



TECHNISCHE  
UNIVERSITÄT  
DRESDEN

Fakultät Informatik, Institut Künstliche Intelligenz, Professur Computational Logic

# Deduction Systems

Sebastian Rudolph  
Computational Logic



DRESDEN  
concept  
Exzellenz aus  
Wissenschaft  
und Kultur

# About this Lecture

- Wednesdays, 16:40 – 18:10, INF E005  
(exceptions will be announced)
- content: algorithmic aspects of practically deployed deduction systems
  - tableau and hypertableau systems for reasoning in description logics
  - reasoning algorithms in answer set programming
- lecture and tutorial sessions (will be announced)
- webpage with material, schedule, and announcements:  
[https://ddl.inf.tu-dresden.de/web/Deduction\\_Systems\\_%28SS2016%29](https://ddl.inf.tu-dresden.de/web/Deduction_Systems_%28SS2016%29)

# About the Lecturer

## Sebastian Rudolph



- from 04/2013 Full Professor for Computational Logic  
Computer Science Department, TU Dresden
- 2006 – 2013 Research Assistant → Project Leader → Privatdozent  
Chair of Knowledge Management, Institute AIFB  
University of Karlsruhe → Karlsruhe Institute of Technology
- 2003 – 2005 Research Assistant  
Chair of Psychology of Teaching and Learning, TU Dresden
- 2000 – 2003 PhD Scholarship Holder Graduate School, TU Dresden
- 1995 – 2000 Studies for high-school teaching (Math, Physics, CS), TU Dresden

scientific interests:

semantic technologies – logic-based knowledge representation – decidability and complexity analysis of logic formalisms – ontological modeling – formal concept analysis – theory of databases – computational linguistics



TECHNISCHE  
UNIVERSITÄT  
DRESDEN

Fakultät Informatik, Institut Künstliche Intelligenz, Professur Computational Logic

# Foundations of Description Logics

Sebastian Rudolph  
Computational Logic  
[sebastian.rudolph@tu-dresden.de](mailto:sebastian.rudolph@tu-dresden.de)



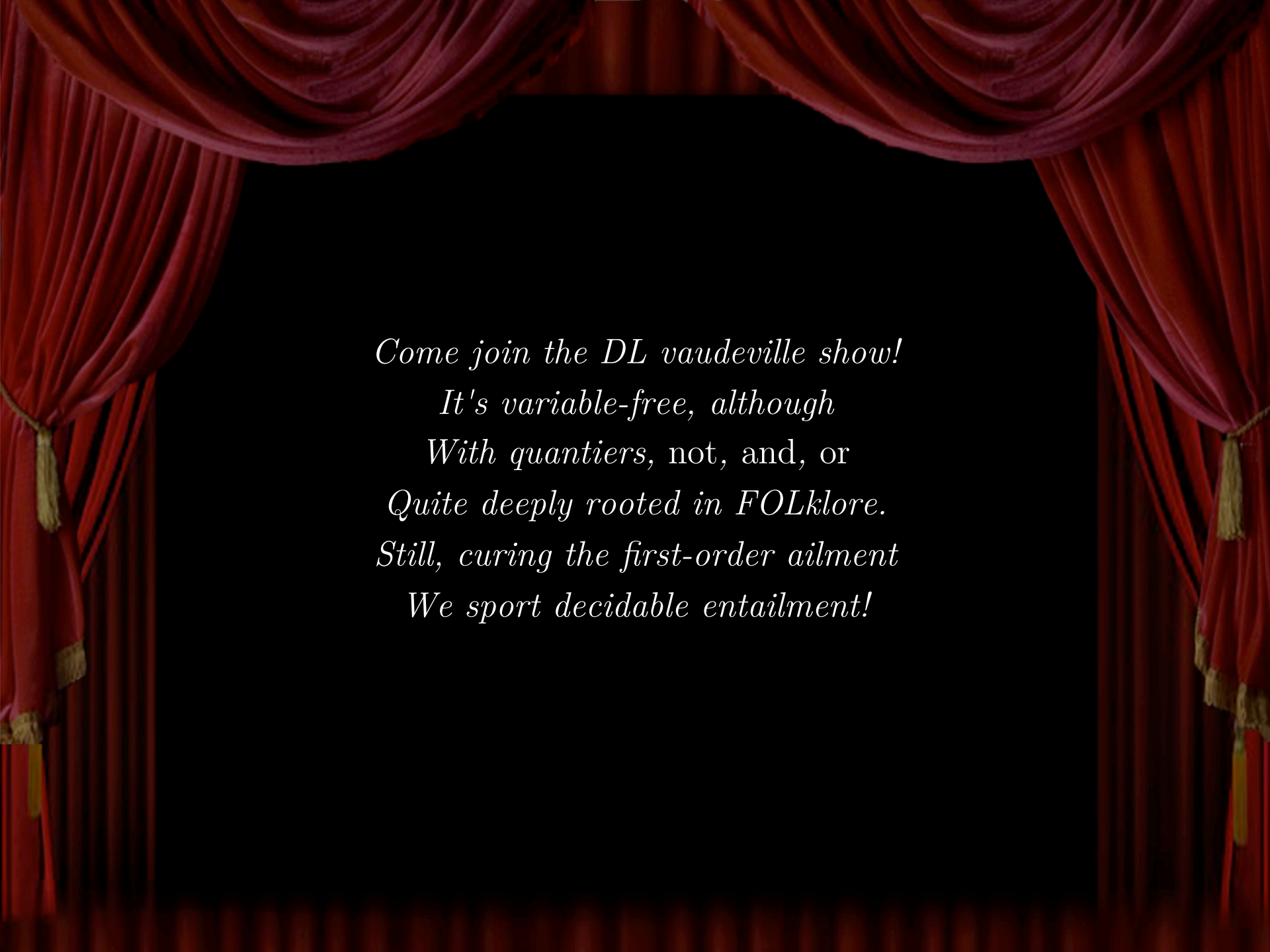
DRESDEN  
concept  
Exzellenz aus  
Wissenschaft  
und Kultur



# Outline

- Introduction: about DLs and the Semantic Web
- Syntax of Description Logics
- Semantics of Description Logics
- Description Logic Nomenclature
- Equivalences, Normalization, Emulation
- Modeling Power of DLs
- DL Reasoning Tasks

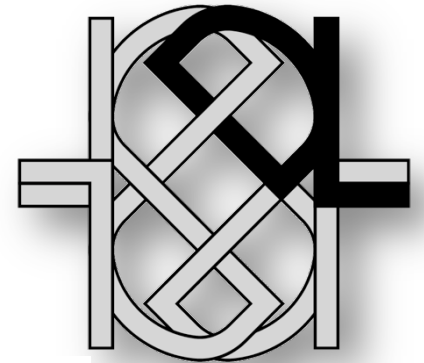


The image features a pair of rich red, velvet-like curtains with gold tassels, framing a central black area. The text is written in a white, elegant, cursive font, centered within the black space.

*Come join the DL vaudeville show!  
It's variable-free, although  
With quantifiers, not, and, or  
Quite deeply rooted in FOLklore.  
Still, curing the first-order ailment  
We sport decidable entailment!*

# Description Logics

- Description Logics (DLs) one of today's main KR paradigms
- influenced standardization of Semantic Web languages, in particular the web ontology language OWL



