TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik, Institut Künstliche Intelligenz, Professur Computational Logic

Deduction Systems

Sebastian Rudolph
Computational Logic
sebastian.rudolph@tu-dresden.de

About this Lecture

- Mondays, 14:50 – 16:20, APB 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://ddll.inf.tu-dresden.de/web/Deduction_Systems_%28SS2019%29

Chair for Computational Logic
Institute for Artificial Intelligence

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik, Institut Künstliche Intelligenz, Professur Computational Logic

Foundations of Description Logics

Sebastian Rudolph
Computational Logic
sebastian.rudolph@tu-dresden.de

DRESDEN concept
Exzellenz aus Wissenschaft und Kultur

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

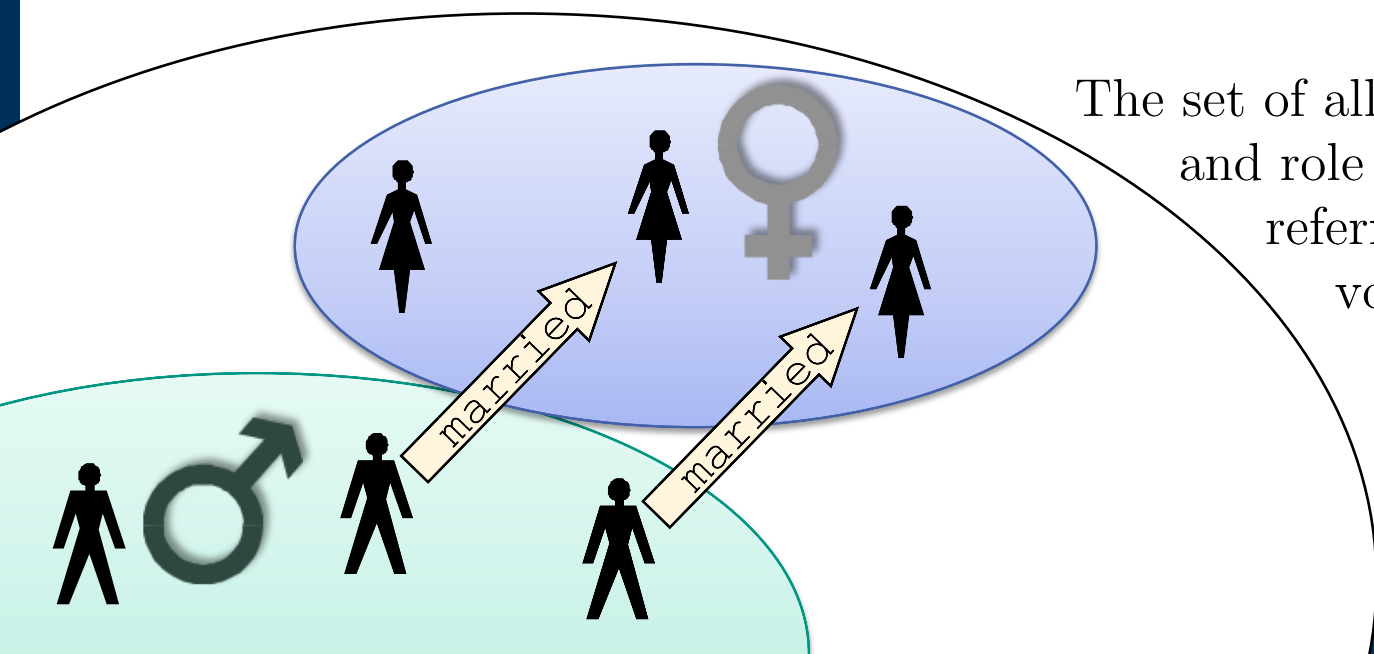
HermiT

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

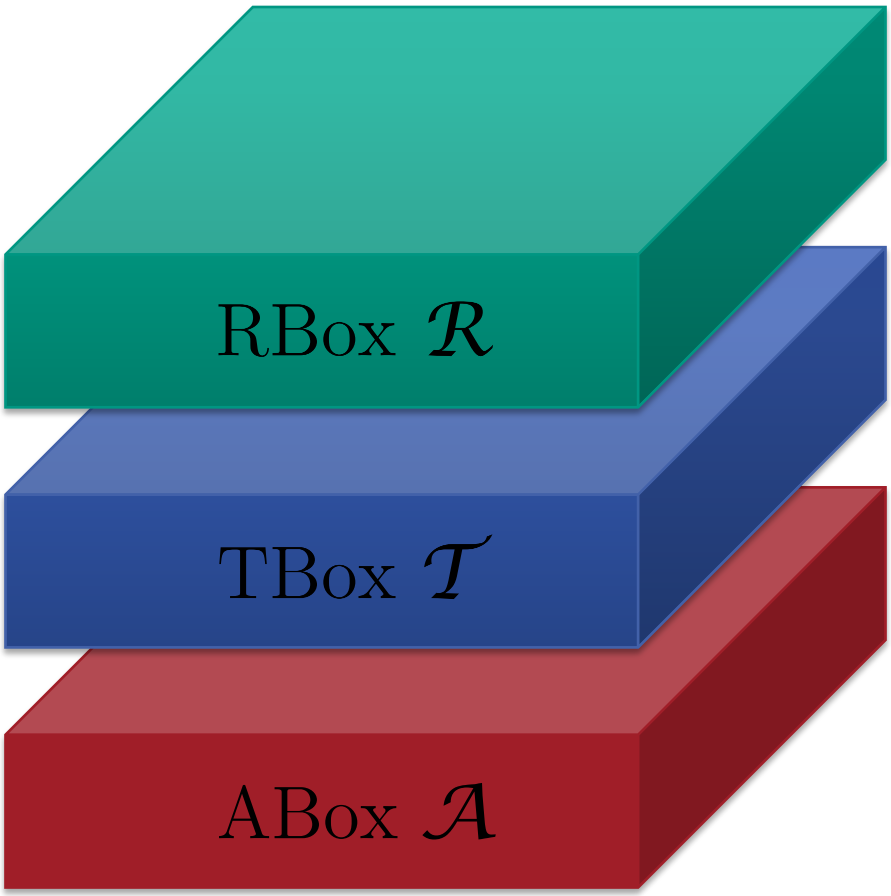
DL Building Blocks

- **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



information about roles and their dependencies

information about concepts and their taxonomic dependencies

information about individuals and their concept and role memberships

Roles and Role Inclusion Axioms

- A *role* can be
  - a role name **r** or
  - an inverted role name **r**⁻ or
  - the universal role *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 $\mathtt{r}_1 \circ \mathtt{r}_2 \circ \ldots \circ \mathtt{r}_n \sqsubseteq \mathtt{r}$ with n>1, $\mathtt{r}$ is non-simple,
  - for any RIA $\mathtt{s} \sqsubseteq \mathtt{r}$ with $\mathtt{s}$ non-simple, $\mathtt{r}$ is non-simple, and
  - all other roles are simple.

- Example:

  $\mathtt{q} \circ \mathtt{p} \sqsubseteq \mathtt{r}$ $\quad$ $\mathtt{r} \circ \mathtt{p} \sqsubseteq \mathtt{r}$ $\quad$ $\mathtt{r} \sqsubseteq \mathtt{s}$ $\quad$ $\mathtt{p} \sqsubseteq \mathtt{r}$ $\quad$ $\mathtt{q} \sqsubseteq \mathtt{s}$

  non-simple: $\mathtt{r}$, $\mathtt{s}$ $\quad$ simple: $\mathtt{p}$, $\mathtt{q}$

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 $\prec$ on the roles such that
  - every RIA has one of the following forms with $\mathtt{s}_i \prec \mathtt{r}$ for all i=1,2,...,n:

$\mathtt{r} \circ \mathtt{r} \sqsubseteq \mathtt{r}$ $\qquad$ $\mathtt{r}^- \sqsubseteq \mathtt{r}$ $\qquad$ $\mathtt{s}_1 \circ \mathtt{s}_2 \circ \ldots \circ \mathtt{s}_n \sqsubseteq \mathtt{r}$

$\mathtt{r} \circ \mathtt{s}_1 \circ \mathtt{s}_2 \circ \ldots \circ \mathtt{s}_n \sqsubseteq \mathtt{r}$ $\qquad$ $\mathtt{s}_1 \circ \mathtt{s}_2 \circ \ldots \circ \mathtt{s}_n \circ \mathtt{r} \sqsubseteq \mathtt{r}$

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

RBox

- A *role disjointness* statement has the form

$$\mathrm{Dis}(\mathtt{s}_1,\mathtt{s}_2)$$

  where $\mathtt{s}_1$ and $\mathtt{s}_2$ 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 $\bot$ are concept expressions,
  - for $\mathsf{a}_1,\ldots,\mathsf{a}_n$ individual names, $\{\mathsf{a}_1,\ldots,\mathsf{a}_n\}$ 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, $\exists s.\mathsf{Self}$ and $\leq n s.C$ and $\geq n s.C$ 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
  - $C(\mathsf{a})$, called *concept assertion*,
  - $r(\mathsf{a},\mathsf{b})$, called *role assertion*,
  - $\neg r(\mathsf{a},\mathsf{b})$, called *negated role assertion*,
  - $\mathsf{a} \approx \mathsf{b}$, called *equality statement*, or
  - $\mathsf{a} \not\approx \mathsf{b}$, called *inequality statement*.

