Semantic Web Modeling Languages Lecture II: OWL Basics

Markus Krötzsch & Sebastian Rudolph

ESSLLI 2009 Bordeaux

slides available at http://semantic-web-book.org/page/ESSLLI_2009:_Ontology_Modeling_Languages

Outline

- OWL: Overview and Rationale
- „Basic" OWL Syntax and Semantics
  - Axioms
  - Constructors
- Modeling
- Tableaux-Based Reasoning

OWL – Overview

- Web Ontology Language
  - W3C Recommendation for the Semantic Web, 2004
  - OWL 2 (revised W3C Recommendation) forthcoming in 2009
    - we already present this here
- Semantic Web KR language based on description logics (DLs)
  - OWL DL is essentially DL SROIQ(D)
  - KR for web resources, using URIs.
  - Using web-enabled syntaxes, e.g. based on XML or RDF
  - We present RDF Turtle syntax

Rationale behind OWL

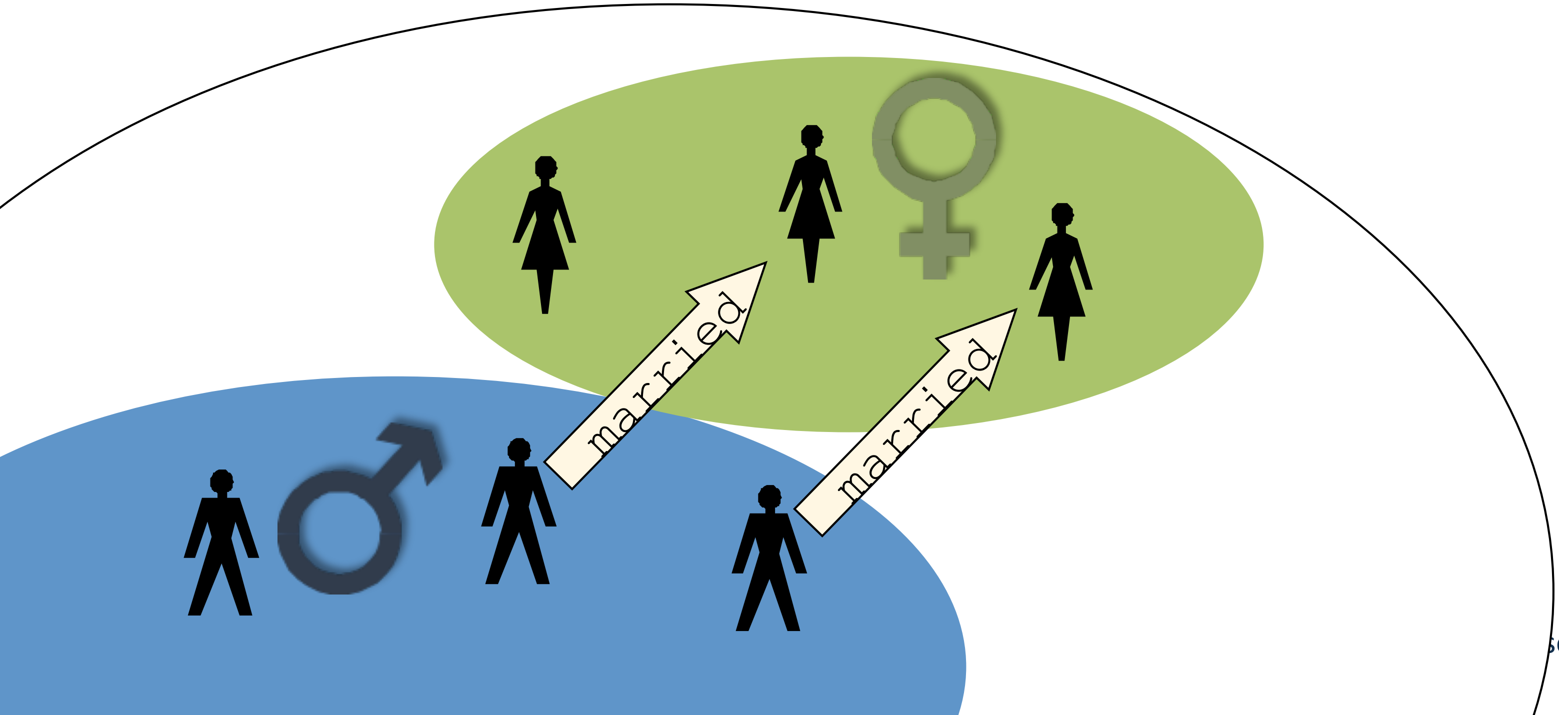
- Open World Assumption
- Favourable trade-off between expressivity and scalability
- Integrates with RDFS
- Purely declarative semantics

Features (for OWL 2 DL):

- Fragment of first-order predicate logic (FOL)
- Decidable
- Known complexity classes (N2ExpTime for OWL 2 DL)
- Reasonably efficient for real KBs

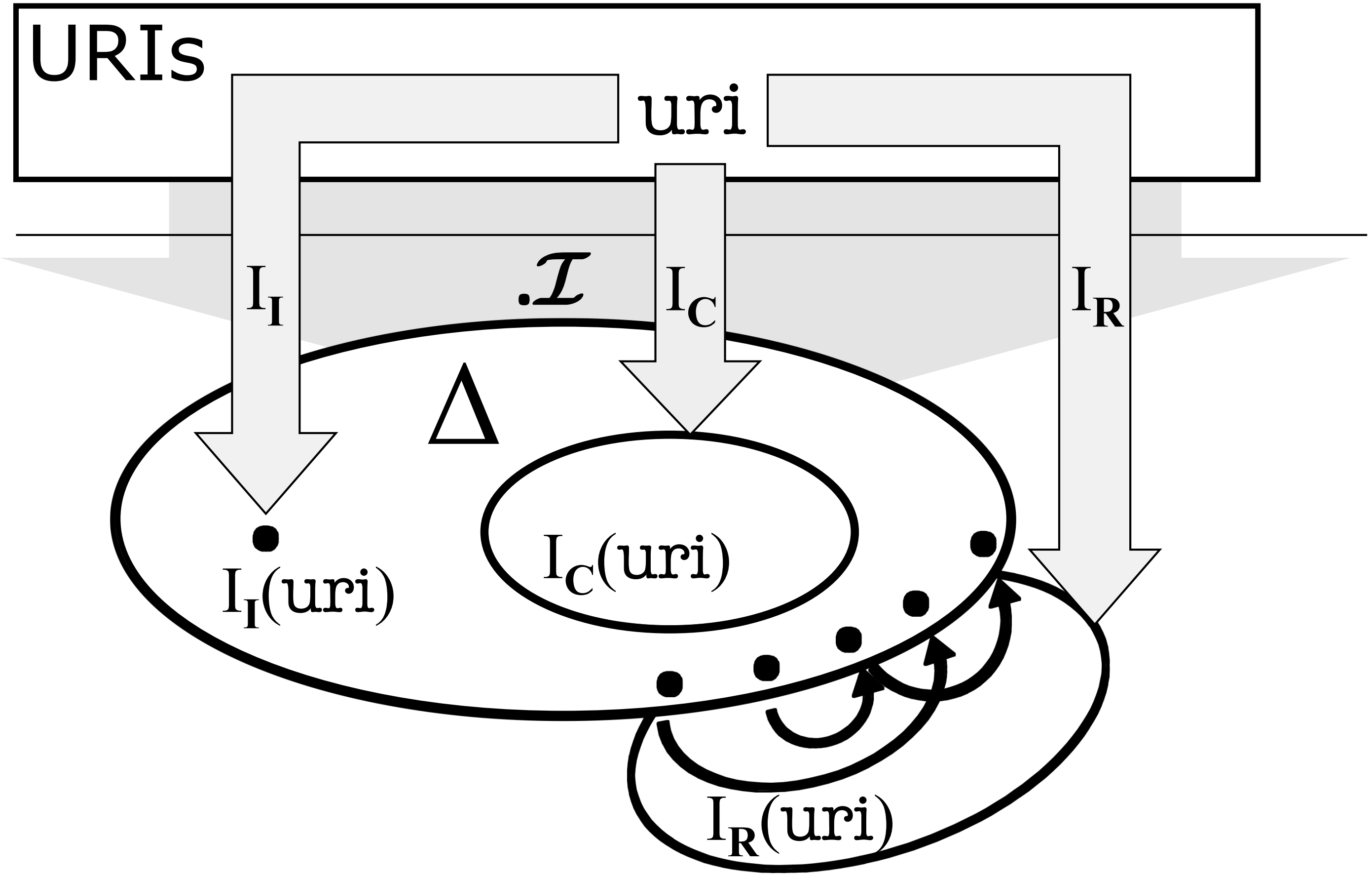
OWL Building Blocks

- individuals (written as URIs): `ex:markus`
  - aka: constants (FOL), ressources (RDF)
- classes (also written as URIs): `ex:Female`
  - aka: concepts, unary predicates (FOL)
- properties (also written as URIs): `ex:married`
  - aka: roles (DL), binary predicates (FOL)



