Description Logic: A Formal Foundation for Ontology Languages and Tools

Ian Horrocks
<ian.horrocks@comlab.ox.ac.uk>
Information Systems Group
Oxford University Computing Laboratory

What Are Description Logics?

What Are Description Logics?

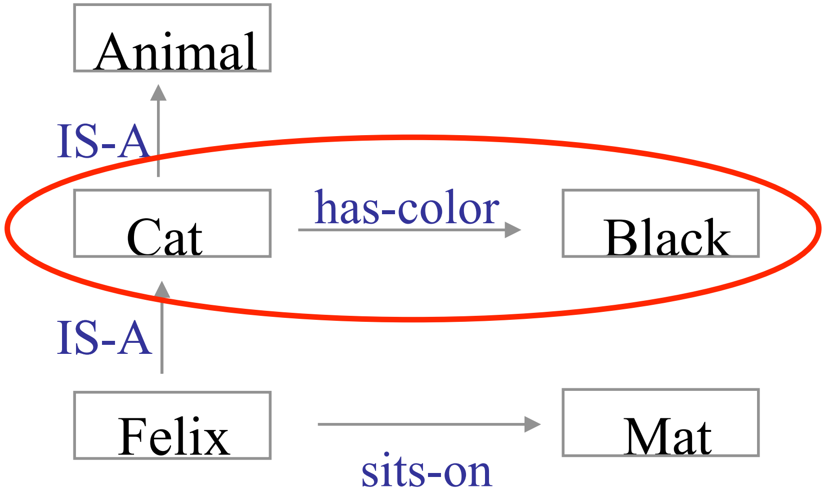
- Decidable fragments of First Order Logic

Thank you for listening

Any questions?

What Are Description Logics?

- A family of logic based Knowledge Representation formalisms
  - Originally descended from semantic networks and KL-ONE
  - Describe domain in terms of concepts (aka classes), roles (aka properties, relationships) and individuals



[Quillian, 1967]

What Are Description Logics?

- Modern DLs (after Baader et al) distinguished by:
  - Fully fledged logics with formal semantics
    - Decidable fragments of FOL (often contained in C_2)
    - Closely related to Propositional Modal/Dynamic Logics & Guarded Fragment
  - Computational properties well understood (worst case complexity)
  - Provision of inference services
    - Practical decision procedures (algorithms) for key problems (satisfiability, subsumption, query answering, etc)
    - Implemented systems (highly optimised)
- The basis for widely used ontology languages

Web Ontology Language OWL (2)

- W3C **recommendation(s)**
- Motivated by **Semantic Web** activity

  Add meaning to web content by annotating it with terms defined in ontologies

- Supported by **tools and infrastructure**
  - APIs (e.g., OWL API, Thea, OWLink)
  - Development environments (e.g., Protégé, Swoop, TopBraid Composer, Neon)
  - Reasoners & Information Systems (e.g., Pellet, Racer, HermiT, Quonto, …)

- Based on **Description Logics** ($\mathcal{SHOIN}$ / $\mathcal{SROIQ}$)

DL Syntax

- Signature
  - **Concept** (aka class) names, e.g., Cat, Animal, Doctor
    - Equivalent to FOL unary predicates
  - **Role** (aka property) names, e.g., sits-on, hasParent, loves
    - Equivalent to FOL binary predicates
  - **Individual** names, e.g., Felix, John, Mary, Boston, Italy
    - Equivalent to FOL constants

DL Syntax

- Operators
  - Many kinds available, e.g.,
    - Standard FOL Boolean operators (⊓, ⊔, ¬)
    - Restricted form of quantifiers (∃, ∀)
    - Counting (≥, ≤, =)
    - …

DL Syntax

- Concept expressions, e.g.,
  - Doctor $\sqcup$ Lawyer
  - Rich $\sqcap$ Happy
  - Cat $\sqcap$ $\exists$sits-on.Mat
- Equivalent to FOL formulae with one free variable
  - $\mathrm{Doctor}(x) \lor \mathrm{Lawyer}(x)$
  - $\mathrm{Rich}(x) \land \mathrm{Happy}(x)$
  - $\exists y.(\mathrm{Cat}(x) \land \text{sits-on}(x, y))$