- comprehensive tool support available

Fact++  
Pellet

Hermit



# 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 behaviour

# Syntax of Description Logics

*Deluxe DL delivery*

*Will come in boxes (number: three),*

*Precisely marked with  $\mathcal{A}$ ,  $\mathcal{T}$ ,  $\mathcal{R}$ .*

*The first exhibits solid grounding,*

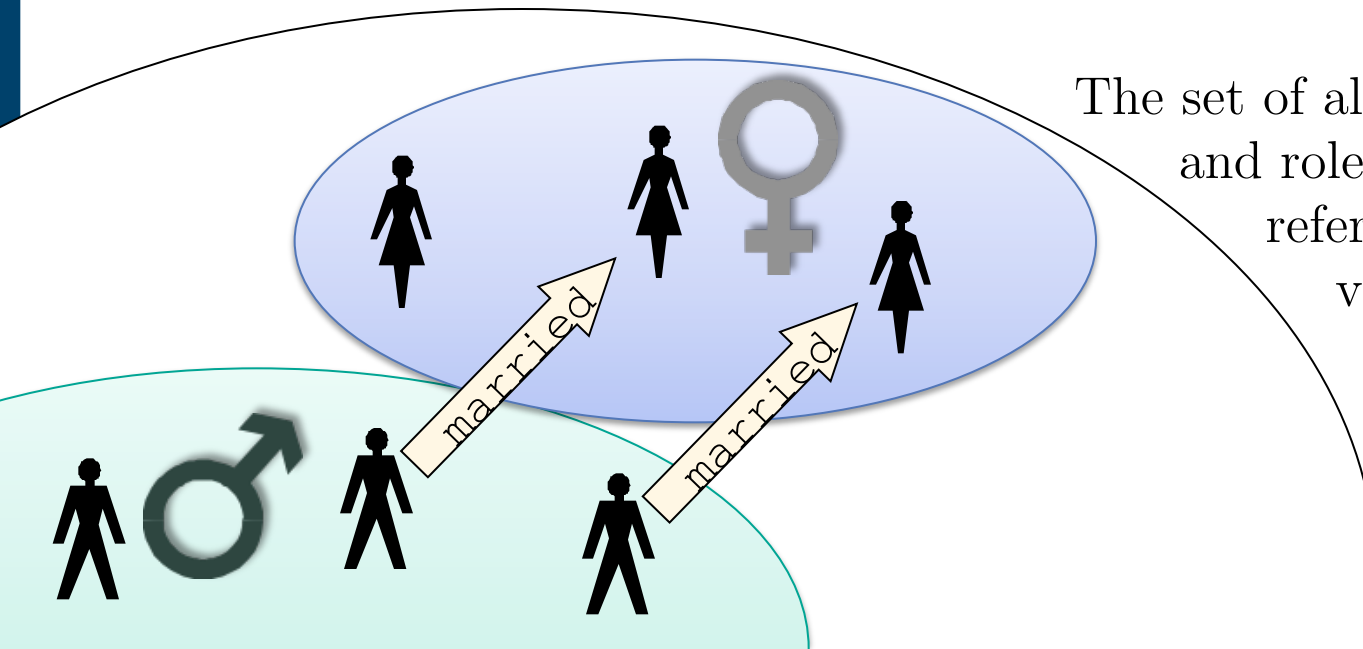
*The next allows for simple counting,*

*The third one's strictly regular.*



# DL Building Blocks

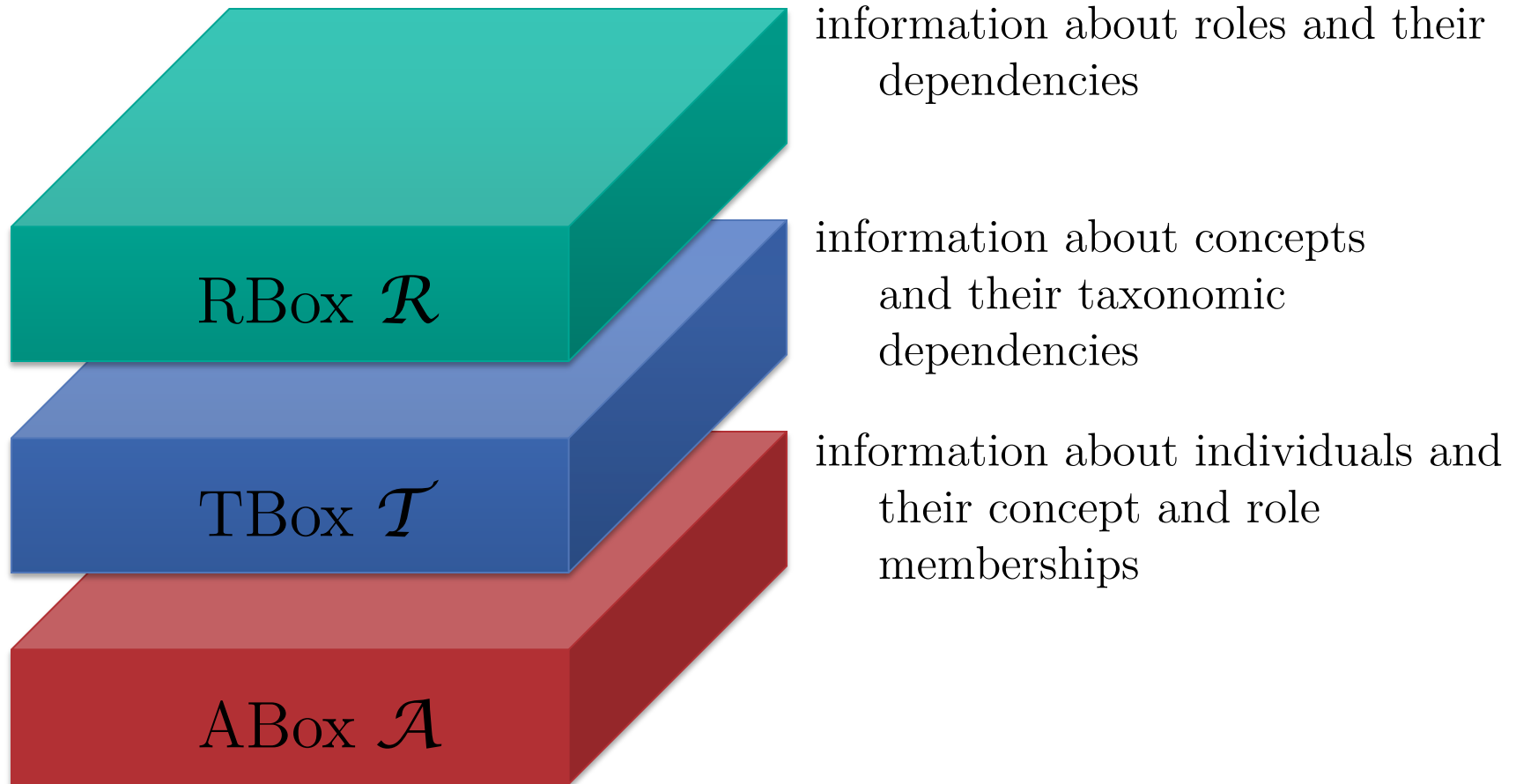
- **individual names:** markus, rhine, sun, excalibur
  - aka: constants (FOL), resources (RDF)
- **concept names:** Female, Mammal, Country
  - aka: unary predicates (FOL), classes (RDFS)
- **role names:** married, fatherOf, locatedIn
  - aka: binary predicates (FOL), properties (RDFS)



The set of all individual, concept and role names is commonly referred to as signature or vocabulary.



# Constituents of a DL Knowledge Base



# Roles and Role Inclusion Axioms

- A *role* can be
  - a role name  $\mathbf{r}$  or
  - an inverted role name  $\mathbf{r}^-$  or
  - the universal role  $\mathbf{u}$ .
- A *role inclusion axiom* (RIA) is a statement of the form

$$r_1 \circ \dots \circ r_n \sqsubseteq r$$

where  $r_1, \dots, r_n, r$  are roles.



# Role Simplicity

- Given a set of RIAs, roles are divided into *simple* and *non-simple* roles.
- Roughly, roles are non-simple if they may occur on the rhs of a complex RIA.
- More precisely,
  - for any RIA  $r_1 \circ r_2 \circ \dots \circ r_n \sqsubseteq r$  with  $n > 1$ ,  $r$  is non-simple,
  - for any RIA  $s \sqsubseteq r$  with  $s$  non-simple,  $r$  is non-simple, and
  - all other properties are simple.

- Example:

$$q \circ p \sqsubseteq r \quad r \circ p \sqsubseteq r \quad r \sqsubseteq s \quad p \sqsubseteq r \quad q \sqsubseteq s$$

non-simple:  $r, s$     simple:  $p, q$

# The Regularity Condition on RIA sets

- For technical reasons, the set of all RIAs of a knowledge base is required to be *regular*.
- regularity restriction:
  - there must be a strict linear order  $<$  on the roles such that
  - every RIA has one of the following forms with  $\mathbf{s}_i < \mathbf{r}$  for all  $i=1,2,\dots,n$ :

$$\mathbf{r} \circ \mathbf{r} \sqsubseteq \mathbf{r}$$

$$\mathbf{r}^- \sqsubseteq \mathbf{r}$$

$$\mathbf{s}_1 \circ \mathbf{s}_2 \circ \dots \circ \mathbf{s}_n \sqsubseteq \mathbf{r}$$

$$\mathbf{r} \circ \mathbf{s}_1 \circ \mathbf{s}_2 \circ \dots \circ \mathbf{s}_n \sqsubseteq \mathbf{r}$$

$$\mathbf{s}_1 \circ \mathbf{s}_2 \circ \dots \circ \mathbf{s}_n \circ \mathbf{r} \sqsubseteq \mathbf{r}$$

- Example 1:  $\mathbf{r} \circ \mathbf{s} \sqsubseteq \mathbf{r}$        $\mathbf{s} \circ \mathbf{s} \sqsubseteq \mathbf{s}$        $\mathbf{r} \circ \mathbf{s} \circ \mathbf{r} \sqsubseteq \mathbf{t}$ 
  - regular with order  $\mathbf{s} < \mathbf{r} < \mathbf{t}$
- Example 2:  $\mathbf{r} \circ \mathbf{t} \circ \mathbf{s} \sqsubseteq \mathbf{t}$ 
  - not regular because form not admissible
- Example 3:  $\mathbf{r} \circ \mathbf{s} \sqsubseteq \mathbf{s}$        $\mathbf{s} \circ \mathbf{r} \sqsubseteq \mathbf{r}$ 
  - not regular because no adequate order exists

# RBox

- A *role disjointness* statement has the form

$$\text{Dis}(\mathbf{s}_1, \mathbf{s}_2)$$

where  $\mathbf{s}_1$  and  $\mathbf{s}_2$  are simple roles.

- An *RBox* consists of regular set of RIAs and a set of role disjointness statements.



# Concept Expressions

- We define *concept expressions* inductively as follows:
  - every concept name is a concept expression,
  - $\top$  and  $\perp$  are concept expressions,
  - for  $\mathbf{a}_1, \dots, \mathbf{a}_n$  individual names,  $\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$  is a concept expression,
  - for  $C$  and  $D$  concept expressions,  $\neg C$  and  $C \sqcap D$  and  $C \sqcup D$  are concept expressions,
  - for  $r$  a role and  $C$  a concept expression,  $\exists r.C$  and  $\forall r.C$  are concept expressions,
  - for  $s$  a simple role,  $C$  a concept expression and  $n$  a natural number,  $\exists s.\text{Self}$  and  $\leq ns.C$  and  $\geq ns.C$  are concept expressions.

# TBox

- A *general concept inclusion* (GCI) has the form

$$C \sqsubseteq D$$

where  $C$  and  $D$  are concept expressions.

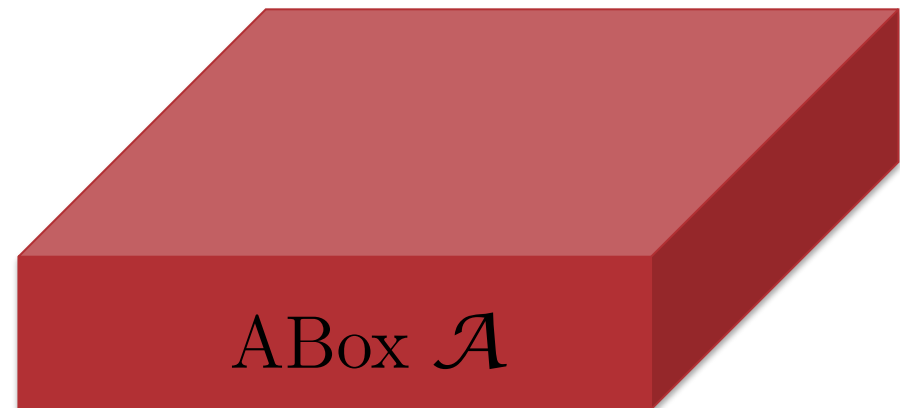
- A *TBox* consists of a set of GCIs.

N.B.: Definition of TBox  
presumes already known  
RBox due to role simplicity  
constraints.



# ABox

- An *individual assertion* can have any of the following forms
  - $C(\mathbf{a})$ , called *concept assertion*,
  - $r(\mathbf{a}, \mathbf{b})$ , called *role assertion*,
  - $\neg r(\mathbf{a}, \mathbf{b})$ , called *negated role assertion*,
  - $\mathbf{a} \approx \mathbf{b}$ , called *equality statement*, or
  - $\mathbf{a} \not\approx \mathbf{b}$ , called *inequality statement*.
  
- An *ABox* consists of a set of individual assertions.



# An Example Knowledge Base

RBox  $\mathcal{R}$

$\text{owns} \sqsubseteq \text{caresFor}$

“If somebody owns something, they care for it.”

TBox  $\mathcal{T}$

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

“Healthy beings are not dead.”

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

“Every cat is dead or alive.”

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

“A happy cat owner owns a cat and all beings he cares for are healthy.”

ABox  $\mathcal{A}$

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

“Schrödinger is a happy cat owner.”

# Semantics of Description Logics

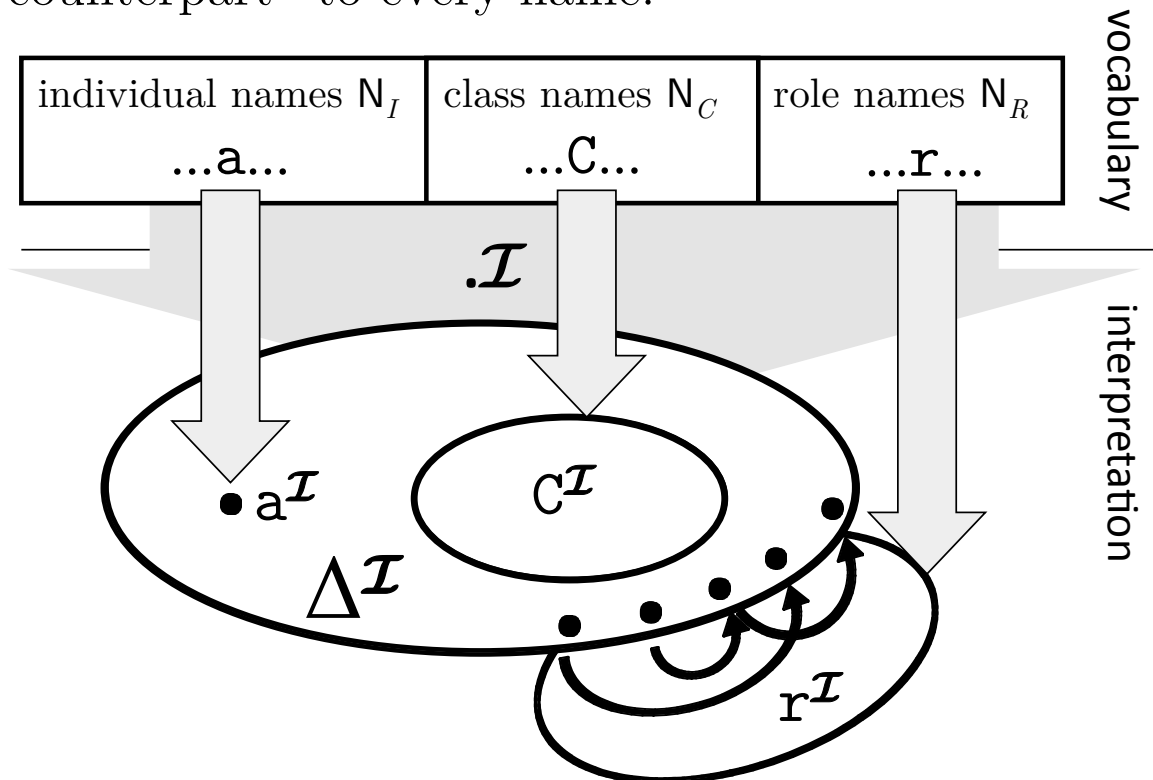
*Semantics has wide applications  
To relationship-based altercations,  
For semantics unveils  
What a statement entails  
Depending on interpretations.*





# Interpretations

- Semantics for DLs is defined in a **model theoretic** way, i.e. based on „abstract possible worlds“, called **interpretations**.
- A DL interpretation  $\mathcal{I}$  fixes a domain set  $\Delta^{\mathcal{I}}$  and a mapping  $\cdot^{\mathcal{I}}$  associating a „semantic counterpart“ to every name.



