Beschreibungslogik

- Description logic (DL)
- Vorläufer sind Formalismen zur Wissensrepräsentation wie
  - Frames (Minsky)
  - Semantische Netzwerke
  - KL-ONE [Brachman, Schmolze 1985]
- DL ist entscheidbares Fragment der Prädikatenlogik erster Stufe (PL1) mit
  - ein- und zweistelligen Prädikaten
  - Operatoren der Aussagenlogik
  - eingeschränkte Quantifizierung

Beschreibungslogik

- Was heißt “entscheidbares Fragment der PL1”?
- Russells Paradoxon:

  Der Barbier von Sevilla rasiert alle Männer von Sevilla, die sich nicht selbst rasieren.

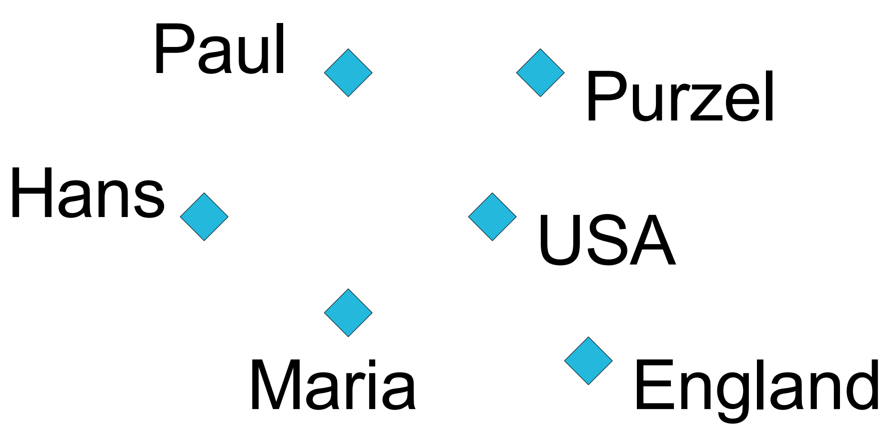
  Rasiert sich der Barbier von Sevilla selbst?
- Problem Selbstbezüglichkeit: Mengen die sich selbst als Element enthalten
- Russells Lösung: Klassenhierarchien
- DL: keine Klassen von Klassen möglich

Beschreibungslogik

DL umfasst Individuen, Eigenschaften und Klassen.

- **Individuen**: konkrete Objekte der Domäne, Instanzen von Klassen.

  Es muss explizit festgelegt werden, ob zwei Individuen dasselbe Objekt repräsentieren oder nicht.



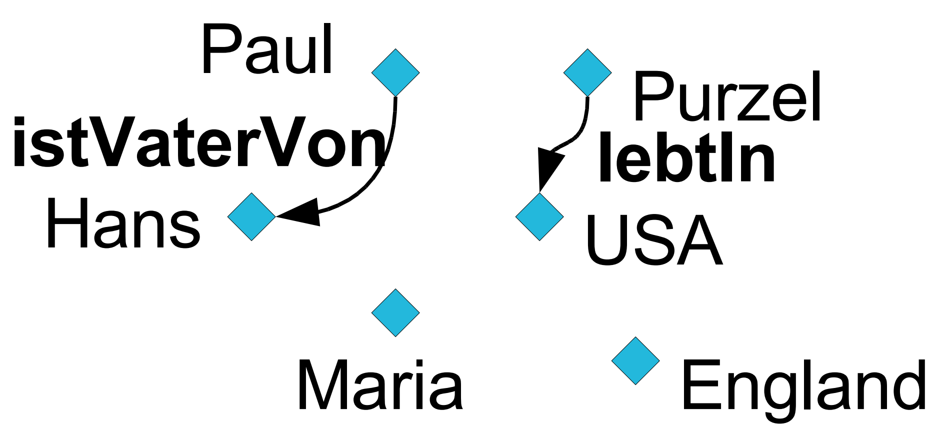
Beschreibungslogik

- **Eigenschaften** (Rollen, Slots, Relationen, Attribute) sind binäre Relationen über Individuen

  inverse Eigenschaften: x hatBesitzer y <-> y gehört x

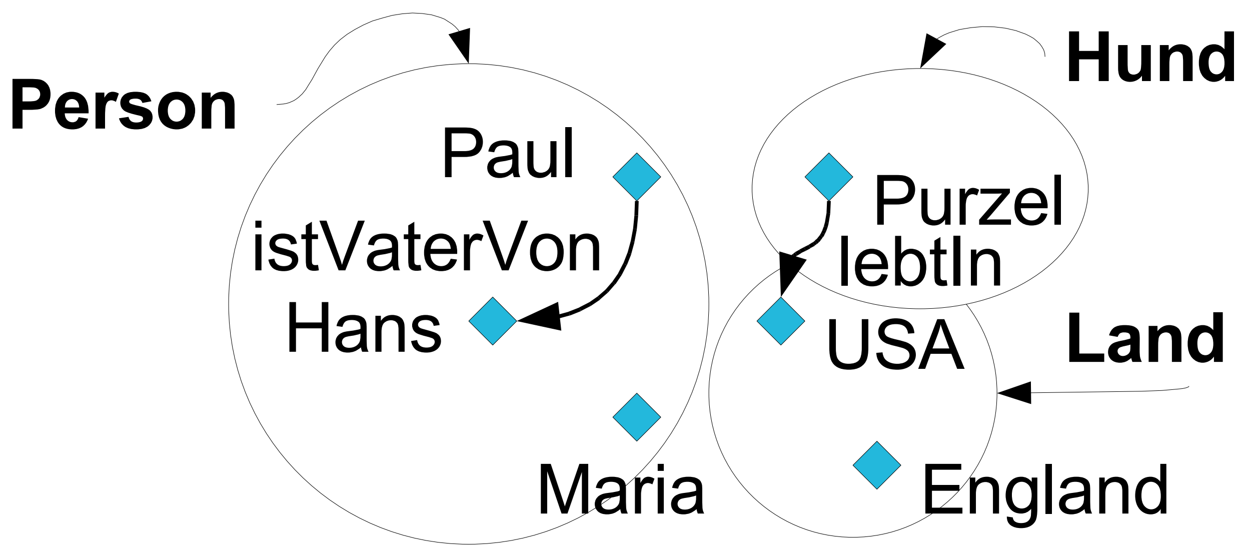
  transitiv, symmetrisch, reflexiv, ...

  Wertebereiche einschränkbar



Beschreibungslogik

- **Klassen** (Konzepte) sind Mengen aus Individuen
- definiert durch Angabe von Bedingungen, die ein Individuum erfüllen muss, um Mitglied der Klasse zu sein
- Superklassen - Subklassen (Taxonomie)



