Modul Semantik Web (SS2011)

Dr. Sören Auer
Dr. Jens Lehmann
Prof. Dr. Gerhard Brewka
Frank Loebe

Institut für Informatik · Universität Leipzig

OWL Semantik und Reasoning

17. Mai 2011

Semantic Web

Gliederung

1. Beschreibungslogiken
2. $\mathcal{ALC}$
3. OWL als $\mathcal{SHOIN}(\mathcal{D})$
4. Inferenzprobleme
5. Tableau-Beweiser
6. OWL2

Gliederung

Beschreibungslogiken

- engl.: description logics (DLs)
- **Familie von** Wissensrepräsentations**sprachen**
- **Fragmente von FOL**
- meist **entscheidbar**
- vergleichsweise ausdrucksstark
- entwickelt aus semantischen Netzwerken
- intuitive Syntax
- **variablenfrei**

- W3C-Standard OWL DL basiert auf der Beschreibungslogik $\mathcal{SHOIN}(\mathcal{D})$
- wir besprechen zunächst $\mathcal{ALC}$ (Basis für komplexere DLs)

Gliederung

1. Beschreibungslogiken
2. $\mathcal{ALC}$
3. OWL als $\mathcal{SHOIN}(\mathcal{D})$
4. Inferenzprobleme
5. Tableau-Beweiser
6. OWL2

$\mathcal{ALC}$ - Grundbausteine und ABox-Axiome

Grundbausteine:

- **Klassen**namen (auch als Konzepte bezeichnet)
- **Rollen**namen
- **Individuen**namen (auch als Objekte bezeichnet)

Wissensbasis = Menge von Axiomen

Axiome für Instanzdaten:

- $Professor(Faehnrich)$
  $\widehat{=}$ Individuum *Faehnrich* ist in Klasse *Professor*
- $zugehoerigkeit(Faehnrich, BIS)$
  $\widehat{=}$ *Faehnrich* ist dem *BIS* zugehoerig

$\mathcal{ALC}$ - TBox-Axiome

$Professor \sqsubseteq Fakultaetsmitglied$

- "Jeder Professor ist ein Fakultätsmitglied."
- entspricht $(\forall x)(Professor(x) \rightarrow Fakultaetsmitglied(x))$
- entspricht `rdfs:subClassOf`

$Professor \equiv Fakultaetsmitglied$

- "Die Fakultätsmitglieder sind genau die Professoren."
- entspricht $(\forall x)(Professor(x) \leftrightarrow Fakultaetsmitglied(x))$
- entspricht `owl:equivalentClass`

$\mathcal{ALC}$ - komplexe Klassen

Konjunktion $\sqcap$ entspricht `owl:intersectionOf`

Disjunktion $\sqcup$ entspricht `owl:unionOf`

Negation $\neg$ entspricht `owl:complementOf`

Beispiel:
$\mathit{Professor} \sqsubseteq (\mathit{Person} \sqcap \mathit{Universitaetsangehoeriger}) \sqcup (\mathit{Person} \sqcap \neg\mathit{Doktorand} \sqcap \mathit{Hochschulabsolvent}))$

Prädikatenlogik:
$(\forall x)(\mathit{Professor}(x) \to ((\mathit{Person}(x) \land \mathit{Universitaetsangehoeriger}(x)) \lor \mathit{Person}(x) \land \neg\mathit{Doktorant}(x) \land \mathit{Hochschulabsolvent}(x))$

$\mathcal{ALC}$ - Quantoren auf Rollen

$Pruefung \sqsubseteq \forall hatPruefung.Professor$

- "Jede Prüfung hat nur Professoren als Prüfer."
- $(\forall x)(Pruefung(x) \to (\forall y)hatPruefer(x, y) \to Professor(y)))$
- entspricht `owl:allValuesFrom`

$Professor \sqsubseteq \exists hatPruefer.Person$

- "Jede Prüfung hat mindestens einen Prüfer."
- $(\forall x)(Pruefung(x) \to (\exists y)(hatPruefer(x, y) \land Person(y)))$
- entspricht `owl:someValuesFrom`

(Weitere) OWL-Konstrukte in $\mathcal{ALC}$

`owl:Nothing`:	$\bot \equiv C \sqcap \neg C$
`owl:Thing`:	$\top \equiv C \sqcup \neg C$
`owl:disjointWith`:	$C \sqcap D \equiv \bot$
(gleichbedeutend:)	$C \sqsubseteq \neg D$
`rdfs:range`:	$\top \sqsubseteq \forall R.C$
`rdfs:domain`:	$\exists R.\top \sqsubseteq C$

$\mathcal{ALC}$ - formale Syntax

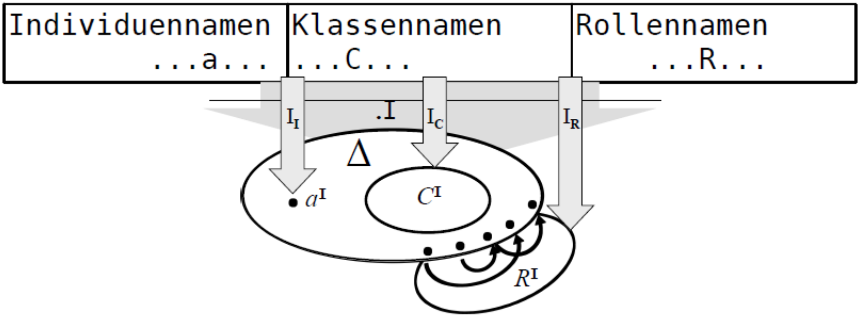
- Folgende Syntaxregeln erzeugen Klassen in $\mathcal{ALC}$. Dabei ist A eine atomare Klasse und R eine Rolle
  $C, D \rightarrow A|\top|\bot|\neg C|C \sqcap D|C \sqcup D|\forall R.C|\exists R.C$
- Eine $\mathcal{ALC}$-TBox besteht aus Aussagen der Form $C \sqsubseteq D$ und $C \equiv D$, wobei C, D Klassen sind.
- Eine $\mathcal{ALC}$-ABox besteht aus Aussagen der Form $C(a)$ und $R(a, b)$, wobei C eine komplexe Klassen, R eine Rolle und a, b Individuen sind.
- Eine $\mathcal{ALC}$-Wissensbasis besteht aus einer ABox und einer TBox.

$\mathcal{ALC}$ - Semantik (Interpretationen)

- wir definieren modelltheoretische Semantik für $\mathcal{ALC}$ (d.h. Folgerung wird über Interpretationen definiert)
- eine Interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ besteht aus
  - einer Menge $\triangle^{\mathcal{I}}$, genannt Domäne und
  - einer Funktion $\cdot^{\mathcal{I}}$, die abbildet von
    - Individuennamen a auf Domänenelemente $a \in \triangle^{\mathcal{I}}$
    - Klassennamen C auf Mengen von Domänenelementen $C^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}}$
    - Rollennamen R auf Mengen von Paaren von Domänenelementen $R^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$

$\mathcal{ALC}$- Semantik (Interpretationen)

- schematisch:



$\mathcal{ALC}$ - Semantik (komplexe Klassen)

- wird auf komplexe Klassen erweitert:

$\top^{\mathcal{I}} = \triangle^{\mathcal{I}}$

$\bot^{\mathcal{I}} = \emptyset$

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$

$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$

$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall(x, y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$

$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists(x, y) \in R^{\mathcal{I}} \text{ mit } y \in C^{\mathcal{I}}\}$

$(\neg C)^{\mathcal{I}} = \triangle^{\mathcal{I}} \setminus C^{\mathcal{I}}$

$\mathcal{ALC}$ - Semantik (Axiome, Modelle)

... und schließlich auf Axiome:

- $C(a)$ gilt in $\mathcal{I}$, wenn: $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $R(a, b)$ gilt in $\mathcal{I}$, wenn: $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- $C \sqsubseteq D$ gilt in $\mathcal{I}$, wenn: $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $C \equiv D$ gilt in $\mathcal{I}$, Wenn: $C^{\mathcal{I}} = D^{\mathcal{I}}$

Interpretationen, die ein Axiom (bzw. eine Menge von Axiomen) erfüllen, nennt man **Modelle.**

Interpretationen - Beispiele

TBox $\mathcal{T}$:

$$\begin{aligned}
\texttt{Man} &\equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
\texttt{Woman} &\sqsubseteq \texttt{Person} \\
\texttt{Mother} &\equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top
\end{aligned}$$

ABox $\mathcal{A}$:

$$\begin{aligned}
&\texttt{Man}(\texttt{STEPHEN}). \\
&\neg\texttt{Man}(\texttt{MONICA}). \\
&\texttt{Woman}(\texttt{JESSICA}). \\
&\texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{aligned}$$

Für alle folgenden Interpretationen $\mathcal{I}$:

- Domäne $\Delta^{\mathcal{I}} = \{\mathit{MONICA}, \mathit{JESSICA}, \mathit{STEPHEN}\}$
- Objekte werden aus sich selbst abgebildet ($a^{\mathcal{I}} = a$)

Interpretationen - Beispiele

TBox $\mathcal{T}$:

$$\begin{aligned}\texttt{Man} &\equiv \neg\texttt{Woman} \sqcap \texttt{Person}\\ \texttt{Woman} &\sqsubseteq \texttt{Person}\\ \texttt{Mother} &\equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top\end{aligned}$$