- An *ABox* consists of a set of individual assertions.

An Example Knowledge Base

RBox $\mathcal{R}$

$$\texttt{owns} \sqsubseteq \texttt{caresFor}$$

"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 all beings he cares for are healthy."

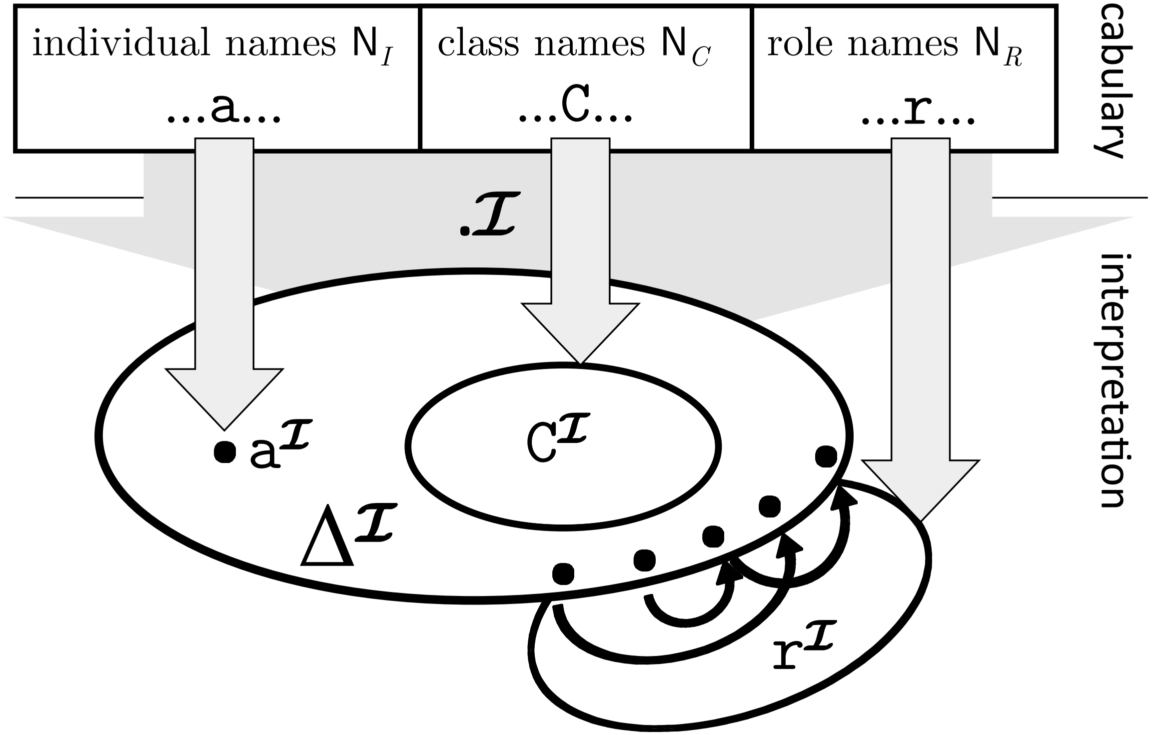
ABox $\mathcal{A}$

HappyCatOwner (schrödinger)

"Schrödinger is a happy cat owner."

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

vocabulary

N_I = {sun, morning_star, evening_star, moon, home}.
N_C = {Planet, Star}.
N_R = {orbitsAround, shinesOn}.