DL Syntax

- Special concepts
  - $\top$ (aka top, Thing, most general concept)
  - $\bot$ (aka bottom, Nothing, inconsistent concept)

  used as abbreviations for
  - $(A \sqcup \neg A)$ for any concept A
  - $(A \sqcap \neg A)$ for any concept A

DL Syntax

- Role expressions, e.g.,
  - loves^{-}
  - $\text{hasParent} \circ \text{hasBrother}$
- Equivalent to FOL formulae with two free variables
  - $\text{loves}(y, x)$
  - $\exists z.(\text{hasParent}(x, z) \wedge \text{hasBrother}(z, y))$

DL Syntax

- “Schema” Axioms, e.g.,
  - $\mathrm{Rich} \sqsubseteq \neg\mathrm{Poor}$ (concept inclusion)
  - $\mathrm{Cat} \sqcap \exists\mathrm{sits\text{-}on}.\mathrm{Mat} \sqsubseteq \mathrm{Happy}$ (concept inclusion)
  - $\mathrm{BlackCat} \equiv \mathrm{Cat} \sqcap \exists\mathrm{hasColour}.\mathrm{Black}$ (concept equivalence)
  - $\mathrm{sits\text{-}on} \sqsubseteq \mathrm{touches}$ (role inclusion)
  - $\mathrm{Trans}(\mathrm{part\text{-}of})$ (transitivity)
- Equivalent to (particular form of) FOL sentence, e.g.,
  - $\forall x.(\mathrm{Rich}(x) \rightarrow \neg\mathrm{Poor}(x))$
  - $\forall x.(\mathrm{Cat}(x) \wedge \exists y.(\mathrm{sits\text{-}on}(x,y) \wedge \mathrm{Mat}(y)) \rightarrow \mathrm{Happy}(x))$
  - $\forall x.(\mathrm{BlackCat}(x) \leftrightarrow (\mathrm{Cat}(x) \wedge \exists y.(\mathrm{hasColour}(x,y) \wedge \mathrm{Black}(y)))$
  - $\forall x,y.(\mathrm{sits\text{-}on}(x,y) \rightarrow \mathrm{touches}(x,y))$
  - $\forall x,y,z.((\mathrm{sits\text{-}on}(x,y) \wedge \mathrm{sits\text{-}on}(y,z)) \rightarrow \mathrm{sits\text{-}on}(x,z))$

DL Syntax

- “Data” Axioms (aka Assertions or Facts), e.g.,
  - BlackCat(Felix) (concept assertion)
  - Mat(Mat1) (concept assertion)
  - Sits-on(Felix,Mat1) (role assertion)
- Directly equivalent to FOL “ground facts”
  - Formulae with no variables

DL Syntax

- A set of axioms is called a TBox, e.g.:

$$\{\text{Doctor} \sqsubseteq \text{Person},$$
$$\text{Parent} \equiv \text{Person} \sqcap \exists\text{hasChild.Pers}$$
$$\text{HappyParent} \equiv \text{Parent} \sqcap \forall\text{hasChil}$$

- A set of facts is called an AB

$$\{\text{HappyParent(John)},$$
$$\text{hasChild(John,Mary)}\}$$

Note

Facts sometimes written

John:HappyParent,

John hasChild Mary,

$\langle$John,Mary$\rangle$:hasChild

- A Knowledge Base (KB) is just a TBox plus an Abox
  - Often written $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

The DL Family

- Many different DLs, often with “strange” names
  - E.g., $\mathcal{EL}$, $\mathcal{ALC}$, $\mathcal{SHIQ}$
- Particular DL defined by:
  - Concept operators ($\sqcap$, $\sqcup$, $\neg$, $\exists$, $\forall$, etc.)
  - Role operators ($^-$, $\circ$, etc.)
  - Concept axioms ($\sqsubseteq$, $\equiv$, etc.)
  - Role axioms ($\sqsubseteq$, Trans, etc.)