N.B.: Different names can be mapped to the same semantic counterpart: no unique name assumption.

N.B.:  $\Delta^{\mathcal{I}}$  can be infinite.

# Interpretations: an Example

$$N_I = \{\text{sun, morning\_star, evening\_star, moon, home}\}.$$

$$N_C = \{\text{Planet, Star}\}.$$

$$N_R = \{\text{orbitsAround, shinesOn}\}.$$

$$\Delta^I = \{\odot, \text{♁, ♀, ♂, ☾, ☽, ♃, ♄, ♅, ♆, ♇, ♁\}$$

$$\text{sun}^I = \odot$$

$$\text{morning\_star}^I = \text{♀}$$

$$\text{evening\_star}^I = \text{♀}$$

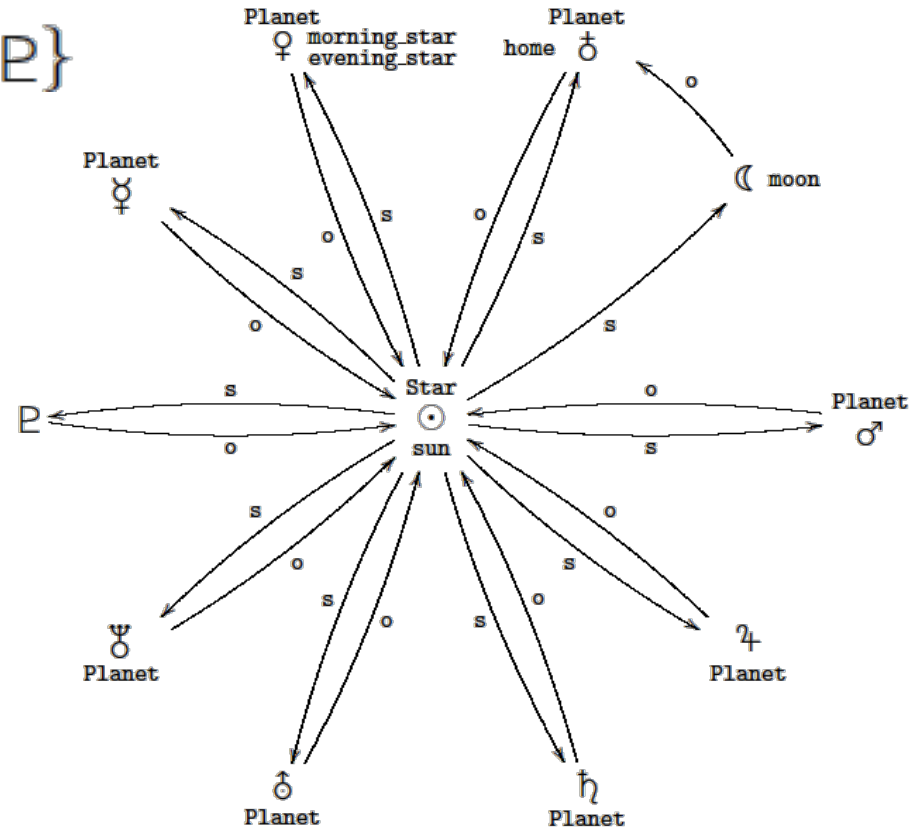
$$\text{moon}^I = \text{☾}$$

$$\text{home}^I = \text{♁}$$

$$\text{Planet}^I = \{\text{♁, ♀, ♂, ☽, ♃, ♄, ♅, ♆, ♇}\}$$

$$\text{Star}^I = \{\odot\}$$

$$\text{orbitsAround}^I = \{\langle \text{♁, } \odot \rangle, \langle \text{♀, } \odot \rangle, \langle \text{♂, } \odot \rangle, \langle \text{☽, } \odot \rangle, \langle \text{☾, } \odot \rangle, \langle \text{♃, } \odot \rangle, \langle \text{♄, } \odot \rangle, \langle \text{♅, } \odot \rangle, \langle \text{♆, } \odot \rangle, \langle \text{♇, } \odot \rangle, \langle \text{♁, } \text{♁} \rangle\}$$

$$\text{shinesOn}^I = \{\langle \odot, \text{♀} \rangle, \langle \odot, \text{♀} \rangle, \langle \odot, \text{♁} \rangle, \langle \odot, \text{☾} \rangle, \langle \odot, \text{☽} \rangle, \langle \odot, \text{♃} \rangle, \langle \odot, \text{♄} \rangle, \langle \odot, \text{♅} \rangle, \langle \odot, \text{♆} \rangle, \langle \odot, \text{♇} \rangle, \langle \odot, \text{♁} \rangle\}$$


# Interpretation of Concept Expressions

- Given an interpretation, we can determine the semantic counterparts for concept expressions along the following inductive definitions:
- mapping is extended to complex class expressions:

$$\top^I = \Delta^I$$

$$\perp^I = \{ \}$$

$$\{ \mathbf{a}_1, \dots, \mathbf{a}_n \}^I = \{ \mathbf{a}_1^I, \dots, \mathbf{a}_n^I \}$$

$$(\neg C)^I = \Delta^I \setminus C^I$$

$$(C \sqcap D)^I = C^I \cap D^I$$

$$(C \sqcup D)^I = C^I \cup D^I$$

$$\exists r.C = \{ x \mid \exists y. (x,y) \in r^I \wedge y \in C^I \}$$

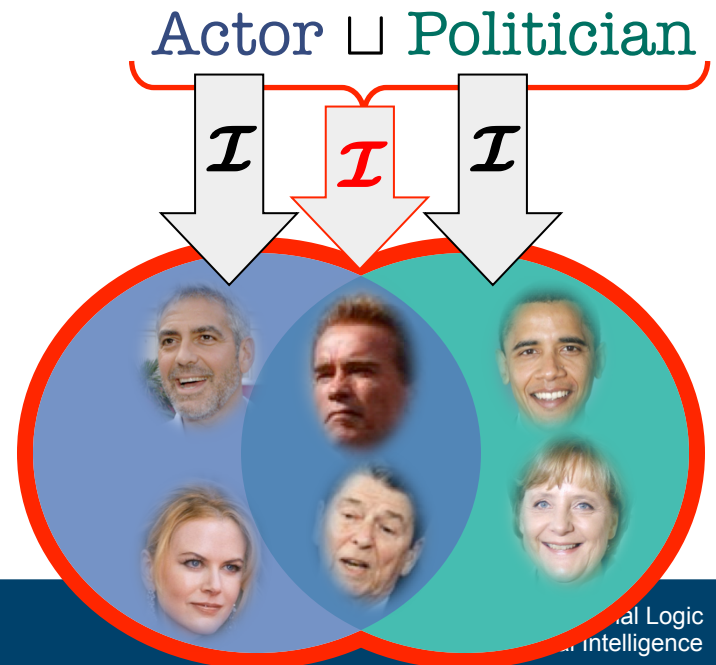
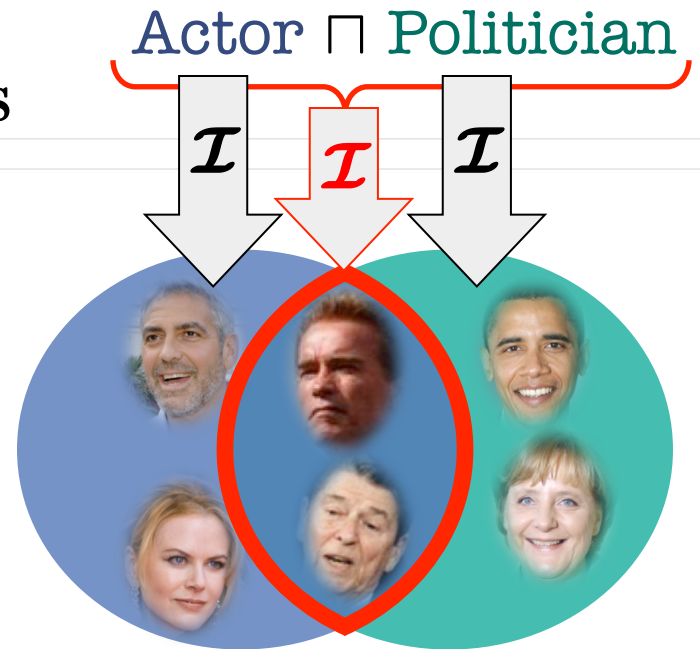
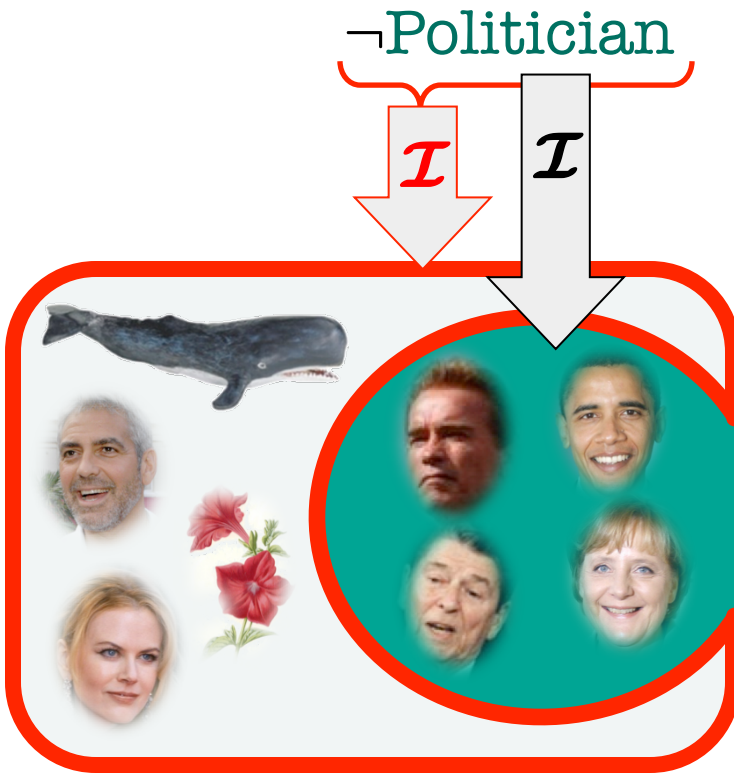
$$\forall r.C = \{ x \mid \forall y. (x,y) \in r^I \rightarrow y \in C^I \}$$