domain

$\Delta^{\mathcal{I}}$ = {☉, ☿, ♀, ♁, ☾, ♂, ♃, ♄, ⛢, ♆, ♇}

interpretation of individual names

sun$^{\mathcal{I}}$ = ☉
morning_star$^{\mathcal{I}}$ = ♀
evening_star$^{\mathcal{I}}$ = ♀
moon$^{\mathcal{I}}$ = ☾
home$^{\mathcal{I}}$ = ♁

interpretation of concept names

Planet$^{\mathcal{I}}$ = {☿, ♀, ♁, ♂, ♃, ♄, ⛢, ♆}
Star$^{\mathcal{I}}$ = {☉}

interpretation of role names

orbitsAround$^{\mathcal{I}}$ = {⟨☿, ☉⟩, ⟨♀, ☉⟩, ⟨♁, ☉⟩, ⟨♂, ☉⟩, ⟨♃, ☉⟩,
⟨♄, ☉⟩, ⟨⛢, ☉⟩, ⟨♆, ☉⟩, ⟨♇, ☉⟩, ⟨☾, ♁⟩}
shinesOn$^{\mathcal{I}}$ = {⟨☉, ☿⟩, ⟨☉, ♀⟩, ⟨☉, ♁⟩, ⟨☉, ☾⟩, ⟨☉, ♂⟩,
⟨☉, ♃⟩, ⟨☉, ♄⟩, ⟨☉, ⛢⟩, ⟨☉, ♆⟩, ⟨☉, ♇⟩}

Interpretations: an Example

TECHNISCHE UNIVERSITÄT DRESDEN

vocabulary

$\mathsf{N}_I = \{$sun, morning_star, evening_star, moon, home$\}$.
$\mathsf{N}_C = \{$Planet, Star$\}$.
$\mathsf{N}_R = \{$orbitsAround, shinesOn$\}$.

domain

$\Delta^{\mathcal{I}}$ = {⊙, ☿, ♀, ♁, ☾, ♂, ♃, ♄, ♅, ♆, ♇}

interpretation of individual names

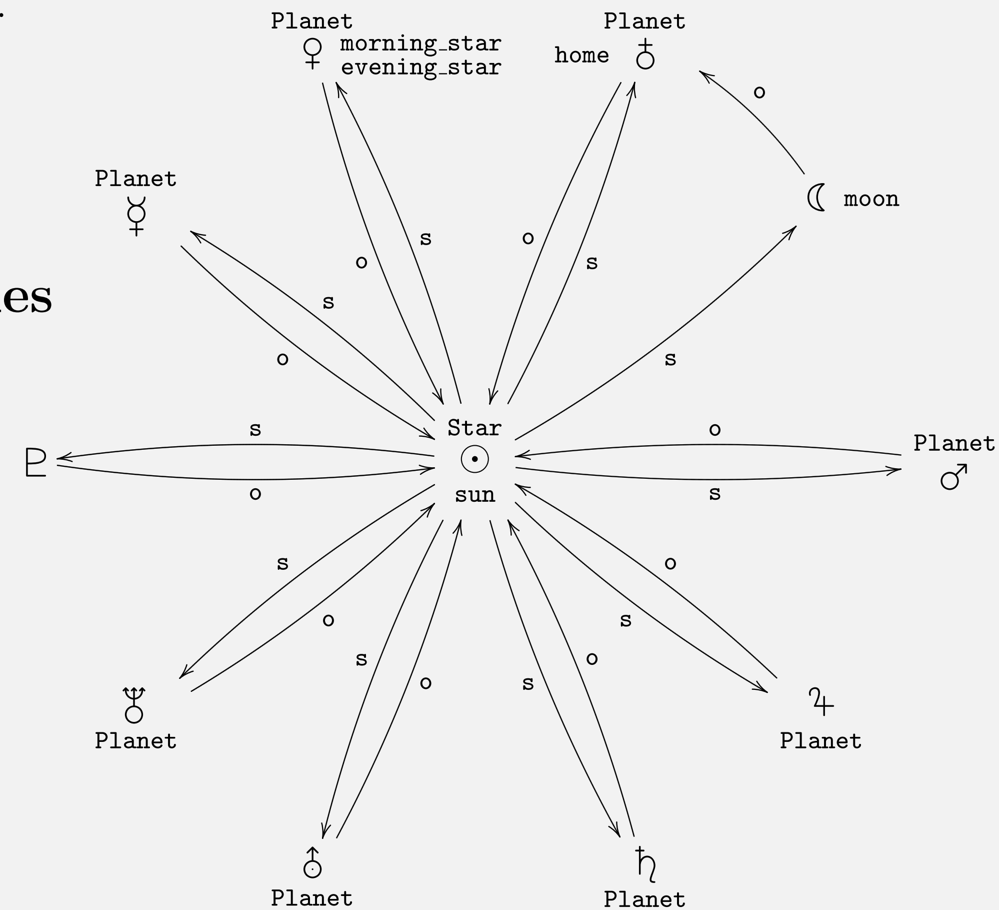
sun$^{\mathcal{I}}$ = ⊙
morning_star$^{\mathcal{I}}$ = ♀
evening_star$^{\mathcal{I}}$ = ♀
moon$^{\mathcal{I}}$ = ☾
home$^{\mathcal{I}}$ = ♁