The DL Family

- E.g., $\mathcal{EL}$ is a well known “sub-Boolean” DL
  - Concept operators: $\sqcap$, $\neg$, $\exists$
  - No role operators (only atomic roles)
  - Concept axioms: $\sqsubseteq$, $\equiv$
  - No role axioms
- E.g.:

$$\text{Parent} \equiv \text{Person} \sqcap \exists \text{hasChild.Person}$$

The DL Family

- $\mathcal{ALC}$ is the smallest propositionally closed DL
  - Concept operators: $\sqcap$, $\sqcup$, $\neg$, $\exists$, $\forall$
  - No role operators (only atomic roles)
  - Concept axioms: $\sqsubseteq$, $\equiv$
  - No role axioms
- E.g.:

$$\text{ProudParent} \equiv \text{Person} \sqcap \forall \text{hasChild}.(\text{Doctor} \sqcup \exists \text{hasChild}.\text{Doctor})$$

The DL Family

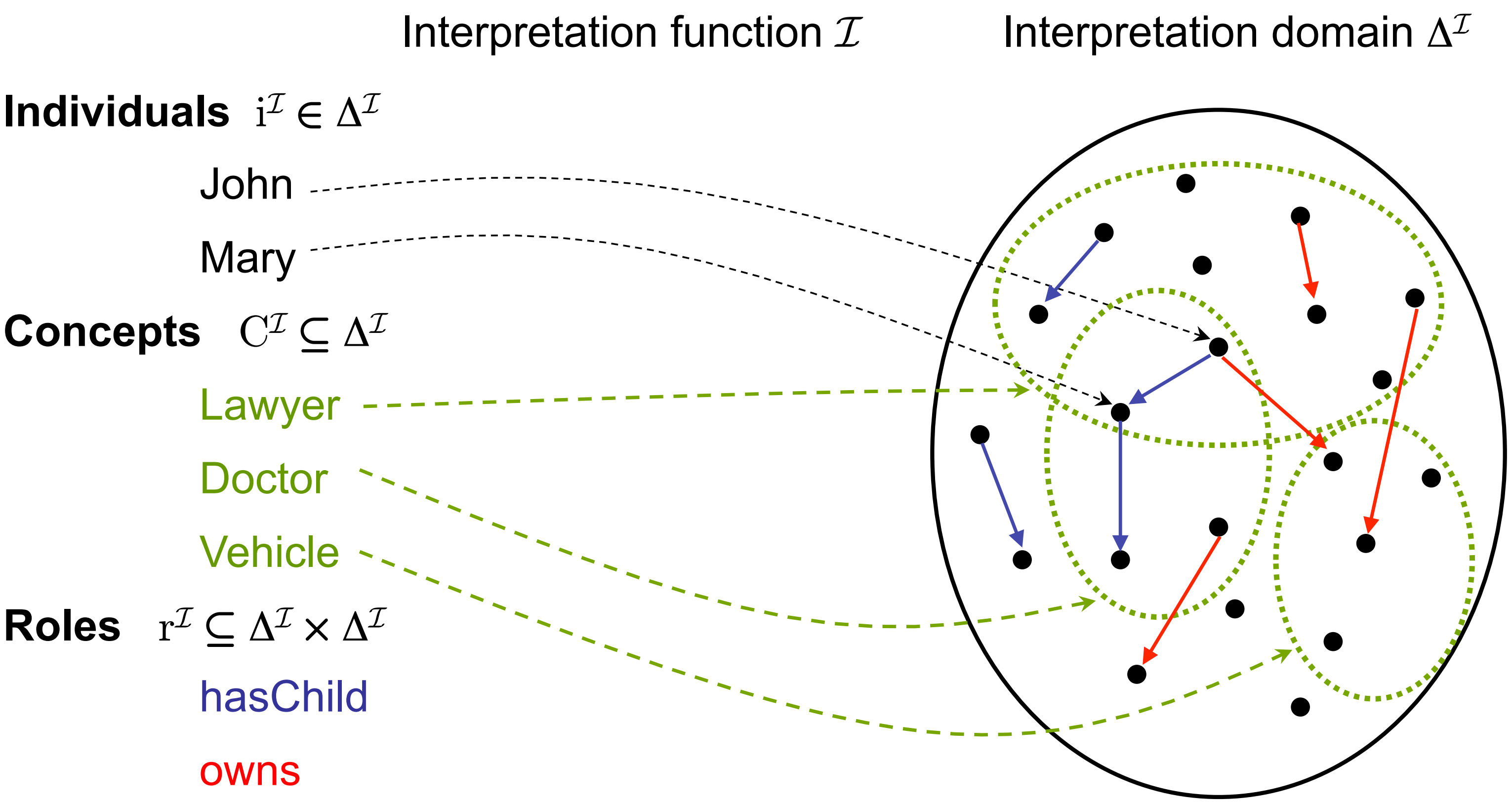
- $\mathcal{S}$ used for $\mathcal{ALC}$ extended with (role) transitivity axioms
- Additional letters indicate various extensions, e.g.:
  - $\mathcal{H}$ for role hierarchy (e.g., hasDaughter $\sqsubseteq$ hasChild)
  - $\mathcal{R}$ for role box (e.g., $\mathrm{hasParent} \circ \mathrm{hasBrother}$ $\sqsubseteq$ hasUncle)
  - $\mathcal{O}$ for nominals/singleton classes (e.g., {Italy})
  - $\mathcal{I}$ for inverse roles (e.g., isChildOf $\equiv$ hasChild$^-$)
  - $\mathcal{N}$ for number restrictions (e.g., $\geqslant 2$hasChild, $\leqslant 3$hasChild)
  - $\mathcal{Q}$ for qualified number restrictions (e.g., $\geqslant 2$hasChild.Doctor)
  - $\mathcal{F}$ for functional number restrictions (e.g., $\leqslant 1$hasMother)
- E.g., $\mathcal{SHIQ}$ = $\mathcal{S}$ + role hierarchy + inverse roles + QNRs