$$\exists s.\mathbf{Self} = \{ x \mid (x,x) \in s^I \}$$

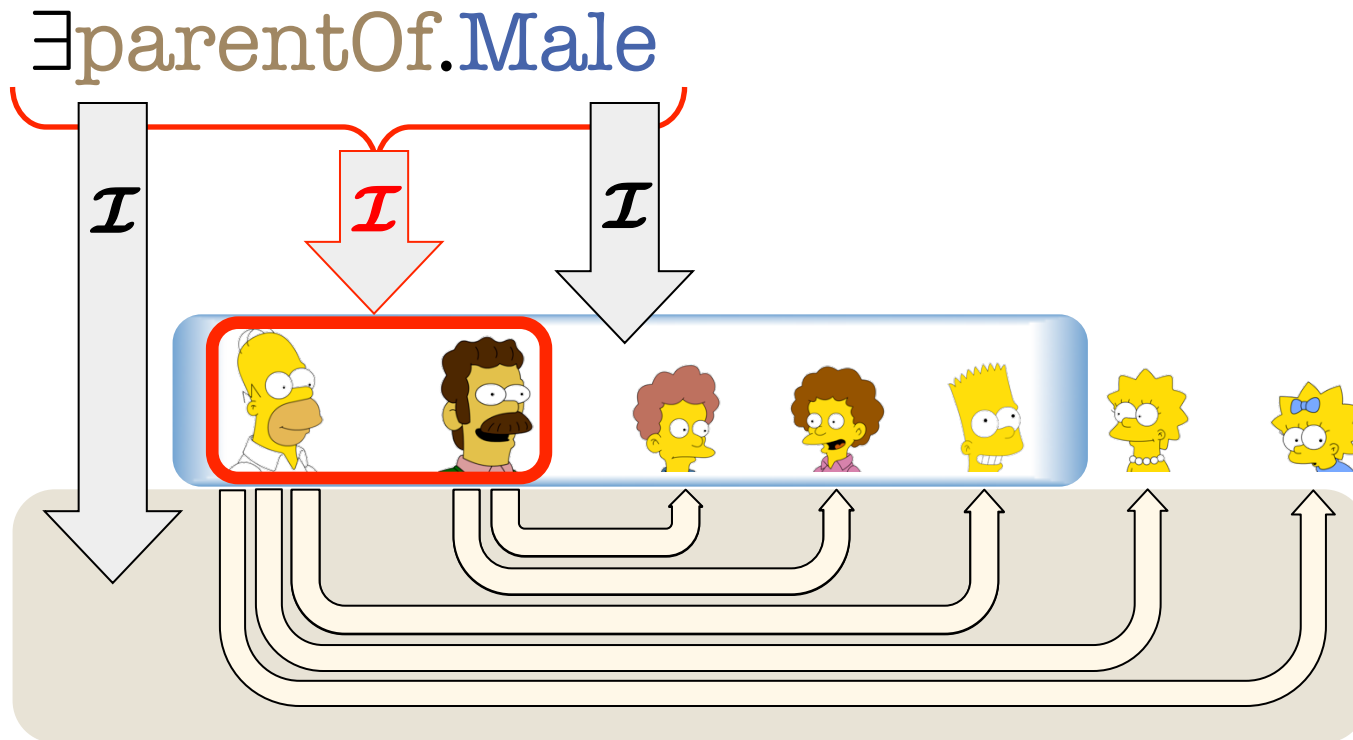
$$\geq ns.C = \{ x \mid \#\{ y \mid (x,y) \in s^I \wedge y \in C^I \} \geq n \}$$

$$\leq ns.C = \{ x \mid \#\{ y \mid (x,y) \in s^I \wedge y \in C^I \} \leq n \}$$

