Karlsruhe Institute of Technology

OWL 2 Rules (Part 1)

Tutorial at ESWC2009
May 31, 2009

Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Universität Karlsruhe (TH)
Research University • founded 1825

Pascal Hitzler 

Markus Krötzsch 

Sebastian Rudolph 

AIFB, Universität Karlsruhe (TH)

http://www.pascal-hitzler.de
http://korrekt.org
http://www.sebastian-rudolph.de

Outline Part 1

- The Early Days of KR: Rule-Based Formalisms
- OWL 2 DL – the new DL-based Web Ontology Language
- Semantics of OWL DL
- Tractable Fragments

The Early Days of KR: Rule-Based Formalisms

- rules provide a natural way of modelling “if-then“ knowledge
- general form of a (Horn) rule:

$$\text{Body} \rightarrow \text{Head}$$

- body: (possibly empty) conjunction of atoms, head: at most one atom
- Examples:

$$\text{married}(x,y) \wedge \text{Woman}(x) \rightarrow \text{Man}(y)$$

$$\text{Man}(x) \wedge \text{Woman}(x) \rightarrow$$

$$\rightarrow \text{married}(\text{pascal},\text{anne})$$

The Early Days of KR: Rule-Based Formalisms

- rules provide a natural way of modelling “if-then“ knowledge
- general form of a (Horn) rule:

$$\text{Body} \rightarrow \text{Head}$$

- body: (possibly empty) conjunction of atoms, head: at most one atom
- Examples:

$$\forall x \forall y (\text{married}(x,y) \wedge \text{Woman}(x) \rightarrow \text{Man}(y))$$

$$\forall x (\text{Man}(x) \wedge \text{Woman}(x) \rightarrow \text{false})$$

$$\text{true} \rightarrow \text{married}(\text{pascal},\text{anne})$$

On the Semantics of Rules

Karlsruhe Institute of Technology

- syntactically, rules are just FOL formulae
- hence they can be interpreted under FOL standard semantics
- other (non-monotonic) interpretations are possible:
  - well-founded semantics
  - stable model semantics
  - answer set semantics
- in the case of Horn rules, they all coincide (differences if negation of atoms is allowed)
- in this tutorial, we strictly adhere to FOL (=open-world) semantics

What We Cannot Say with Rules

Karlsruhe Institute of Technology

- with rules, one cannot require the existence of individuals with certain properties except by explicitly naming them
- i.e. we can express that there are two persons that are married by giving them names (say, person1 and person2):

true → married(person1,person2)

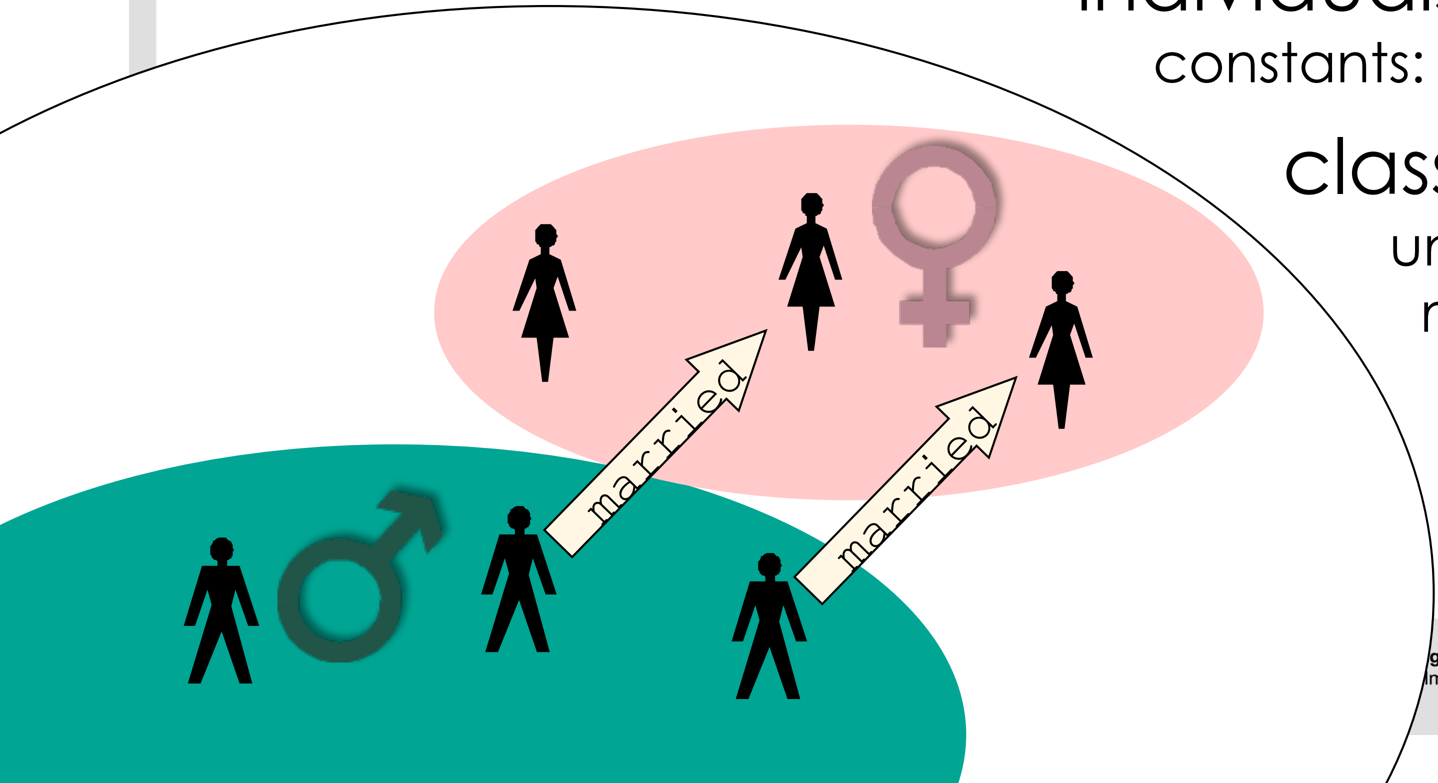
- but we cannot express something like:
  "every husband is married to somebody"

wrong:
husband(x) → married(x,person)

That's where OWL comes in!

