(bill)	$\models^?$	$(\forall$ child.Man)(bill)	yes	yes

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: $\mathcal{K} \models^? \text{false}$? $\mathcal{K} \models^? \top \sqsubseteq \bot$?
  - 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: $\mathcal{K} \models^? C \sqcap D \sqsubseteq \bot$?
  - 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