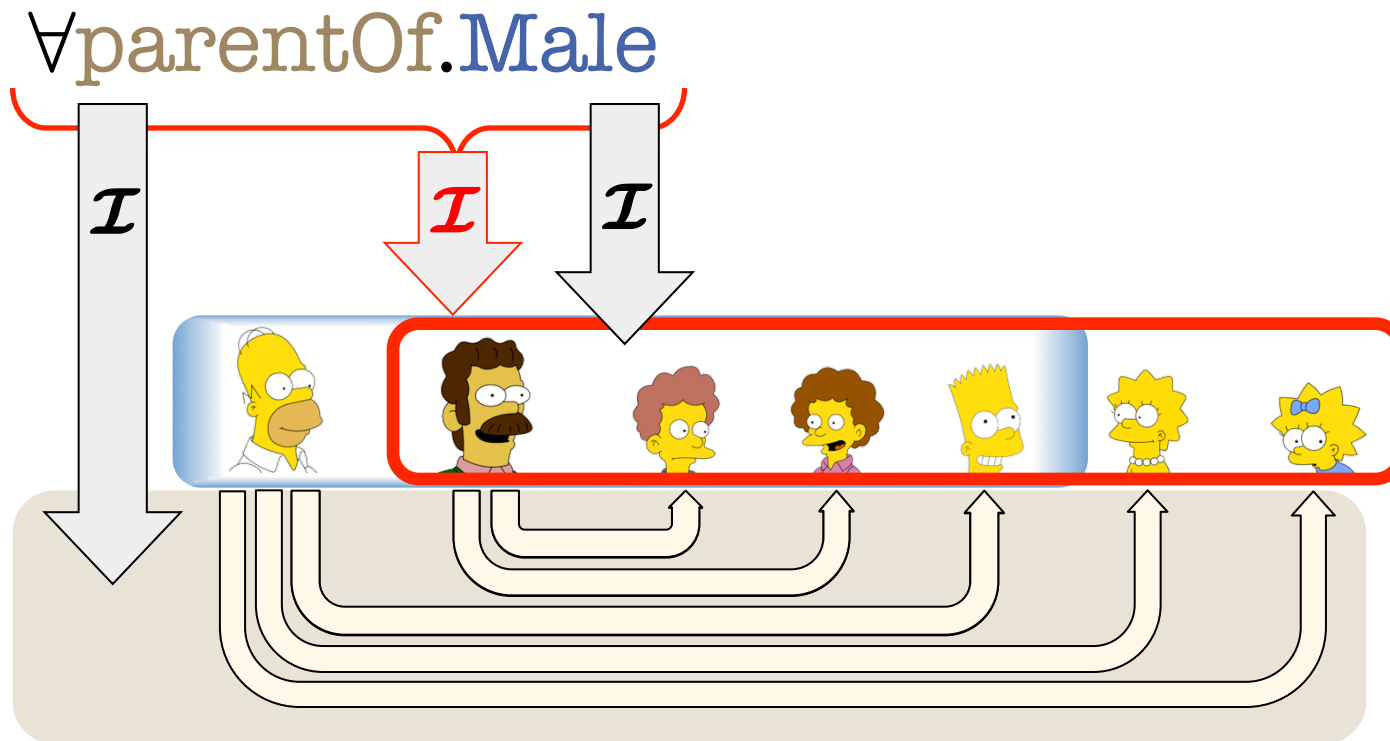
# Boolean Concept Expressions



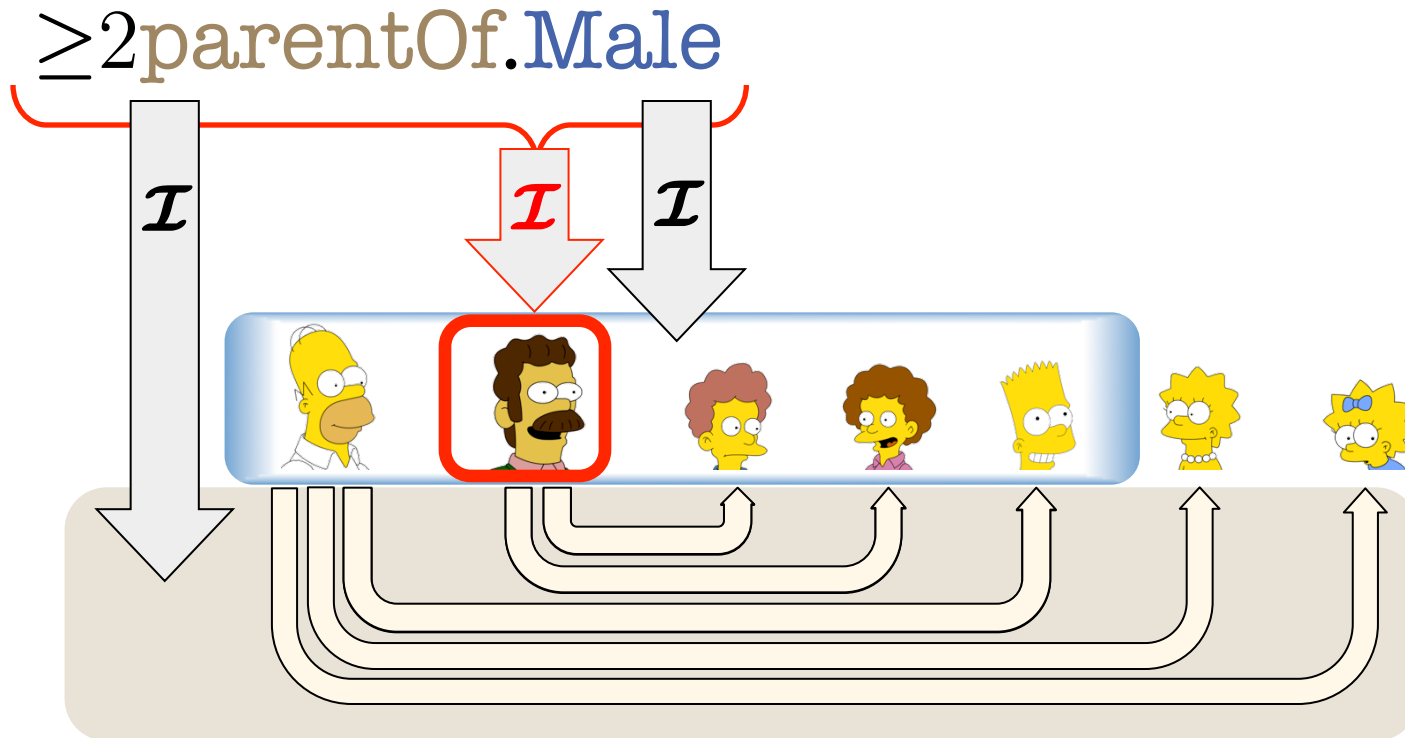
# Existential Role Restrictions



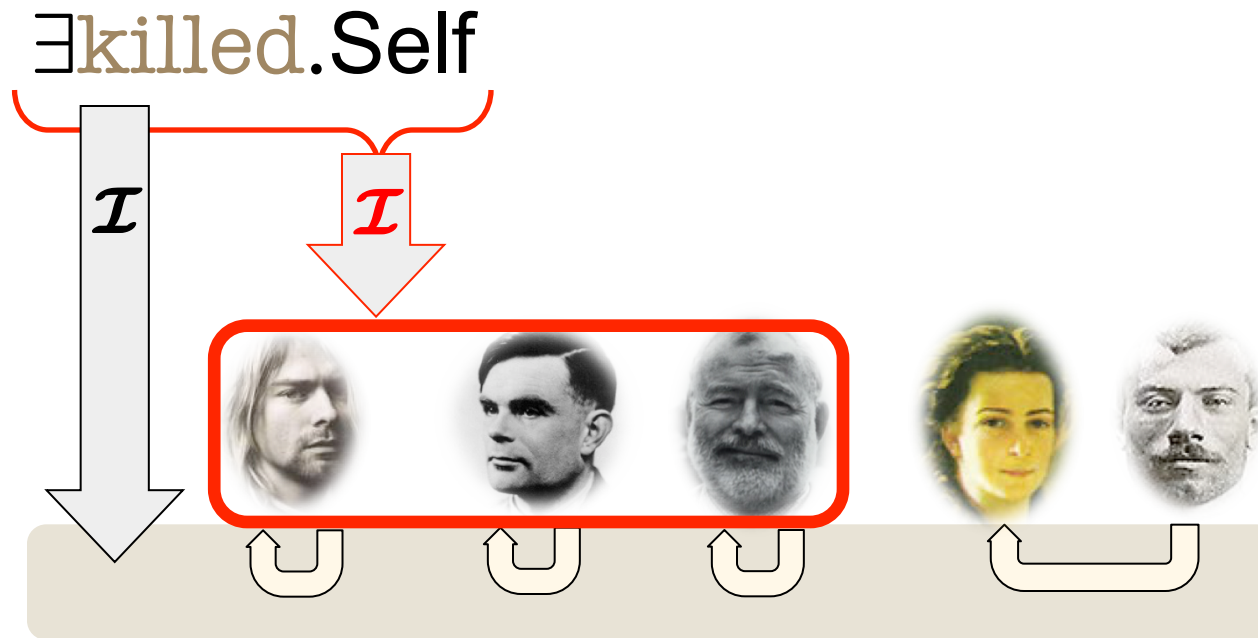
# Universal Role Restrictions



# Qualified Number Restrictions



# Self-Restrictions





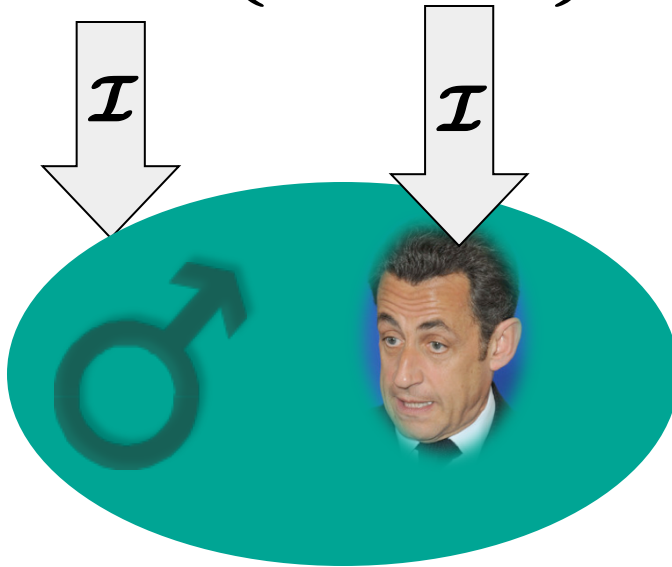
# Semantics of Axioms

Given a way to determine a semantic counterpart for all expressions, we now define the criteria for checking if an interpretation  $\mathcal{I}$  satisfies an axiom alpha  $\alpha$  (written:  $\mathcal{I} \models \alpha$ ).

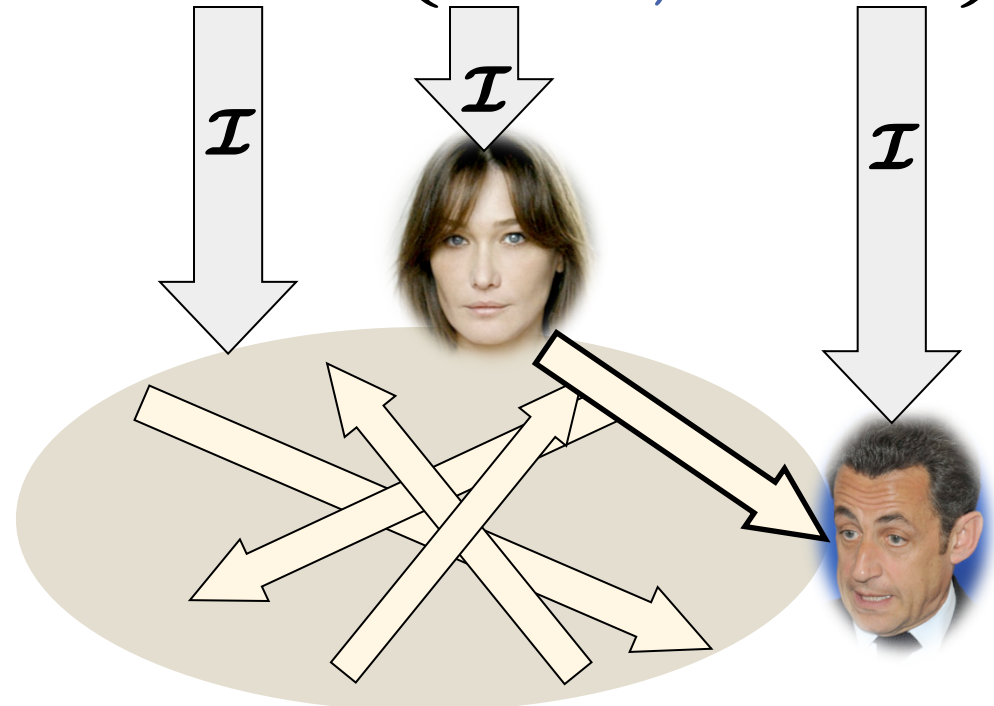
- $\mathcal{I} \models r_1 \circ \dots \circ r_n \sqsubseteq r$       if  $r_1^{\mathcal{I}} \circ \dots \circ r_n^{\mathcal{I}} \subseteq r^{\mathcal{I}}$
- $\mathcal{I} \models \text{Dis}(s_1, s_2)$       if  $s_1^{\mathcal{I}} \cap s_2^{\mathcal{I}} = \{\}$
- $\mathcal{I} \models C \sqsubseteq D$       if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I} \models C(a)$       if  $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I} \models r(a, b)$       if  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$
- $\mathcal{I} \models \neg_r(a, b)$       if  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin r^{\mathcal{I}}$
- $\mathcal{I} \models a \approx b$       if  $a^{\mathcal{I}} = b^{\mathcal{I}}$
- $\mathcal{I} \models a \not\approx b$       if  $a^{\mathcal{I}} \neq b^{\mathcal{I}}$

# Concept and Role Membership

Male(nicolas)

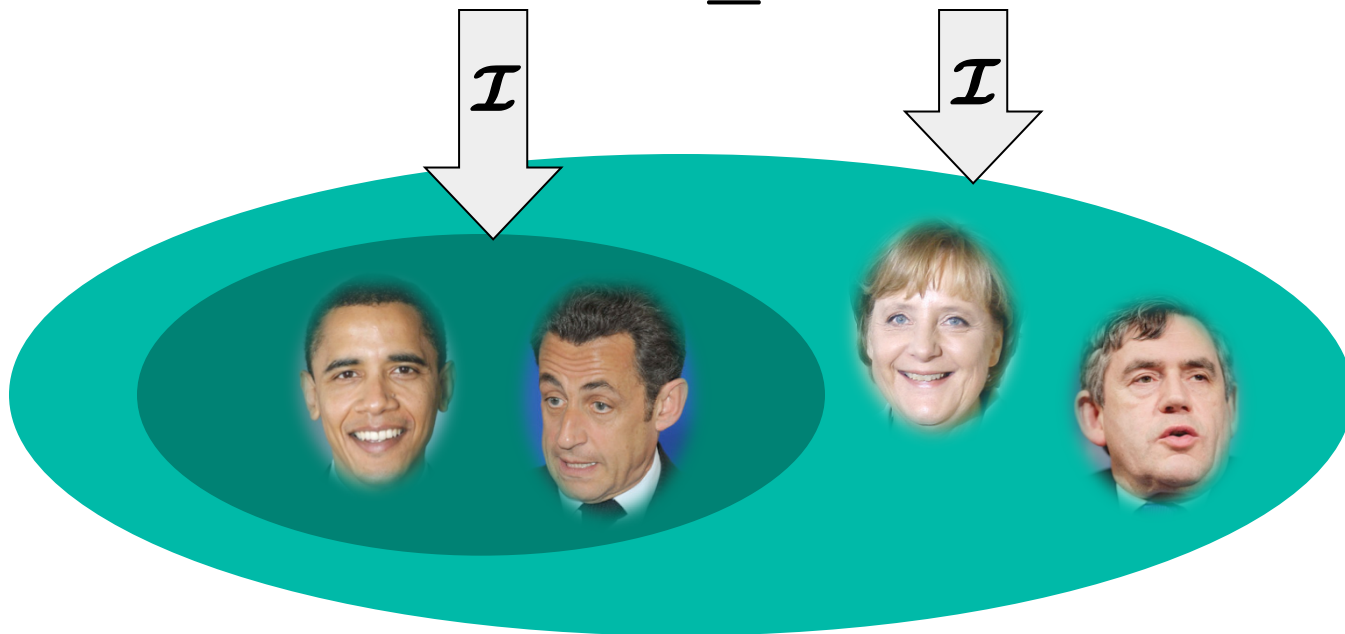


married(carla,nicolas)



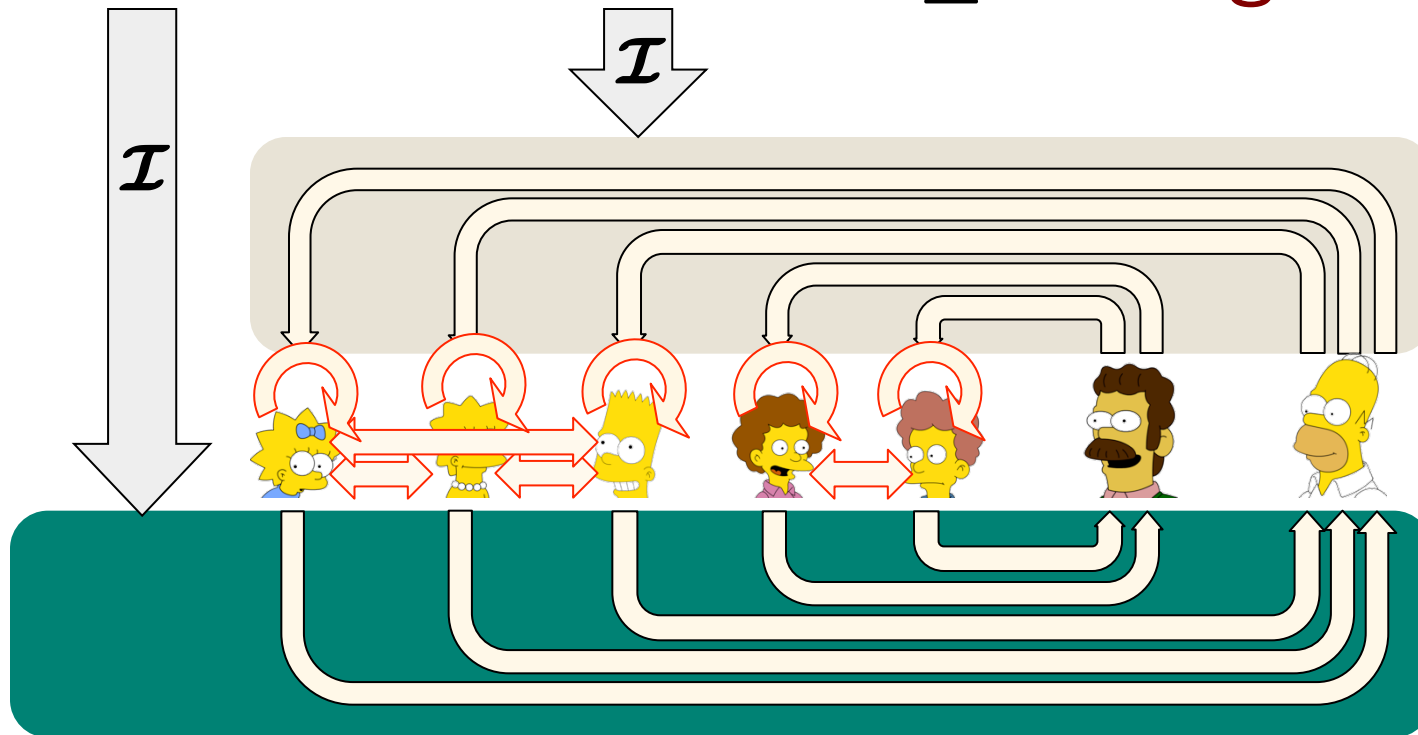
# General Inclusion Axioms

President  $\sqsubseteq$  Politician



# Role Inclusion Axioms

$\text{hasParent} \circ \text{hasChild} \sqsubseteq \text{siblingOf}$



# (Un)Satisfiability of Knowledge Bases

- A KB is *satisfiable* (also: *consistent*) if there exists an interpretation that satisfies all its axioms (a *model* of the KB). Otherwise it is *unsatisfiable* (also: *inconsistent* or *contradictory*).
- Is the following KB satisfiable?