OWL Direct Semantics

- model theory (aka extensional semantics)
- OWL DL Interpretation $\mathcal{I}$:



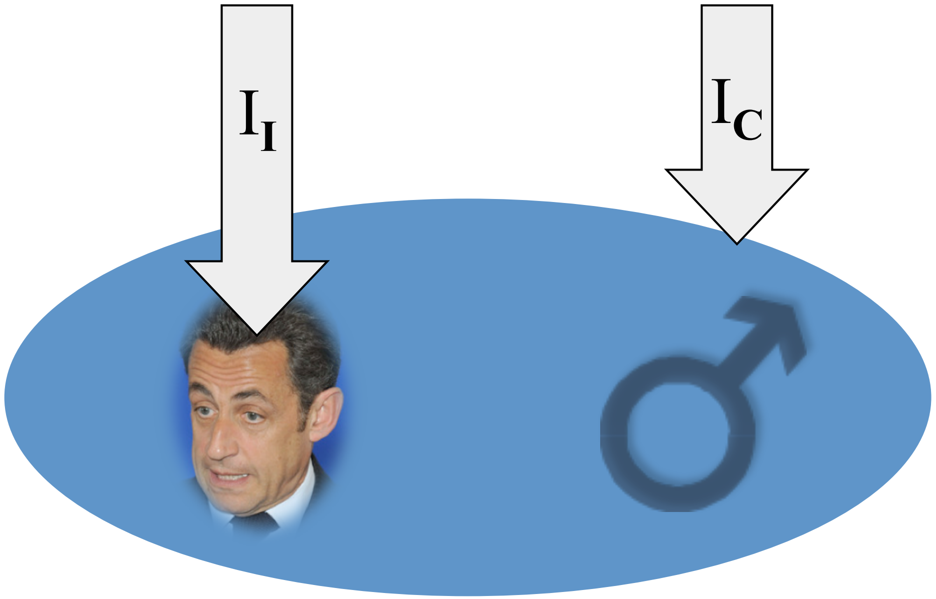
On the OWL Syntax

- OWL statements are written down as (sets of) RDF triples
- OWL facts (aka: assertions) are written down like in RDF
- some RDF language elements are reused
- new language elements from the OWL namespace
- more complex statements are constructed by using bnodes (we “hide“ them for convenience)

Class Membership

- induri rdf:type classuri .
- true in $\mathcal{I}$, if $I_I(\text{induri}) \in I_C(\text{classuri})$
- Example:

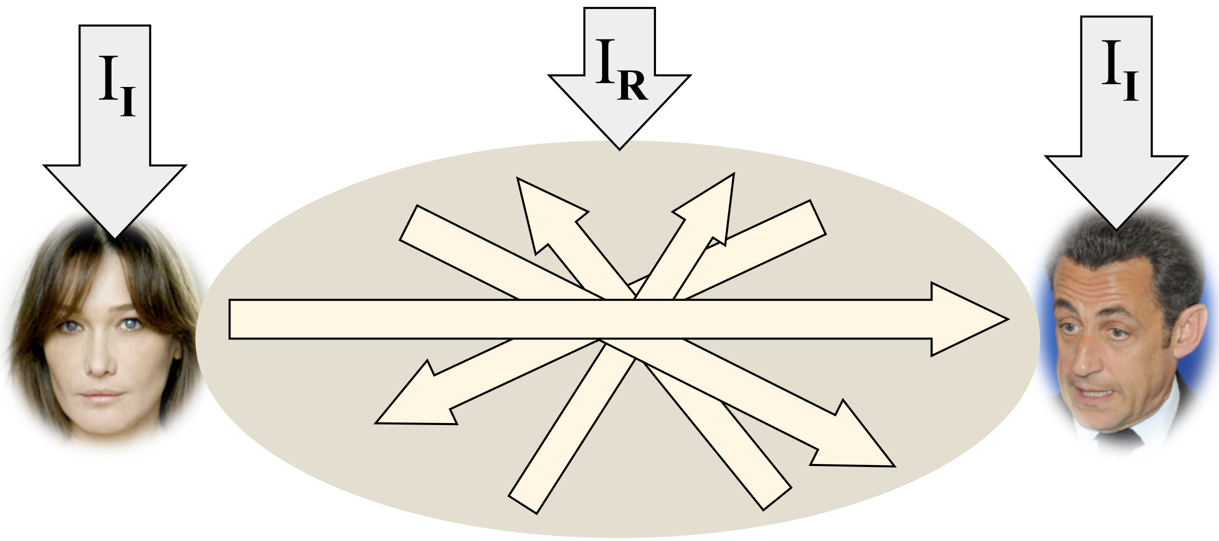
ex:nicolas rdf:type ex:Male



Property Membership

- induri1 propuri induri2 .
- true in $\mathcal{I}$, if
  $\langle I_I(\text{induri1}), I_I(\text{induri2})\rangle \in I_P(\text{propuri})$
- Example:

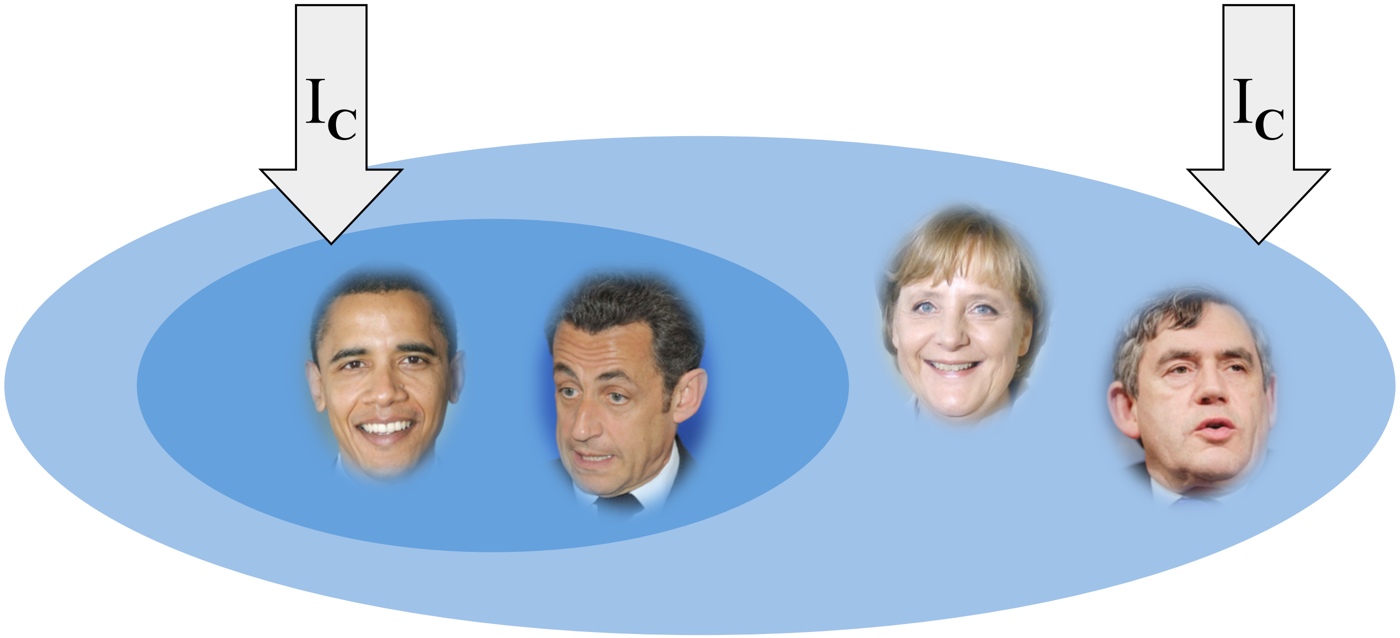
  ex:carla ex:marriedWith ex:nicolas



Class Inclusion

- classuri1 rdfs:subClassOf classuri2 .
- true in $\mathcal{I}$, if $\mathrm{I}_C(\text{classuri1}) \subseteq \mathrm{I}_C(\text{classuri2})$
- Example:

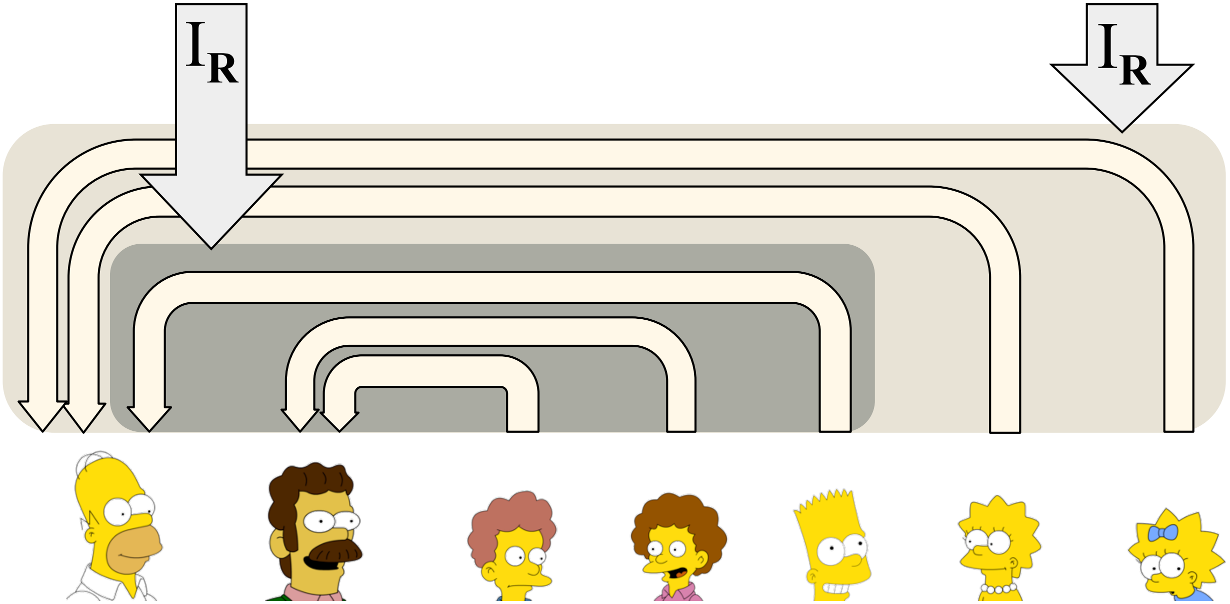
  ex:President rdfs:subClassOf ex:Politician



Property Inclusion

- propuri1 rdfs:subPropertyOf propuri2 .
- true in $\mathcal{I}$, if $\mathrm{I_R}$(propuri1) $\subseteq$ $\mathrm{I_R}$(propuri2)
- Example:

  ex:sonOf rdfs:subPropertyOf ex:childOf



Predefined Classes & Properties

- owl:Thing – the class containing everything
  - $I_C(\text{owl:Thing}) = \Delta$

- owl:Nothing – the empty class
  - $I_C(\text{owl:Nothing}) = \emptyset$

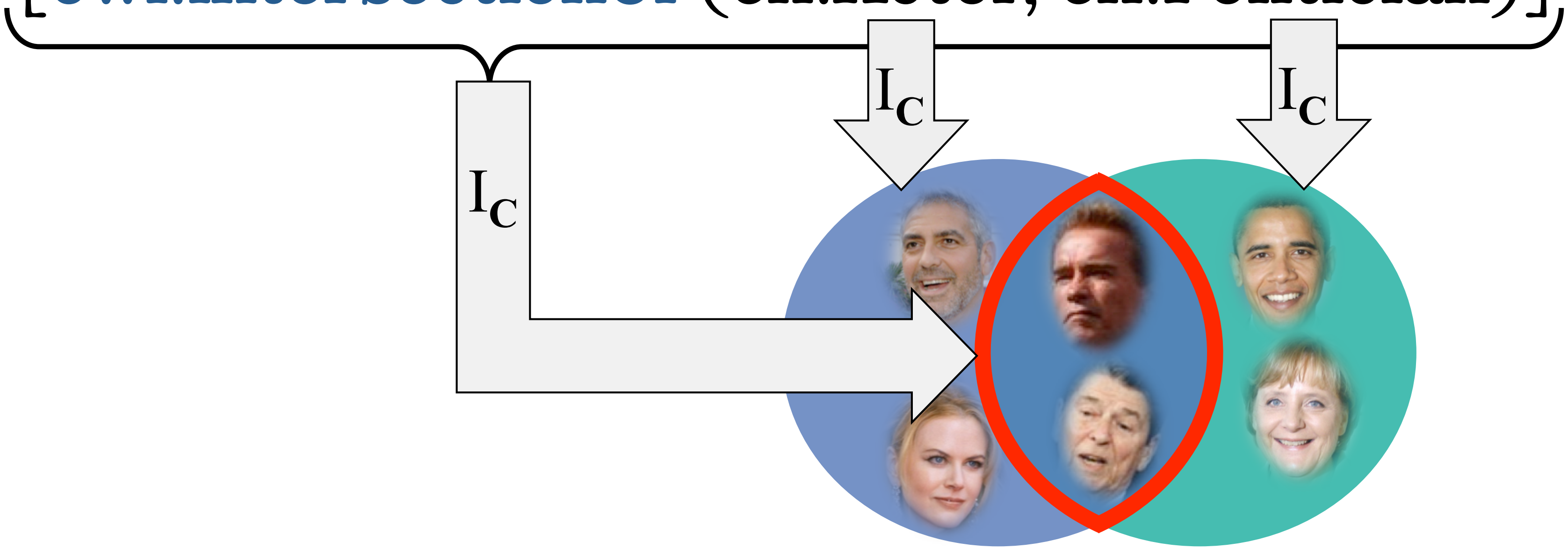
- owl:topProperty – the property connecting everything
  - $I_R(\text{owl:topProperty}) = \Delta \times \Delta$

- owl:bottomProperty – the empty property
  - $I_R(\text{owl:bottomProperty}) = \emptyset$

Complex Classes: Intersection

- [owl:intersectionOf (*class1* , … , *classn*)]
- I_C([owl:intersectionOf (*class1* , … , *classn*)]) $= I_C(\mathit{class1}) \cap \ldots \cap I_C(\mathit{classn})$
- Example:

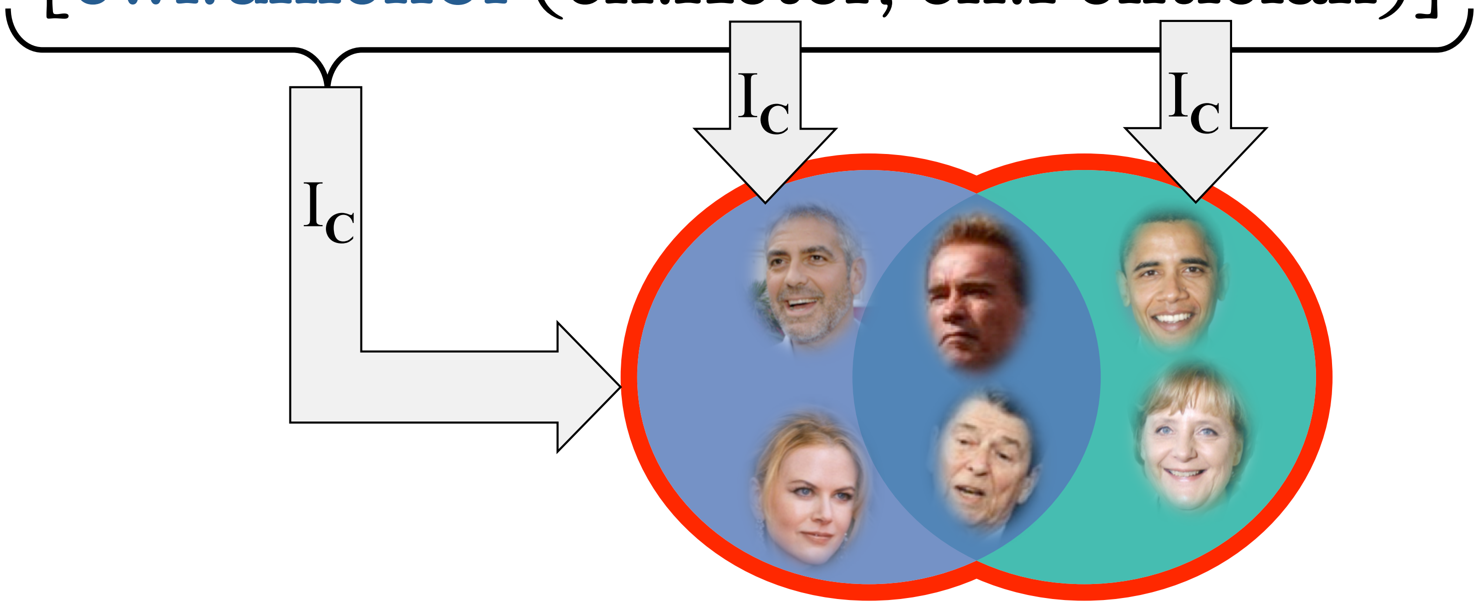
[owl:intersectionOf (ex:Actor, ex:Politician)]



Complex Classes: Union

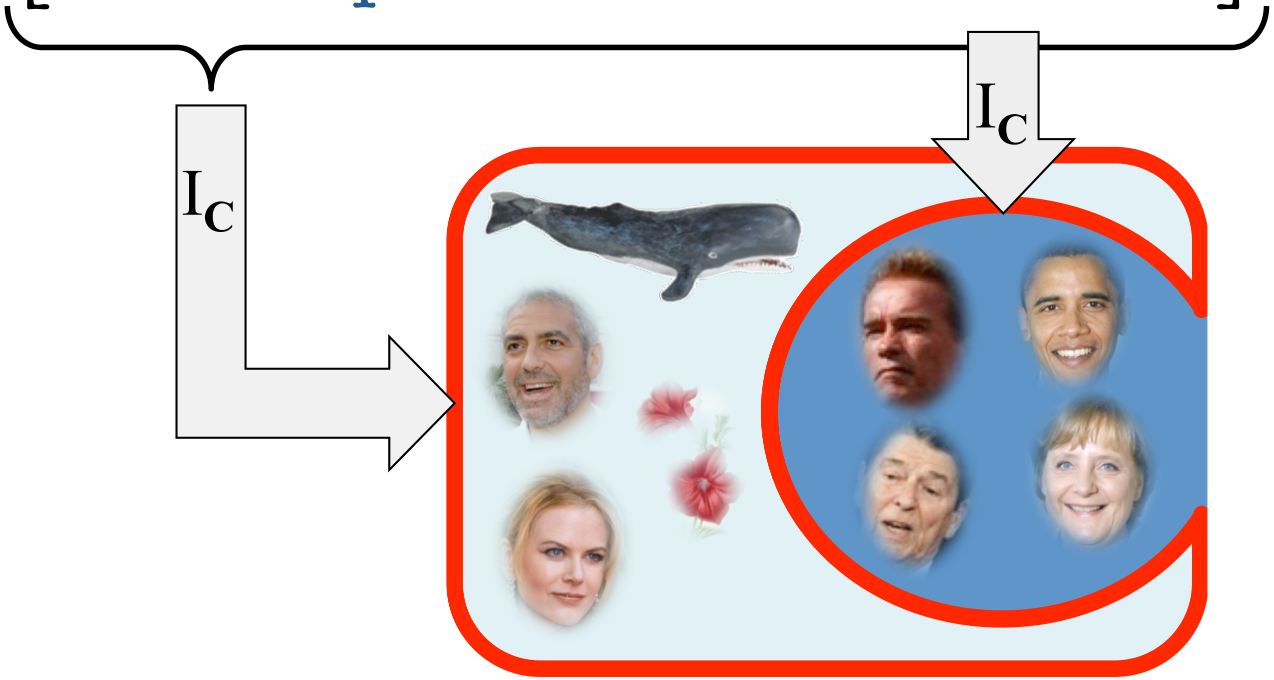
- [owl:unionOf (*class1* , … , *classn*)]
- I_C([owl:unionOf (*class1* , … , *classn*)])
  $= I_C(\mathit{class1}) \cup \ldots \cup I_C(\mathit{classn})$
- Example:

  [owl:unionOf (ex:Actor, ex:Politician)]



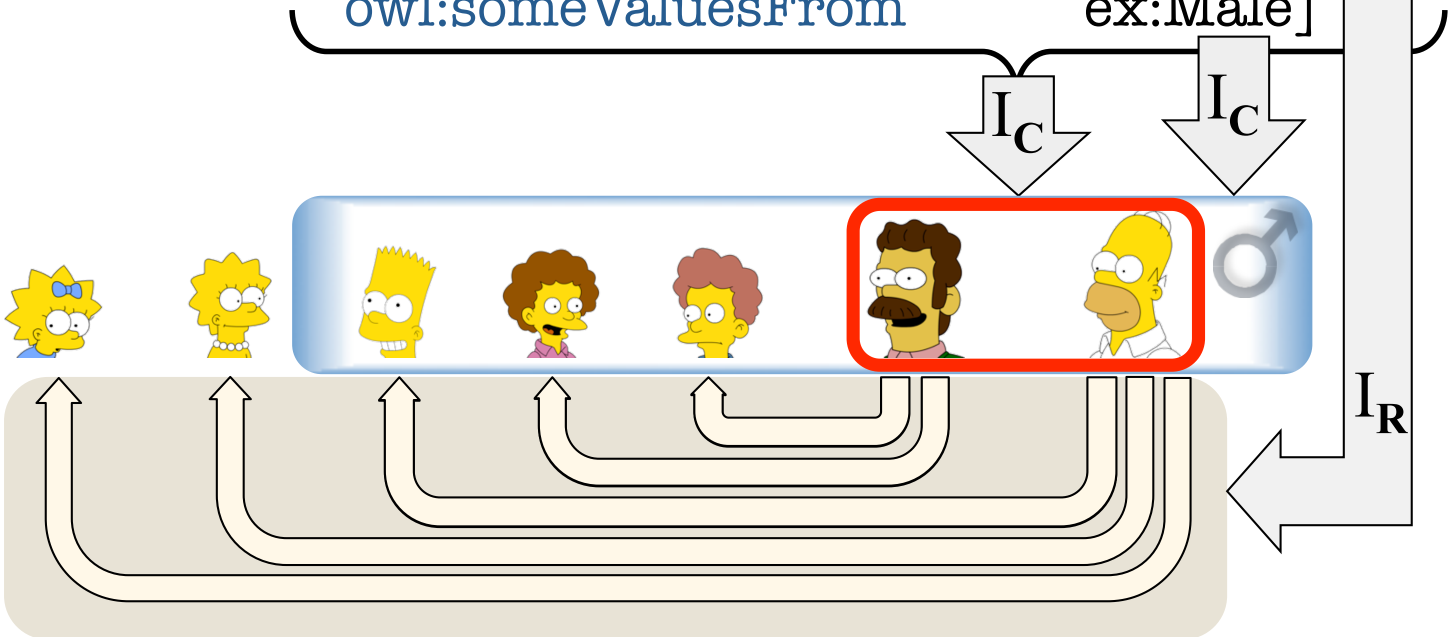
Complex Classes: Complement

- [owl:complementOf *class*]
- I_C([owl:complementOf *class*]) = Δ - I_C(*class*)
- Example:
  [owl:complementOf ex:Politician]