interpretation of concept names

Planet$^{\mathcal{I}}$ = {☿, ♀, ♁, ♂, ♃, ♄, ♅, ♆}
Star$^{\mathcal{I}}$ = {⊙}

interpretation of role names

orbitsAround$^{\mathcal{I}}$ = {⟨☿, ⊙⟩, ⟨♀, ⊙⟩, ⟨♁, ⊙⟩, ⟨♂, ⊙⟩, ⟨♃, ⊙⟩,
⟨♄, ⊙⟩, ⟨♅, ⊙⟩, ⟨♆, ⊙⟩, ⟨♇, ⊙⟩, ⟨☾, ♁⟩}
shinesOn$^{\mathcal{I}}$ = {⟨⊙, ☿⟩, ⟨⊙, ♀⟩, ⟨⊙, ♁⟩, ⟨⊙, ☾⟩, ⟨⊙, ♂⟩,
⟨⊙, ♃⟩, ⟨⊙, ♄⟩, ⟨⊙, ♅⟩, ⟨⊙, ♆⟩, ⟨⊙, ♇⟩}



Interpretation of Concept Expressions

Given an interpretation, we can determine the semantic counterparts for concept expressions along the following inductive definitions:

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

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

$$\{\mathsf{a}_1,\ldots,\mathsf{a}_n\}^{\mathcal{I}} = \{\mathsf{a}_1{}^{\mathcal{I}},\ldots,\mathsf{a}_n{}^{\mathcal{I}}\}$$

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

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

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

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

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

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

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

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

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 α (written: $\mathcal{I} \models \alpha$).

$\mathcal{I} \models r_1 \circ \ldots \circ r_n \sqsubseteq r$	if $r_1^{\mathcal{I}} \circ \ldots \circ r_n^{\mathcal{I}} \subseteq r^{\mathcal{I}}$
$\mathcal{I} \models \mathrm{Dis}(\mathtt{s}_1,\mathtt{s}_2)$	if $\mathtt{s}_1^{\mathcal{I}} \cap \mathtt{s}_2^{\mathcal{I}} = \{\}$
$\mathcal{I} \models C \sqsubseteq D$	if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$\mathcal{I} \models C(\mathsf{a})$	if $\mathsf{a}^{\mathcal{I}} \in C^{\mathcal{I}}$
$\mathcal{I} \models r(\mathsf{a},\mathsf{b})$	if $(\mathsf{a}^{\mathcal{I}},\mathsf{b}^{\mathcal{I}}) \in r^{\mathcal{I}}$
$\mathcal{I} \models \neg \mathrm{r}(\mathsf{a},\mathsf{b})$	if $(\mathsf{a}^{\mathcal{I}},\mathsf{b}^{\mathcal{I}}) \notin r^{\mathcal{I}}$
$\mathcal{I} \models \mathsf{a} \approx \mathsf{b}$	if $\mathsf{a}^{\mathcal{I}} = \mathsf{b}^{\mathcal{I}}$
$\mathcal{I} \models \mathsf{a} \not\approx \mathsf{b}$	if $\mathsf{a}^{\mathcal{I}} \neq \mathsf{b}^{\mathcal{I}}$

(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*).

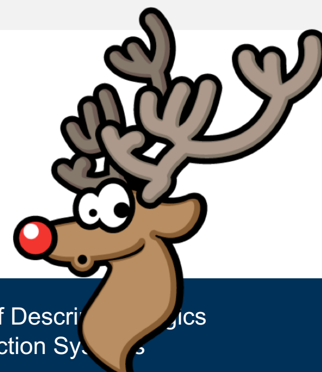
- Is the following KB satisfiable?

$$\mathtt{Reindeer}\sqcap\exists\mathtt{hasNose}.\mathtt{Red}(\mathtt{rudolph}) \qquad \mathtt{Reindeer} \sqsubseteq \mathtt{Mammal}$$

$$\forall\mathtt{worksFor}^{-}.(\neg\mathtt{Reindeer}\sqcup\mathtt{Flies})(\mathtt{santa}) \qquad \mathtt{Mammal}\sqcap\mathtt{Flies} \sqsubseteq \mathtt{Bat}$$