<code>Reindeer <math>\sqcap</math> <math>\exists</math>hasNose.Red(rudolph)</code>	<code>Reindeer <math>\sqsubseteq</math> Mammal</code>
<code><math>\forall</math>worksFor.<math>\neg</math>(<math>\neg</math>Reindeer <math>\sqcup</math> Flies)(santa)</code>	<code>Mammal <math>\sqcap</math> Flies <math>\sqsubseteq</math> Bat</code>
<code>worksFor(rudolph, santa)</code>	<code>Bat <math>\sqsubseteq</math> <math>\forall</math>worksFor.{batman}</code>
<code>santa <math>\neq</math> batman</code>	

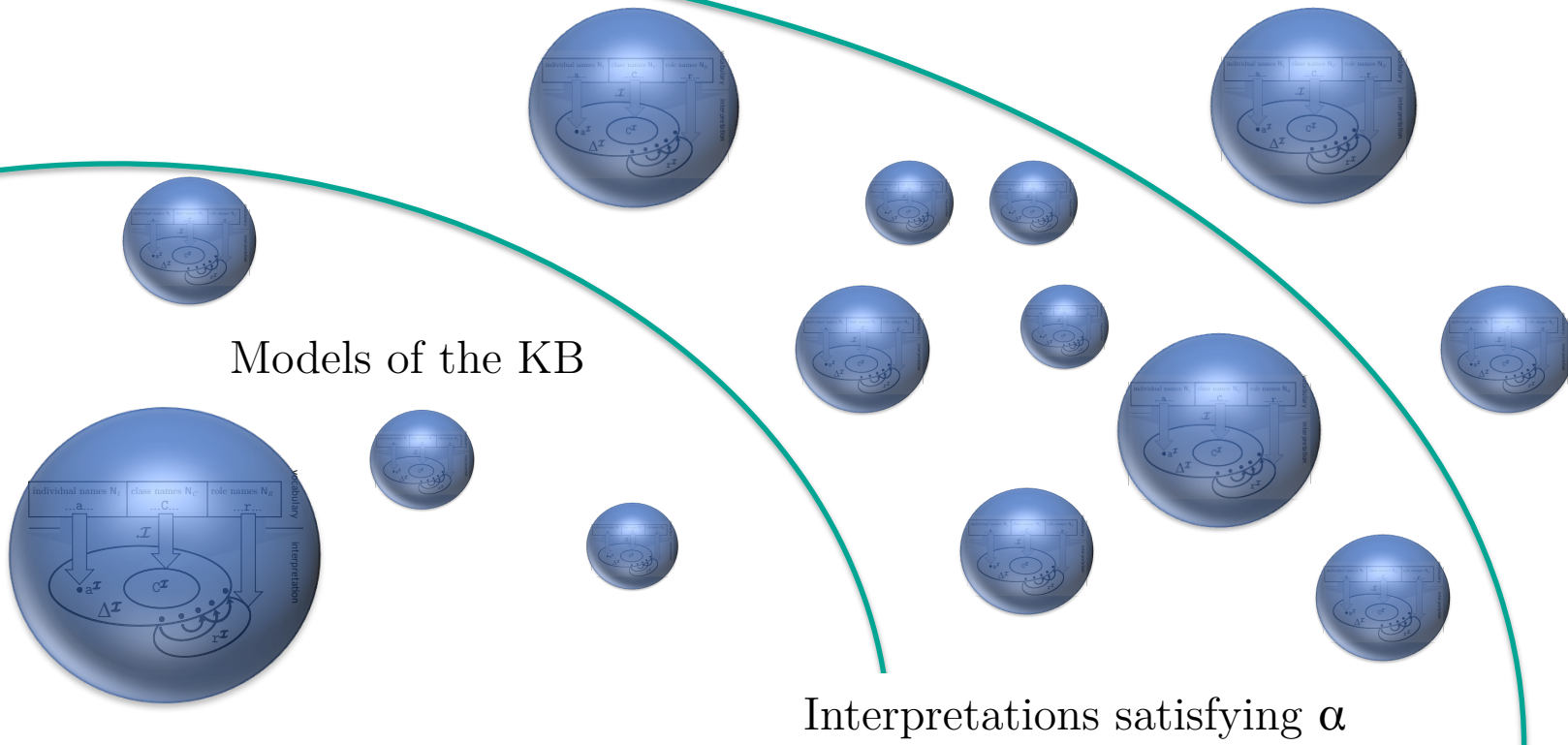


Sebastian Rudolph



# Entailment of Axioms

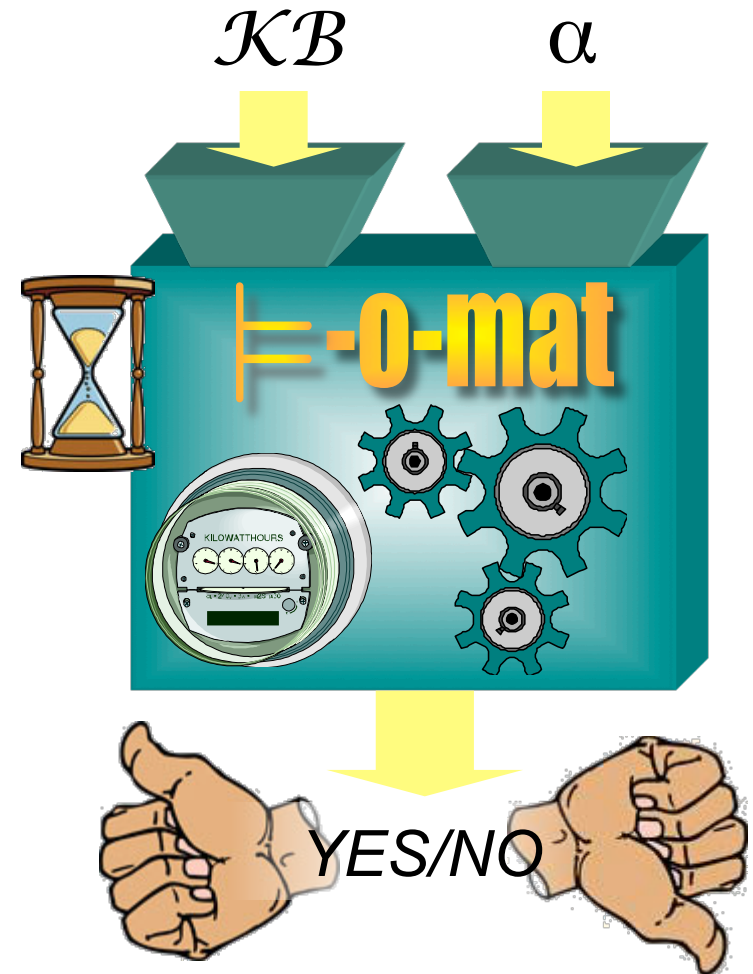
- A KB entails an axiom  $\alpha$  if the axiom  $\alpha$  is satisfied by every model of the knowledge base.



# Decidability of DLs

DLs are *decidable*, i.e. there exists an algorithm that

- takes a knowledge base and an axiom as input,
- terminates after finite time,
- provides as output the correct answer to the question whether the KB entails the axiom.



# Semantics via Translation into FOL

Since DLs can be seen as fragments of FOL, we can alternatively define the semantics by providing a translation of DL axioms into FOL formulae.

$$\begin{aligned}
 \tau(r_1 \circ \dots \circ r_n \sqsubseteq r) &= \forall x_0 \dots x_n (\bigwedge_{1 \leq i \leq n} \tau_{\mathbf{R}}(r_i, x_{i-1}, x_i)) \rightarrow \tau_{\mathbf{R}}(r, x_0, x_n) \\
 \tau(\text{Dis}(r, r')) &= \forall x_0 x_1 (\tau_{\mathbf{R}}(r, x_0, x_1) \rightarrow \neg \tau_{\mathbf{R}}(r', x_0, x_1)) \\
 \tau(C \sqsubseteq D) &= \forall x_0 (\tau_{\mathbf{C}}(C, x_0) \rightarrow \tau_{\mathbf{C}}(D, x_0)) \\
 \tau(C(\mathbf{a})) &= \tau_{\mathbf{C}}(C, x_0)[x_0/\mathbf{a}] \\
 \tau(r(\mathbf{a}, \mathbf{b})) &= \tau_{\mathbf{R}}(r, x_0, x_1)[x_0/\mathbf{a}][x_1/\mathbf{b}] \\
 \tau(\neg r(\mathbf{a}, \mathbf{b})) &= \neg \tau(r(\mathbf{a}, \mathbf{b})) \\
 \tau(a \approx b) &= a = b \\
 \tau(a \not\approx b) &= \neg(a = b)
 \end{aligned}$$



# Semantics via Translation into FOL (ctd.)

Concept/role expressions are translated into formulae with one/two free variable(s).

$$\tau_{\mathbf{C}}(\mathbf{A}, x_i) = \mathbf{A}(x_i)$$

$$\tau_{\mathbf{R}}(u, x_i, x_j) = \mathbf{true}$$

$$\tau_{\mathbf{C}}(\top, x_i) = \mathbf{true}$$

$$\tau_{\mathbf{R}}(\mathbf{r}, x_i, x_j) = \mathbf{r}(x_i, x_j)$$

$$\tau_{\mathbf{C}}(\perp, x_i) = \mathbf{false}$$

$$\tau_{\mathbf{R}}(\mathbf{r}^-, x_i, x_j) = \mathbf{r}(x_j, x_i)$$

$$\tau_{\mathbf{C}}(\{\mathbf{a}_1, \dots, \mathbf{a}_n\}, x_i) = \bigvee_{1 \leq j \leq n} x_i = \mathbf{a}_j$$

$$\tau_{\mathbf{C}}(\neg C, x_i) = \neg \tau_{\mathbf{C}}(C, x_i)$$

$$\tau_{\mathbf{C}}(C \sqcap D, x_i) = \tau_{\mathbf{C}}(C, x_i) \wedge \tau_{\mathbf{C}}(D, x_i)$$

$$\tau_{\mathbf{C}}(C \sqcup D, x_i) = \tau_{\mathbf{C}}(C, x_i) \vee \tau_{\mathbf{C}}(D, x_i)$$

$$\tau_{\mathbf{C}}(\exists r.C, x_i) = \exists x_{i+1}. (\tau_{\mathbf{R}}(r, x_i, x_{i+1}) \wedge \tau_{\mathbf{C}}(C, x_{i+1}))$$

$$\tau_{\mathbf{C}}(\forall r.C, x_i) = \forall x_{i+1}. (\tau_{\mathbf{R}}(r, x_i, x_{i+1}) \rightarrow \tau_{\mathbf{C}}(C, x_{i+1}))$$