What OWL Talks About (Semantics)

- both OWL 1 DL and OWL 2 DL are based on description logics
- here, we will treat OWL from the “description logic viewpoint“:
  - we use DL syntax
  - we won‘t talk about datatypes and non-semantic features of OWL
- OWL (DL) ontologies talk about worlds that contain



individuals

constants: pascal, anne

classes / concepts

unary predicates:
male(_), female(_)

properties / roles

binary predicates:
married(_,_)

Assertional Knowledge

- asserts information about concrete named individuals
  - class membership: Male(pascal)

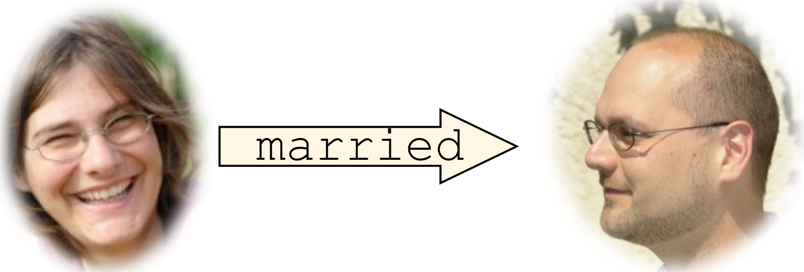
```
<Male rdf:about="pascal"/>
```

rule version: $\rightarrow$ Male(pascal)

  - property membership: married(anne,pascal)

```
<rdf:Description rdf:about="anne">
  <married rdf:resource="pascal"/>
</rdf:Description>
```

rule version: $\rightarrow$ married (anne,pascal)

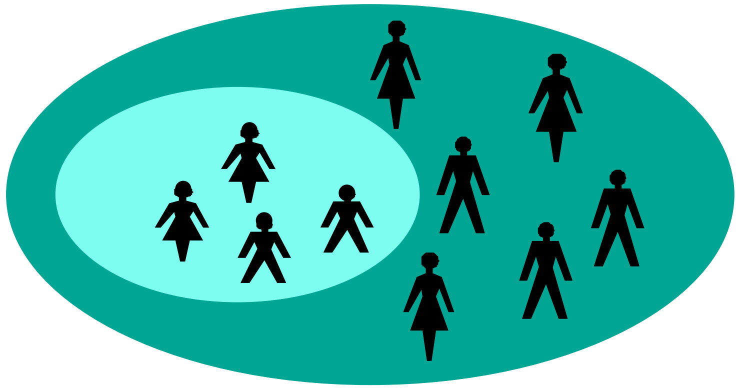


That's all what can be said with RDF!

Terminological Knowledge – Subclasses and Subproperties

- information about how classes and properties relate in general

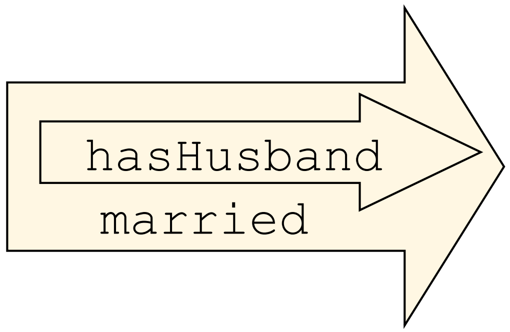
  - subclass: Child ⊑ Person



```
<owl:Class rdf:about="Child">
  <rdfs:subClassOf rdf:resource="Person"/>
</owl:Class>
```

rule version: Child(x) → Person(x)

  - subproperty: hasHusband ⊑ married




```
<owl:ObjectProperty rdf:about="hasHusband">
  <rdfs:subPropertyOf rdf:resource="married"/>
</owl:ObjectProperty>
```

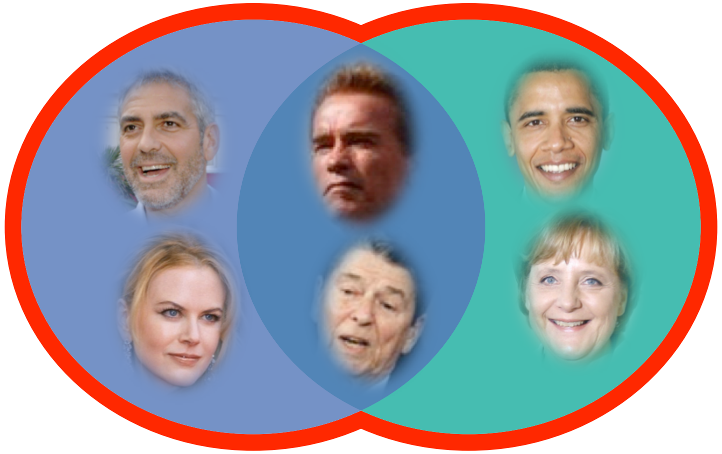
rule version: hasHusband(x,y) → married (x,y)