The DL Family

- Numerous other extensions have been investigated
  - Concrete domains (numbers, strings, etc)
  - DL-safe rules (Datalog-like rules)
  - Fixpoints
  - Role value maps
  - Additional role constructors ($\cap$, $\cup$, $\neg$, $\circ$, id, …)
  - Nary (i.e., predicates with arity >2)
  - Temporal
  - Fuzzy
  - Probabilistic
  - Non-monotonic
  - Higher-order
  - …

DL Semantics

Via translaton to FOL, or directly using FO model theory:



Interpretation function $\mathcal{I}$

Interpretation domain $\Delta^{\mathcal{I}}$

Individuals $i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

John

Mary

Concepts $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$

Lawyer

Doctor

Vehicle

Roles $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

hasChild

owns

DL Semantics

- Interpretation function extends to **concept expressions** in the obvious(ish) way, e.g.:

$$
\begin{aligned}
&(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}\\
&(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}\\
&(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}\\
&\{x\}^{\mathcal{I}} = \{x^{\mathcal{I}}\}\\
&(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}\\
&(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.(x, y) \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}\\
&(\leqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \leqslant n\}\\
&(\geqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \geqslant n\}
\end{aligned}
$$

DL Semantics

- Given a model M = $\langle D, \cdot^I \rangle$
  - $M \models C \sqsubseteq D$ iff $C^I \subseteq D^I$
  - $M \models C \equiv D$ iff $C^I = D^I$
  - $M \models C(a)$ iff $a^I \in C^I$
  - $M \models R(a, b)$ iff $\langle a^I, b^I \rangle \in R^I$
  - $M \models \langle \mathcal{T}, \mathcal{A} \rangle$ iff for every axiom $ax \in \mathcal{T} \cup \mathcal{A}$, $M \models ax$