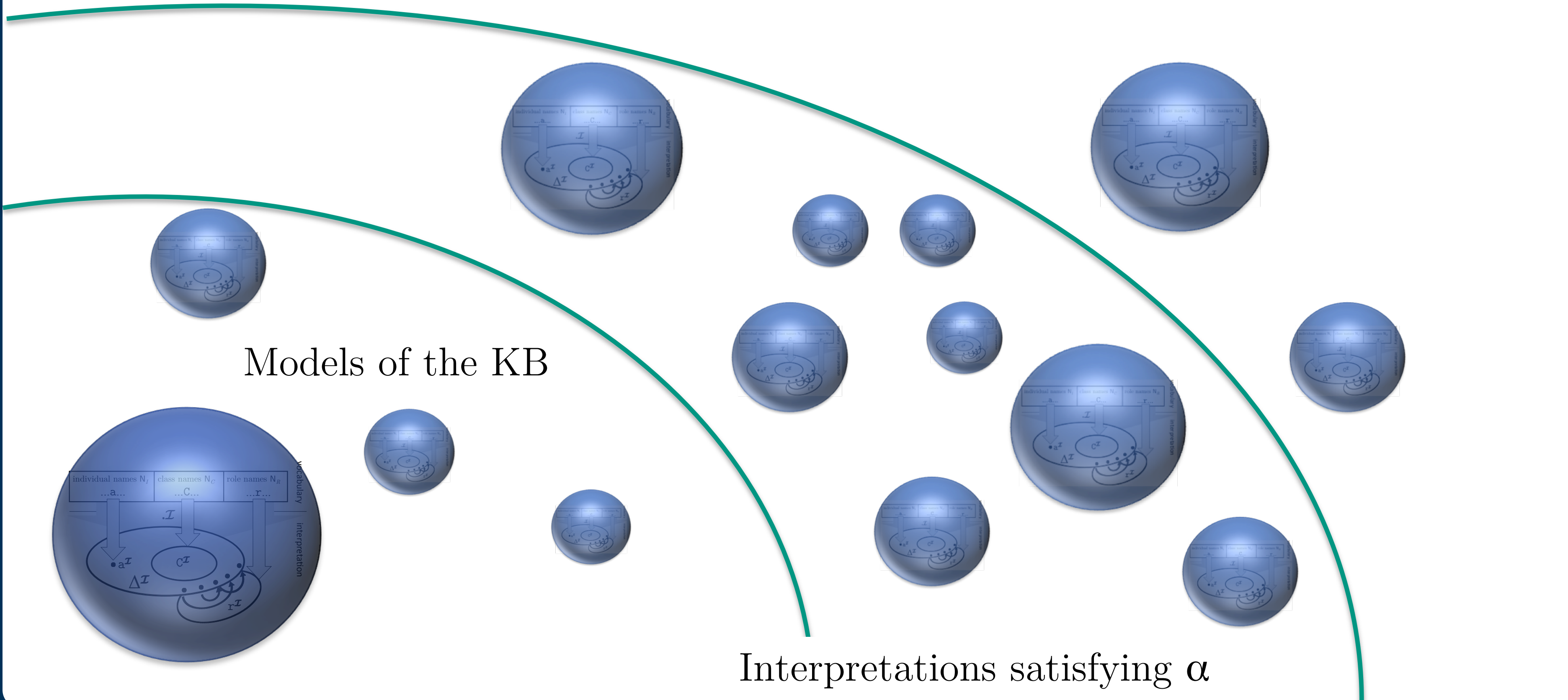
$$\mathtt{worksFor}(\mathtt{rudolph}, \mathtt{santa}) \qquad \mathtt{Bat} \sqsubseteq \forall\mathtt{worksFor}.\{\mathtt{batman}\}$$