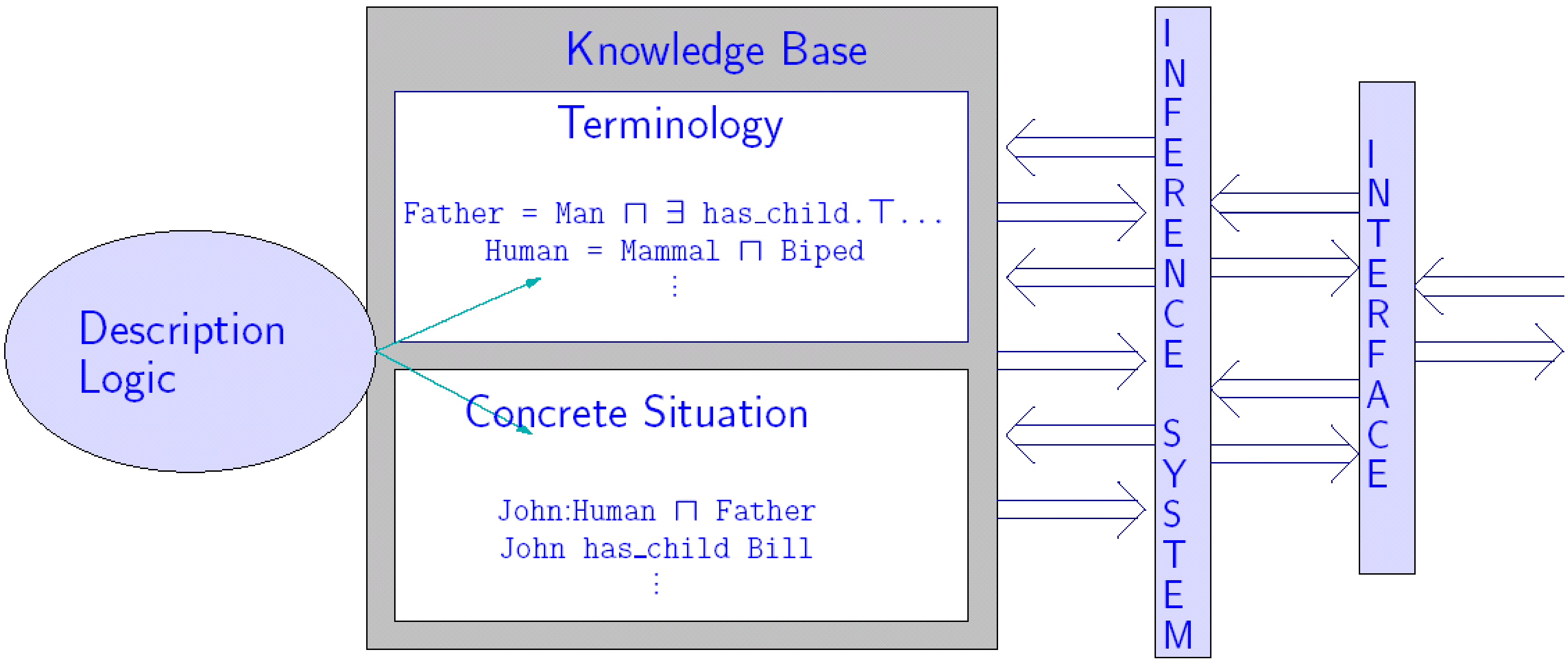
Semantik der DL

Interpretation I = (Δ^I, $\bullet^I$):

- Domäne (nichtleere Menge von Individuen) Δ^I,
- Funktion $\bullet^I$ (Interpretationsfunktion) bildet
  - jedes Konzept auf Untermenge von Δ^I,
  - jede Eigenschaft auf Untermenge von $\Delta^I \times \Delta^I$,
  - jedes Individuum auf Element von Δ^I

  ab.

Architecture of a Standard DL System



Source: Description Logics Tutorial, Ian Horrocks and Ulrike Sattler, ECAI-2002, Lyon, France, July 23rd, 2002

DL: Notation

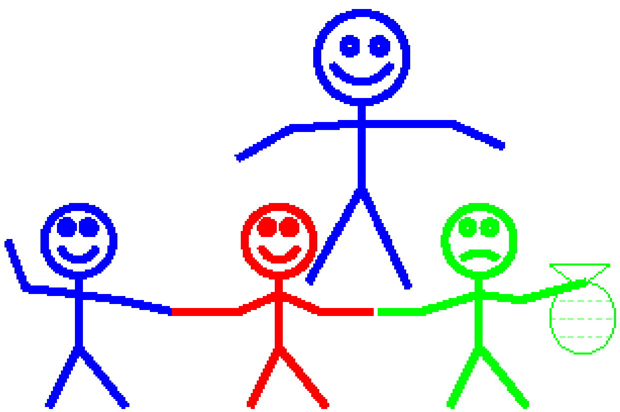
Student

Person	
name:	[String]
address:	[String]
enrolled:	[Course]

$$\begin{aligned}\{x \mid \texttt{Student}(x)\} = \{x \mid\, & \texttt{Person}(x) \wedge \\ & (\exists y.\ \texttt{NAME}(x,y) \wedge \texttt{String}(y)) \wedge \\ & (\exists z.\ \texttt{ADDRESS}(x,z) \wedge \texttt{String}(z)) \wedge \\ & (\exists w.\ \texttt{ENROLLED}(x,w) \wedge \texttt{Course}(w))\ \}\end{aligned}$$

$$\begin{aligned}\texttt{Student} \doteq\ & \texttt{Person} \sqcap \\ & \exists \texttt{NAME}.\texttt{String} \sqcap \\ & \exists \texttt{ADDRESS}.\texttt{String} \sqcap \\ & \exists \texttt{ENROLLED}.\texttt{Course}\end{aligned}$$

Example: "Happy Father"



$$\text{Man} \sqcap (\exists \text{has-child.Blue}) \sqcap (\exists \text{has-child.Green}) \sqcap (\forall \text{has-child.Happy} \sqcup \text{Rich})$$

Example TBox

1. P
2. F
3. $\mathsf{W} \equiv \mathsf{P} \sqcap \mathsf{F}$
4. $\mathsf{M1} \equiv \mathsf{P} \sqcap \neg\mathsf{W}$
5. $\mathsf{M2} \equiv \mathsf{W} \sqcap \exists h1.\mathsf{P}$
6. $\mathsf{F2} \equiv \mathsf{M1} \sqcap \exists h1.\mathsf{P}$
7. $\mathsf{P2} \equiv (\mathsf{F2} \sqcup \mathsf{M2})$
8. $\mathsf{G} \equiv \mathsf{M2} \sqcap \exists h1.\mathsf{P2}$
9. $\mathsf{W2} \equiv \mathsf{W} \sqcap \exists h2.\mathsf{M1}$
10. $\mathsf{M3} \equiv \mathsf{M2} \sqcap \forall h1.\neg\mathsf{W}$

Atomic concepts = {P,F,W, M1,…}

Base concepts = {P,F}

Defined concepts = {W, M1, M2, …}

Roles = {*h1*,*h2*}

Concept Definition

Axiom

$\mathsf{A} \equiv \mathsf{C}$

where A atomic concept

$\mathsf{C} \sqsubseteq \mathsf{D}$

C, D complex concept expressions

Exercise

1. P
2. F
3. $\mathsf{W} \equiv \mathsf{P} \sqcap \mathsf{F}$
4. $\mathsf{M1} \equiv \mathsf{P} \sqcap \neg\mathsf{W}$
5. $\mathsf{M2} \equiv \mathsf{W} \sqcap \exists h1.\mathsf{P}$
6. $\mathsf{F2} \equiv \mathsf{M1} \sqcap \exists h1.\mathsf{P}$
7. $\mathsf{P2} \equiv (\mathsf{F2} \sqcup \mathsf{M2})$
8. $\mathsf{G} \equiv \mathsf{M2} \sqcap \exists h1.\mathsf{P2}$
9. $\mathsf{W2} \equiv \mathsf{W} \sqcap \exists h2.\mathsf{M1}$
10. $\mathsf{M3} \equiv \mathsf{M2} \sqcap \forall h1.\neg\mathsf{W}$

Starting with the base interpretation of

- I(P) := “the class of persons”
- I(F) := “the class of females”

… what is the meaning of the defined concepts?

… what role play the *roles* in this process?

Example Tbox spelled out

1. Person
2. Female
3. Woman $\equiv$ Person $\sqcap$ Female
4. Man $\equiv$ Person $\sqcap$ $\neg$Woman
5. Mother $\equiv$ Woman $\sqcap$ $\exists hasChild$.Person
6. Father $\equiv$ Man $\sqcap$ $\exists hasChild$.Person
7. Parent $\equiv$ (Father $\sqcup$ Mother)
8. Grandmother $\equiv$ Mother $\sqcap$ $\exists hasChild$.Parent
9. Wife $\equiv$ Woman $\sqcap$ $\exists hasHusband$.Man
10. MotherWithoutDaughter $\equiv$ Mother $\sqcap$ $\forall hasChild$.$\neg$Woman

- atomic concept
- atomic concept
- concept def. w/ intersection
- … plus negation
- … existential restriction
- … value restriction

DL Knowledge Base

- DL Knowledge Base (KB) normally separated into 2 parts:
  - TBox is a set of axioms describing structure of domain (i.e., a conceptual schema), e.g.:
    - HappyFather Man AND hasChild.Female AND …
    - Elephant Animal AND Large AND Grey
    - transitive(ancestor)
  - ABox is a set of axioms describing a concrete situation (data), e.g.:
    - John:HappyFather
    - <John,Mary>:hasChild
- Separation has no logical significance
  - But is conceptually and implementationally convenient

Introduction to DL: Syntax and Semantics of $\mathcal{ALC}$

Semantics given by means of an **interpretation** $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

Constructor	Syntax	Example	Semantics
atomic concept	A	Human	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
atomic role	R	likes	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
For C, D concepts and R a role name			
conjunction	$C \sqcap D$	Human $\sqcap$ Male	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	Nice $\sqcup$ Rich	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
negation	$\neg C$	$\neg$ Meat	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
exists restrict.	$\exists R.C$	$\exists$has-child.Human	$\{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
value restrict.	$\forall R.C$	$\forall$has-child.Blond	$\{x \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}$

Introduction to DL: Other DL Constructors

Constructor	Syntax	Example	Semantics
number restriction	$(\geq n\ R)$	$(\geq 7$ has-child$)$	$\{x \mid \lvert\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}\rvert \geq n\}$
	$(\leq n\ R)$	$(\leq 1$ has-mother$)$	$\{x \mid \lvert\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}\rvert \leq n\}$
inverse role	R^{-}	has-child^{-}	$\{\langle x, y\rangle \mid \langle y, x\rangle \in R^{\mathcal{I}}\}$
trans. role	R^{*}	has-child*	$(R^{\mathcal{I}})^{*}$
concrete domain	$u_1, \ldots, u_n.P$	h-father·age, age. $>$	$\{x \mid \langle u_1^{\mathcal{I}}, \ldots, u_n^{\mathcal{I}}\rangle \in P\}$
etc.			