Class Constructors

Karlsruhe Institute of Technology

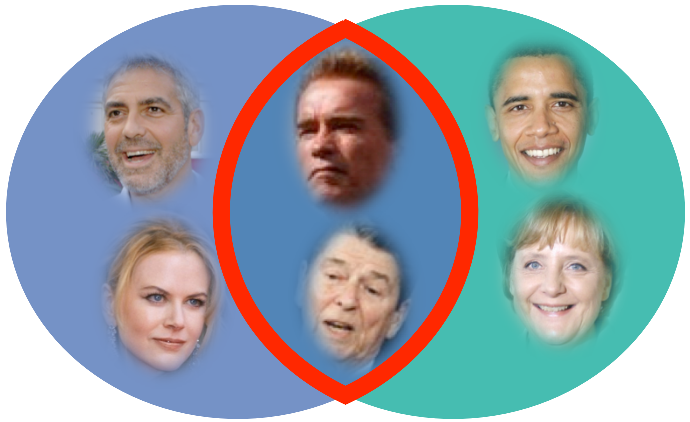
- build new classes from class, property and individual names
  - union: Actor ⊔ Politician

```
<owl:unionOf  rdf:parseType="Collection“>
  <owl:Class rdf:about="Actor"/>
  <owl:Class rdf:about="Politician"/>
</owl:unionOf>
```



  - intersection: Actor ⊓ Politician

```
<owl:intersectionOf  rdf:parseType="Collection“>
  <owl:Class rdf:about="Actor"/>
  <owl:Class rdf:about="Politician"/>
</owl:intersectionOf>
```



Class Constructors

- build new classes from class, property and individual names
  - complement: ¬Politician

```
<owl:complementOf
          rdf:resource="Politician">
```

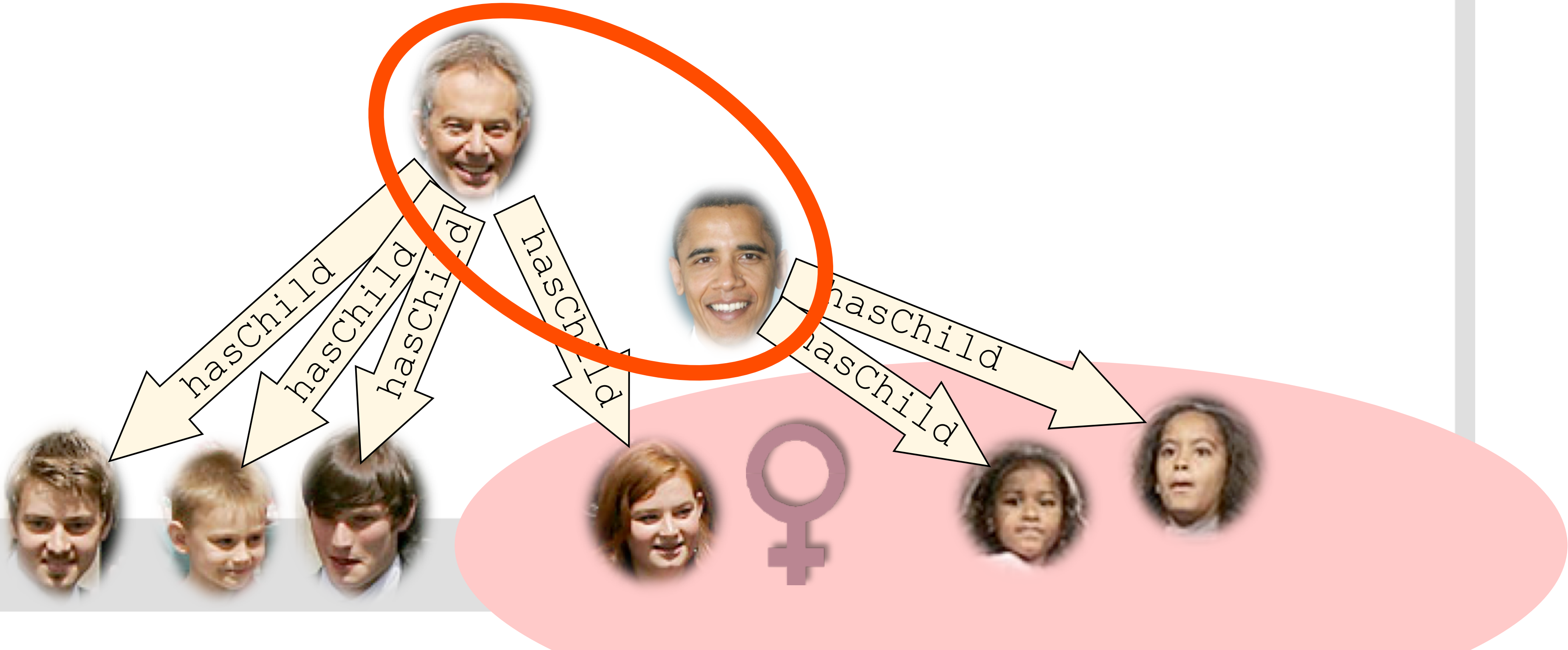
  - closed classes: {anne,merula,pascal}

```
<owl:oneOf rdf:parseType="Collection">
  <rdf:Description rdf:about="anne"/>
  <rdf:Description rdf:about="merula"/>
  <rdf:Description rdf:about="pascal"/>
</owl:oneOf>
```

Class Constructors

- build new classes from class, property and individual names
  - existential quantification: ∃hasChild.Female