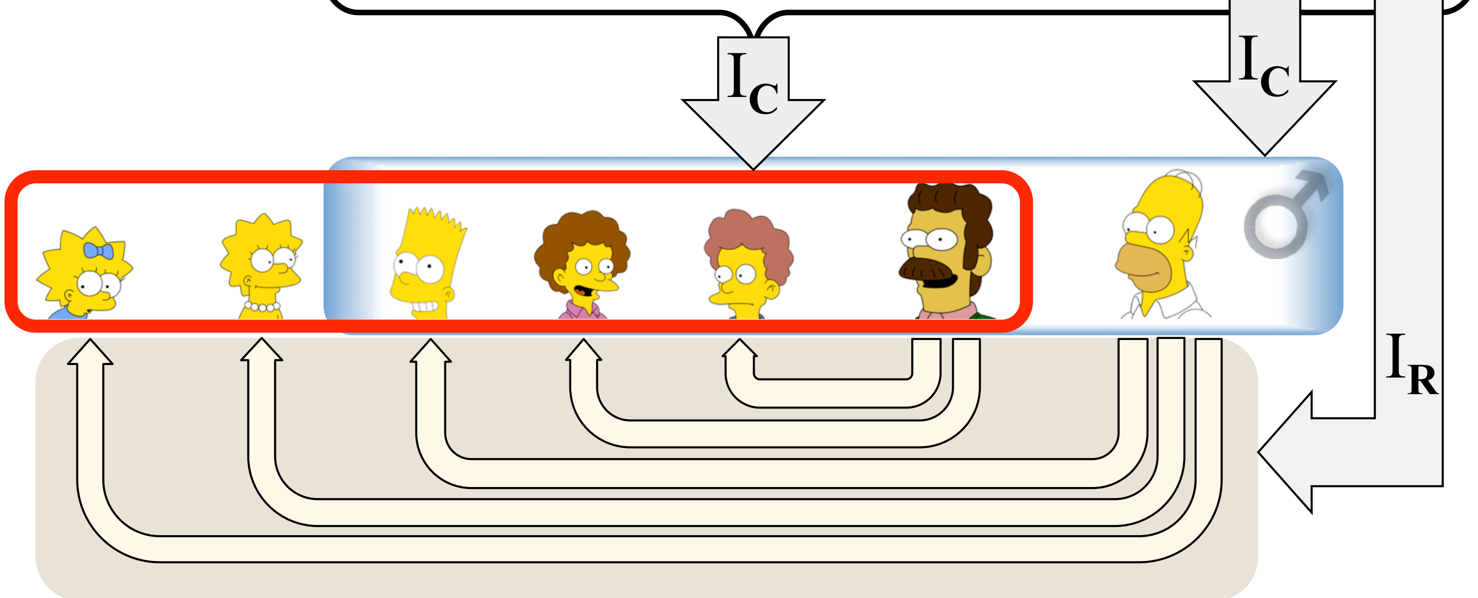
Complex Classes: Existential Property Restriction

- [rdf:type owl:Restriction ;
  owl:onProperty *prop* ;
  owl:someValuesFrom *class*]
- $I_C(\ldots)=\{x \mid \langle x,y\rangle \in I_R(\mathit{prop}) \text{ for some } y \in I_C(\mathit{class})\}$
- Example: [rdf:type owl:Restriction ;
  owl:onProperty ex:parentOf ;
  owl:someValuesFrom ex:Male]



Complex Classes: Universal Property Restriction

- [rdf:type owl:Restriction ;
  owl:onProperty *prop* ;
  owl:allValuesFrom *class*]
- $I_C(\ldots)=\{x \mid \langle x,y\rangle \in I_R(\textit{prop}) \text{ implies } y \in I_C(\textit{class})\}$
- Example: [rdf:type owl:Restriction ;
  owl:onProperty ex:parentOf ;
  owl:allValuesFrom ex:Male]



Alternative Semantics: Translation into FOL

trans(induri rdf:type *class* .)
  = trans_x(*class*)[x/induri]

trans(induri1 propuri induri2 .)
  = propuri (induri1, induri2)

trans(*class1* rdfs:subClassOf *class2* .)
  = $\forall$x: trans_x(*class1*) $\rightarrow$ trans_x(*class2*)

trans(propuri1 rdfs:subPropertyOf propuri2 .)
  = $\forall$x,y: propuri1(x,y) $\rightarrow$ propuri2(x,y)

Alternative Semantics: Translation into FOL

trans_x(owl:Thing) = true $\quad\quad$ trans_x(owl:Nothing) = false

trans_x(classuri) = classuri(x)

trans_x([owl:intersectionOf (*class1* , … , *classn*)]) = trans_x(*class1*) ∧ … ∧ trans_x(*classn*)

trans_x([owl:unionOf (*class1* , … , *classn*)]) = trans_x(*class1*) ∨ … ∨ trans_x(*classn*)

trans_x([owl:complementOf *class*]) = ¬trans_x(*class*)

trans_x([rdf:type owl:Restriction ; owl:onProperty propuri ; owl:someValuesFrom *class*])
= ∃y: propuri(x,y) ∧ trans_y(*class*)

trans_x([rdf:type owl:Restriction ; owl:onProperty propuri ; owl:someValuesFrom *class*])
= ∀y: propuri(x,y) → trans_y(*class*)

Syntactic Sugar: Disjointness, Domain & Range Statements

class1 owl:disjointWith *class2* .

- same as:
  [owl:intersectionOf (*class1* , *class2*)]
  rdfs:subClassOf owl:Nothing .

propuri rdf:domain *class* .

- same as:
  [rdf:type owl:Restriction ;
  owl:onProperty propuri ;
  owl:someValuesFrom owl:Thing] rdfs:subClassOf *class* .

propuri rdf:range *class* .

- same as:
  owl:Thing rdfs:subClassOf [rdf:type owl:Restriction ;
  owl:onProperty propuri ;
  owl:allValuesFrom *class*] .

A simple modeling example

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .

Healthy beings are not dead.

ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .

Every cat is alive or dead.

ex:owns rdfs:subPropertyOf ex:caresFor .

If somebody owns something, (s)he cares for it.

```
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [ rdf:type owl:Restriction ;  owl:onProperty ex:owns ;
                                  owl:someValuesFrom ex:Cat],
    [ rdf:type owl:Restriction ;  owl:onProperty ex:caresFor ;
                                  owl:someValuesFrom ex:Healthy]
    ) ] .
```

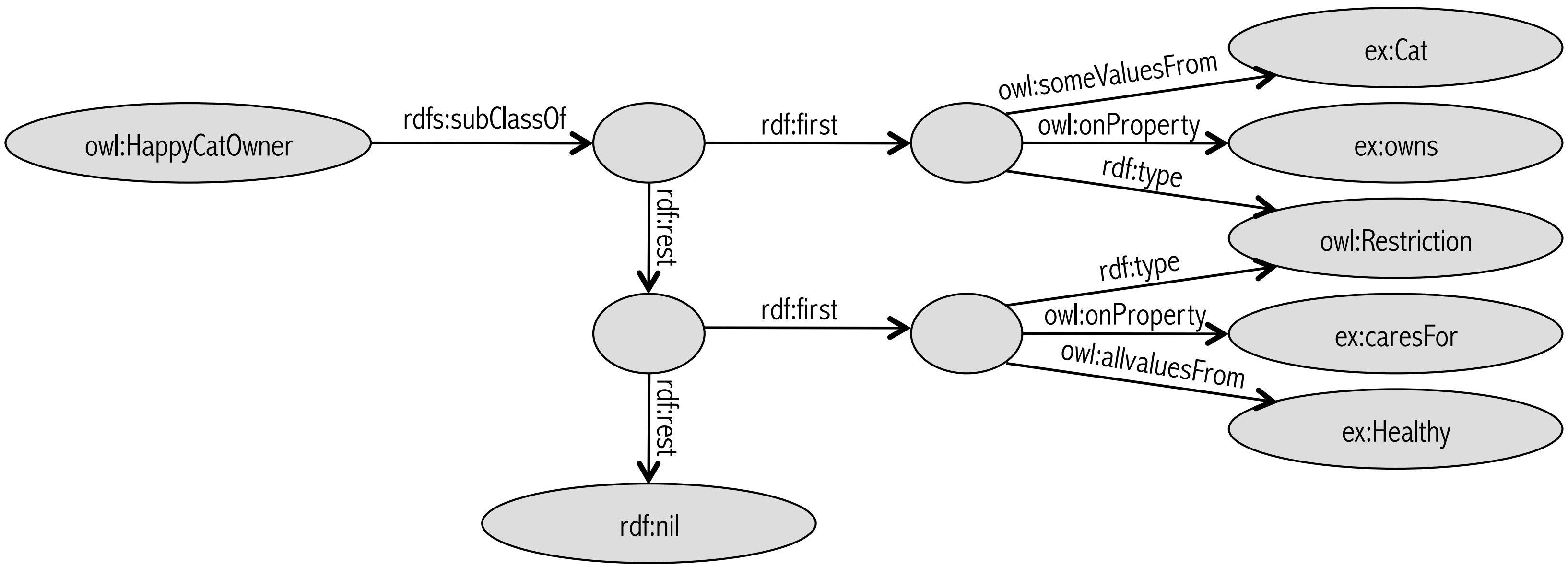
A happy cat owner owns a cat and all beings he cares for are healthy.

ex:schrödinger rdf:type ex:HappyCatOwner .