ABox $\mathcal{A}$:

$$\begin{aligned}&\texttt{Man}(\texttt{STEPHEN}).\\ &\neg\texttt{Man}(\texttt{MONICA}).\\ &\texttt{Woman}(\texttt{JESSICA}).\\ &\texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).\end{aligned}$$

$$\begin{aligned}\texttt{Man}^{\mathcal{I}_1} &= \{\texttt{JESSICA}, \texttt{STEPHEN}\}\\ \texttt{Woman}^{\mathcal{I}_1} &= \{\texttt{MONICA}, \texttt{JESSICA}\}\\ \texttt{Mother}^{\mathcal{I}_1} &= \emptyset\\ \texttt{Person}^{\mathcal{I}_1} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\}\\ \texttt{hasChild}^{\mathcal{I}_1} &= \{(\texttt{STEPHEN}, \texttt{JESSICA})\}\end{aligned}$$

Interpretationen - Beispiele

TBox $\mathcal{T}$: $\mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person}$

$\mathcal{I}_1 \not\models \mathcal{T}$ $\mathtt{Woman} \sqsubseteq \mathtt{Person}$

$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists\mathtt{hasChild}.\top$

ABox $\mathcal{A}$: $\mathtt{Man}(\mathtt{STEPHEN}).$

$\mathcal{I}_1 \models \mathcal{A}$ $\neg\mathtt{Man}(\mathtt{MONICA}).$

$\mathtt{Woman}(\mathtt{JESSICA}).$

$\mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).$

$$
\begin{aligned}
\mathtt{Man}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{STEPHEN}\} \\
\mathtt{Woman}^{\mathcal{I}_1} &= \{\mathtt{MONICA}, \mathtt{JESSICA}\} \\
\mathtt{Mother}^{\mathcal{I}_1} &= \emptyset \\
\mathtt{Person}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\} \\
\mathtt{hasChild}^{\mathcal{I}_1} &= \{(\mathtt{STEPHEN}, \mathtt{JESSICA})\}
\end{aligned}
$$

Interpretationen - Beispiele

TBox $\mathcal{T}$:

$$\begin{aligned}
\texttt{Man} &\equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
\texttt{Woman} &\sqsubseteq \texttt{Person} \\
\texttt{Mother} &\equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top
\end{aligned}$$

ABox $\mathcal{A}$:

$$\begin{aligned}
&\texttt{Man}(\texttt{STEPHEN}). \\
&\neg\texttt{Man}(\texttt{MONICA}). \\
&\texttt{Woman}(\texttt{JESSICA}). \\
&\texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{aligned}$$

$$\begin{aligned}
\texttt{Man}^{\mathcal{I}_2} &= \{\texttt{STEPHEN}\} \\
\texttt{Woman}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}\} \\
\texttt{Mother}^{\mathcal{I}_2} &= \emptyset \\
\texttt{Person}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\} \\
\texttt{hasChild}^{\mathcal{I}_2} &= \emptyset
\end{aligned}$$

Interpretationen - Beispiele

TBox $\mathcal{T}$: $\quad \mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person}$

$\mathcal{I}_2 \models \mathcal{T}$ $\quad \mathtt{Woman} \sqsubseteq \mathtt{Person}$

$$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists\mathtt{hasChild}.\top$$

ABox $\mathcal{A}$: $\quad \mathtt{Man}(\mathtt{STEPHEN}).$

$\mathcal{I}_2 \not\models \mathcal{A}$ $\quad \neg\mathtt{Man}(\mathtt{MONICA}).$

$$\mathtt{Woman}(\mathtt{JESSICA}).$$

$$\mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).$$

$$\begin{aligned}
\mathtt{Man}^{\mathcal{I}_2} &= \{\mathtt{STEPHEN}\} \\
\mathtt{Woman}^{\mathcal{I}_2} &= \{\mathtt{JESSICA}, \mathtt{MONICA}\} \\
\mathtt{Mother}^{\mathcal{I}_2} &= \emptyset \\
\mathtt{Person}^{\mathcal{I}_2} &= \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\} \\
\mathtt{hasChild}^{\mathcal{I}_2} &= \emptyset
\end{aligned}$$

Interpretationen - Beispiele

TBox $\mathcal{T}$:

$$\begin{aligned}
\texttt{Man} &\equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
\texttt{Woman} &\sqsubseteq \texttt{Person} \\
\texttt{Mother} &\equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top
\end{aligned}$$

ABox $\mathcal{A}$:

$$\begin{aligned}
&\texttt{Man}(\texttt{STEPHEN}). \\
&\neg\texttt{Man}(\texttt{MONICA}). \\
&\texttt{Woman}(\texttt{JESSICA}). \\
&\texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{aligned}$$

$$\begin{aligned}
\texttt{Man}^{\mathcal{I}_3} &= \{\texttt{STEPHEN}\} \\
\texttt{Woman}^{\mathcal{I}_3} &= \{\texttt{JESSICA}, \texttt{MONICA}\} \\
\texttt{Mother}^{\mathcal{I}_3} &= \{\texttt{MONICA}\} \\
\texttt{Person}^{\mathcal{I}_3} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\} \\
\texttt{hasChild}^{\mathcal{I}_3} &= \{(\texttt{MONICA}, \texttt{STEPHEN}), (\texttt{STEPHEN}, \texttt{JESSICA})\}
\end{aligned}$$

Interpretationen - Beispiele

TBox $\mathcal{T}$:

$\mathcal{I}_3 \models \mathcal{T}$

$$\mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person}$$
$$\mathtt{Woman} \sqsubseteq \mathtt{Person}$$
$$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists\mathtt{hasChild}.\top$$

ABox $\mathcal{A}$:

$\mathcal{I}_3 \models \mathcal{A}$

$$\mathtt{Man}(\mathtt{STEPHEN}).$$
$$\neg\mathtt{Man}(\mathtt{MONICA}).$$
$$\mathtt{Woman}(\mathtt{JESSICA}).$$
$$\mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).$$

$$\mathtt{Man}^{\mathcal{I}_3} = \{\mathtt{STEPHEN}\}$$
$$\mathtt{Woman}^{\mathcal{I}_3} = \{\mathtt{JESSICA}, \mathtt{MONICA}\}$$
$$\mathtt{Mother}^{\mathcal{I}_3} = \{\mathtt{MONICA}\}$$
$$\mathtt{Person}^{\mathcal{I}_3} = \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\}$$
$$\mathtt{hasChild}^{\mathcal{I}_3} = \{(\mathtt{MONICA}, \mathtt{STEPHEN}), (\mathtt{STEPHEN}, \mathtt{JESSICA})\}$$

$\mathcal{ALC}$ alternative Semantik

- Übersetzung von TBox-Aussagen in die Prädikatenlogik mittels der Abbildung π (rechts).
- Dabei sind C, D komplexe Klassen, R eine Rolle und A eine atomare Klasse.

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

OWL und $\mathcal{ALC}$

Folgende OWL DL Sprachelemente sind in $\mathcal{ALC}$ repräsentierbar:

- Klassen, Rollen, Individuen
- Klassenzugehörigkeit, Rollenverknüpfungen
- `owl:Thing` und `owl:Nothing`
- Klasseninklusionen, -äquivalenz, -disjunktheit
- `owl:intersectionOf`, `owl:unionOf`
- `owl:complementOf`
- `owl:allValuesFrom`, `owl:someValuesFrom`
- `rdfs:range` und `rdfs:domain`

Gliederung

OWL als $\mathcal{SHOIN}$(D) - Individuen

owl:sameAs

- gibt an dass zwei Individuennamen dasselbe Domänenelement bezeichnen
- DL: $a = b$
- FOL: Erweiterung durch Gleichheitsprädikat

owl:differentFrom

- gibt an dass zwei Individuennamen unterschiedliche Domänenelemente bezeichnen
- DL: $a \neq b$
- FOL: $\neg(a = b)$

OWL als $\mathcal{SHOIN}$(D) - Nominals

Abgeschlossene Klassen

- **owl:oneOf**
  - definiert eine Klasse durch vollständige Aufzählung ihrer Instanzen
  - DL: $C \equiv \{a, b, c\}$
  - FOL: $(\forall x)(C(x) \leftrightarrow (x = a \vee x = b \vee x = c))$
- **owl:hasValue**
  - "erzwingt" Rolle zu einem bestimmten Individuum
  - darstellbar mittels *owl:someValuesFrom* und *owl:oneOf*
  - DL: $C \equiv \exists r.\{a\}$