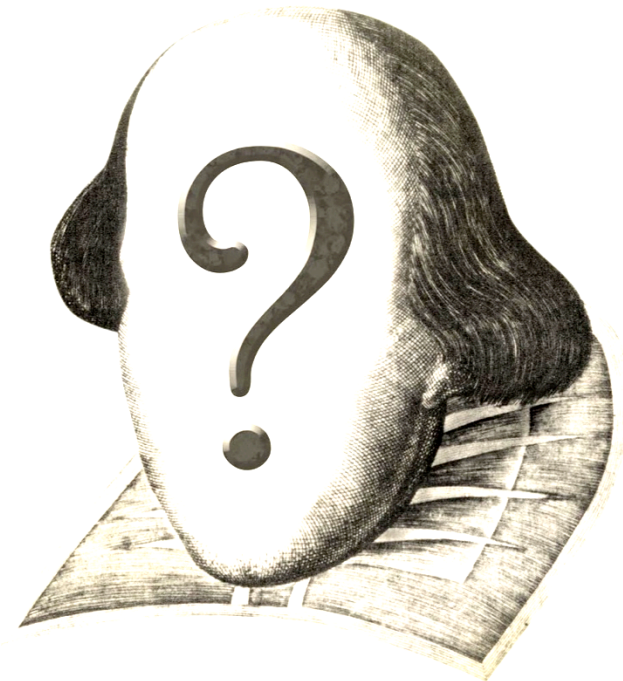
$$\tau_{\mathbf{C}}(\exists r.\mathbf{Self}, x_i) = \tau_{\mathbf{R}}(r, x_i, x_i)$$

$$\tau_{\mathbf{C}}(\geq nr.C, x_i) = \exists x_{i+1} \dots x_{i+n}. \left( \bigwedge_{i+1 \leq j < k \leq i+n} (x_j \neq x_k) \right. \\ \left. \wedge \bigwedge_{i+1 \leq j \leq i+n} (\tau_{\mathbf{R}}(r, x_i, x_j) \wedge \tau_{\mathbf{C}}(C, x_j)) \right)$$

$$\tau_{\mathbf{C}}(\leq nr.C, x_i) = \neg \tau_{\mathbf{C}}(\geq (n+1)r.C, x_i)$$

# Description Logics Nomenclature

*What's in a name? That which we call, say, SHIQ,  
By any other name would do the trick.  
While DL names might leave the novice SHOQed,  
Some principles of ALCHemy unlocked  
Enable understanding in a minute:  
Though it be madness, yet there's method in it.*



# Naming Scheme for Expressive DLs

$$((ALC | S)[H] | SR)[O][I][F | N | Q]$$

- $S$  subsumes  $ALC$
- $SR$  subsumes  $S$ ,  $SH$ ,  $ALC$  and  $ALCH$
- $N$  makes  $F$  obsolete
- $Q$  makes  $N$  (and  $F$ ) obsolete

We treat here the very expressive description logic  $SRQIQ$  which subsumes all the other ones in this scheme.

# DL Syntax – Overview

Concepts		
ALC	Atomic	$A, B$
	Not	$\neg C$
	And	$C \sqcap D$
	Or	$C \sqcup D$
	Exists	$\exists r.C$
	For all	$\forall r.C$
Q(N)	At least	$\geq n r.C$ ( $\geq n r$ )
	At most	$\leq n r.C$ ( $\leq n r$ )
O	Closed class	$\{i_1, \dots, i_n\}$
R	Self	$\exists r.\text{Self}$

Roles		
I	Atomic	$r$
	Inverse	$r^-$

## Ontology (=Knowledge Base)

### Concept Axioms (TBox)

Subclass	$C \sqsubseteq D$
Equivalent	$C \equiv D$

### Role Axioms (RBox)

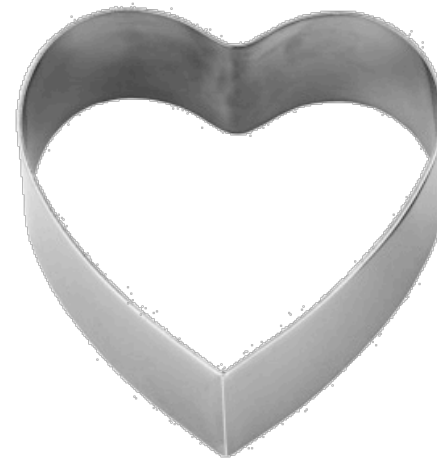
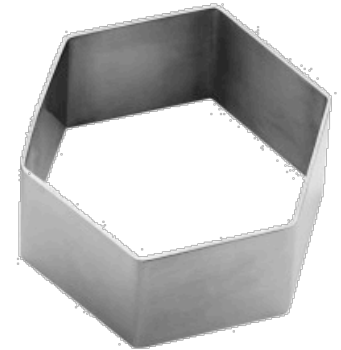
SH	Subrole	$r \sqsubseteq s$
	Transitivity	$\text{Trans}(r)$
SR	Role Chain	$r \circ r' \sqsubseteq s$
	R. Disjointness	$\text{Disj}(s, r)$

### Assertional Axioms (ABox)

Instance	$C(a)$
Role	$r(a, b)$
Same	$a \approx b$
Different	$a \not\approx b$

# Equivalences, Emulation, Normalization

*Don't give told consequences lip,  
Nor 'bout equivalences quip,  
'Cause often it's the formal norm  
That statements be in normal form.*



# Concept Equivalences

Two concept expressions  $C$  and  $D$  are called *equivalent* (written:  $C \equiv D$ ), if for **every** interpretation  $\mathcal{I}$  holds  $C^{\mathcal{I}} = D^{\mathcal{I}}$ .

$$\begin{array}{ll}
 C \sqcap D \equiv D \sqcap C & C \sqcup D \equiv D \sqcup C \\
 (C \sqcap D) \sqcap E \equiv C \sqcap (D \sqcap E) & (C \sqcup D) \sqcup E \equiv C \sqcup (D \sqcup E) \\
 C \sqcap C \equiv C & C \sqcup C \equiv C
 \end{array}$$

$$\begin{array}{ll}
 (C \sqcup D) \sqcap E \equiv (C \sqcap E) \sqcup (D \sqcap E) & (C \sqcup D) \sqcap C \equiv C \\
 (C \sqcap D) \sqcup E \equiv (C \sqcup E) \sqcap (D \sqcup E) & (C \sqcap D) \sqcup C \equiv C
 \end{array}$$

$$\begin{array}{lll}
 \neg\neg C \equiv C & \neg\exists r.C \equiv \forall r.\neg C & \geq 0r.C \equiv \top \\
 \neg(C \sqcap D) \equiv \neg D \sqcup \neg C & \neg\forall r.C \equiv \exists r.\neg C & \geq 1r.C \equiv \exists r.C \\
 \neg(C \sqcup D) \equiv \neg D \sqcap \neg C & \neg\leq nr.C \equiv \geq (n+1)r.C & \leq 0r.C \equiv \forall r.\neg C \\
 & \neg\geq (n+1)r.C \equiv \leq nr.C &
 \end{array}$$

# Negation Normal Form

- Iterated rewriting of concept expressions along the mentioned equivalences allows to convert every concept expression into one with negation only in front of concept names, nominal concepts and Self-restrictions.

$$\text{nnf}(C) := C \text{ if } C \in \{A, \neg A, \{a_1, \dots, a_n\}, \neg\{a_1, \dots, a_n\}, \exists r.\text{Self}, \neg\exists r.\text{Self}, \top, \perp\}$$

$$\text{nnf}(\neg\neg C) := \text{nnf}(C)$$

$$\text{nnf}(\neg\top) := \perp$$

$$\text{nnf}(\neg\perp) := \top$$

$$\text{nnf}(C \sqcap D) := \text{nnf}(C) \sqcap \text{nnf}(D) \quad \text{nnf}(\neg(C \sqcap D)) := \text{nnf}(\neg C) \sqcup \text{nnf}(\neg D)$$

$$\text{nnf}(C \sqcup D) := \text{nnf}(C) \sqcup \text{nnf}(D) \quad \text{nnf}(\neg(C \sqcup D)) := \text{nnf}(\neg C) \sqcap \text{nnf}(\neg D)$$

$$\text{nnf}(\forall r.C) := \forall r.\text{nnf}(C)$$

$$\text{nnf}(\neg\forall r.C) := \exists r.\text{nnf}(\neg C)$$

$$\text{nnf}(\exists r.C) := \exists r.\text{nnf}(C)$$

$$\text{nnf}(\neg\exists r.C) := \forall r.\text{nnf}(\neg C)$$

$$\text{nnf}(\leq n r.C) := \leq n r.\text{nnf}(C)$$

$$\text{nnf}(\neg\leq n r.C) := \geq (n + 1) r.\text{nnf}(C)$$

$$\text{nnf}(\geq n r.C) := \geq n r.\text{nnf}(C)$$

$$\text{nnf}(\neg\geq n r.C) := \leq (n - 1) r.\text{nnf}(C)$$

# Axiom and KB Equivalences

- Lloyd-Topor equivalences

$$\{A \sqcup B \sqsubseteq C\} \iff \{A \sqsubseteq C, B \sqsubseteq C\}$$

$$\{A \sqsubseteq B \sqcap C\} \iff \{A \sqsubseteq B, A \sqsubseteq C\}$$

- turning GCIs into universally valid concept descriptions

$$C \sqsubseteq D \iff \top \sqsubseteq \neg C \sqcup D$$

- internalisation of ABox into TBox

$$C(a) \iff \{a\} \sqsubseteq C$$

$$r(a, b) \iff \{a\} \sqsubseteq \exists r. \{b\}$$

$$\neg r(a, b) \iff \{a\} \sqsubseteq \neg \exists r. \{b\}$$

$$a \approx b \iff \{a\} \sqsubseteq \{b\}$$

$$a \not\approx b \iff \{a\} \sqsubseteq \neg \{b\}$$



# Emulation

- Sometimes the knowledge base is required to be in some specific form which cannot be obtained by equivalent transformations alone. In that cases, one can try to obtain a KB that is equivalent “up to additional vocabulary“ (called *fresh names*).

Example:

- ABox is *extensionally reduced* if all concept assertions contain concept names only.
- Any KB can be turned into one with extensionally reduced ABox by repeating the following procedure:
  - Pick a concept assertion  $C(\mathbf{a})$  where  $C$  is not a concept name
  - remove  $C(\mathbf{a})$  from the Abox and add  $\mathbf{A}(\mathbf{a})$  instead, where  $\mathbf{A}$  is not used elsewhere in the KB
  - add  $\mathbf{A} \sqsubseteq C$  to the TBox

# Emulation

A knowledge base  $\mathcal{KB}'$  *emulates* a knowledge base  $\mathcal{KB}$  if two conditions are satisfied:

- Every model of  $\mathcal{KB}'$  is a model of  $\mathcal{KB}$ , formally: given an interpretation  $\mathcal{I}$ , we have that  $\mathcal{I} \models \mathcal{KB}'$  implies  $\mathcal{I} \models \mathcal{KB}$ .
- For every model  $\mathcal{I}$  of  $\mathcal{KB}$  there is a model  $\mathcal{I}'$  of  $\mathcal{KB}'$  that has the same domain as  $\mathcal{I}$ , and coincides with  $\mathcal{I}$  on the vocabulary used in  $\mathcal{KB}$ .

Using emulation allows to model many things that are not directly expressible in the used DL.