Schrödinger is a happy cat owner.

Behind the scenes...

- ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
  [rdf:type owl:Restriction ; owl:onProperty ex:owns ;
  owl:someValuesFrom ex:Cat],
  [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ;
  owl:someValuesFrom ex:Healthy]
)] .



Typical Inference Problems

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

- whether the knowledge in KB is consistent,
- whether KB entails a class membership (e.g. ex:schrödinger rdf:type ex:Alive .),
- whether a class is (un)satisfiable (e.g. [owl:intersectionOf (ex:Dead , ex:Alive)]),
- whether KB entails a subclass statement (e.g. ex:Alive rdfs:subClassOf ex:Healthy .),
- etc.

Reducing Inference Problems

- Many inference problems can be reduced to knowledge base consistency checking.
- Technique: claim the opposite and look what happens...

- **Class membership:**
  KB entails

  ex:schrödinger rdf:type ex:Alive .

  iff adding

  ex:schrödinger rdf:type [owl:complementOf ex:Alive].

  to KB makes it inconsistent.

Reducing Inference Problems

- Many inference problems can be reduced to knowledge base consistency checking.
- Technique: claim the opposite and look what happens...

- **Class (un)satisfiability:**
  KB entails unsatisfiability of
  [owl:intersectionOf (ex:Dead , ex:Alive)]
  iff adding
  ex:n rdf:type [owl:intersectionOf (ex:Dead , ex:Alive)].
  to KB makes it inconsistent.

Reducing Inference Problems

- Many inference problems can be reduced to knowledge base consistency checking.
- Technique: claim the opposite and look what happens...

- **Subclass entailment:**
  KB entails

  ex:Alive rdfs:subClassOf ex:Healthy .

  iff adding

  ex:n rdf:type [owl:intersectionOf
  (ex:Alive , [owl:complementOf ex:Healthy])].

  to KB makes it inconsistent.

Reasoning in OWL

- But how to determine whether a KB is consistent?
- One option: translate to FOL and use standard methods.
- But: OWL is decidable while FOL isn‘t.
- Still: FOL inferencing techniques (tableaux, resolution, type elimination) can be turned into decision procedures for OWL.

OWL Reasoning with Tableaux

- Tableaux methods are most frequent.
- Basic idea: try to build a model of the given KB. If this fails, the KB is inconsistent, otherwise consistent.
- Warning! The following example is simplified for better presentation (but demonstrates the essential features of tableaux-based methods). Consult the literature for a comprehensive treatment.

OWL Reasoning with Tableaux

Knowledge Base

```
ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [ rdf:type owl:Restriction ; owl:onProperty ex:owns ;      owl:someValuesFrom ex:Cat],
    [ rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy]) ] .
ex:schrödinger rdf:type ex:HappyCatOwner .
```

Tableau

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau

ex:HappyCatOwner

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau

ex:HappyCatOwner
[owl:intersectionOf (,)]

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .

ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .

ex:owns rdfs:subPropertyOf ex:caresFor .

ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .

ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau

ex:HappyCatOwner

[owl:intersectionOf (,)]

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .

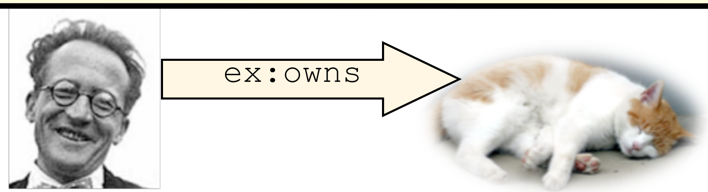
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .

ex:owns rdfs:subPropertyOf ex:caresFor .

ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .

ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



ex:HappyCatOwner
[owl:intersectionOf (,)]

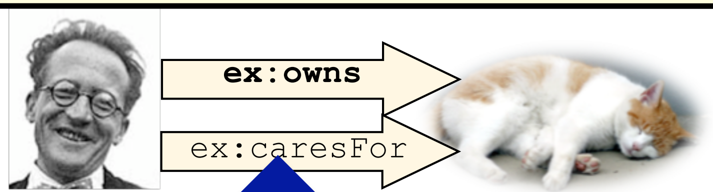
ex:Cat

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



ex:HappyCatOwner ex:Cat
[owl:intersectionOf (,)]

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau

ex:HappyCatOwner
[owl:intersectionOf (,)]

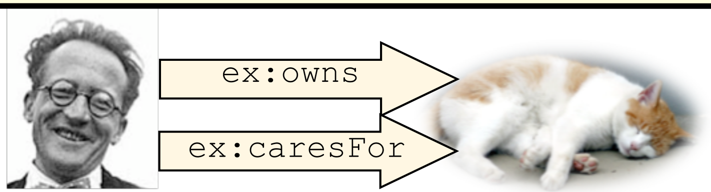
ex:Cat ex:Healthy

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



ex:HappyCatOwner
[owl:intersectionOf (,)]

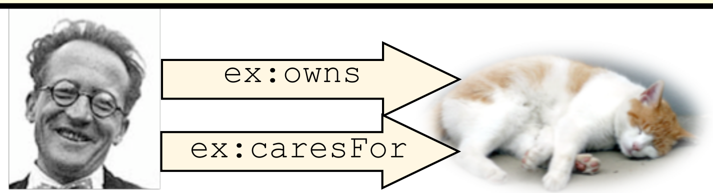
ex:Cat ex:Healthy
[owl:unionOf
(ex:Dead, ex:Alive)]

OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



ex:HappyCatOwner
[owl:intersectionOf (,)]

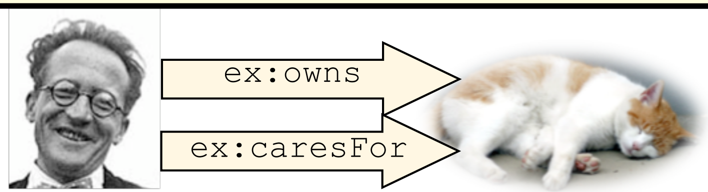
ex:Cat ex:Healthy
[owl:unionOf
(ex:Dead, ex:Alive)]

OWL Reasoning with Tableaux

Knowledge Base

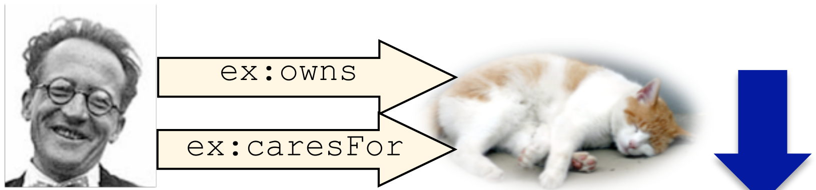
ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



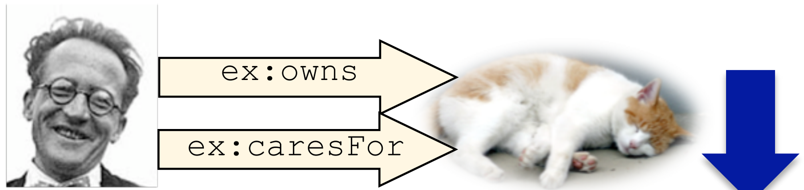
ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy
[owl:unionOf (ex:Dead, ex:Alive)]



ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy ex:Dead
[owl:unionOf (ex:Dead, ex:Alive)]



ex:HappyCatOwner
[owl:intersectionOf (,)]

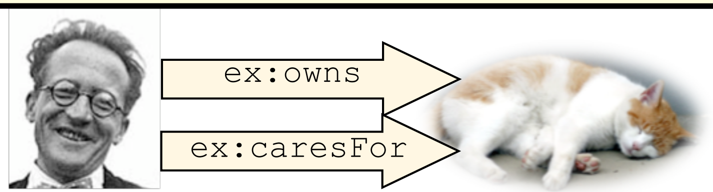
ex:Cat ex:Healthy ex:Alive
[owl:unionOf (ex:Dead, ex:Alive)]

OWL Reasoning with Tableaux

Knowledge Base

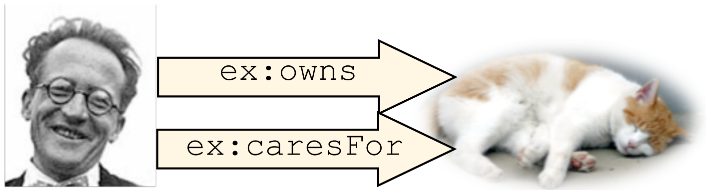
ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



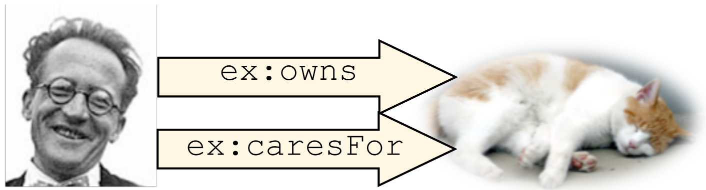
ex:HappyCatOwner
[owl:intersectionOf (■, ■)]
■ ■

ex:Cat ex:Healthy
[owl:unionOf
(ex:Dead, ex:Alive)]



ex:HappyCatOwner
[owl:intersectionOf (■, ■)]
■ ■

ex:Cat **ex:Healthy** ex:Dead
[owl:unionOf
(ex:Dead, ex:Alive)]
[owl:complementOf ex:Dead]



ex:HappyCatOwner
[owl:intersectionOf (■, ■)]
■ ■

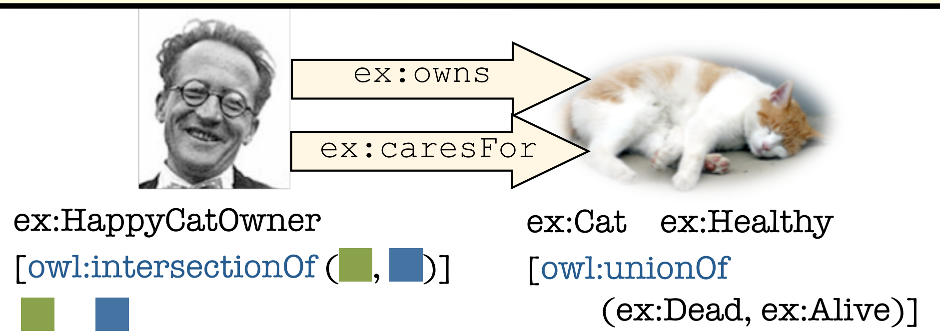
ex:Cat ex:Healthy ex:Alive
[owl:unionOf
(ex:Dead, ex:Alive)]

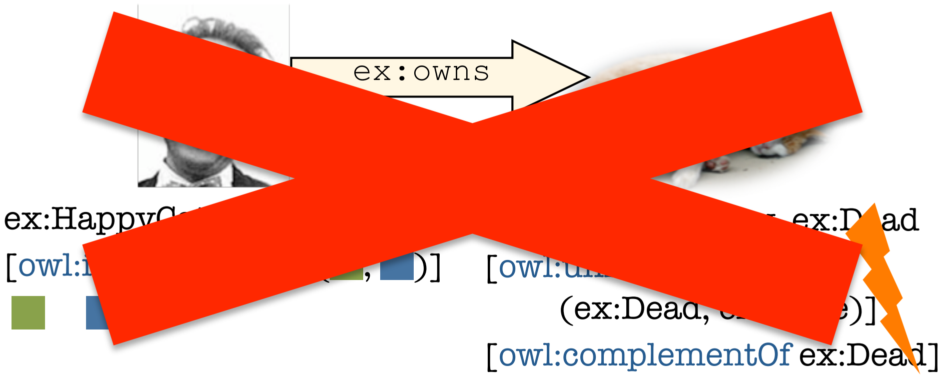
OWL Reasoning with Tableaux

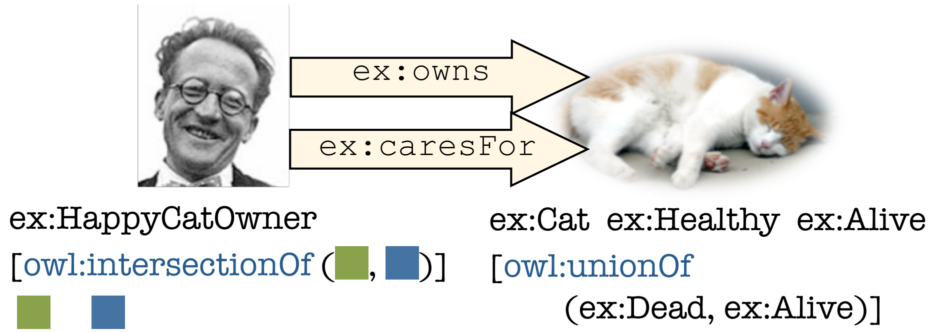
Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau





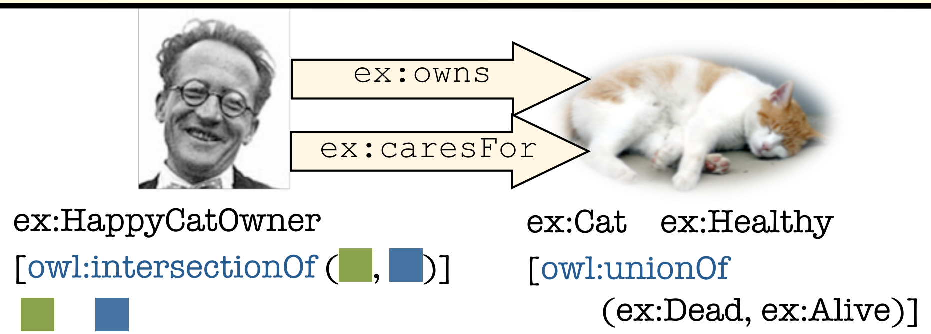


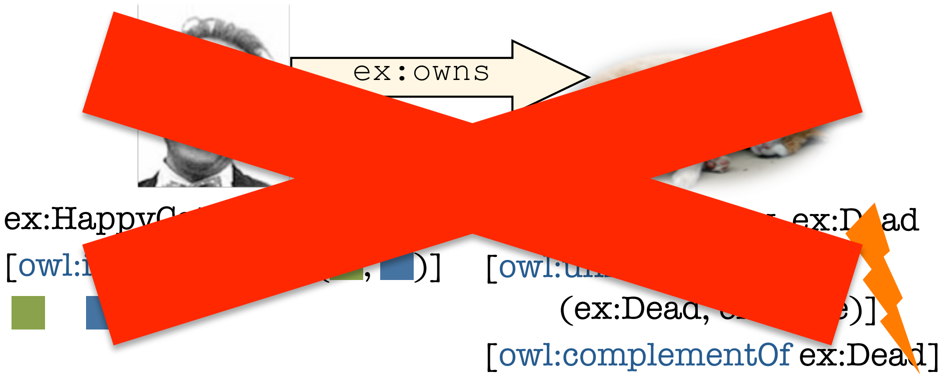
OWL Reasoning with Tableaux

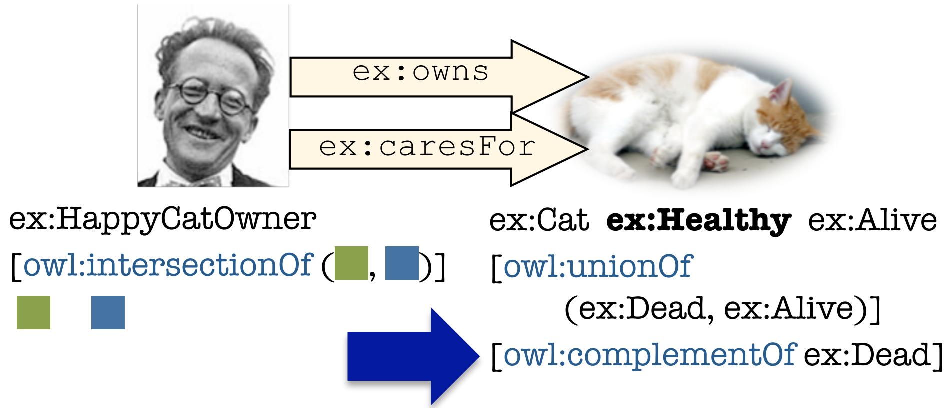
Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
    [rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
    [rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau







OWL Reasoning with Tableaux

Knowledge Base

ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .

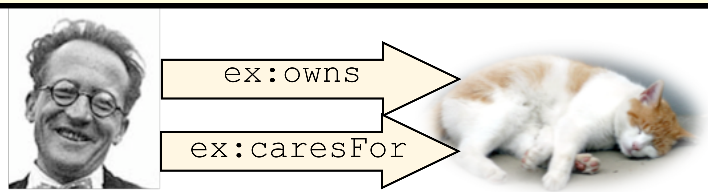
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .

ex:owns rdfs:subPropertyOf ex:caresFor .

ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .

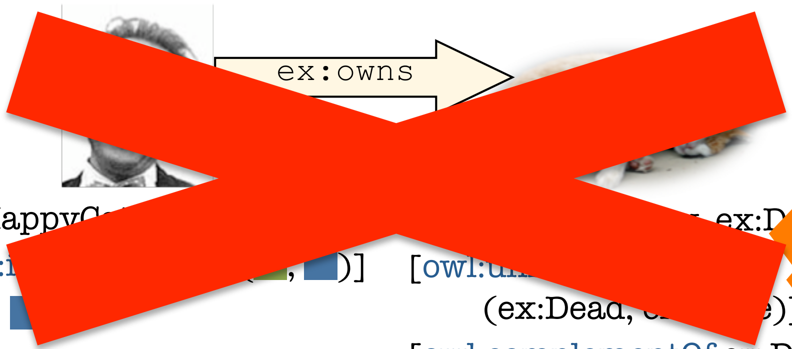
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau

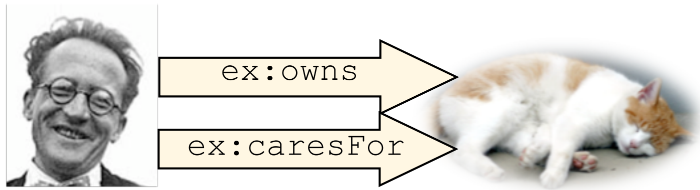


ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy
[owl:unionOf
(ex:Dead, ex:Alive)]



ex:HappyC… ex:D ad
[owl:i… ,)] [owl:u…
(ex:Dead, …)]
[owl:complementOf ex:Dead]



ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy ex:Alive
[owl:unionOf
(ex:Dead, ex:Alive)]
[owl:complementOf ex:Dead]

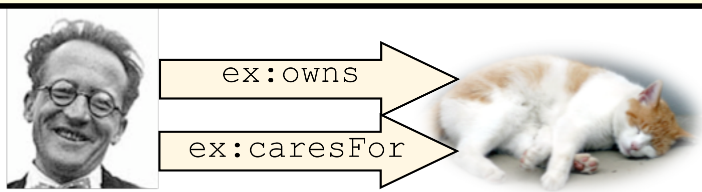
OWL Reasoning with Tableaux

Knowledge Base

Satisfiable

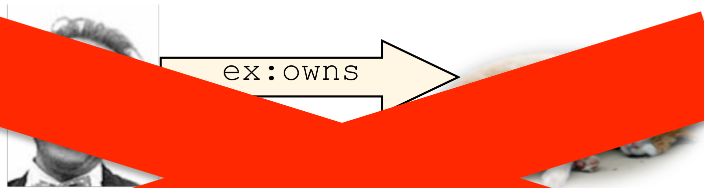
ex:Healthy rdfs:subClassOf [owl:complementOf ex:Dead] .
ex:Cat rdfs:subClassOf [owl:unionOf (ex:Dead, ex:Alive)] .
ex:owns rdfs:subPropertyOf ex:caresFor .
ex:HappyCatOwner rdfs:subClassOf [owl:intersectionOf (
[rdf:type owl:Restriction ; owl:onProperty ex:owns ; owl:someValuesFrom ex:Cat],
[rdf:type owl:Restriction ; owl:onProperty ex:caresFor ; owl:someValuesFrom ex:Healthy])] .
ex:schrödinger rdf:type ex:HappyCatOwner .

Tableau



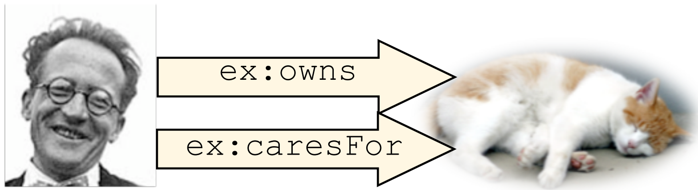
ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy
[owl:unionOf
(ex:Dead, ex:Alive)]



ex:HappyC
[owl:i ,)]

ex:D ad
[owl:u
(ex:Dead, e)]
[owl:complementOf ex:Dead]



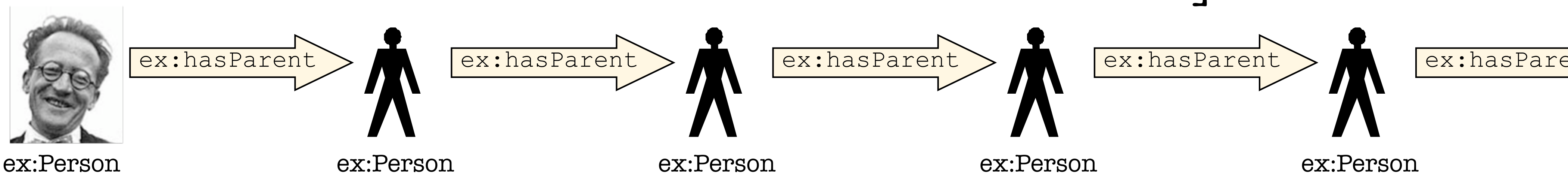
ex:HappyCatOwner
[owl:intersectionOf (,)]

ex:Cat ex:Healthy ex:Alive
[owl:unionOf
(ex:Dead, ex live)
[owl:complement Dead]

OWL Reasoning with Tableaux

- possible termination problem:

```
ex:schrödinger rdf:type ex:Person .
ex:Person rdfs:subClassOf
          [ rdf:type owl:Restriction ;
            owl:onProperty ex:hasParent ;
            owl:someValuesFrom ex:Person].
```



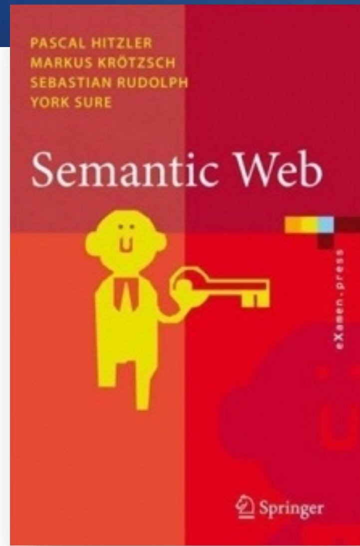
- can be solved by *blocking*

References

- W3C OWL Working Group, OWL 2 Web Ontology Language: Document Overview. http://www.w3.org/TR/owl2-overview/
- Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter Patel-Schneider, Sebastian Rudolph, OWL 2 Web Ontology Language: Primer. http://www.w3.org/TR/owl2-primer/
- Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider, The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2nd edition, 2007.

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/ (Ask for a flyer from us.)