Many different DLs/DL constructors have been investigated

For terminological knowledge: **TBox** contains

Concept definitions $A \doteq C$ (A a concept name, C a complex concept)

$\text{Father} \doteq \text{Man} \sqcap \exists\text{has-child.Human}$

$\text{Human} \doteq \text{Mammal} \sqcap \forall\text{has-child}^{-}.\text{Human}$

$\leadsto$ introduce macros/names for concepts, can be (a)cyclic

Axioms $C_1 \sqsubseteq C_2$ (C_i complex concepts)

$\exists\text{favourite.Brewery} \sqsubseteq \exists\text{drinks.Beer}$

$\leadsto$ restrict your models

An interpretation $\mathcal{I}$ **satisfies**

a **concept definition** $A \doteq C$ iff $A^{\mathcal{I}} = C^{\mathcal{I}}$

an **axiom** $C_1 \sqsubseteq C_2$ iff $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$

a **TBox** $\mathcal{T}$ iff $\mathcal{I}$ satisfies all definitions and axioms in $\mathcal{T}$

$\leadsto$ $\mathcal{I}$ is a **model** of $\mathcal{T}$

Introduction to DL: Knowledge Bases: ABoxes

For assertional knowledge: **ABox** contains

Concept assertions $a : C$ (a an individual name, C a complex concept)

$\mathsf{John} : \mathsf{Man} \sqcap \forall \mathsf{has\text{-}child}.(\mathsf{Male} \sqcap \mathsf{Happy})$

Role assertions $\langle a_1, a_2 \rangle : R$ (a_i individual names, R a role)

$\langle \mathsf{John}, \mathsf{Bill} \rangle : \mathsf{has\text{-}child}$

An interpretation $\mathcal{I}$ **satisfies**

a **concept assertion** $a : C$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$

a **role assertion** $\langle a_1, a_2 \rangle : R$ iff $\langle a_1^{\mathcal{I}}, a_2^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$

an **ABox** $\mathcal{A}$ iff $\mathcal{I}$ satisfies all assertions in $\mathcal{A}$

$\leadsto$ $\mathcal{I}$ is a **model** of $\mathcal{A}$

Introduction to DL: Basic Inference Problems

Subsumption: $C \sqsubseteq D$ Is $C^\mathcal{I} \subseteq D^\mathcal{I}$ in all interpretations $\mathcal{I}$?

w.r.t. TBox $\mathcal{T}$: $C \sqsubseteq_\mathcal{T} D$ Is $C^\mathcal{I} \subseteq D^\mathcal{I}$ in all models $\mathcal{I}$ of $\mathcal{T}$?

$\leadsto$ structure your knowledge, compute taxonomy

Consistency: Is C consistent w.r.t. $\mathcal{T}$? Is there a model $\mathcal{I}$ of $\mathcal{T}$ with $C^\mathcal{I} \neq \emptyset$?

of ABox $\mathcal{A}$: Is $\mathcal{A}$ consistent? Is there a model of $\mathcal{A}$?

of KB $(\mathcal{T},\mathcal{A})$: Is $(\mathcal{T},\mathcal{A})$ consistent? Is there a model of both $\mathcal{T}$ and $\mathcal{A}$?

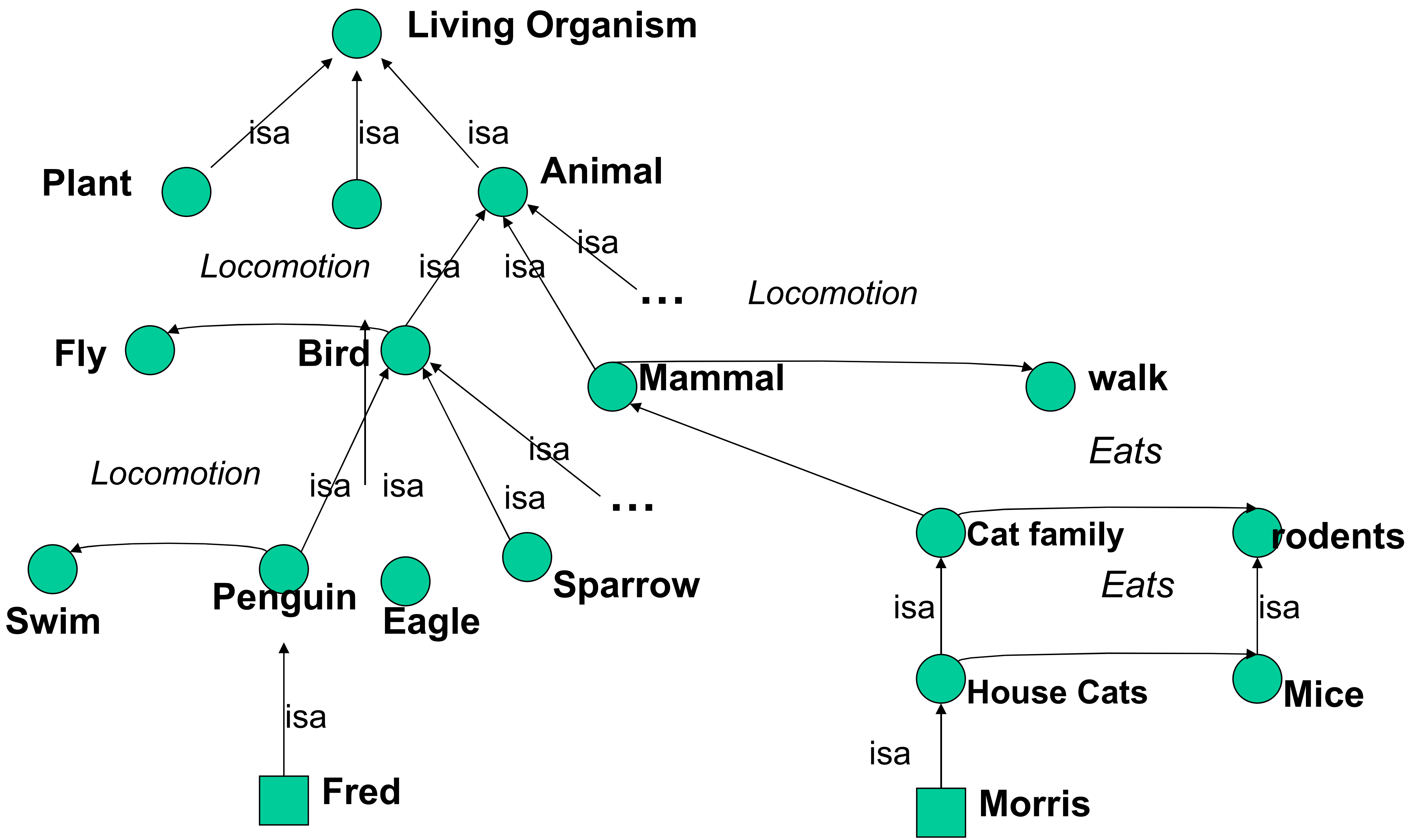
Inference Problems are **closely related:**

$C \sqsubseteq_\mathcal{T} D$ iff $C \sqcap \neg D$ is **in**consistent w.r.t. $\mathcal{T}$,
(no model of $\mathcal{I}$ has an instance of $C \sqcap \neg D$)