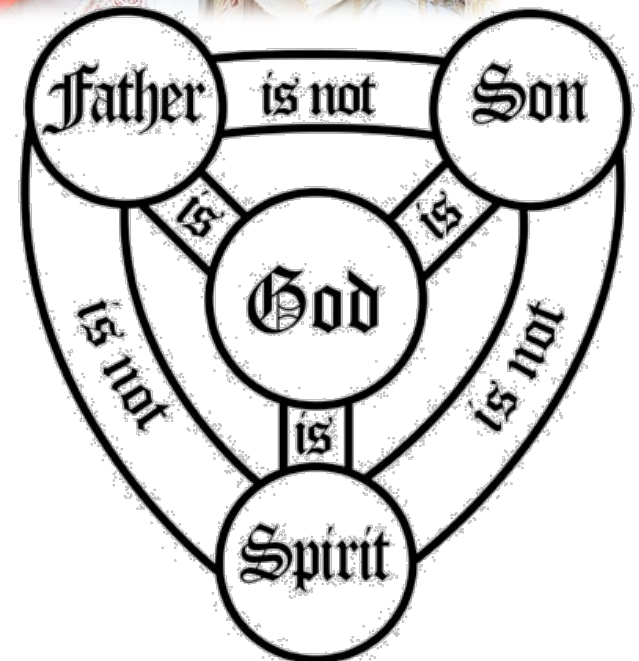
Example: “ $\mathbf{A} \sqsubseteq \mathbf{B}$  holds or  $\mathbf{C} \sqsubseteq \mathbf{D}$  holds” can be emulated by

$$\top \sqsubseteq \exists \mathbf{r}. \{ \mathbf{o} \} \quad \{ \mathbf{o} \} \sqsubseteq \forall \mathbf{r}^-. (\neg \mathbf{A} \sqcup \mathbf{B}) \sqcup \forall \mathbf{r}^-. (\neg \mathbf{C} \sqcup \mathbf{D})$$

where  $\mathbf{o}$  is a fresh individual name and  $\mathbf{r}$  a fresh role name.

# Modeling with DLs

*While frowning on plurality,  
The pope likes cardinality:  
It can enforce infinity,  
And hence endorse divinity.  
But, theologically speaking,  
The papal theory needs tweaking  
For it demands divine assistance  
to prove "the three are one"-consistence.*



# Frequent Modeling Features

- domain:  $\exists \text{authorOf}.T \sqsubseteq \text{Person}$
- range:  $T \sqsubseteq \forall \text{authorOf}.\text{Publication}$   
 or  $\exists \text{authorOf}^-.T \sqsubseteq \text{Publication}$
- concept disjointness:  $\text{Male} \sqcap \text{Female} \sqsubseteq \perp$   
 or  $\text{Male} \sqsubseteq \neg \text{Female}$
- role symmetry:  $\text{marriedWith}^- \sqsubseteq \text{marriedWith}$
- role transitivity:  $\text{partOf} \circ \text{partOf} \sqsubseteq \text{partOf}$

# Number Restrictions

- allow for defining that a role is functional:

$$\top \sqsubseteq \leq 1 \text{hasFather} . \top$$

- ...or inverse functional:

$$\top \sqsubseteq \leq 1 \text{fatherOf}^{-} . \top$$

- allow for enforcing an infinite domain:

$$(\forall \text{succ}^{-} . \perp)(\text{zero}) \quad \top \sqsubseteq \exists \text{succ} . \top \quad \top \sqsubseteq \leq 1 . \text{succ}^{-} . \top$$

- Consequently, DLs with number restrictions and inverses do not have the finite model property.

# Nominal Concept and Universal Role

- allow to restrict the size of concepts:

$$\text{AtMostTwo} \sqsubseteq \{\text{one}, \text{two}\} \quad \top \sqsubseteq \leq 2u. \text{AtMostTwo}$$

- even allow to restrict the size of the domain:

$$\top \sqsubseteq \{\text{one}, \text{two}\} \quad \top \sqsubseteq \leq 2u. \top$$

# Self-Restriction

- allows to define a role as reflexive

$$\top \sqsubseteq \exists \text{knows.Self}$$

- allows to define a role as irreflexive

$$\exists \text{betterThan.Self} \sqsubseteq \perp$$

- together with number restrictions, we can even axiomatize equality:

$$\top \sqsubseteq \exists \text{equals.Self}$$

$$\top \sqsubseteq \leq 1 \text{equals.}\top$$

# Open vs. Closed World Assumption

- CWA: Closed World Assumption  
The knowledge base contains all information, non-derivable axioms are assumed to be false.
- OWA: Open World Assumption  
The knowledge base may be incomplete. The truth of non-derivable axioms is simply unknown.
- With DLs, the OWA is applied (as for FOL in general), certain closed-world information can be axiomatized via number restrictions and nominals

*Are all children of Bill male?*

*No idea, since we do not know all children of Bill.*

*If we assume that we know everything about Bill, then all of his children are male.*

child(bill,bob)

Man(bob)

$\leq 1\text{child}.\top(\text{Bill})$

?  $\models \forall \text{child}.\text{Man}(\text{Bill})$

?  $\models \forall \text{child}.\text{Man}(\text{Bill})$

**DL answers**

**don't know**

**yes**

**Prolog**

**yes**

*Now we know everything about Bill's children.*



# Reasoning Tasks and Their Reducibility

*A knowledge base with statements in it*

*Seeks a model sound and nice*

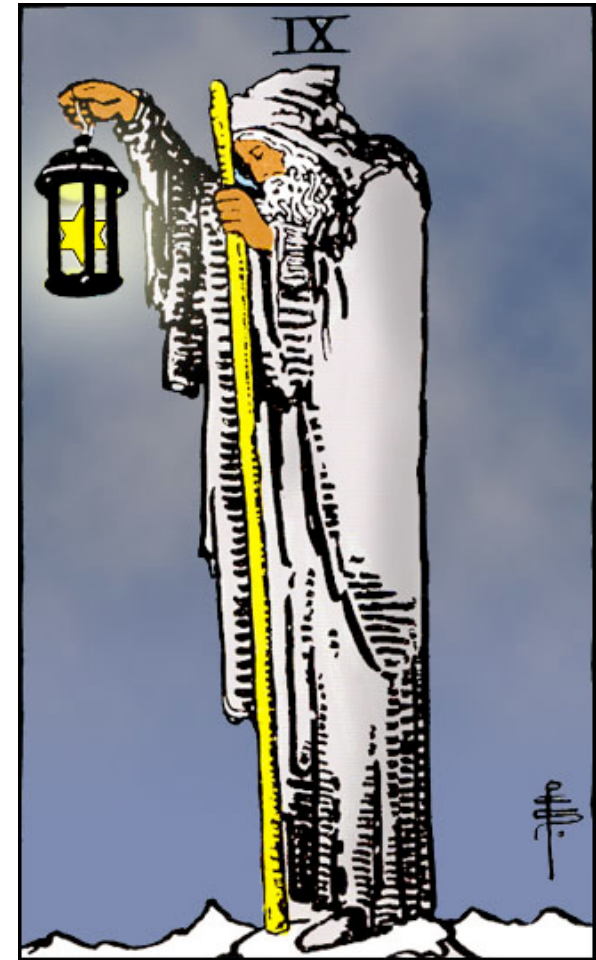
*No matter, finite or infinite,*

*It asks a hermit for advice.*

*Yet, shattering is the reaction:*

*“Inconsistency detection,*

*You can't get no satisfaction.”*



# Standard DL Inference Problems

Given a knowledge base KB, we might want to know:

- whether the KB is consistent,
- whether the KB entails a certain axiom  
( such as **Alive(schrödinger)** ),
- whether a given concept is (un)satisfiable  
( such as **Dead  $\sqcap$  Alive** ),
- all the individuals known to be instances a certain concept
- the subsumption hierarchy of all atomic concepts

# Knowledge Base Consistency

- basic inferencing task
- directly needed in the process of KB engineering in order to detect severe modelling errors
- other tasks can be reduced to checking KB (in)consistency

# Entailment Checking

- used in the KB modelling process to check, whether the specified knowledge has the intended consequences
- used for querying the KB if certain propositions are necessarily true
- can be reduced to checking KB inconsistency (along the idea of indirect proof) by
  - negating the axiom the entailment of which is to be checked
  - adding the negated axiom to the knowledge base
  - checking for inconsistency of the KB
- if axiom cannot be negated directly, its negation can be emulated

# Entailment Checking

$\alpha$	$\mathcal{A}_\alpha$
$r_1 \circ \dots \circ r_n \sqsubseteq r$	$\{\neg r(c_0, c_n), r_1(c_0, c_1), \dots, r_n(c_{n-1}, c_n)\}$
$\text{Dis}(r, r')$	$\{r(c_1, c_2), r'(c_1, c_2)\}$
$C \sqsubseteq D$	$\{(C \sqcap \neg D)(c)\}$ or: $\{\top \sqsubseteq \exists u(C \sqcap \neg D)\}$
$C(a)$	$\{\neg C(a)\}$
$r(a, b)$	$\{\neg r(a, b)\}$
$\neg r(a, b)$	$\{r(a, b)\}$
$a \approx b$	$\{a \not\approx b\}$
$a \not\approx b$	$\{a \approx b\}$

**Table 1.** Definition of axiom sets  $\mathcal{A}_\alpha$  such that  $\mathcal{KB} \models \alpha$  exactly if  $\mathcal{KB} \cup \mathcal{A}_\alpha$  is unsatisfiable. Individual names  $c$  with possible subscripts are supposed to be fresh. For GCIs (third line), the first variant is normally employed, however, we also give a variant which is equivalent instead of just emulating.

# Concept satisfiability

- A concept expression  $C$  is called *satisfiable* with respect to a knowledge base, if there is a model of this KB where  $C^{\mathcal{I}}$  is not empty.
- Unsatisfiable atomic concepts normally indicate modeling errors in the KB.
- Checking concept satisfiability can be reduced to checking (non-)entailment:  $C$  is satisfiable wrt. a KB if the KB does **not** entail the axiom  $C \sqsubseteq \perp$ .

# Instance Retrieval

- Asking for all the named individuals known to be in a certain concept (role) is a typical querying or retrieval task.
- It can be reduced to checking entailment of as many individual assertions as there are named individuals in the knowledge base.
- Depending on the used system and inferencing algorithm, this can be done in a much more efficient way (e.g. by translation into a database query).

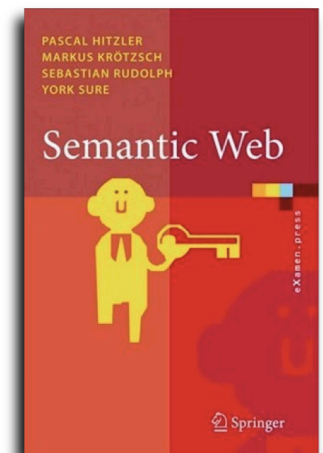
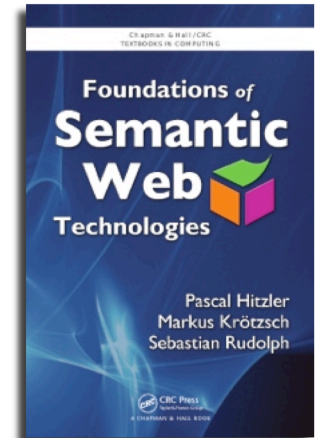
# Classification

- Classification of a knowledge base aims at determining for any two concept names  $A$ ,  $B$ , whether  $A \sqsubseteq B$  is a consequence of the KB.
- This is useful at KB design time for checking the inferred concept hierarchy. Also, computing this hierarchy once and storing it can speed up further queries.
- Classification can be reduced to checking entailment of GCI's.
- While this requires quadratically many checks, one can often do much better in practice by applying optimizations and exploiting that subsumption is a preorder.



# References – Textbooks

- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure, Semantic Web – Grundlagen. Springer, 2008.  
<http://www.semantic-web-grundlagen.de/>  
(In German.)
- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies. Chapman & Hall/CRC, 2009.  
<http://www.semantic-web-book.org/>



# Thank You!