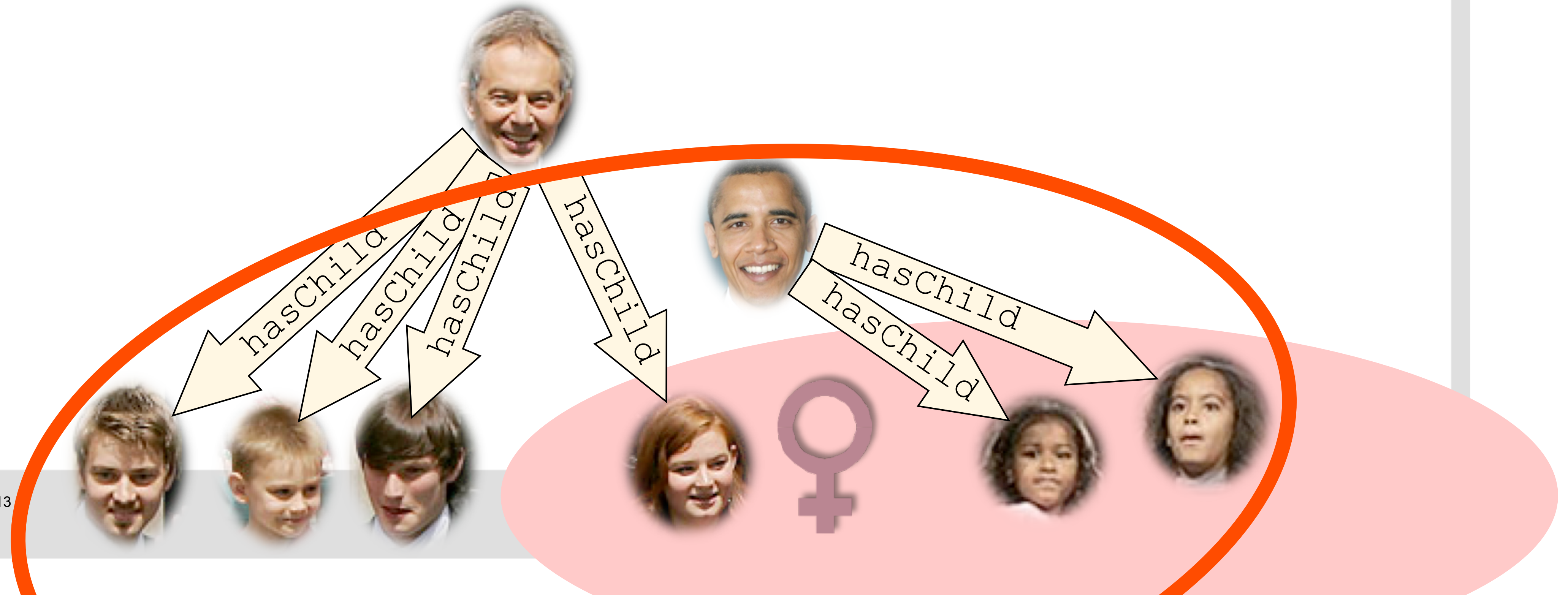
```
<owl:Restriction>
    <owl:onProperty rdf:resource="hasChild"/>
    <owl:someValuesFrom rdf:resource="Female"/>
</owl:Restriction>
```



Class Constructors

- build new classes from class, property and individual names
  - universal quantification: ∀hasChild.Female

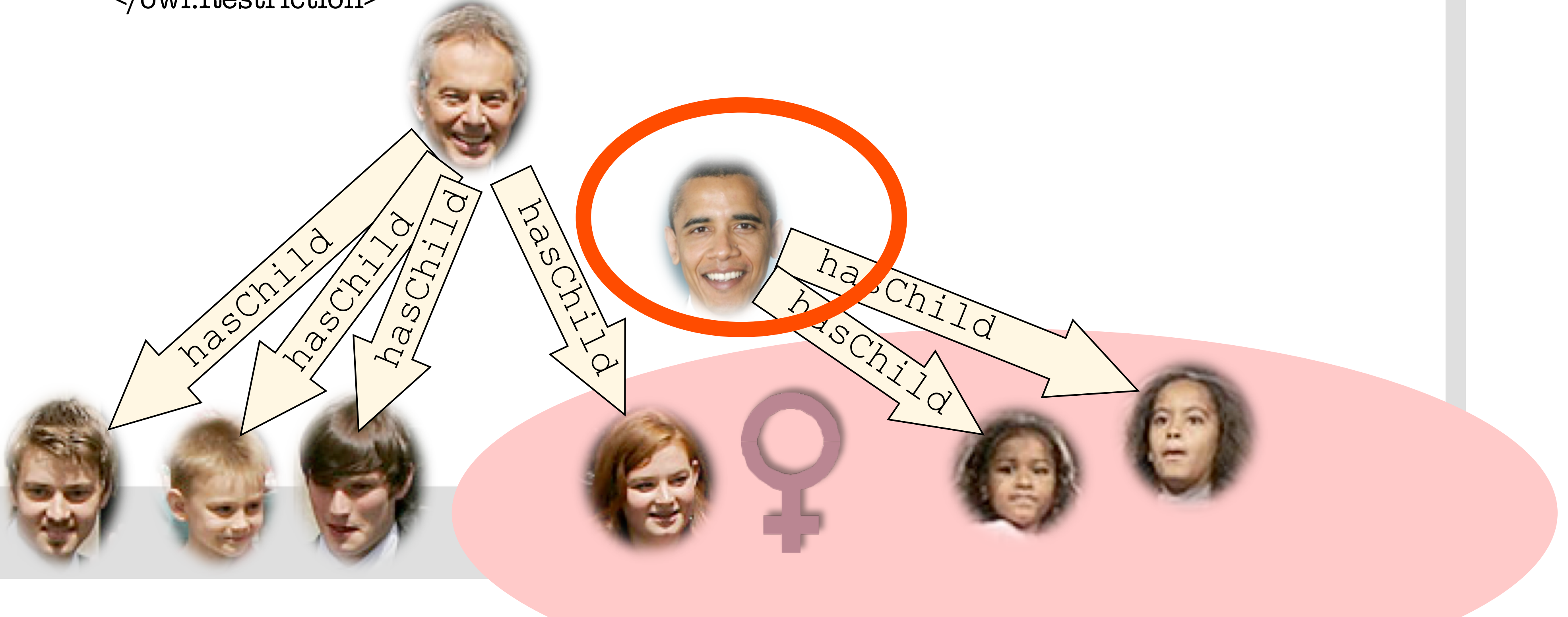
```
<owl:Restriction>
    <owl:onProperty rdf:resource="hasChild"/>
    <owl:allValuesFrom rdf:resource="Female"/>
</owl:Restriction>
```