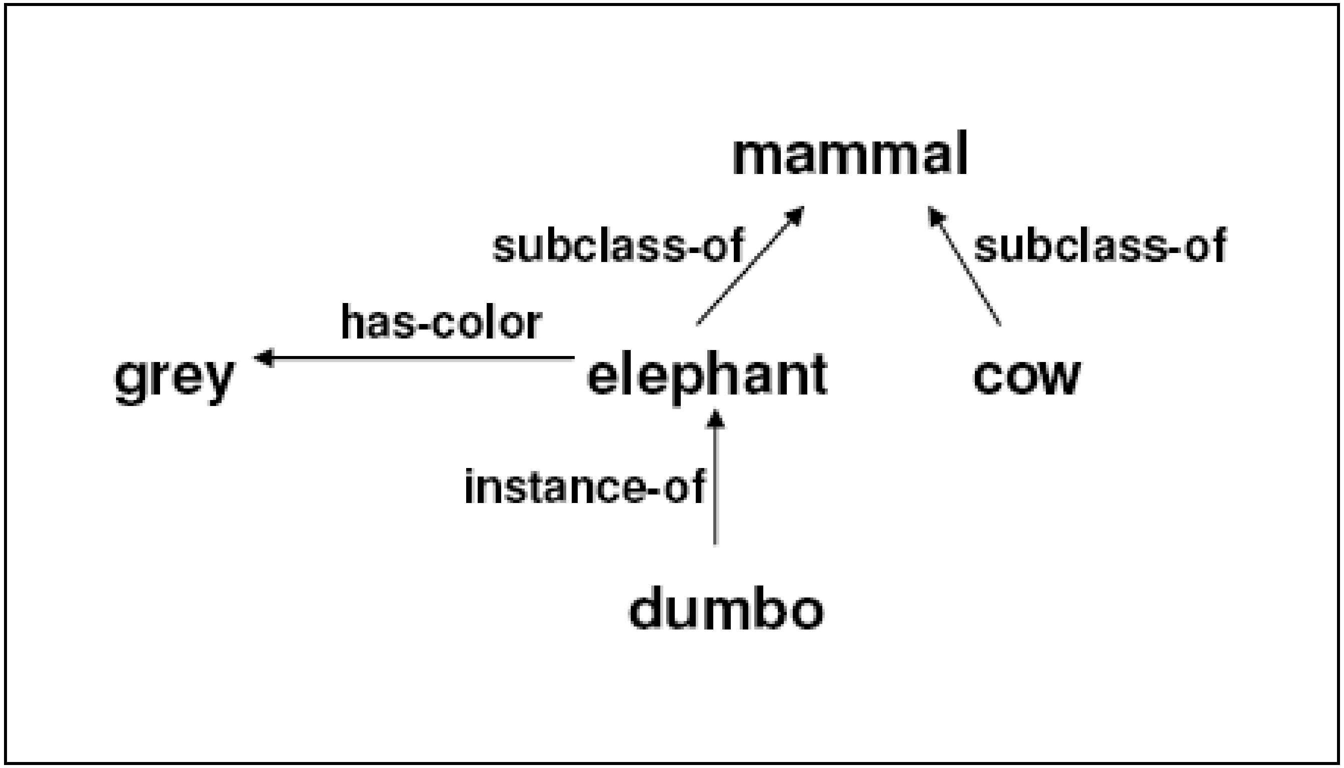
C is consistent w.r.t. $\mathcal{T}$ iff **not** $C \sqsubseteq_\mathcal{T} A \sqcap \neg A$

Source: Description Logics Tutorial, Ian Horrocks and Ulrike Sattler, ECAI-2002, Lyon, France, July 23rd, 2002

Fragment of a Semantic Network



DL: Semantic Net made precise



elephant $\sqsubseteq$ mammal

cow $\sqsubseteq$ mammal

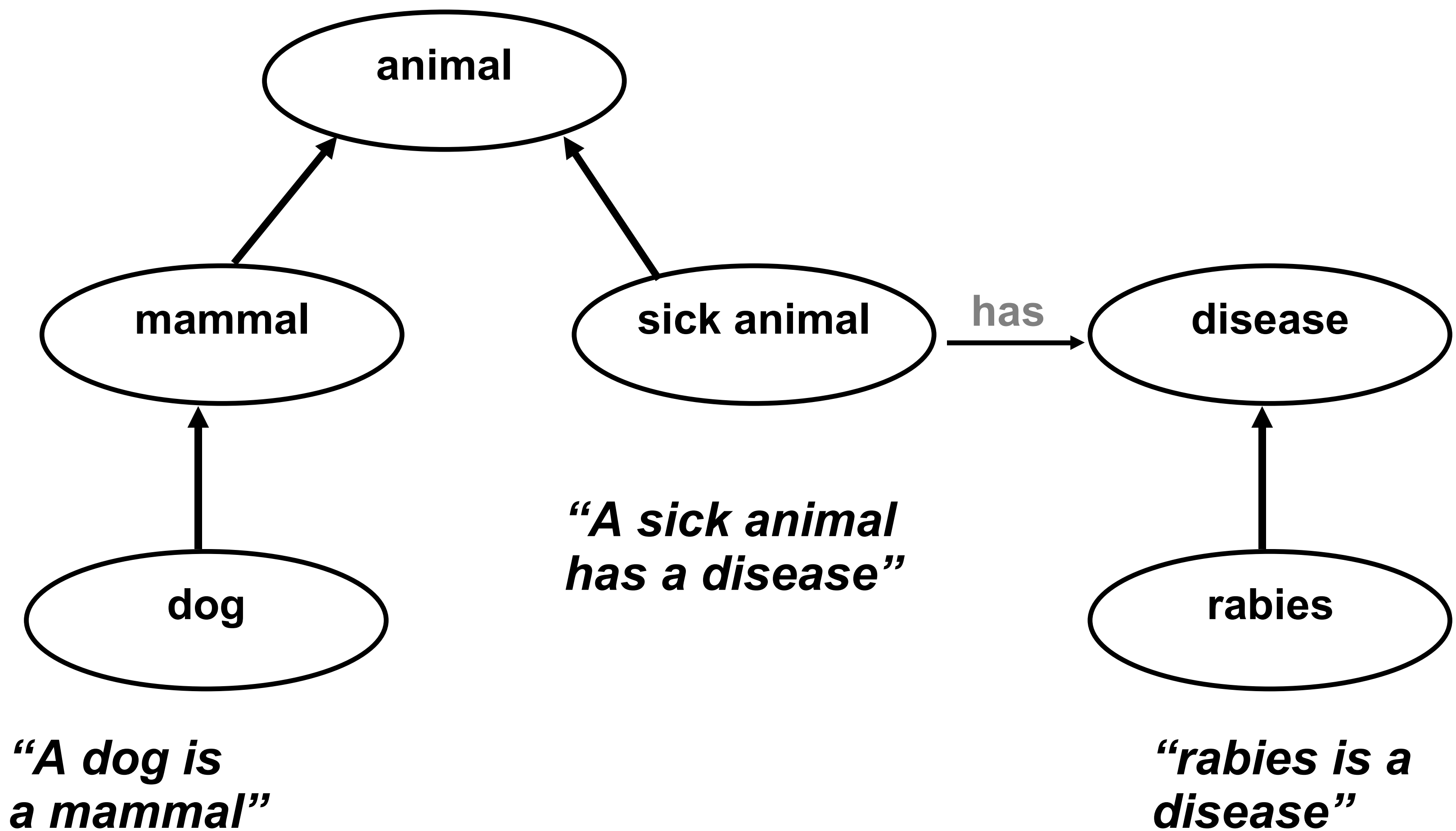
elephant(dumbo)