$$\mathtt{santa} \not\approx \mathtt{batman}$$


Entailment of Axioms

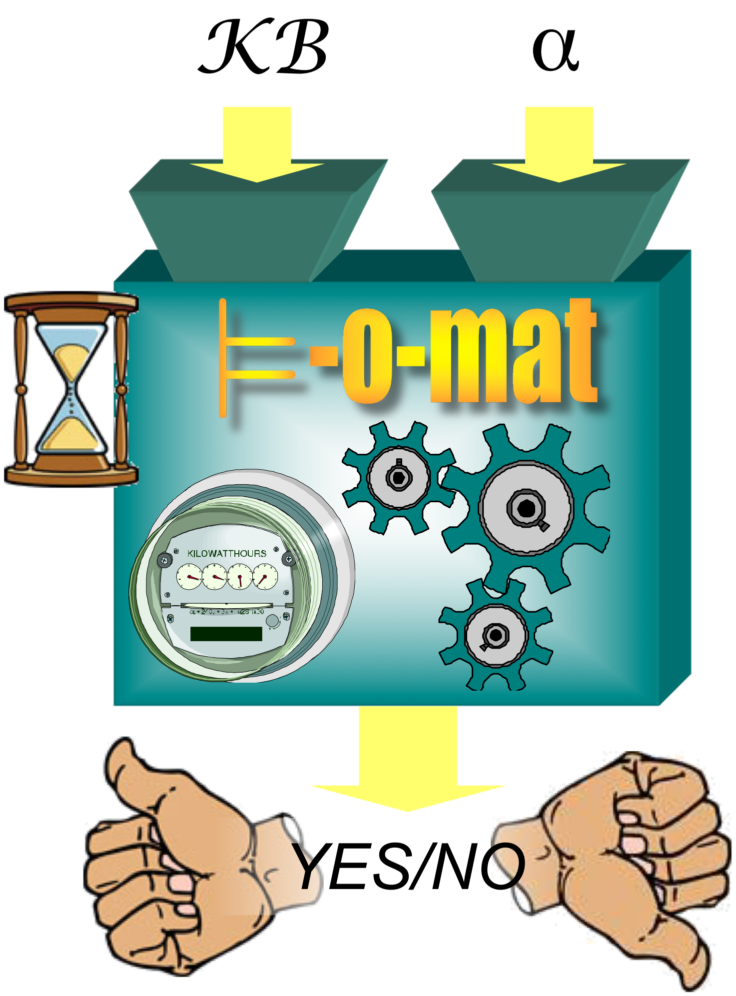
- A KB *entails* an axiom α if the axiom α is satisfied by every model of the knowledge base.



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.



Naming Scheme for Expressive DLs

$$((\mathcal{ALC} \mid \mathcal{S})[\mathcal{H}] \mid \mathcal{SR})[\mathcal{O}][\mathcal{I}][\mathcal{F} \mid \mathcal{N} \mid \mathcal{Q}]$$

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

We treat here the very expressive description logic $\mathcal{SROIQ}$ which subsumes all the other ones in this scheme.

DL Syntax – Overview

TECHNISCHE UNIVERSITÄT DRESDEN

	Concepts	
$\mathcal{ALC}$	Atomic	A, B
	Not	¬C
	And	C ⊓ D
	Or	C ⊔ D
	Exists	∃r.C
	For all	∀r.C
$\mathcal{Q}$ $(\mathcal{N})$	At least	≥n r.C (≥n r)
	At most	≤n r.C (≤n r)
$\mathcal{O}$	Closed class	$\{i_1, \dots, i_n\}$
$\mathcal{R}$	Self	∃r.Self

	Roles	
	Atomic	r
$\mathcal{I}$	Inverse	r^-

Ontology (=Knowledge Base)

Concept Axioms (TBox)	
Subclass	C ⊑ D
Equivalent	C ≡ D

	Role Axioms (RBox)	
$\mathcal{H}$	Subrole	r ⊑ s
$\mathcal{S}$	Transitivity	Trans(r)
$\mathcal{SR}$	Role Chain	r ∘ r′ ⊑ s
	R. Disjointness	Disj(s,r)

Assertional Axioms (ABox)	
Instance	C(a)
Role	r(a,b)
Same	a ≈ b
Different	a ≉ b

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

Concept Equivalences

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

$$C \sqcap D \equiv D \sqcap C \qquad C \sqcup D \equiv D \sqcup C$$
$$(C \sqcap D) \sqcap E \equiv C \sqcap (D \sqcap E) \qquad (C \sqcup D) \sqcup E) \equiv C \sqcup (D \sqcup E)$$
$$C \sqcap C \equiv C \qquad C \sqcup C \equiv C$$

$$(C \sqcup D) \sqcap E \equiv (C \sqcap E) \sqcup (D \sqcap E) \qquad (C \sqcup D) \sqcap C \equiv C$$
$$(C \sqcap D) \sqcup E \equiv (C \sqcup E) \sqcap (D \sqcup E) \qquad (C \sqcap D) \sqcup C \equiv C$$

$$\neg\neg C \equiv C$$
$$\neg(C \sqcap D) \equiv \neg D \sqcup \neg C$$
$$\neg(C \sqcup D) \equiv \neg D \sqcap \neg C$$

$$\neg\exists r.C \equiv \forall r.\neg C$$
$$\neg\forall r.C \equiv \exists r.\neg C$$
$$\neg{\leqslant} n r.C \equiv {\geqslant}(n+1) r.C$$
$$\neg{\geqslant}(n+1) r.C \equiv {\leqslant} n r.C$$

$${\geqslant} 0 r.C \equiv \top$$
$${\geqslant} 1 r.C \equiv \exists r.C$$
$${\leqslant} 0 r.C \equiv \forall r.\neg C$$