OWL als $\mathcal{SHOIN}$(D) - Kardinalität

Zahlenrestriktionen mittels Gleichheitsprädikat

```
<owl:Class rdf:about="#Pruefung">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hatPruefer"/>
      <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">
      2
      </owl:maxCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

“Eine Prüfung kann höchstens zwei Prüfer haben.”
DL: $Pruefung \sqsubseteq\ \leq 2 hatPruefer$

In FOL: (P...Pruefung, h...hatPruefer)
$(\forall x)(P(x) \rightarrow \neg(\exists x1)(\exists x2)(\exists x3)(x1 \neq x2 \wedge x2 \neq x3 \wedge x1 \neq x3 \wedge h(x, x1) \wedge h(x, x2) \wedge h(x, x3)))$
Entsprechend für die anderen Zahlenrestriktionen

OWL als $\mathcal{SHOIN}$(D) - Rollen

rdfs:subPropertyOf
spezifiziert Unterrolle-Oberrolle-Beziehung
DL: $R \sqsubseteq S$
FOL: $(\forall x)(\forall y)(R(x, y) \rightarrow S(x, y))$

Rollenäquivalenz: $R \equiv S$
inverse Rollen: $R \equiv S^{-1}$
Konstruktor für Rollen zur Bildung der Inversen
FOL: $(\forall x)(\forall y)(R(x, y) \leftrightarrow S(x, y))$

transitive Rollen: Trans(R)
FOL: $(\forall x)(\forall y)(\forall z)(R(x, y) \wedge R(y, z) \rightarrow R(x, z))$

Symmetrie: $R \equiv R^{-1}$
Funtionalität: $\top \sqsubseteq \leq 1 R$
Inverse Funktionalität: $\top \sqsubseteq \leq 1 R^{-1}$

OWL als $\mathcal{SHOIN}$(D) - Überblick

Erlaubt sind:

- $\mathcal{ALC}$
- Gleichheit und Ungleichheit zwischen Individuen
- Abgeschlossene Klassen
- Zahlenrestriktionen
- Subrollen und Rollenäquivalenz
- Inverse und transitive Rollen
- Datentypen

DLs - Nomenklatur

- $\mathcal{ALC}$: Attribute Language with Complement
- $\mathcal{S}$: $\mathcal{ALC}$ + Rollentransitivität
- $\mathcal{H}$: Subrollenbeziehung
- $\mathcal{O}$: abgeschlossene Klassen
- $\mathcal{I}$: inverse Rollen
- $\mathcal{N}$: Zahlenrestrktionen $\leq n\, R$ etc.
- $\mathcal{Q}$: Qualifizierende Zahlenrestriktionen $\leq n\, R.C$ etc.
- (D): Datentypen
- $\mathcal{F}$: Funktionale Rollen

- OWL DL ist $\mathcal{SHOIN}$(D)
- OWL Lite ist $\mathcal{SHIF}$(D)

DL-Syntax - Übersicht

Concepts	
Atomic	A, B
Not	$\neg C$
And	$C \sqcap D$
Or	$C \sqcup D$
Exists	$\exists R.C$
For all	$\forall R.C$
At least	$\geq n\, R.C (\geq nR)$
At most	$\leq n\, R.C (\leq nR)$
Nominal	$i_1, ..., i_n$
Roles	
Atomic	R
Inverse	R^-

Concept Axioms (TBox)	
Subclass	$C \sqsubseteq D$
Equivalent	$C \equiv D$
Role Axioms (RBox)	
Subrole	$R \sqsubseteq S$
Transitivity	$Trans(S)$
Assertional Axioms (ABox)	
Instance	$C(a)$
Role	$R(a, b)$
same	$a = b$
different	$a \neq b$

$\mathcal{S}$ = $\mathcal{ALC}$ + Transitivity **OWL DL =** $\mathcal{SHOIN}$**(D)** (D: concrete domain)

DL-Syntax - Klassenkonstruktoren

Constructor	**DL-Syntax**	**Example**	**FOL Syntax**
intersectionOf	$C_1 \sqcap ... \sqcap C_n$	$Human \sqcap Male$	$C_1 \wedge ... \wedge C_n$
unionOf	$C_1 \sqcup ... \sqcup C_n$	$Doctor \sqcup Lawyer$	$C_1 \vee ... \vee C_n$
complementOf	$\neg C$	$\neg Male$	$\neg C(x)$
oneOf	$\{x_1\} \sqcup ... \sqcup \{x_n\}$	john $\sqcup$ mary	$x = x_1 \vee ... \vee x = x_n$
allValuesFrom	$\forall P.C$	$\forall$ hasChild.Doctor	$\forall y.P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	$\exists$ hasChild.Lawyer	$\exists y.P(x,y) \wedge C(y)$
maxCardinality	$\leq n\ P$	$\leq$1 hasChild	$\exists^{\leq n} y.P(x,y)$
minCardinality	$\geq n\ P$	$\geq$2 hasChild	$\exists^{\geq n} y.P(x,y)$

Beliebig komplexes Schachteln von Konstruktoren erlaubt.

Person$\sqcap\forall hasChild.(Doctor \sqcup \exists hasChild.Doctor)$

DL-Syntax - Axiome

Axiom	**DL-Syntax**	**Example**
subClassOf	$C_1 \sqsubseteq C_2$	$Human \sqsubseteq Animale \sqcap Biped$
equivalentClass	$C_1 \equiv C_2$	$Man \equiv Human \sqcap Male$
disjointWith	$C_1 \sqsubseteq \neg C_2$	$Male \sqsubseteq \neg Female$
sameAs	$x_1 \equiv x_2$	$President\_Bush \equiv G\_W\_Bush$
differentFrom	$x_1 \sqsubseteq \neg x_2$	$john \sqsubseteq \neg peter$