Class Constructors

- build new classes from class, property and individual names
  - cardinality restriction: ≥2hasChild.Female

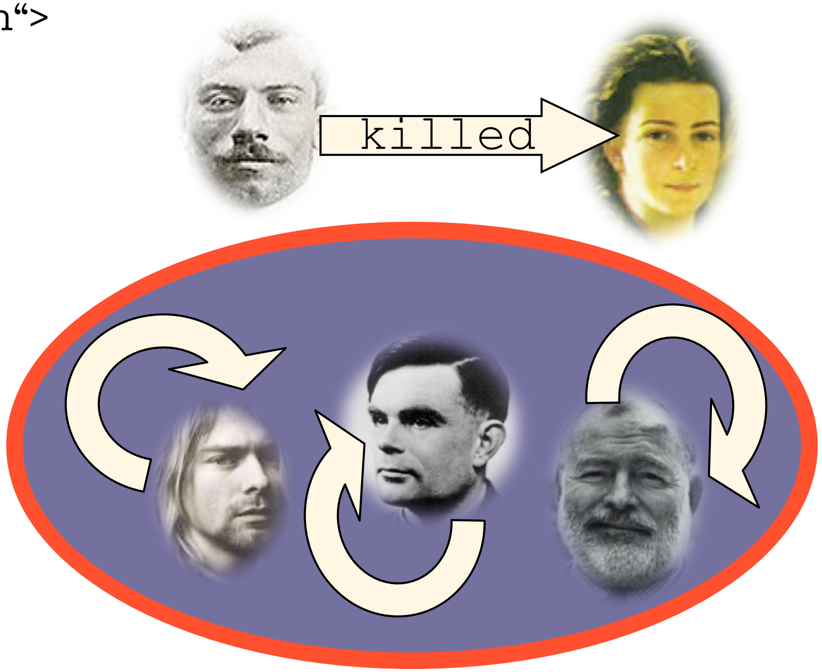
```
<owl:Restriction>
      <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger“>
       2 </owl:minQualifiedCardinality>
      <owl:onProperty rdf:about="hasChild"/>
      <owl:onClass rdf:about=“Female"/>
</owl:Restriction>
```



Class Constructors

- build new classes from class, property and individual names
  - Self-restriction: ∃killed.Self

```
<owl:Restriction>
    <owl:onProperty rdf:resource="killed"/>
    <owl:hasSelf rdf:datatype="&xsd;boolean">
     true
    </owl:hasSelf>
</owl:Restriction>
```



Special Classes and Properties

- special classes
  - top class: $\top$
    ...class containing all individuals of the domain

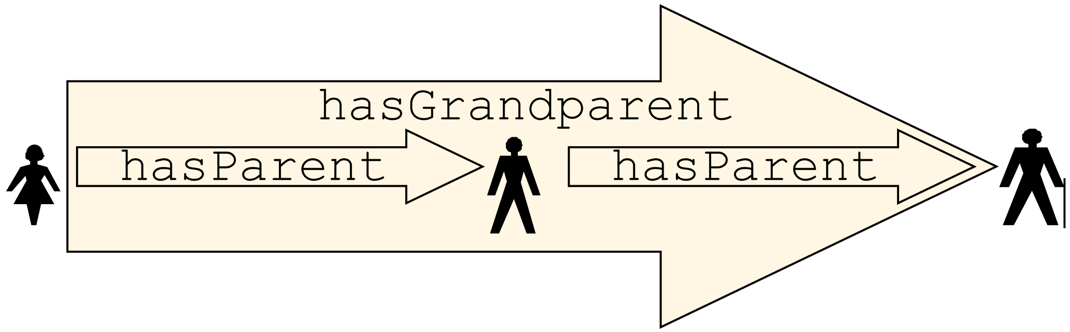
    owl:Thing
  - bottom class: $\bot$
    ...“empty“ class containing no individuals

    owl:Nothing
  - universal property: U
    ...property linking every individual to every individual

    owl:topObjectProperty

Property Chain Axioms

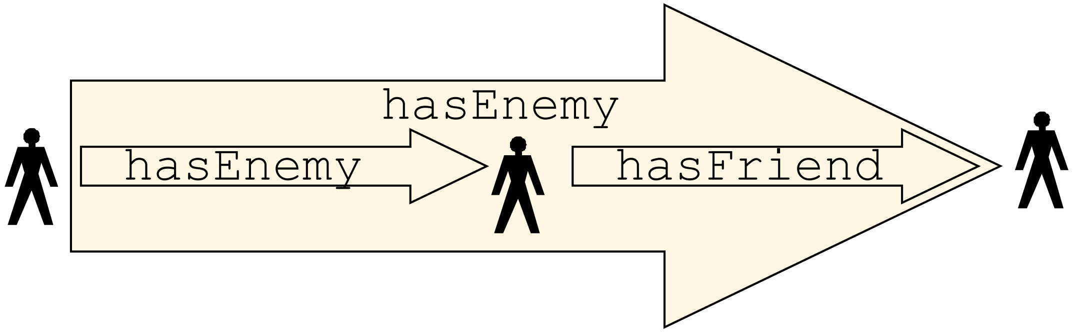
- allow to infer the existence of a property from a chain of properties:
  - hasParent ∘ hasParent ⊑ hasGrandparent
    rule version: hasParent(x,y) ∧ hasParent(y,z) → hasGrandparent(x,z)




```
<rdf:Description rdf:about="hasGrandparent">
    <owl:propertyChainAxiom rdf:parseType="Collection">
        <owl:ObjectProperty rdf:about="hasParent"/>
        <owl:ObjectProperty rdf:about="hasParent"/>
    </owl:propertyChainAxiom>
</rdf:Description>
```