DL Semantics

- Satisfiability and entailment
  - A KB $\mathcal{K}$ is satisfiable iff there exists a model M s.t. $\mathrm{M} \models \mathcal{K}$
  - A concept C is satisfiable w.r.t. a KB $\mathcal{K}$ iff there exists a model $\mathrm{M} = \langle \mathrm{D}, \cdot^{\mathrm{I}} \rangle$ s.t. $\mathrm{M} \models \mathcal{K}$ and $\mathrm{C}^{\mathrm{I}} \neq \emptyset$
  - A KB $\mathcal{K}$ entails an axiom ax (written $\mathcal{K} \models \mathrm{ax}$) iff for every model M of $\mathcal{K}$, $\mathrm{M} \models \mathrm{ax}$ (i.e., $\mathrm{M} \models \mathcal{K}$ implies $\mathrm{M} \models \mathrm{ax}$)

DL Semantics

E.g.,

$\mathcal{T}$ = {Doctor $\sqsubseteq$ Person, Parent $\equiv$ Person $\sqcap$ $\exists$hasChild.Person,
HappyParent $\equiv$ Parent $\sqcap$ $\forall$hasChild.(Doctor $\sqcup$ $\exists$hasChild.Doctor)}

$\mathcal{A}$ = {John:HappyParent, John hasChild Mary, John hasChild Sally,
Mary:$\neg$Doctor, Mary hasChild Peter, Mary:($\leq$ 1 hasChild)

✓ – $\mathcal{K} \models$ John:Person ?

✓ – $\mathcal{K} \models$ Peter:Doctor ?

✓ – $\mathcal{K} \models$ Mary:HappyParent ?

– What if we add “Mary hasChild Jane” ?