subPropertyOf	$P_1 \sqsubseteq P_2$	$hasDaughter \sqsubseteq hasChild$
equivalentProperty	$P_1 \equiv P_2$	$cost \equiv price$
inverseOf	$P_1 \equiv P_2^-$	$hasChild \equiv hasParent^-$
transitiveProperty	$P^+ \sqsubseteq P$	$ancestor^+ \sqsubseteq ancestor$
functionalProperty	$\top \sqsubseteq \leq 1 P$	$\top \sqsubseteq \leq 1 hasMother$
inverseFunctionalProperty	$\top \sqsubseteq \leq 1 P^-$	$\top \sqsubseteq \leq 1 hasSSN^-$

General Class Inclusion ($\sqsubseteq$) genügt: $C \equiv D$ gdw ($C \sqsubseteq D$ und $D \sqsubseteq C$)

Offensichtliche FOL-Äquivalenzen:

- $C \equiv D \Leftrightarrow (\forall x)(C(x) \leftrightarrow D(x))$
- $C \sqsubseteq D \Leftrightarrow (\forall x)(C(x) \rightarrow D(x))$

Wissensmodellierung OWA vs. CWA

OWA = Open World Assumption
Die Existenz von weiteren Individuen ist möglich, sofern sie nicht explizit ausgeschlossen wird.
OWL verwendet OWA!

CWA = Closed World Assumption
Es wird angenommen, dass die Wissensbasis alle Individuen enthält.

Wissensmodellierung OWA vs. CWA

OWA = Open World Assumption
Die Existenz von weiteren Individuen ist möglich, sofern sie nicht explizit ausgeschlossen wird.
OWL verwendet OWA!

CWA = Closed World Assumption
Es wird angenommen, dass die Wissensbasis alle Individuen enthält.

$\mathcal{K}$	query	DL answer	Prolog
child(Bill,Bob) Man(Bob)	$\mathcal{K} \models \forall \mathit{child}.\mathit{Man}(\mathit{Bill})$	don't know	yes

Wissensmodellierung OWA vs. CWA

OWA = Open World Assumption
Die Existenz von weiteren Individuen ist möglich, sofern sie nicht explizit ausgeschlossen wird.
OWL verwendet OWA!

CWA = Closed World Assumption
Es wird angenommen, dass die Wissensbasis alle Individuen enthält.

$\mathcal{K}$	query	DL answer	Prolog
child(Bill,Bob) Man(Bob)	$\mathcal{K} \models \forall child.Man(Bill)$	don't know	yes
$+ \leq 1 child.T(Bill)$	$\mathcal{K} \models \forall child.Man(Bill)$	yes	yes

Reasoning - Beispiele

Wissensbasen:

- K1 = { hasChild(Anna, Bob) }
- K2 = { Female(Anna) }

Ist Anna Instanz von $\exists hasChild$?

Reasoning - Beispiele

Wissensbasen:

- K1 = { hasChild(Anna, Bob) }
- K2 = { Female(Anna) }

Ist Anna Instanz von $\exists \mathit{hasChild}$?

Ja, in K1. Nein, in K2.

Reasoning - Beispiele

Wissensbasen:

- K1 = { hasChild(Anna, Bob) }
- K2 = { Female(Anna) }

Ist Anna Instanz von $\exists hasChild$?

Ja, in K1. Nein, in K2.

Ist Anna Instanz von $\neg\exists hasChild$?

Reasoning - Beispiele

Wissensbasen:

- K1 = { hasChild(Anna, Bob) }
- K2 = { Female(Anna) }

Ist Anna Instanz von $\exists hasChild$?

Ja, in K1. Nein, in K2.

Ist Anna Instanz von $\neg\exists hasChild$?

Nein, in K1. Nein, in K2.

Grund: Open World Assumption (Wir wissen nicht, ob Anna in K2 ein Kind hat.)

Reasoning - Beispiele

K3 = {hasChild(Anna, Bob), hasChild(Anna, Charlie)}

Ist Anna Instanz von = 2 hasChild in K3?

Reasoning - Beispiele

K3 = {hasChild(Anna, Bob), hasChild(Anna, Charlie)}

Ist Anna Instanz von = 2 hasChild in K3?

Nein, Anna könnte mehr Kinder haben.

Reasoning - Beispiele

K3 = {hasChild(Anna, Bob), hasChild(Anna, Charlie)}

Ist Anna Instanz von = 2 hasChild in K3?

Nein, Anna könnte mehr Kinder haben.

K4 =
$\{(\leq 2 hasChild)(Anna), hasChild(Anna, Bob), hasChild(Anna, Charlie)\}$

Ist Anna Instanz von = 2 hasChild in K4?

Reasoning - Beispiele

K3 = {hasChild(Anna, Bob), hasChild(Anna, Charlie)}

Ist Anna Instanz von = 2 hasChild in K3?

Nein, Anna könnte mehr Kinder haben.

K4 =
$\{(\leq 2 hasChild)(Anna), hasChild(Anna, Bob), hasChild(Anna, Charlie)\}$

Ist Anna Instanz von = 2 hasChild in K4?

Nein, da Bob und Charlie sich auf die gleiche Person beziehen können (OWL macht keine **unique names assumption** - siehe `owl:sameAs` oder `owl:differentFrom`).

Gliederung

Wichtige Inferenzprobleme

Globale Konsistenz der Wissensbasis $\mathcal{K} \models \text{false}?$

- Ist Wissensbasis sinnvoll? (Semantik: Hat $\mathcal{K}$ ein Modell?)