Property Chain Axioms

- allow to infer the existence of a property from a chain of properties:
  - hasEnemy ∘ hasFriend ⊑ hasEnemy
    rule version: hasEnemy(x,y) ∧ hasFriend(y,z) → hasEnemy(x,z)




```
<rdf:Description rdf:about="hasEnemy">
    <owl:propertyChainAxiom rdf:parseType="Collection">
        <owl:ObjectProperty rdf:about="hasEnemy"/>
        <owl:ObjectProperty rdf:about="hasFriend"/>
    </owl:propertyChainAxiom>
</rdf:Description>
```

Property Chain Axioms: Caution! (1/2)

- arbitrary property chain axioms lead to undecidability
- restriction: set of property chain axioms has to be *regular*
  - there must be a strict linear order $\prec$ on the properties
  - every property chain axiom has to have one of the following forms:

    $R \circ R \sqsubseteq R$ $\quad S^{-} \sqsubseteq R$ $\quad S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$

    $R \circ S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$ $\quad S_1 \circ S_2 \circ \ldots \circ S_n \circ R \sqsubseteq R$
  - thereby, $S_i \prec R$ for all i= 1, 2, . . . , *n.*

- Example 1: $R \circ S \sqsubseteq R$ $\quad S \circ S \sqsubseteq S$ $\quad R \circ S \circ R \sqsubseteq T$

  → regular with order $S \prec R \prec T$
- Example 2: $R \circ T \circ S \sqsubseteq T$

  → not regular because form not admissible
- Example 3: $R \circ S \sqsubseteq S$ $\quad S \circ R \sqsubseteq R$

  → not regular because no adequate order exists

Property Chain Axioms: Caution! (2/2)

- combining property chain axioms and cardinality constraints may lead to undecidability
- restriction: use only *simple* properties in cardinality expressions (i.e. those which cannot be – directly or indirectly – inferred from property chains)
- technically:
  - for any property chain axiom $S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$ with n>1, R is non-simple
  - for any subproperty axiom $S \sqsubseteq R$ with S non-simple, R is non-simple
  - all other properties are simple
- Example: $Q \circ P \sqsubseteq R$ $R \circ P \sqsubseteq R$ $R \sqsubseteq S$ $P \sqsubseteq R$ $Q \sqsubseteq S$

  non-simple: R, S simple: P, Q

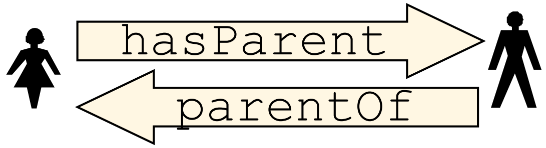
Property Characteristics

- a property can be
  - the inverse of another property: hasParent $\equiv$ parentOf $^-$

    rule version:

    hasParent(x,y) $\rightarrow$ parentOf(y,x)

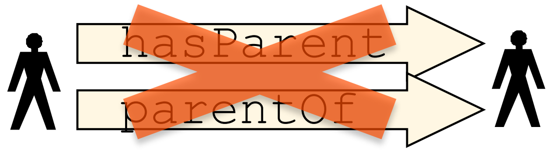
    parentOf(x,y) $\rightarrow$ hasParent(y,x)



  - disjoint with another property: Disj(hasParent,parentOf)

    rule version:

    hasParent(x,y), parentOf(x,y) $\rightarrow$

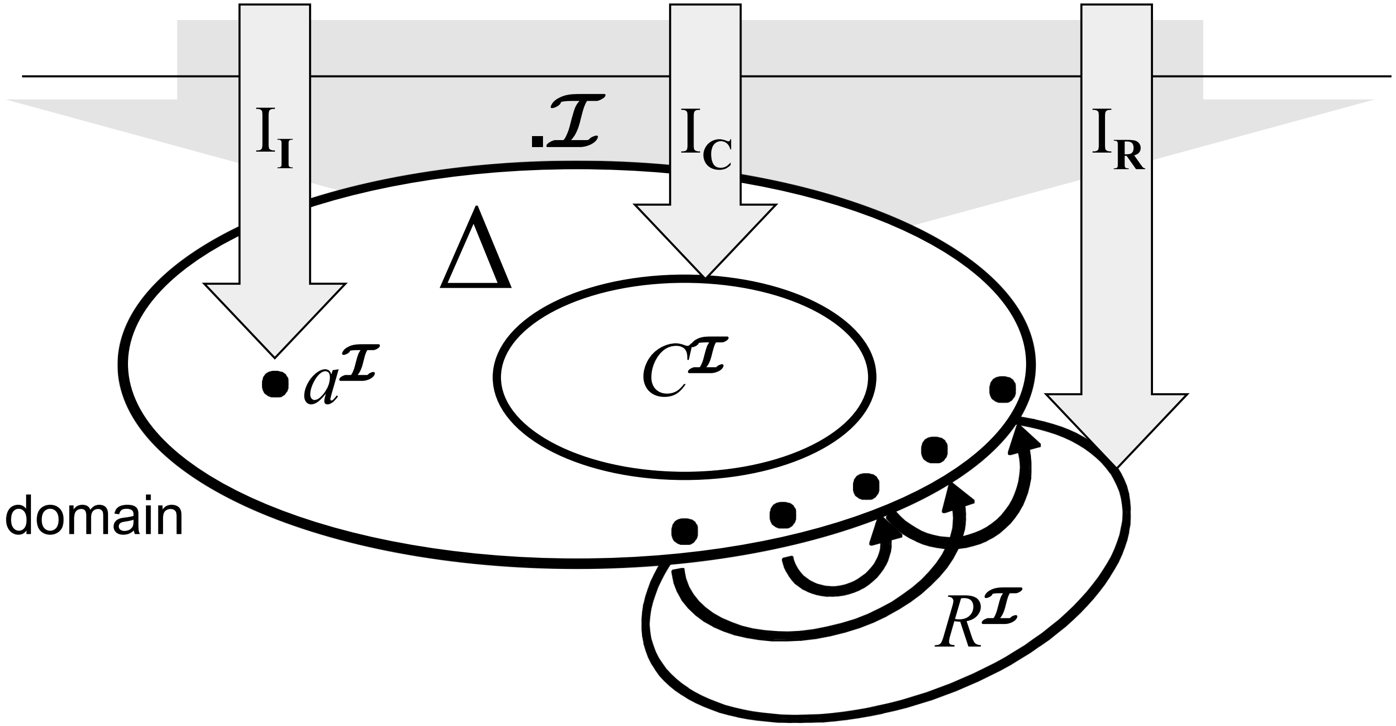


- other property characteristics that can be expressed: (inverse) functionality, transitivity, symmetry, asymmetry, reflexivity, irreflexivity

OWL 2 DL – Semantics

- model-theoretic semantics
- starts with interpretations
- an interpretation maps

individual names, class names and property names...



...into a domain

OWL 2 DL – Semantics

- mapping is extended to complex class expressions:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$ $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$ $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $\forall R.C = \{ x \mid \forall (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}} \}$ $\exists R.C = \{ x \mid \exists (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \}$
  - $\geq nR.C = \{ x \mid \#\{ y \mid (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \} \geq n \}$
  - $\leq nR.C = \{ x \mid \#\{ y \mid (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \} \leq n \}$
- ...and to role expressions:
  - $U^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ $(R^{-})^{\mathcal{I}} = \{ (y,x) \mid (x,y) \in R^{\mathcal{I}} \}$
- ...and to axioms:
  - $C(a)$ holds, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ $R(a,b)$ holds, if $(a^{\mathcal{I}},b^{\mathcal{I}}) \in R^{\mathcal{I}}$
  - $C \sqsubseteq D$ holds, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ $R \sqsubseteq S$ holds, if $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
  - Disj(R,S) holds if $R^{\mathcal{I}} \cap S^{\mathcal{I}} = \emptyset$
  - $S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$ holds if $S_1^{\mathcal{I}} \circ S_2^{\mathcal{I}} \circ \ldots \circ S_n^{\mathcal{I}} \subseteq R^{\mathcal{I}}$

OWL 2 DL – Alternative Semantics

- but often OWL 2 DL is said to be a fragment of FOL...
- yes, there is a translation of OWL 2 DL into FOL

$$\pi(C \sqsubseteq D) = (\forall x)(\pi_x(C) \rightarrow \pi_x(D))$$
$$\pi_x(A) = A(x)$$
$$\pi_x(\neg C) = \neg\pi_x(C)$$
$$\pi_x(C \sqcap D) = \pi_x(C) \wedge \pi_x(D)$$
$$\pi_x(C \sqcup D) = \pi_x(C) \vee \pi_x(D)$$
$$\pi_x(\forall R.C) = (\forall x_1)(R(x, x_1) \rightarrow \pi_{x_1}(C))$$
$$\pi_x(\exists R.C) = (\exists x_1)(R(x, x_1) \wedge \pi_{x_1}(C))$$
$$\pi_x(\geq nS.C) = (\exists x_1)\dots(\exists x_n)\left(\bigwedge_{i \neq j}(x_i \neq x_j) \wedge \bigwedge_i (S(x, x_i) \wedge \pi_{x_i}(C))\right)$$
$$\pi_x(\leq nS.C) = \neg(\exists x_1)\dots(\exists x_{n+1})\left(\bigwedge_{i \neq j}(x_i \neq x_j) \wedge \bigwedge_i (S(x, x_i) \wedge \pi_{x_i}(C))\right)$$
$$\pi_x(\{a\}) = (x = a)$$
$$\pi_x(\exists S.\mathrm{Self}) = S(x, x)$$

$$\pi(R_1 \sqsubseteq R_2) = (\forall x)(\forall y)(\pi_{x,y}(R_1) \rightarrow \pi_{x,y}(R_2))$$
$$\pi_{x,y}(S) = S(x, y)$$
$$\pi_{x,y}(R^-) = \pi_{y,x}(R)$$
$$\pi_{x,y}(R_1 \circ \dots \circ R_n) = (\exists x_1)\dots(\exists x_{n-1})\left(\pi_{x,x_1}(R_1) \wedge \bigwedge_{i=1}^{n-2} \pi_{x_i,x_{i+1}}(R_{i+1}) \wedge \pi_{x_{n-1},y}(R_n)\right)$$
$$\pi(\mathrm{Ref}(R)) = (\forall x)\pi_{x,x}(R)$$
$$\pi(\mathrm{Asy}(R)) = (\forall x)(\forall y)(\pi_{x,y}(R) \rightarrow \neg\pi_{y,x}(R))$$
$$\pi(\mathrm{Dis}(R_1, R_2)) = \neg(\exists x)(\exists y)(\pi_{x,y}(R_1) \wedge \pi_{x,y}(R_2))$$

- ...which (interpreted under FOL semantics) coincides with the definition just given.