"Every elephant is also grey": elephant $\sqsubseteq$ $\exists$has-color.grey

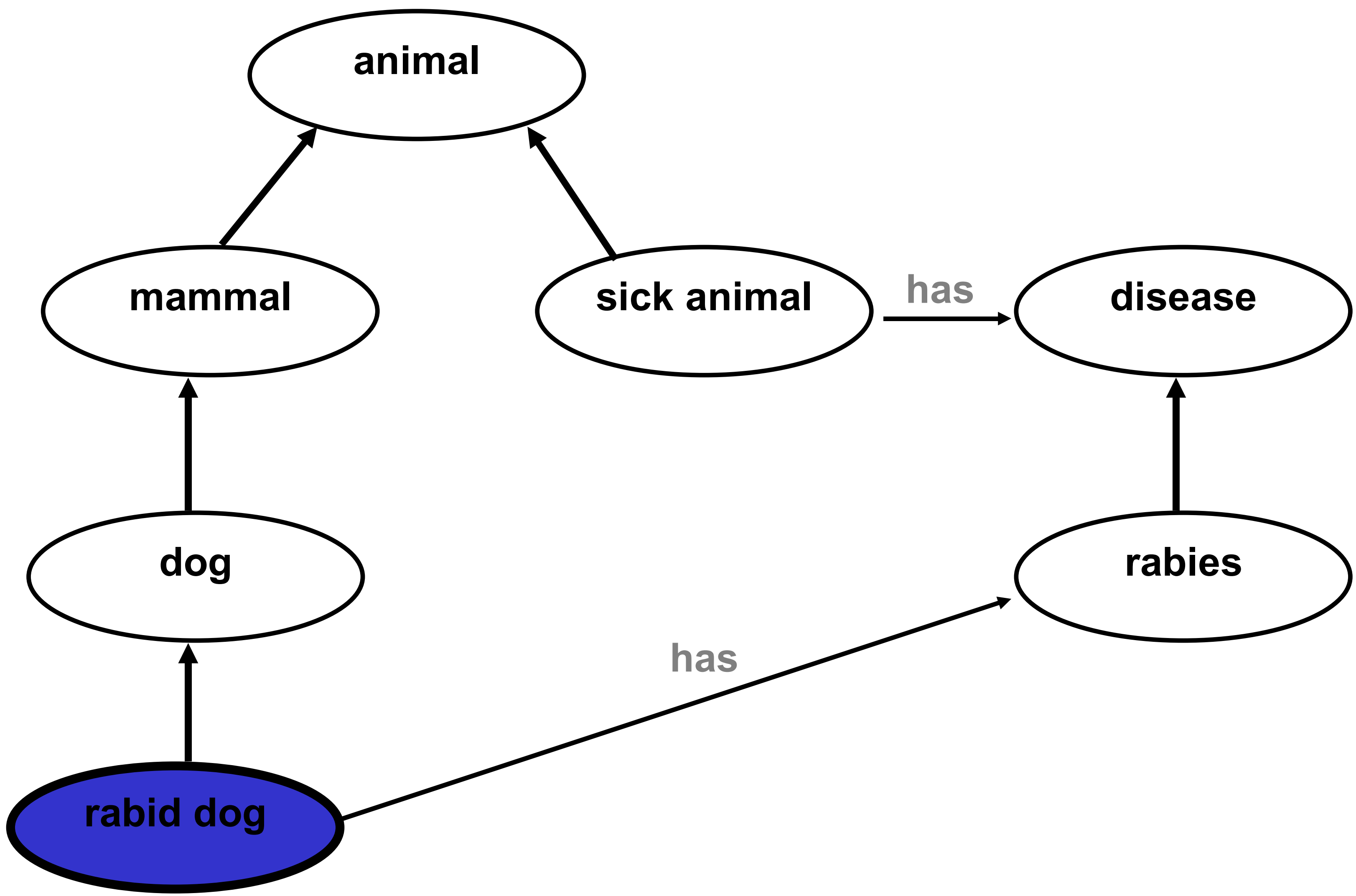
"Every elephant is just grey": elephant $\sqsubseteq$ $\forall$has-color.grey

$\sqsubseteq$ concept introduction: reasoning based on basic concepts and anonymous constructs

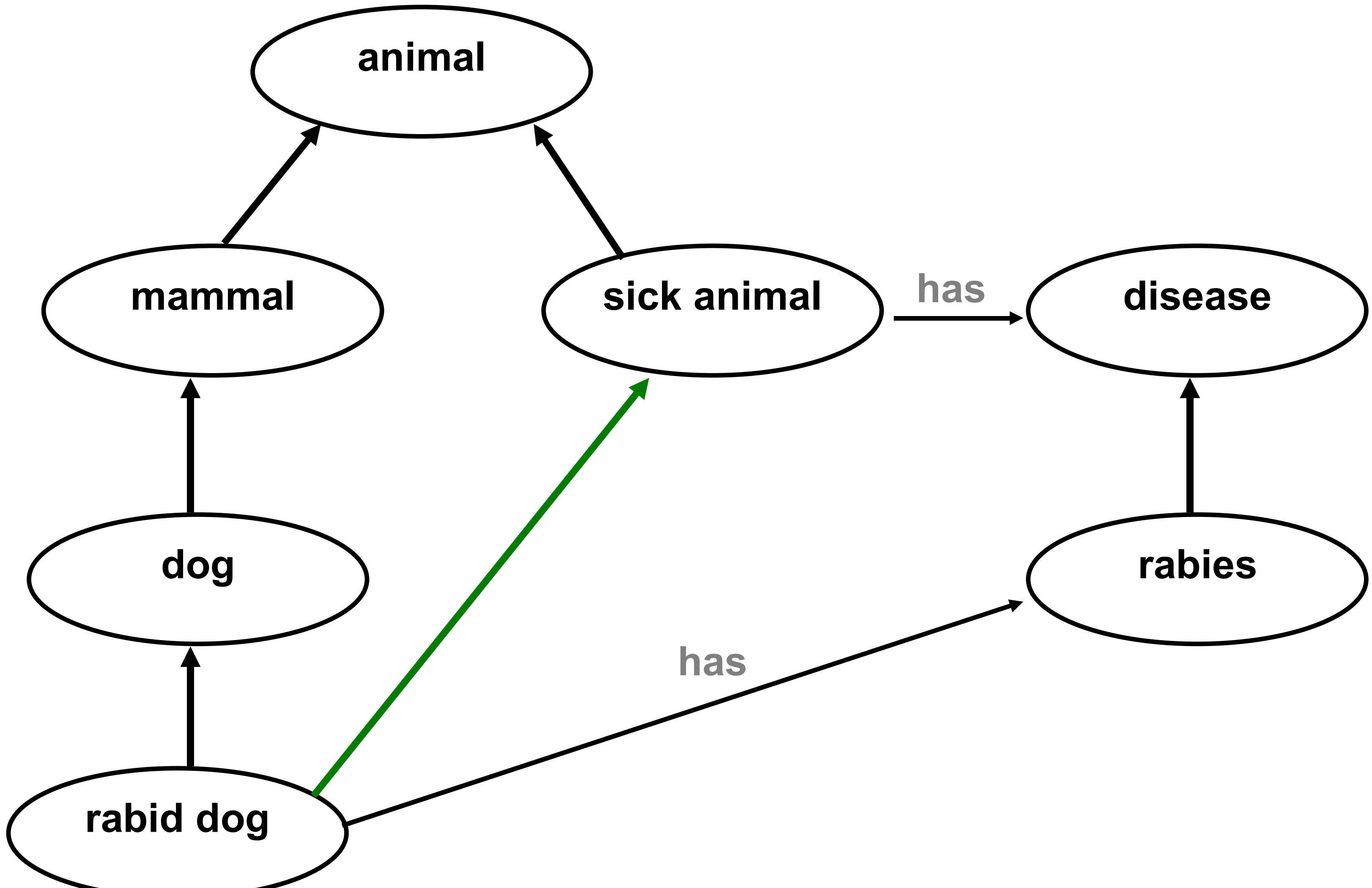
How Does Classification Work?



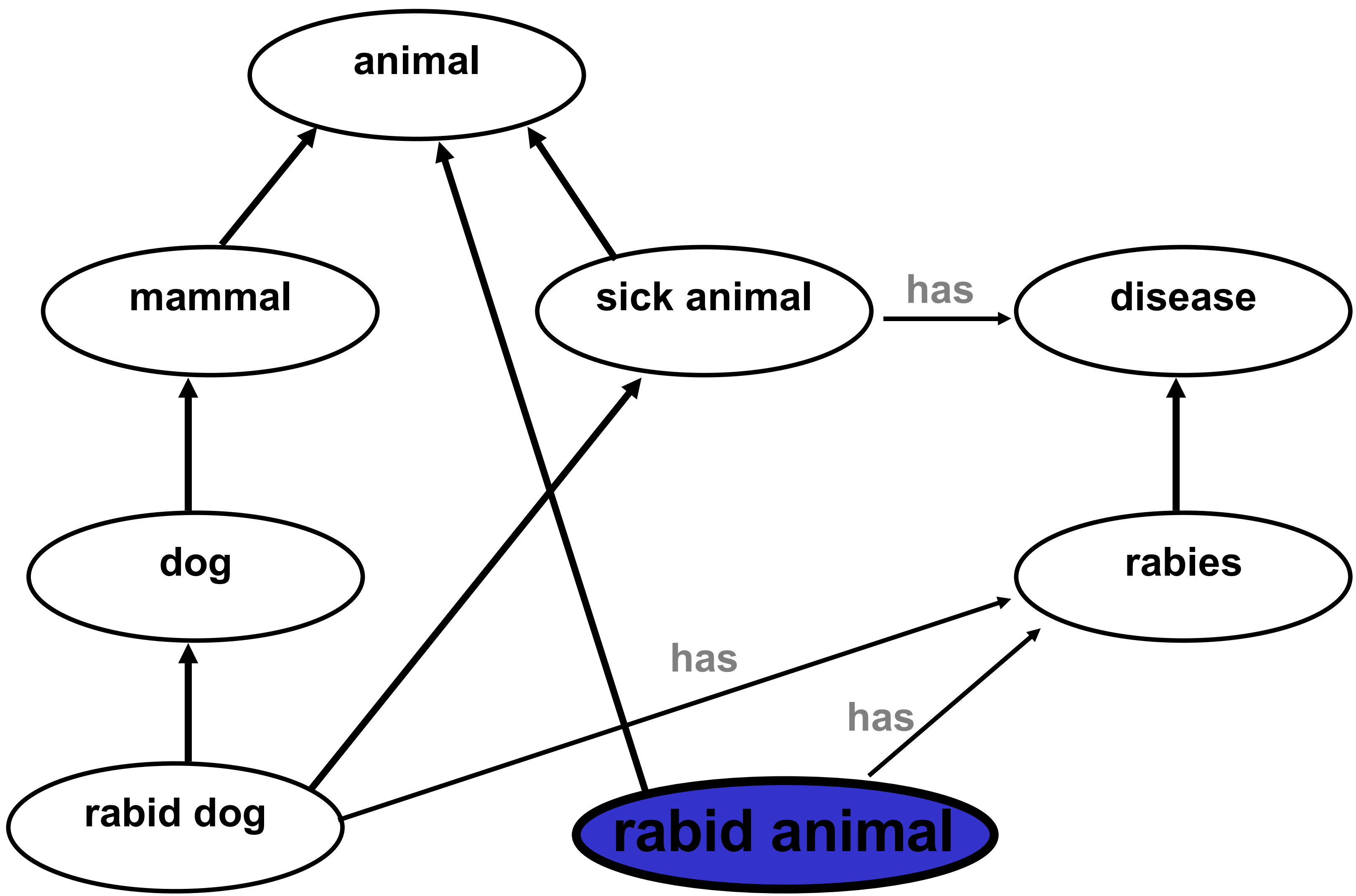
Defining a "rabid dog"