Klassenkonsistenz $C \equiv \bot?$

- Muss Klasse C leer sein?

Klasseninklusion (Subsumption) $C \sqsubseteq D?$

- Strukturierung der Wissensbasis

Klassenäquivalenz $C \equiv D?$

- Sind zwei Klassen eigentlich dieselbe?

Klassendisjunktheit $C \sqcap D = \bot?$

- Sind zwei Klassen disjunkt?

Klassenzugehörigkeit $C(a)$?

- Ist Individuum a in der Klasse C?

Instanzgenerierung (Retrieval) “alle x mit $C(x)$ finden”

- Finde alle (bekannten!) Individuen zur Klasse C.

Entscheidbarkeit von OWL DL

- Entscheidbarkeit: zu jedem Inferenzproblem gibt es einen immer terminierenden Algorithmus
- OWL DL ist Fragment von FOL, also könnten (im Prinzip) FOL-Inferenzalgorithmen (Resolution, Tableaux) verwendet werden.
- Diese terminieren aber nicht immer!
- Problem: Finde immer terminierende Algorithmen! Keine "naiven"Lösungen in Sicht!

Rückführung auf Unerfüllbarkeit

- Wir werden Tableauverfahren für OWL DL bzw. $\mathcal{ALC}$ abwandeln.
- Tableau- und Resolutionsverfahren zeigen Unerfüllbarkeit einer Theorie
- Rückführung der Inferenzprobleme auf das Finden von Inkonsistenzen in der Wissensbasis, d.h. zeigen der Unerfüllbarkeit der Wissensbasis!

Rückführung auf Unerfüllbarkeit

- Klassenkonsistenz $C \equiv \bot$ gdw.
  $KB \cup \{C(a)\}$ unerfüllbar (a neu)
- Klasseninklusion(Subsumption) $C \sqsubseteq D$ gdw.
  $KB \cup \{C \sqcap \neg D(a)\}$ unerfüllbar (a neu)
- Klassenäquivalenz $C \equiv D$ gdw.
  $C \sqsubseteq D$ und $D \sqsubseteq C$
- Klassendisjunktheit $C \sqcap D = \bot$ gdw.
  $KB \cup \{(C \sqcap D)(a)\}$ unerfüllbar (a neu)
- Klassenzugehörigkeit $C(a)$ gdw.
  $KB \cup \{(\neg C(a)\}$ unerfüllbar (a neu)
- Instanzgenerierung (Retrieval) alle $C(x)$ finden
  - Prüfe Klassenzugehörigkeit für alle Individuen.
  - Schwerer, dies gut zu implementieren!

Gliederung

Tableau - Transformation in NNF

Gegeben eine Wissensbasis W.

- Ersetze $C \equiv D$ durch $C \sqsubseteq D$ und $D \sqsubseteq C$.
- Ersetze $C \sqsubseteq D$ durch $\neg C \sqcup D$.
- Wende die Regeln auf der folgenden Folie an, bis es nicht mehr geht.

Resultierende Wissensbasis: NNF(W)

- Negationsnormalform von W.
- Negation steht nur noch direkt vor atomaren Klassen.

Tableau - Transformation in NNF

$$
\begin{aligned}
NNF(C) &= \text{C, falls C atomar ist}\\
NNF(\neg C) &= \neg C\text{, falls C atomar ist}\\
NNF(\neg\neg C) &= NNF(C)\\
NNF(C \sqcup D) &= NNF(C) \sqcup NNF(D)\\
NNF(C \sqcap D) &= NNF(C) \sqcap NNF(D)\\
NNF(\neg(C \sqcup D)) &= NNF(\neg C) \sqcap NNF(\neg D)\\
NNF(\neg(C \sqcap D)) &= NNF(\neg C) \sqcup NNF(\neg D)\\
NNF(\forall R.C) &= \forall R.NNF(C)\\
NNF(\exists R.C) &= \exists R.NNF(C)\\
NNF(\neg\forall R.C) &= \exists R.NNF(\neg C)\\
NNF(\neg\exists R.C) &= \forall R.NNF(\neg C)
\end{aligned}
$$

W und NNF(W) sind logisch äquivalent.

Tableau - NNF - Beispiel

- $P \sqsubseteq (E \sqcap U) \sqcup \neg(\neg E \sqcup D)$

Tableau - NNF - Beispiel

- $P \sqsubseteq (E \sqcap U) \sqcup \neg(\neg E \sqcup D)$
- Subklasse auflösen: $\neg P \sqcup ((E \sqcap U) \sqcup \neg(\neg E \sqcup D))$

Tableau - NNF - Beispiel

- $P \sqsubseteq (E \sqcap U) \sqcup \neg(\neg E \sqcup D)$

- Subklasse auflösen: $\neg P \sqcup ((E \sqcap U) \sqcup \neg(\neg E \sqcup D))$
- Negation nach innen: $\neg P \sqcup (E \sqcap U) \sqcup (\neg\neg E \sqcap \neg D)$

Tableau - NNF - Beispiel

- $P \sqsubseteq (E \sqcap U) \sqcup \neg(\neg E \sqcup D)$

- Subklasse auflösen: $\neg P \sqcup ((E \sqcap U) \sqcup \neg(\neg E \sqcup D))$
- Negation nach innen: $\neg P \sqcup (E \sqcap U) \sqcup (\neg\neg E \sqcap \neg D)$
- Endergebnis Negationsnormalform: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Naives Tableauverfahren

Rückführung auf Unerfüllbarkeit/Widerspruch

Idee:

- Gegeben Wissensbasis W.
- Erzeugung von Konzequenzen der Form $C(a)$ und $\neg C(a)$, bis Widerspruch gefunden.

Tableau einfaches Beispiel

Wissenbasis:
$C(a)$
$(\neg C \sqcap D)(a)$

Tableau einfaches Beispiel

Wissenbasis:
$C(a)$
$(\neg C \sqcap D)(a)$

$\neg C(a)$ ist logische Konzequenz

Widerspruch ist gefunden.

Tableau weiteres Beispiel

$C(a)$ $\neg C \sqcup D$ $\neg D(a)$

Ableitung von Konzequenzen:
$C(a)$
$\neg D(a)$
$(\neg C \sqcup D)(a)$

Nur Fallunterscheidung

1. $\neg C(a)$
   Widerspruch
2. $D(a)$
   Widerspruch

Teilen des Tableaus in zwei Zweige.

Tableau - Definitionen

- Tableauzweig:
  Endliche Menge von Aussagen der Form
  $C(a), \neg C(a), R(a, b)$.
- Tableau: Endliche Menge von Tableauzweigen.
- Tableauzweig ist abgeschlossen wenn er ein Paar Widersprüchlicher Aussagen $C(a)$ und $\neg C(a)$ enthält.

- Tableau ist abgeschlossen, wenn jeder Zweig von ihm abgeschlossen ist.

Tableau - Erzeugung

Name	Auswahl	Aktion
$C_{\mathcal{A}}$	$C(a) \in W$ (ABox)	Füge $C(a)$ hinzu.
$R_{\mathcal{A}}$	$R(a, b) \in W$ (ABox)	Füge $R(a, b)$ hinzu.
C	$C \in W$ (TBox)	Füge $C(a)$ für ein bekanntes Individuum a hinzu.
$\sqcap$	$(C \sqcap D)(a) \in A$	Füge $C(a)$ und $D(a)$ hinzu.
$\sqcup$	$(C \sqcup D)(a) \in A$	Dupliziere den Zweig. Füge zum einen Zweig $C(a)$ und zum anderen Zweig $D(a)$ hinzu.
$\exists$	$(\exists R.C)(a) \in A$	Füge $R(a, b)$ und $C(b)$ für neues Individuum b hinzu.
$\forall$	$(\forall R.C)(a) \in A$	Falls $R(a, b) \in A$, so füge $C(b)$ hinzu.

- Ist das resultierende Tableau abgeschlossen, so ist die ursprüngliche Wissensbasis unerfüllbar.
- Man wählt dabei immer nur solche Elemente aus, die auch wirklich zu neuen Elementen im Tableau führen. Ist dies nicht möglich, so terminiert der Algorithmus und W ist erfüllbar.

Tableau - Beispiel(1/2)

- P ... Professor
  E ... Person
  U ... Universitätsangehöriger
  D ... Doktorand
- Wissensbasis: $P \sqsubseteq (E \sqcap U) \sqcup (E \sqcap \neg D)$

  Ist $P \sqsubseteq E$ logische Konzequenz?

- Wissensbasis (mit Anfrage) in NNF:
  $\{\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E)(a)\}$

Tableau - Beispiel(2/2)

$\boxed{(P \sqcap \neg E)(a)}$

TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

Tableau - Beispiel(2/2)

$$\begin{tikzcd}
\boxed{(P \sqcap \neg E)(a)} \arrow[d, "\sqcap"] \\
\boxed{P(a)} \arrow[d, "\sqcap"] \\
\boxed{\neg E(a)}
\end{tikzcd}$$

TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

Tableau - Beispiel(2/2)

$$
\begin{tikzcd}
\boxed{(P \sqcap \neg E)(a)} \arrow[d, "\sqcap"] \\
\boxed{P(a)} \arrow[d, "\sqcap"] \\
\boxed{\neg E(a)} \arrow[d, "C"] \\
\boxed{(\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a)}
\end{tikzcd}
$$

TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

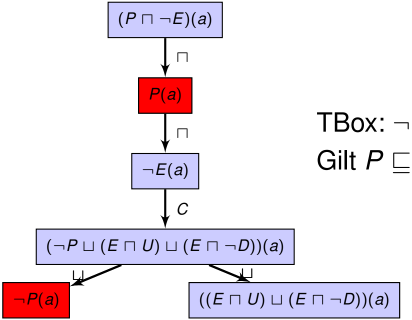
Tableau - Beispiel(2/2)

$$
\begin{tikzcd}
& (P \sqcap \neg E)(a) \arrow[d, "\sqcap"] & \\
& P(a) \arrow[d, "\sqcap"] & \\
& \neg E(a) \arrow[d, "C"] & \\
& (\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a) \arrow[dl, "\sqcup"'] \arrow[dr, "\sqcup"] & \\
\neg P(a) & & ((E \sqcap U) \sqcup (E \sqcap \neg D))(a)
\end{tikzcd}
$$

TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

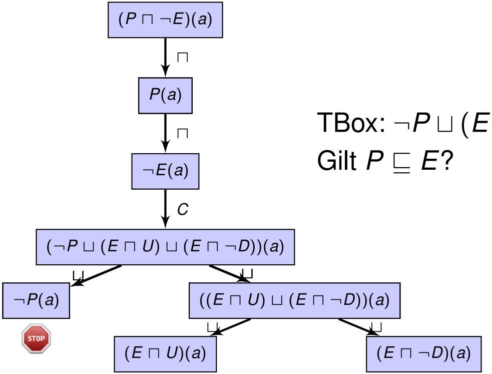
Tableau - Beispiel(2/2)

$$
\begin{tikzcd}
 & (P \sqcap \neg E)(a) \arrow[d, "\sqcap"] & \\
 & P(a) \arrow[d, "\sqcap"] & \\
 & \neg E(a) \arrow[d, "C"] & \\
 & (\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a) \arrow[dl, "\sqcup"'] \arrow[dr, "\sqcup"] & \\
\neg P(a) & & ((E \sqcap U) \sqcup (E \sqcap \neg D))(a)
\end{tikzcd}
$$

TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

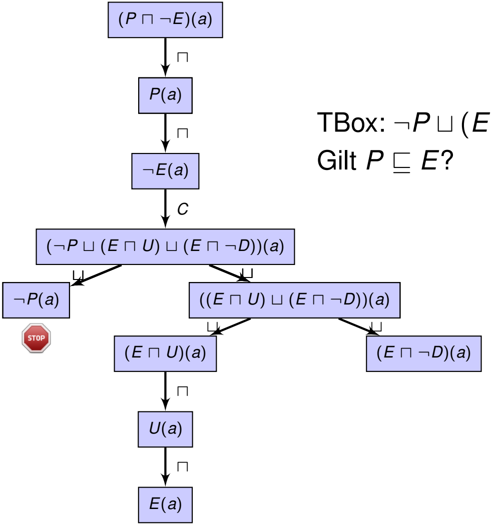
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

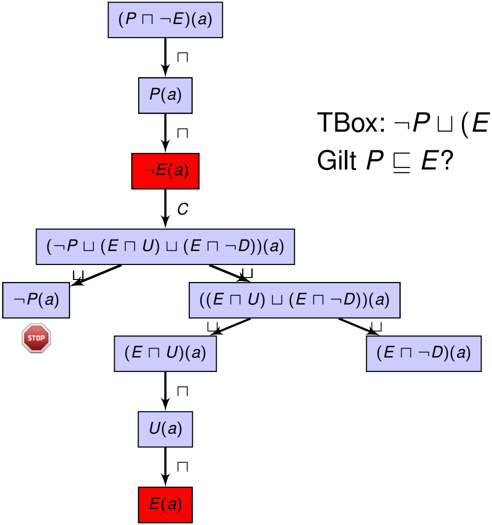
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

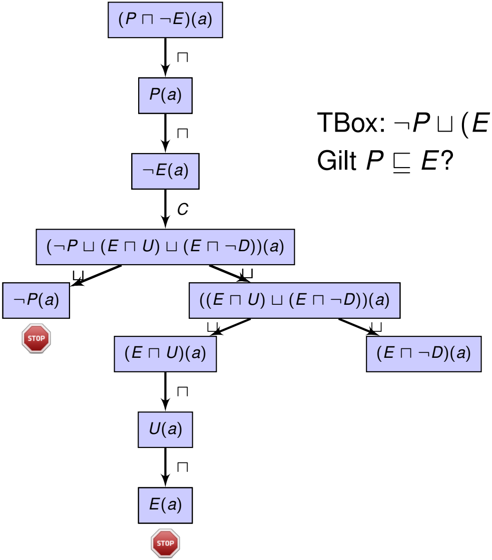
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

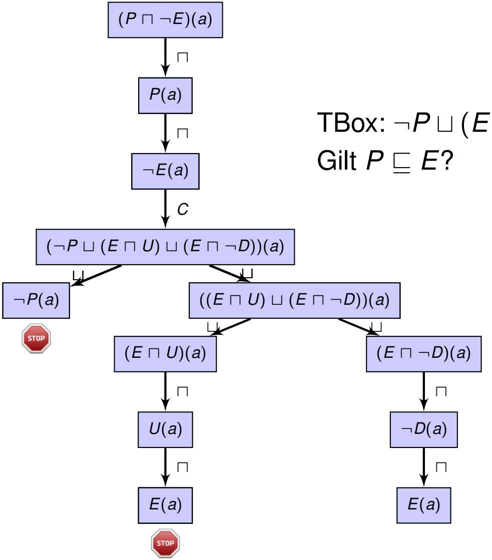
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

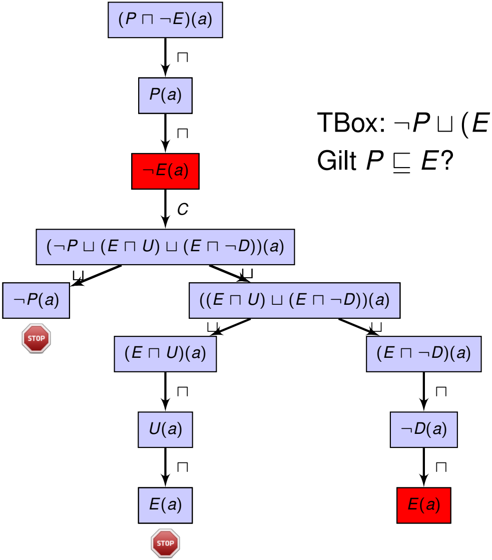