OWL 2 Profiles

- OWL 2 DL is very expressive (although decidable)
  - tool support for full OWL 2 DL difficult to achieve
- complexity for standard reasoning tasks: N2ExpTime
  - scalability cannot be guaranteed
- idea: identify subsets of OWL 2 DL which are
  - still sufficiently expressive
  - of lower complexity (preferably in PTime)
  - computationally easier to handle
- OWL 2 Profiles:
  - OWL EL
  - OWL RL
  - OWL QL

OWL 2 EL

Karlsruhe Institute of Technology

- allowed:
  - subclass axioms with intersection, existential quantification, top, bottom
    - closed classs must have only one member
  - property chain axioms, range restrictions (under certain conditions)
- disallowed:
  - negation, disjunction, arbitrary universal quantification, role inverses

$$\sqcap \exists \top \bot \sqsubseteq \sqcap \exists \top \bot$$

- Reasoning is PTime complete
- Examples: $\exists \text{has.Sorrow} \sqsubseteq \exists \text{has.Liqueur}$ $\top \sqsubseteq \exists \text{hasParent.Person}$

  $\exists \text{married.}\top \sqcap \text{CatholicPriest} \sqsubseteq \bot$ $\text{German} \sqsubseteq \exists \text{knows.\{angela\}}$

  $\text{hasParent} \circ \text{hasParent} \sqsubseteq \text{hasGrandparent}$

OWL 2 RL

- motivated by the question: what fraction of OWL 2 DL can be expressed by rules (with equality)?
- examples:
  - $\exists$parentOf.$\exists$parentOf.$\top \sqsubseteq$ Grandfather

    rule version: parentOf(x,y) $\wedge$ parentOf(y,z) $\rightarrow$ Grandfather(x)
  - Orphan $\sqsubseteq$ $\forall$hasParent.Dead

    rule version: Orphan(x) $\wedge$ hasParent(x,y) $\rightarrow$ Dead(y)
  - Monogamous $\sqsubseteq$ $\leq$1married.Alive

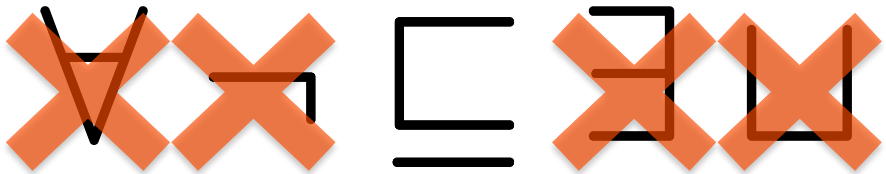
    rule version:
    Monogamous(x) $\wedge$ married(x,y) $\wedge$ Alive(y) $\wedge$ married(x,z) $\wedge$ Alive(z)$\rightarrow$ y=z
  - childOf $\circ$ childOf $\sqsubseteq$ grandchildOf

    rule version: childOf(x,y) $\wedge$ childOf(y,z) $\rightarrow$ grandchildOf(x,z)
  - Disj(childOf,parentOf)

    rule version: childOf(x,y) $\wedge$ parentOf(x,y) $\rightarrow$

OWL 2 RL

- syntactic characterization:
  - essentially, all axiom types are allowed
  - disallow certain constructors on lhs and rhs of subclass statements



  - cardinality restrictions: only on rhs and only ≤1 and ≤0 allowed
  - closed classes: only with one member
- Reasoning is PTime complete
- Example Ontology: SWRC

OWL 2 QL

Karlsruhe Institute of Technology

- motivated by the question: what fraction of OWL 2 DL can be captured by standard database technology?
- formally: query answering LOGSPACE w.r.t. data
  (via translation into SQL)
- allowed:
  - subproperties, domain, range
  - subclass statements with
    - left hand side: class name or expression of type $\exists r.\top$
    - right hand side: intersection of class names, expressions of type $\exists r.C$ and negations of lhs expressions
    - no closed classes!
- Example:

$$\exists \text{married}.\top \sqsubseteq \neg \text{Free} \sqcap \exists \text{has}.\text{Sorrow}$$

OWL 2 Reasoner

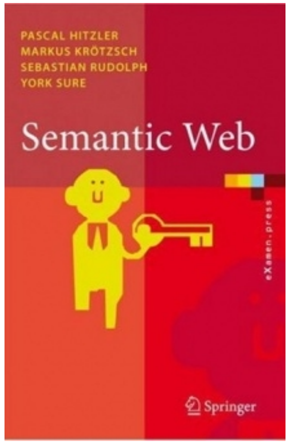
- OWL 2 DL:
  - Pellet http://clarkparsia.com/pellet/
  - HermiT http://www.hermit-reasoner.com/
- OWL 2 EL:
  - CEL http://code.google.com/p/cel/
- OWL 2 RL:
  - essentially any rule engine
- OWL 2 QL:
  - essentially any SQL engine (with a bit of query rewriting on top)

References

Karlsruhe Institute of Technology

- OWL 2 W3C Documentation
  - http://www.w3.org/TR/owl2-overview/

- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure, Semantic Web – Grundlagen. Springer, 2008. http://www.semantic-web-grundlagen.de/



- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies. CRC Press, 2009. http://www.semantic-web-book.org/ (Grab a flyer from us.)

Karlsruhe Institute of Technology

Thanks!

http://semantic-web-grundlagen.de/wiki/ESWC09_Tutorial