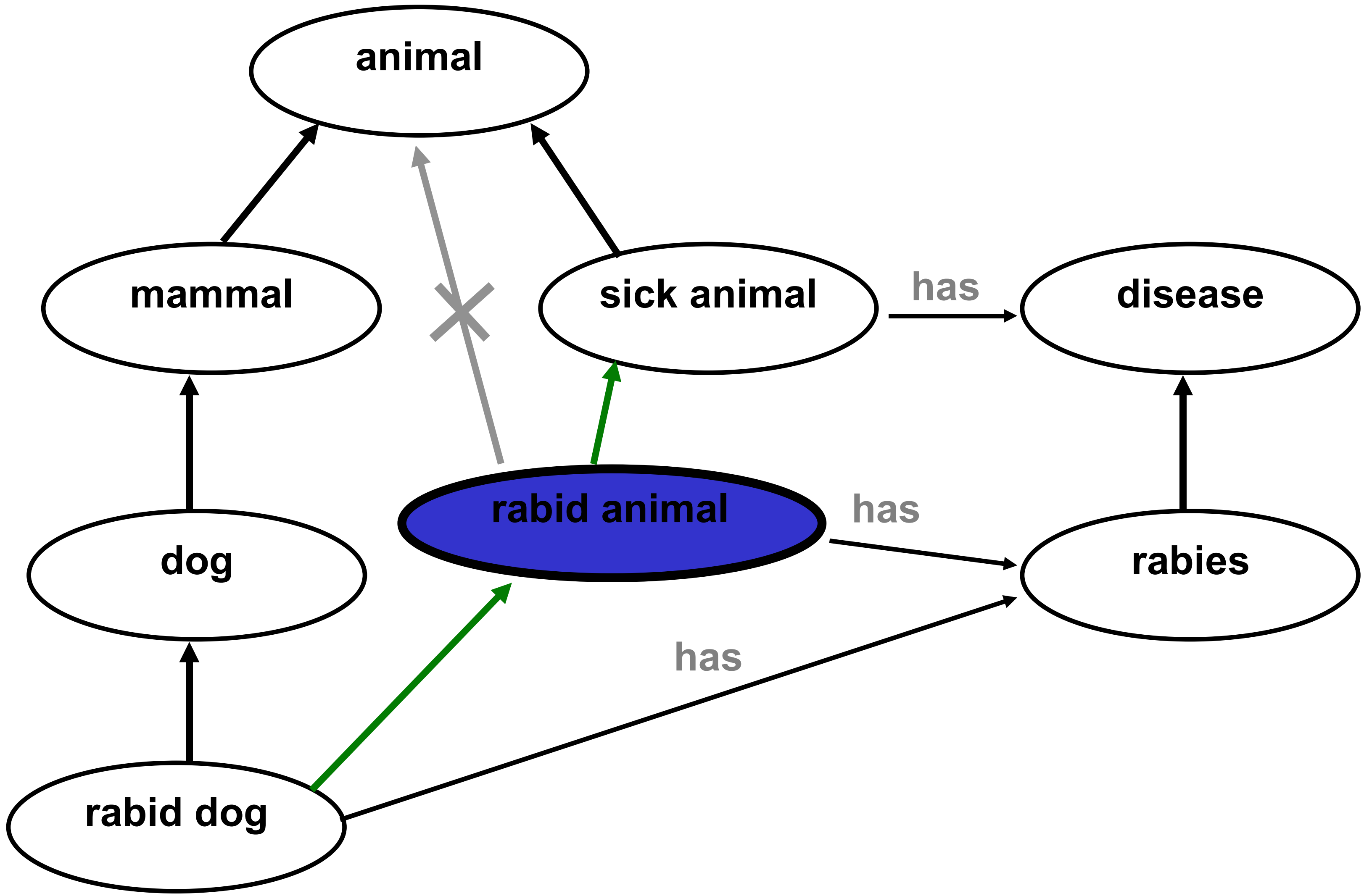
Loom Concludes “sick animal”



Defining "rabid animal"



Loom Places Concept in Hierarchy



OWL and DL

- OWL is a W3C standard to represent ontologies on the Semantic Web
- OWL provides three **increasingly expressive sublanguages**

  OWL Lite, OWL DL, OWL Full
- and **different syntactical representations**

  RDF, OWL/RDF, abstract syntax
- OWL DL (and its easier implementable subset OWL Lite) can be mapped via the OWL abstract syntax to a DL
- In OWL DL concepts are called classes and roles are called properties
- The mapping of OWL DL to description logics allows OWL reasoning based on existing DL reasoners

OWL and DL

OWL Abstract Syntax	DL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$
complementOf	$\neg C$
oneOf	$\{a_1\} \sqcup \ldots \sqcup \{a_n\}$
allValuesFrom	$\forall P.C$
someValuesFrom	$\exists P.C$
maxCardinality	$\leqslant n P$
subClassOf	$C_1 \sqsubseteq C_2$
equivalentClass	$C_1 \equiv C_2$
disjointWith	$C_1 \sqsubseteq \neg C_2$
sameIndividualAs	$\{a_1\} \equiv \{a_2\}$

OWL and DL

OWL/RDF syntax:

```
<owl:Class rdf:ID="Cat">
  <rdfs:subClassOf rdf:resource="#Animal" />
</owl:Class>
```

Abstract syntax:

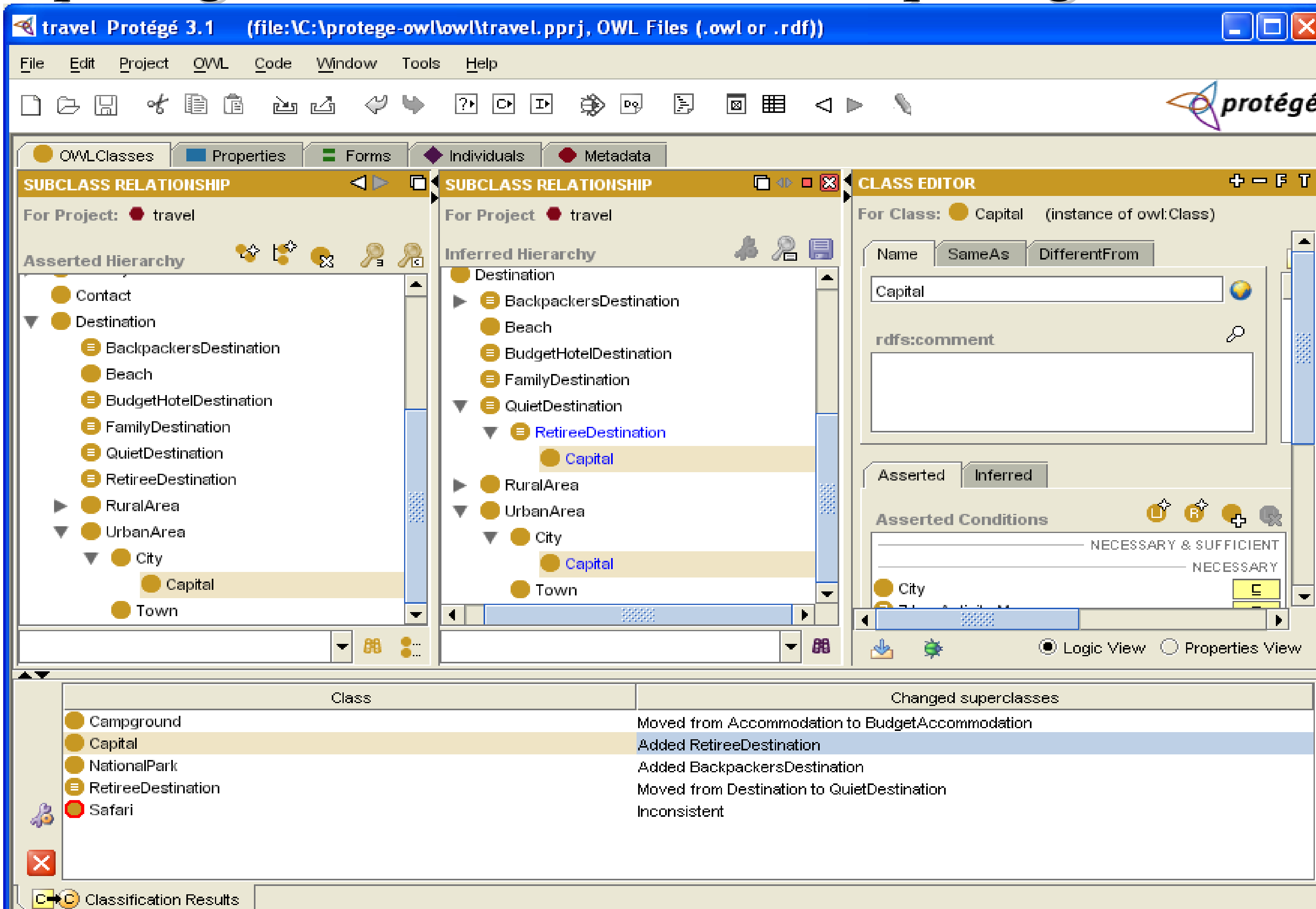
```
class(a:Cat partial
  a:Animal)
```

DL:

$Cat \sqsubseteq Animal$

OWL: Ressourcen

- OWL-Editor: Protegé (http://protege.stanford.edu/overview/protege-owl.html)



OWL: Ressourcen

- OWL-Editor: Protegé (http://protege.stanford.edu/overview/protege-owl.html)
- OWL-Reasoner:
  - Pellet (http://clarkparsia.com/pellet/)
  - FaCT++ (http://owl.man.ac.uk/factplusplus/)
  - RacerPro (http://www.sts.tu-harburg.de/~r.f.moeller/racer/)
- OWL-APIs:
  - OWLAPI (http://owlapi.sourceforge.net)
  - DIG (http://dig.sourceforge.net)

OWLAPI: Beispiel

```
// Connect to a DIG HTTP reasoner, eg Racer, running on localhost:8080
HTTPReasoner racer = new HTTPReasoner(new URL("http://locahost:8080"));

// Wrap it in a TBox reasoner
TReasoner reasoner = new TReasoner(racer);

//Parse a DIG ontology into a TellsDocument xmlbean.
File myDigOntology = new File("myDigOntology.xml");
TellsDocument tells = TellsDocument.Factory.parse(myDigOntology);

// retrieve the children of concept name "driver"
ConceptId driver = ConceptId.Factory.newInstance();
Named d = driver.addNewCatom();
d.setName("driver");
Set equivalents = reasoner.childrenNames(driver, kbURI);
```

Textadventure

- Computerspiel in rein textueller Form ohne Grafik und Sound
- Kommunikation mit Spiel über “Text-Parser”
- z.B. Zork (Infocom)

  Douglas Adams: Hitchhikers Guide to the Galaxy

  The Hobbit
- “interactive fiction”
- http://www.xs4all.nl/~pot/infocom/
- http://c64s.com/game/418/hobbit,_the/

```
West of House                                    Score: 0   Moves: 3
```

ZORK I: The Great Underground Empire

Revision 88 / Serial number 840726

West of House
You are standing in an open field west of a white house, with a boarded front door.
There is a small mailbox here.

> examine mailbox
The small mailbox is closed.

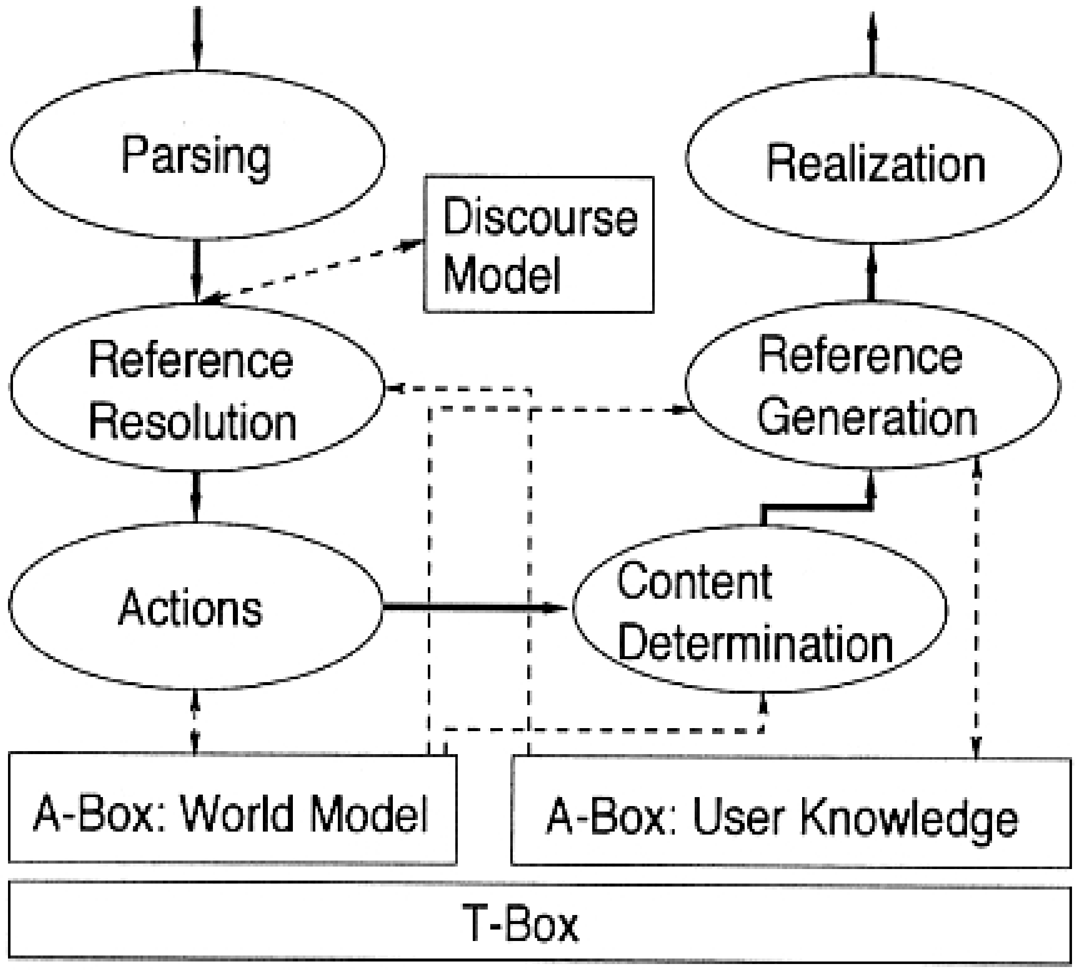
> open mailbox
Opening the small mailbox reveals a leaflet.

> get leaflet
Taken.