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

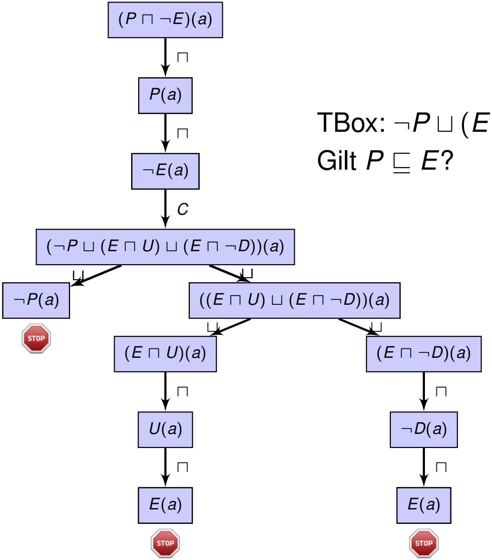
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

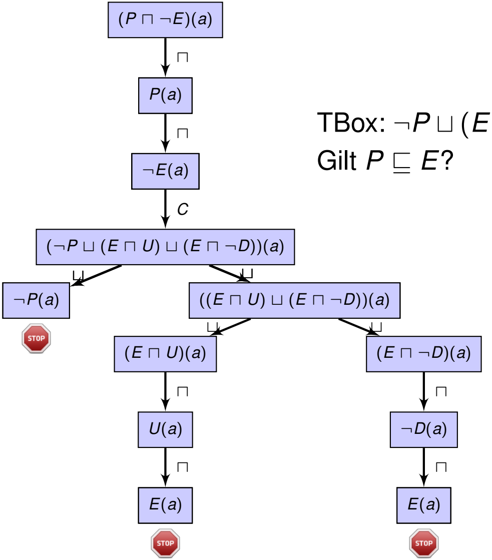
Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

Tableau - Beispiel(2/2)



TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$

Gilt $P \sqsubseteq E$?

Alle Pfade abgeschlossen $\rightarrow$ Wissensbasis unerfüllbar $\rightarrow$ $P \sqsubseteq E$.

Tableau - Terminierungsproblem

$Person(Bill)$

Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

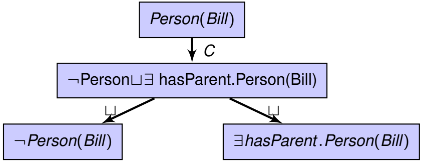
Tableau - Terminierungsproblem

$$
\begin{array}{c}
\boxed{Person(Bill)} \\
\downarrow C \\
\boxed{\neg \mathrm{Person} \sqcup \exists\ \mathrm{hasParent.Person(Bill)}}
\end{array}
$$

Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

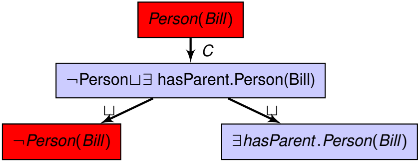
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

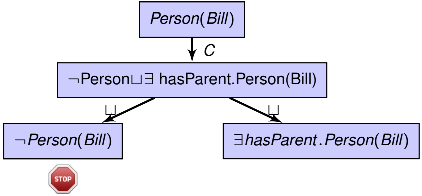
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

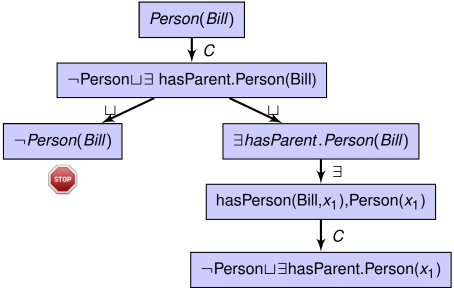
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

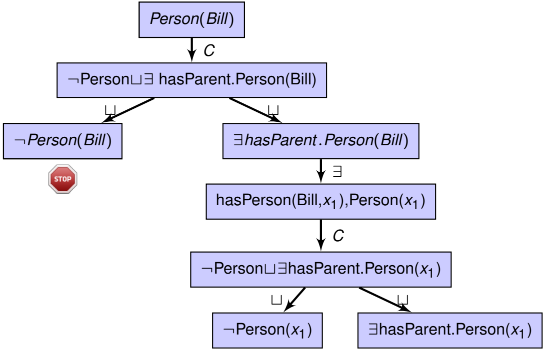
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

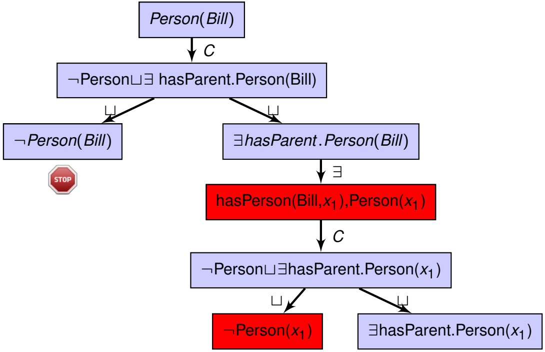
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

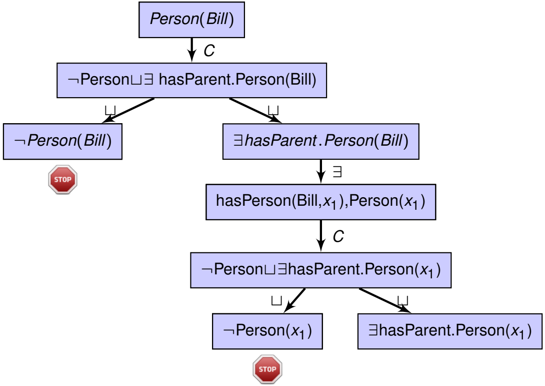
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

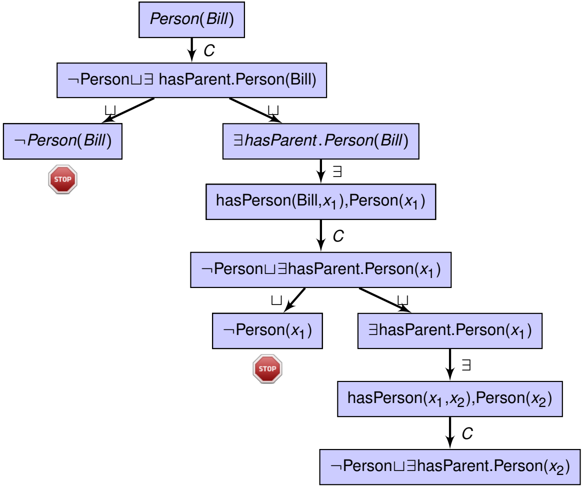
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

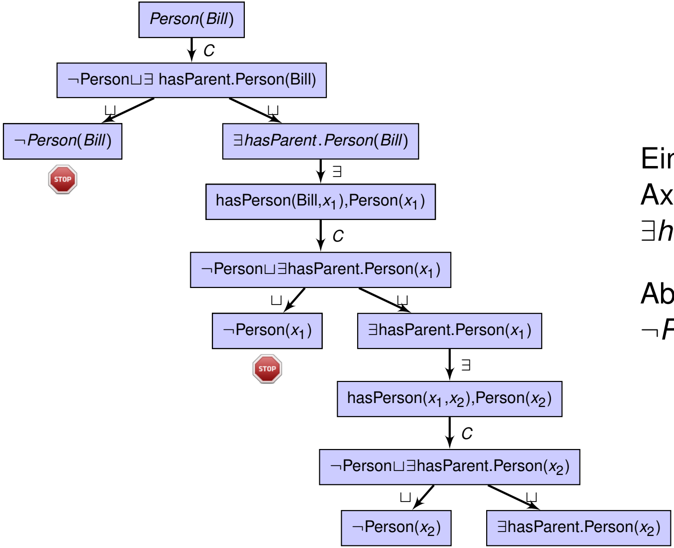
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

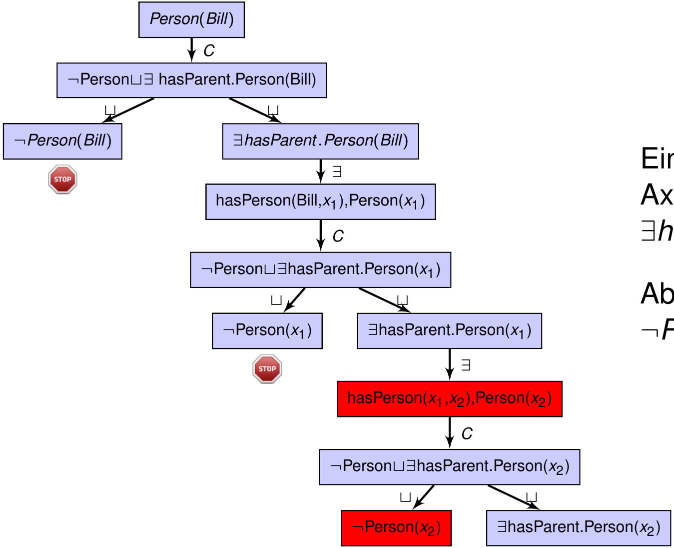
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

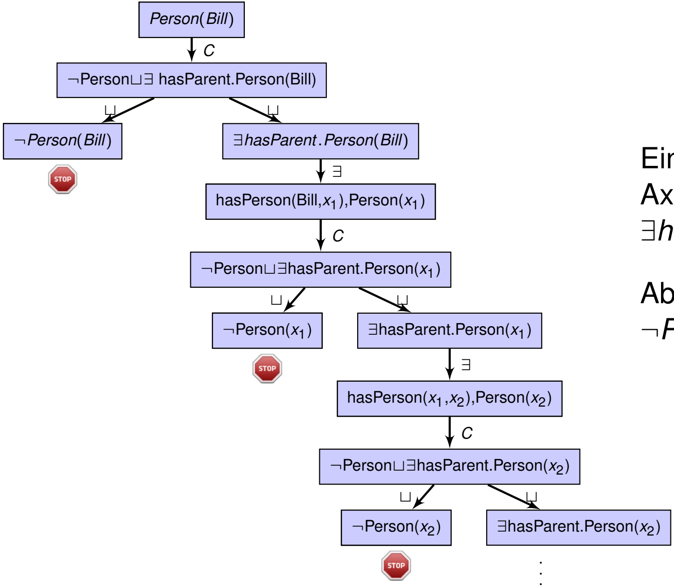
Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

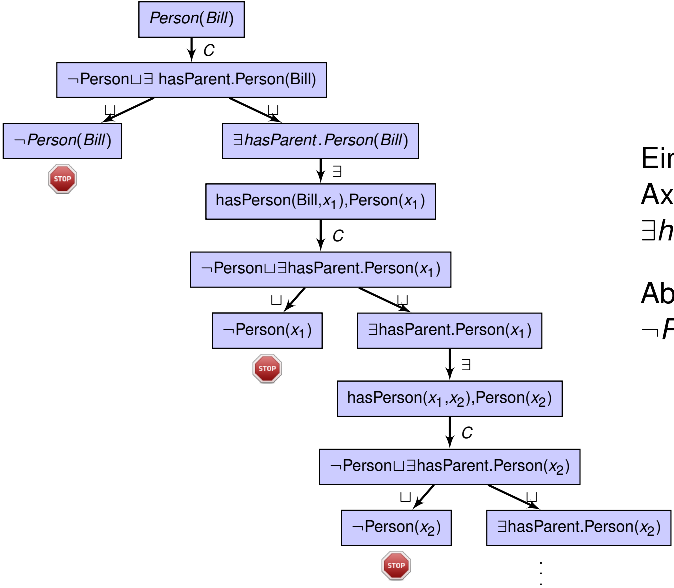
Abzuleiten: $\neg Person(Bill)$

Tableau - Terminierungsproblem



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$



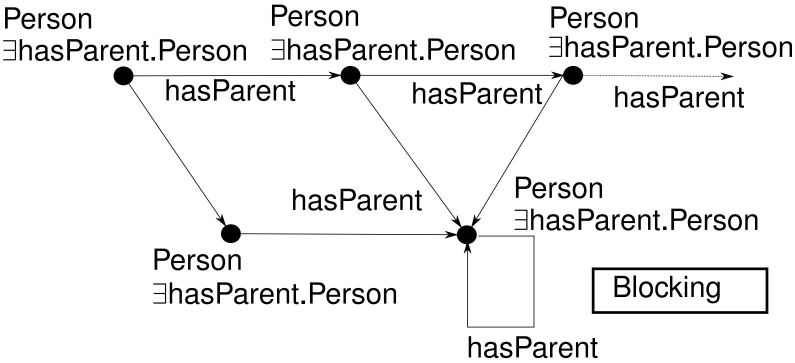
Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

Problem tritt auf durch Existenzquantoren (und minCardinality)

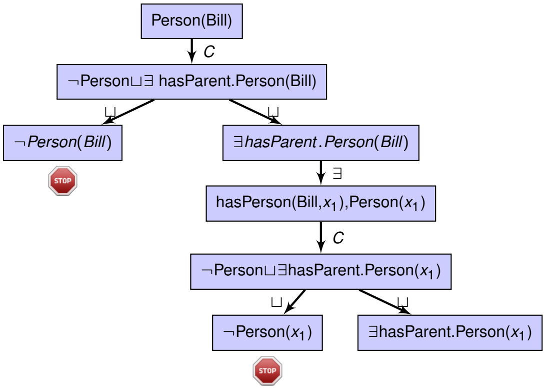
Tableau - Blocking - Idee

Wir haben folgendes konstruiert:



D.h. Wiederverwendung alter Knoten!
Es muss natürlich formal nachgewiesen werden, dass das ausreicht!

Tableau mit Blocking



Einziges Axiom: $\neg Person \sqcup \exists hasParent.Person$

Abzuleiten: $\neg Person(Bill)$

$$\sigma(Bill) = \{Person, \neg Person \sqcup \exists hasParent.Person, \exists hasParent.Person\}$$

$$\sigma(x_1) = \{Person, \neg Person \sqcup \exists hasParent.Person, \exists hasParent.Person\}$$

$\sigma(x_1) \subseteq \sigma(Bill)$, d.h. Bill blockt x_1

Tableau - Blocking - Definition

Die Auswahl von $(\exists R.C)(a)$ im Tableauzweig A ist

blockiert, falls es ein Individuum b gibt, so dass
$\{C | C(a) \in A\} \subseteq \{C | C(b) \in A\}$ ist.

Zwei Möglichkeiten der Terminierung:

1. Abschluss des Tableaus.
   Dann Wissensbasis unerfüllbar.
2. Keine ungeblockte Auswahl führt zu Erweiterung.
   Dann Wissensbasis erfüllbar.

Tableau für OWL DL

- Die Grundidee ist dieselbe!
- Kompliziertere Blockingregeln müssen verwendet werde.
- Reasonerimplementierungen arbeiten oft mit zahlreichen Optimierungen wegen problematischer Performance einer “naiven„ Implementierung.

Tableau - Beweiser

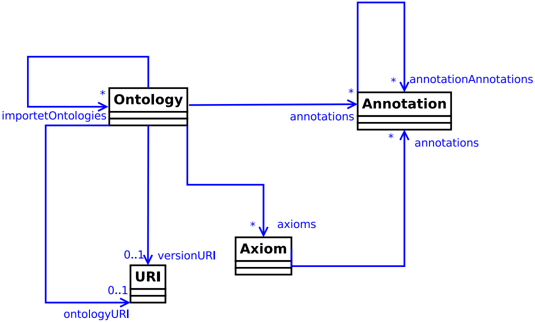
- Fact
  - `http://www.cs.man.ac.uk/~horrocks/FaCT/`
  - $\mathcal{SHIQ}$
- Fact++
  - `http://owl.man.ac.uk/factplusplus/`
  - $\mathcal{SHOIQ}$(D)
- Pellet
  - `http://clarkparsia.com/pellet/`
  - $\mathcal{SHOIN}$(D)
- RacerPro
  - `http://www.sts.tu-harburg.de/~r.f.moeller/racer/`
  - $\mathcal{SHIQ}$(D)
- HermIT
  - `http://www.hermit-reasoner.com`
  - Hypertableau-Kalkül

Gliederung

OWL 2

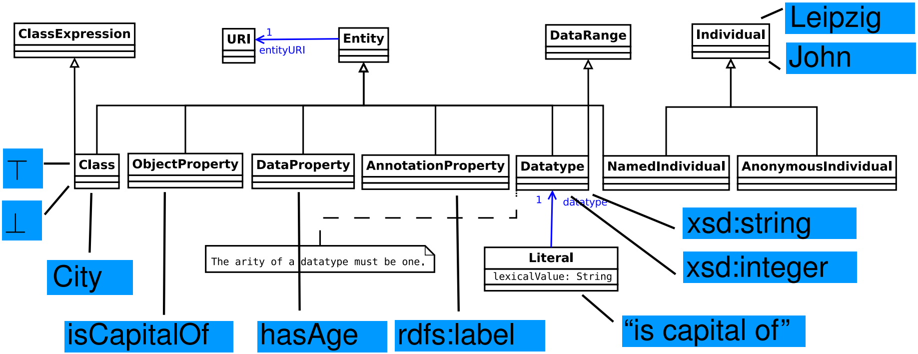