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.

$$nnf(C) := C \text{ if } C \in \{A, \neg A, \{\mathsf{a}_1,\ldots,\mathsf{a}_n\}, \neg\{\mathsf{a}_1,\ldots,\mathsf{a}_n\}, \exists r.\mathsf{Self}, \neg\exists r.\mathsf{Self}, \top, \bot\}$$

$$nnf(\neg\neg C) := nnf(C)$$

$$\begin{array}{llll}
nnf(\neg\top) & := \bot & nnf(\neg\bot) & := \top \\
nnf(C \sqcap D) & := nnf(C) \sqcap nnf(D) & nnf(\neg(C \sqcap D)) & := nnf(\neg C) \sqcup nnf(\neg D) \\
nnf(C \sqcup D) & := nnf(C) \sqcup nnf(D) & nnf(\neg(C \sqcup D)) & := nnf(\neg C) \sqcap nnf(\neg D) \\
nnf(\forall r.C) & := \forall r.nnf(C) & nnf(\neg\forall r.C) & := \exists r.nnf(\neg C) \\
nnf(\exists r.C) & := \exists r.nnf(C) & nnf(\neg\exists r.C) & := \forall r.nnf(\neg C) \\
nnf(\leqslant n\, r.C) & := \leqslant n\, r.nnf(C) & nnf(\neg \leqslant n\, r.C) & := \geqslant (n+1)\, r.nnf(C) \\
nnf(\geqslant n\, r.C) & := \geqslant n\, r.nnf(C) & nnf(\neg \geqslant n\, r.C) & := \leqslant (n-1)\, r.nnf(C)
\end{array}$$

Axiom and KB Equivalences

- Lloyd-Topor equivalences

$$\{A \sqcup B \sqsubseteq C\} \Longleftrightarrow \{A \sqsubseteq C,\ B \sqsubseteq C\}$$
$$\{A \sqsubseteq B \sqcap C\} \Longleftrightarrow \{A \sqsubseteq B,\ A \sqsubseteq C\}$$

- turning GCIs into universally valid concept descriptions

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

- internalisation of ABox into TBox

$$C(\mathsf{a}) \Longleftrightarrow \{\mathsf{a}\} \sqsubseteq C$$
$$r(\mathsf{a},\mathsf{b}) \Longleftrightarrow \{\mathsf{a}\} \sqsubseteq \exists r.\{\mathsf{b}\}$$
$$\neg r(\mathsf{a},\mathsf{b}) \Longleftrightarrow \{\mathsf{a}\} \sqsubseteq \neg\exists r.\{\mathsf{b}\}$$
$$\mathsf{a} \approx \mathsf{b} \Longleftrightarrow \{\mathsf{a}\} \sqsubseteq \{\mathsf{b}\}$$
$$\mathsf{a} \not\approx \mathsf{b} \Longleftrightarrow \{\mathsf{a}\} \sqsubseteq \neg\{\mathsf{b}\}$$

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.

Knowledge base	Query	DL answers	Prolog
child(bill,bob)			
Man(bob)	? $\models \forall$child.Man(Bill)	don't know	yes
≤ 1child.$\top$(Bill)	? $\models \forall$child.Man(Bill)	yes	

Now we know everything about Bill's children.

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 $\mathsf{Dead} \sqcap \mathsf{Alive}$),
- 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 directly negated within the logic, use fresh individual names as Skolem constants, e.g., $C \sqsubseteq D$ "negates" to $C \sqcap \neg D(\mathtt{c})$

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 $C \sqsubseteq \bot$.

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

TECHNISCHE UNIVERSITÄT DRESDEN

- Classification of a knowledge base aims at determining for any two concept names A, B, whether $\mathsf{A} \sqsubseteq \mathsf{B}$ 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 GCIs.
- While this requires quadratically many checks, one can often do much better in practice by applying optimizations and exploiting that subsumption is a preorder.