>

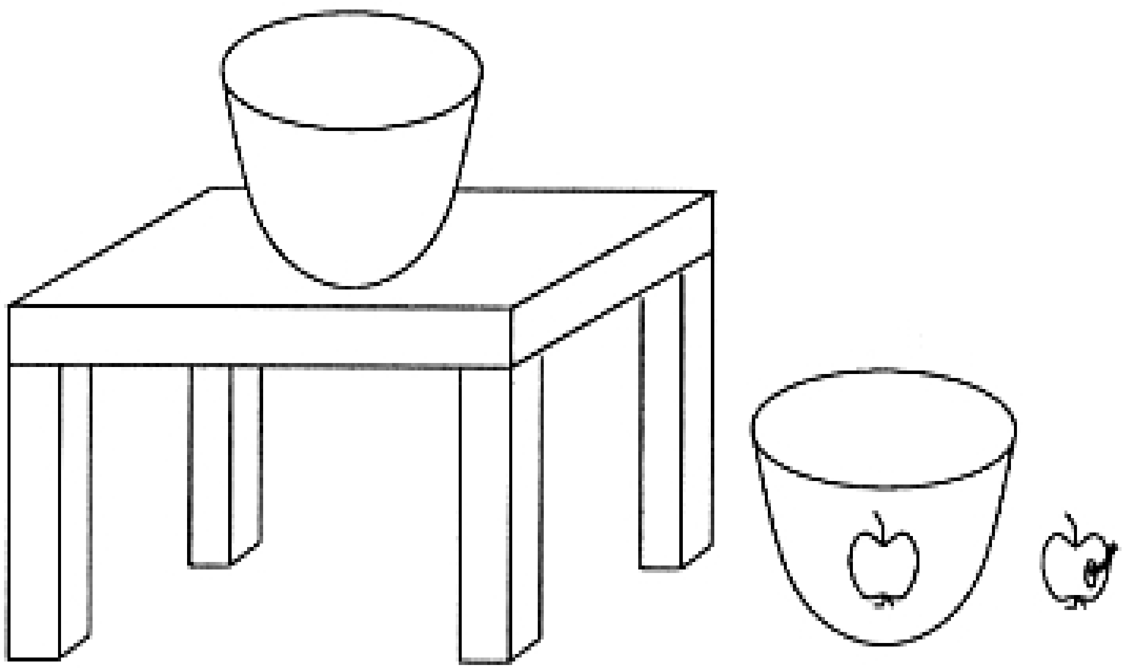
Textadventure mit DL

- A. Koller et al. (2001)



Textadventure mit DL

- Modellierung der Spielwelt in DL



Textadventure mit DL

room(kitchen)

table(t1)

apple(a2)

red(a1)

bowl(b1)

has-location(t1,kitchen)

has-location(b2,kitchen)

has-location(a2,kitchen)

has-location(myself,kitchen)

player(myself)

apple(a1)

worm(w1)

green(a2)

bowl(b2)

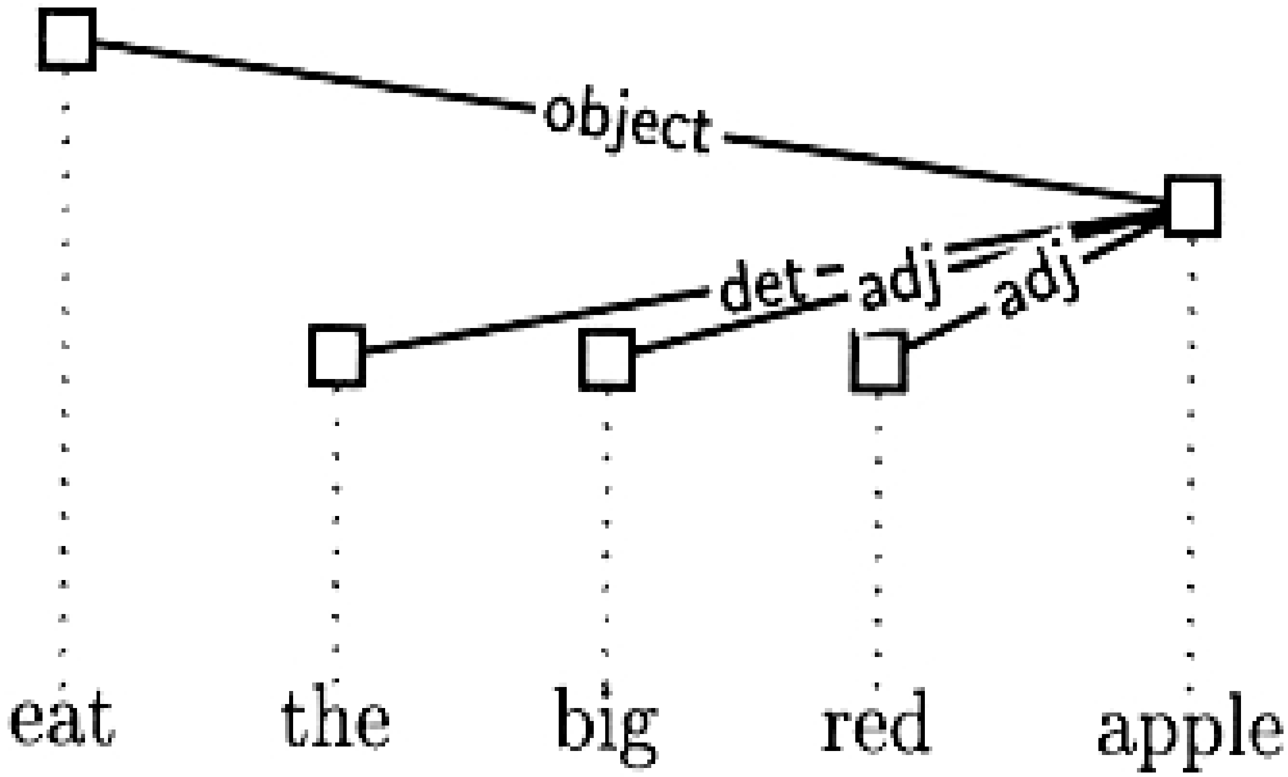
has-location(b1,t1)

has-location(a1,b2)

has-detail(a2,w1)

...

Textadventure mit DL



Textadventure mit DL

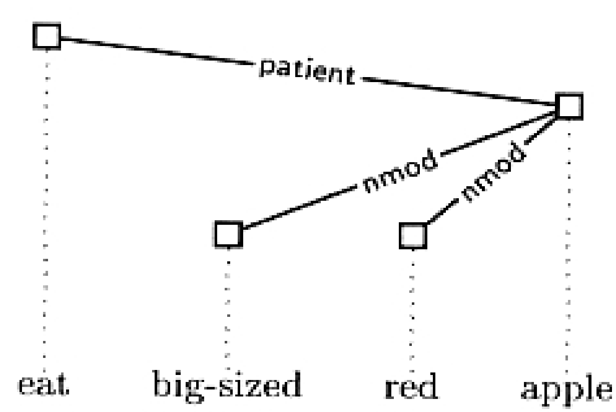
$$eat = \begin{bmatrix} \mathsf{in} & : & \{\} \\ \mathsf{out} & : & \{(\mathsf{subj}, ?), (\mathsf{obj}, !)\} \end{bmatrix}$$

$$the = \begin{bmatrix} \mathsf{in} & : & \{\mathsf{det}\} \\ \mathsf{out} & : & \{\} \end{bmatrix}$$

$$big, red = \begin{bmatrix} \mathsf{in} & : & \{\mathsf{adj}\} \\ \mathsf{out} & : & \{\} \end{bmatrix}$$

$$apple = \begin{bmatrix} \mathsf{in} & : & \{\mathsf{subj}, \mathsf{obj}\} \\ \mathsf{out} & : & \{(\mathsf{det}, ?), (\mathsf{adj}, *)\} \end{bmatrix}$$

FIG. 8. An Example Lexicon.



Textadventure mit DL

Semantik-Lexikon:

```
apple = [ sem:   ‘apple‘
          nmod: {adj} ]

eat(patient:[apple(agr:[unit(gender:[neut]
                             number:[sing]
                             spec:[def])]
                   nmod:[‘big-sized‘ red]
                   pos:[5])])
```

Textadventure mit DL

Referierende Ausdrücke:

- Semantische Repräsentation der Eingabe auf Individuen in der KB abbilden
- vereinfacht, da Spieler nur auf Gegenstände referiert, die er sieht
- definite NPs: finde eindeutiges Individuum, das auf Beschreibung des Spielers passt und das er sehen kann:

  the apple -> apple $\sqcap$ visible

- Frage A-Box nach allen Instanzen dieser Klasse

Textadventure mit DL

Referierende Ausdrücke:

- *the apple with the worm* -> apple $\sqcap$ has-detail.worm $\sqcap$ visible
- wenn nicht eindeutig, Rückfrage
- indefinite NPs: finde **irgendein** Individuum, das auf Beschreibung des Spielers passt und das er sehen kann:

  an apple -> apple $\sqcap$ visible
- Frage A-Box nach allen Instanzen dieser Klasse

Textadventure mit DL

Pronomen:

- *take the apple and eat **it**.*
- Diskurs-Modell
- Pronomen beziehen sich auf salientestes Individuum, das auf Kongruenz-Bedingungen passt
- Salienz:
  - hearer-old > hearer-new
  - innerhalb davon: Position

Textadventure mit DL

```
eat(patient:[apple(agr:[unit(gender:[neut]
                                   number:[sing]
                                   spec:[def])]
                         nmod:[‘big-sized‘ red]
                         pos:[5])])

-->

eat(patient:a2)
```