- Praktische Anwendung von OWL 1 legte Probleme offen, dienicht schwerwiegend sind, aber in der Menge zur Arbeit an neuer Spezifikation geführt haben
- Basiert auf Beschreibungslogik $\mathcal{SROIQ}$**(D)**
- Quellen/Literatur:
  - W3C OWL2 Spezifikation
  - Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The next step for OWL. Journal of Web Semantics, 2008.
- Spezifikation verwendet 22 UML Klassendiagrame (siehe folgende Folien)

OWL 2 - Ontologie Struktur

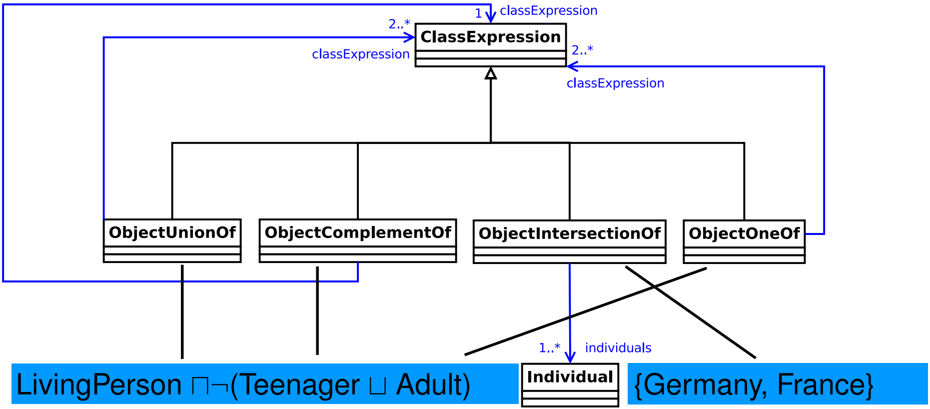


- Wie bisher:
  - Ontologie = **Menge von Axiomen** (+ Kopf)
  - 1 Axiom = 1...n RDF Triple
- Physikalischer Ort muss Versions-URI entsprechen (falls vorhanden) und aktuelle Version muss an ontology URI zu finden sein (falls vorhanden)

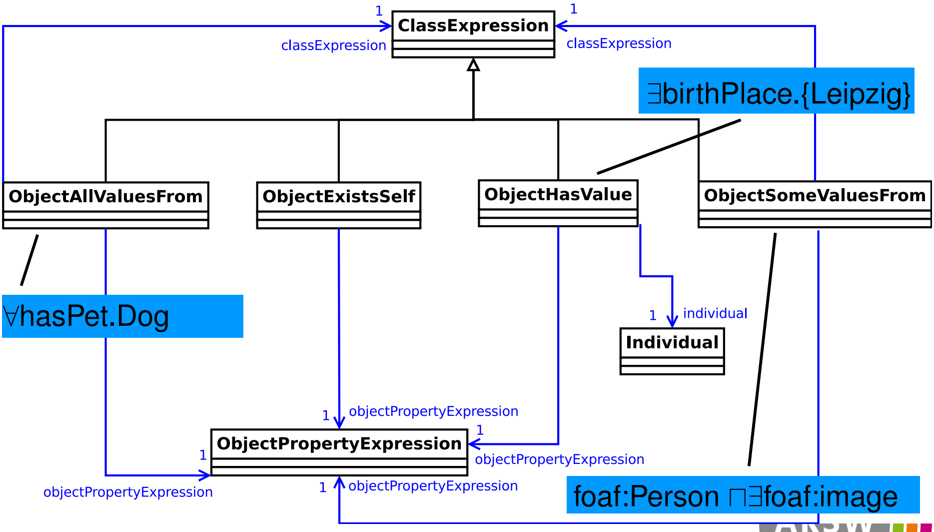
OWl 2 - Entities



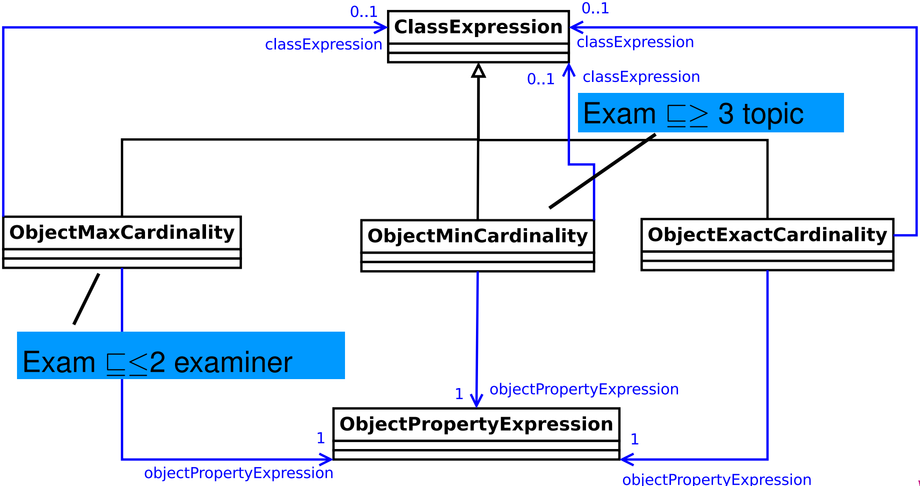
OWL 2 – Class Expressions



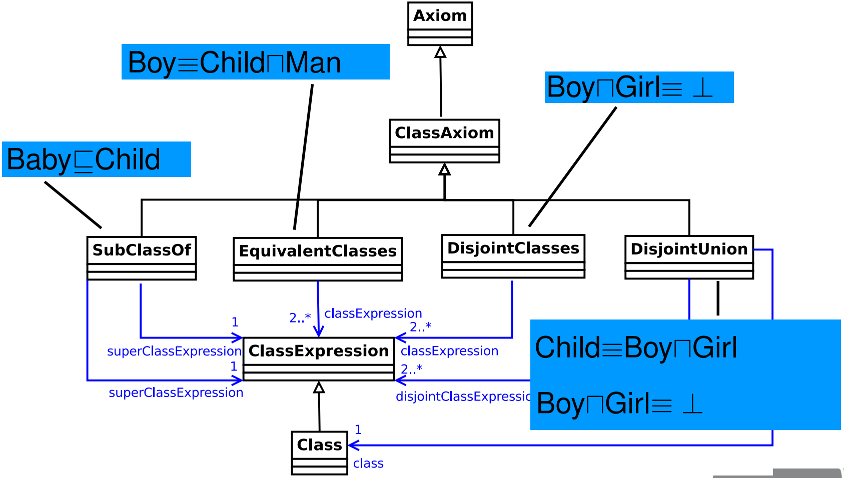
OWL 2 – Class Expressions



OWL 2 – Class Expressions

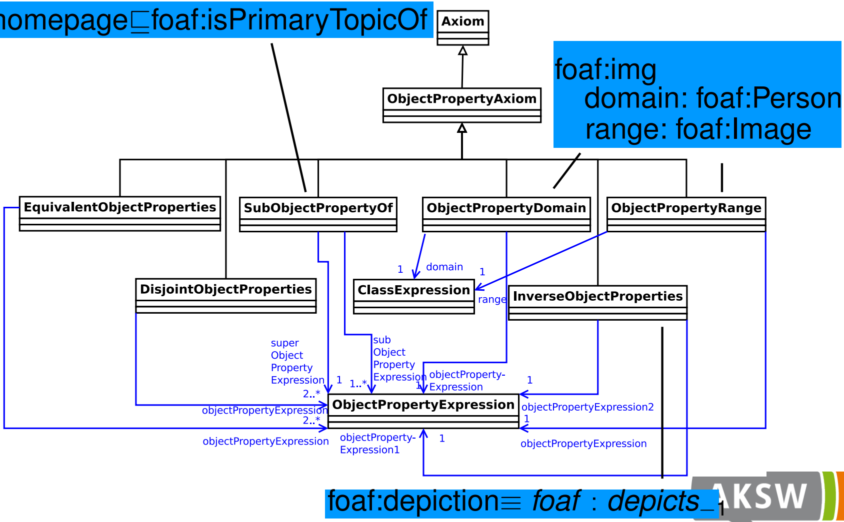


OWL 2 - Axioms

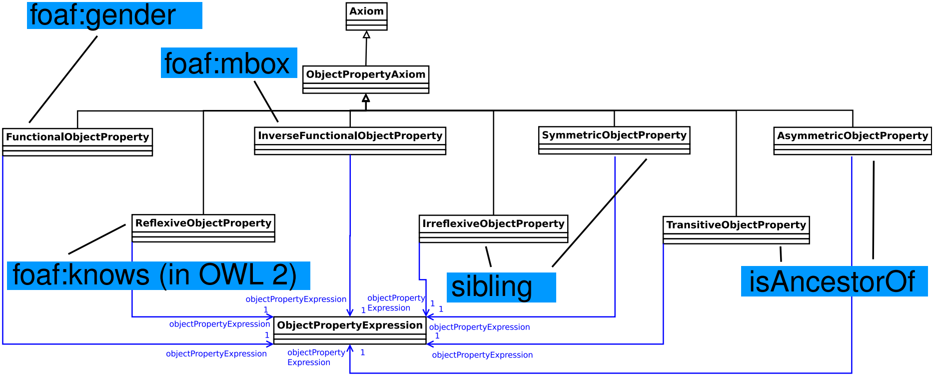


Axiom
ClassAxiom
Boy≡Child⊓Man
Boy⊓Girl≡ ⊥
Baby⊑Child
SubClassOf
EquivalentClasses
DisjointClasses
DisjointUnion
1
superClassExpression
1
superClassExpression
2..*
classExpression
2..*
classExpression
2..*
disjointClassExpressio
ClassExpression
Child≡Boy⊓Girl
Boy⊓Girl≡ ⊥
Class
1
class

OWL 2 - Axioms


foaf:homepage ⊑ foaf:isPrimaryTopicOf
foaf:img
domain: foaf:Person
range: foaf:Image
foaf:depiction ≡ $foaf:depicts^{-1}$
Axiom
ObjectPropertyAxiom
EquivalentObjectProperties
SubObjectPropertyOf
ObjectPropertyDomain
ObjectPropertyRange
DisjointObjectProperties
ClassExpression
InverseObjectProperties
ObjectPropertyExpression
domain
range
1
super Object Property Expression
sub Object Property Expression
1..*
2..*
objectProperty-Expression
objectPropertyExpression
objectPropertyExpression2
objectProperty-Expression1
objectPropertyExpression

OWL 2 - Axioms


foaf:gender
foaf:mbox
Axiom
ObjectPropertyAxiom
FunctionalObjectProperty
InverseFunctionalObjectProperty
SymmetricObjectProperty
AsymmetricObjectProperty
ReflexiveObjectProperty
IrreflexiveObjectProperty
TransitiveObjectProperty
foaf:knows (in OWL 2)
sibling
isAncestorOf
ObjectPropertyExpression
objectPropertyExpression
objectProperty Expression
1

Unterschiede zwischen OWL 1 und OWL 2

- Ausdrucksstärke (Auszug):
  - Qualified number restrictions: $\geq 2 hasChild.male$
  - Property chains: $locatedIn \circ partOf \sqsubseteq locatedIn$
  - Properties of properties: reflexive, irreflexive, asymmetric
  - Datatypes:
    - Facets: $Person \sqcap hasAge \geq 18$
    - Unterstütze Datentypen: owl:boolean, owl:string, xsd:integer, xsd:dateTime, xsd:hexBinary, owl:real, ...
  - Easy keys / unique identifiers:
    HasKey(Person ssn)
    HasKey(Transplantation donor recipient organ)

Unterschiede zwischen OWL 1 und OWL 2

- Verwendung von MOF/UML zur Spezifikation
- Typisierung in OWL 2 DL weniger restriktiv, z.B. kann eine **URI sowohl eine Klasse als auch ein Individual bezeichnen:**
  Eagle(Harry)
  EndangeredSpecies(Eagle)
- **Annotierung von Axiomen** und Entities, z.B. (in funktionaler Syntax) ClassAssertion(
  Annotation(author Jens)
  Annotation(confidence 0.95/wl:real)
  OWL2 GreatOntologyLanguage)

OWL 2 Profile (EL, RL, QL)

- OWL 2 Full
  - OWL-**RL Full**: skalierbares Reasoning mit **Regeln**; Obermenge von RDFS
- OWL 2 DL
  - OWL-**RL DL**: ähnlich wie OWL-R Full, aber Untermenge von OWL 2 DL
  - **EL++**: satisfiability, instance checks, subsumption, classification in **polynomieller Zeit**
  - **QL**: korrektes und vollständiges Reasoning in **logspace**; entspricht ungefähr Features von UML und ER Diagrammen

OWL 2 Tool-Unterstützung

- APIs: OWL API, KAON2
- Editors: Protégé, TopBraid
- Reasoning:
  - OWL 2 DL: Pellet, FaCT++
  - OWL 2 EL: CEL
  - OWL 2 QL: QuONto, Owlgres
  - OWL 2 RL: Oracle 11g
- ...

Danke für die Aufmerksamkeit!