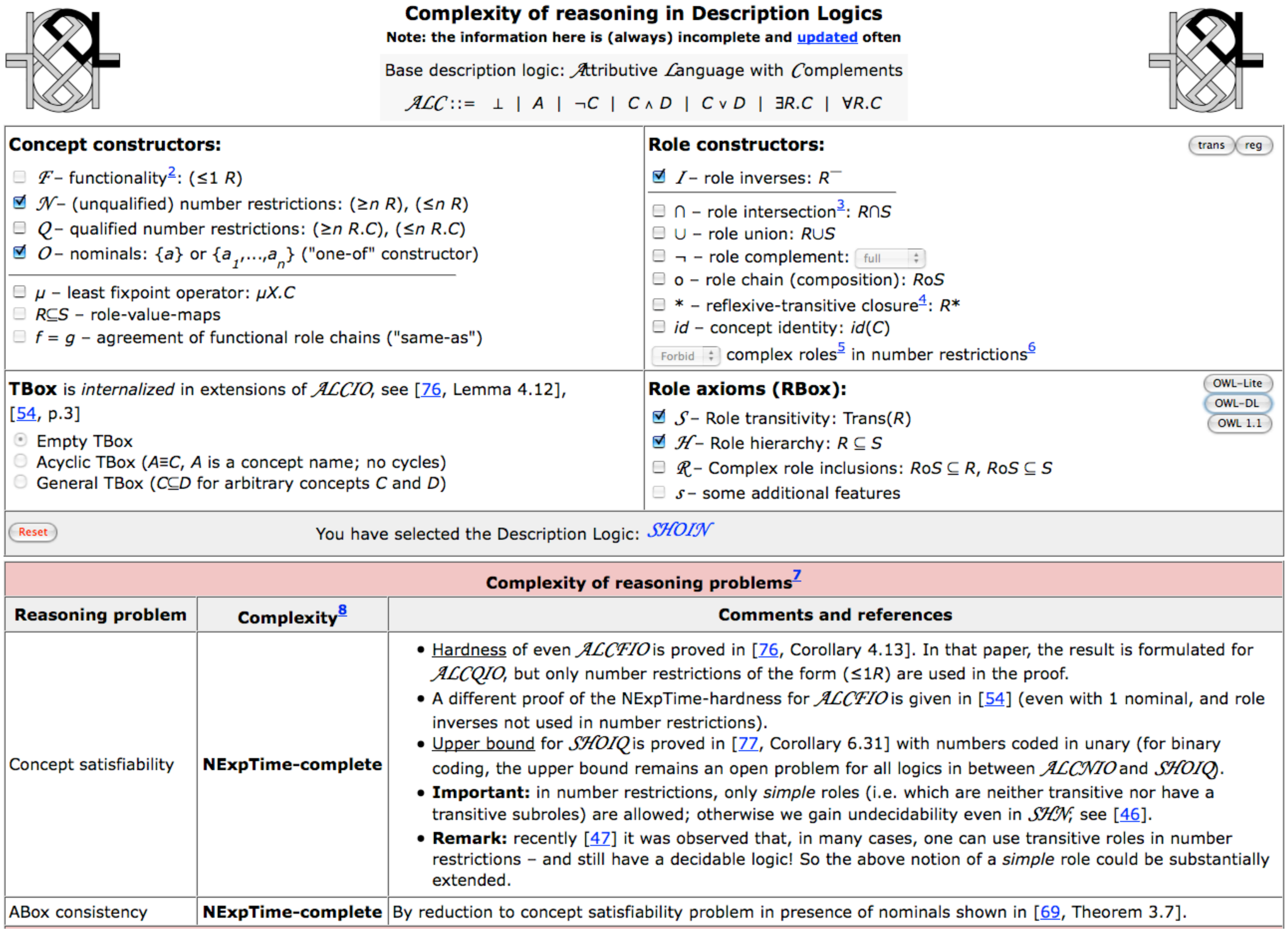
$\mathcal{K} \models$ Peter = Jane

– What if we add “HappyPerson $\equiv$ Person $\sqcap$ $\exists$hasChild.Doctor” ?

$\mathcal{K} \models$ HappyPerson $\sqsubseteq$ Parent

DL and FOL

- Most DLs are subsets of C2
  - But reduction to C2 may be (highly) non-trivial
    - Trans(R) naively reduces to $\forall x, y, z. R(x, y) \wedge R(y, z) \rightarrow R(x, z)$
- Why use DL instead of C2?
  - Syntax is succinct and convenient for KR applications
  - Syntactic conformance guarantees being inside C2
    - Even if reduction to C2 is non-obvious
  - Different combinations of constructors can be selected
    - To guarantee decidability
    - To reduce complexity
  - DL research has mapped out the decidability/complexity landscape in great detail
    - See Evgeny Zolin's DL Complexity Analyzer http://www.cs.man.ac.uk/~ezolin/dl/



Complexity of reasoning in Description Logics

Note: the information here is (always) incomplete and updated often

Base description logic: $\mathcal{A}$ttributive $\mathcal{L}$anguage with $\mathcal{C}$omplements

$\mathcal{ALC} ::= \bot \mid A \mid \neg C \mid C \wedge D \mid C \vee D \mid \exists R.C \mid \forall R.C$

Concept constructors:	**Role constructors:**
☐ $\mathcal{F}$ – functionality[2]: $(\leq 1\ R)$ ☑ $\mathcal{N}$ – (unqualified) number restrictions: $(\geq n\ R)$, $(\leq n\ R)$ ☐ $\mathcal{Q}$ – qualified number restrictions: $(\geq n\ R.C)$, $(\leq n\ R.C)$ ☑ $\mathcal{O}$ – nominals: $\{a\}$ or $\{a_1,\ldots,a_n\}$ ("one-of" constructor) ☐ μ – least fixpoint operator: $\mu X.C$ ☐ $R \subseteq S$ – role-value-maps ☐ $f = g$ – agreement of functional role chains ("same-as")	☑ $\mathcal{I}$ – role inverses: R^- ☐ $\cap$ – role intersection[3]: $R \cap S$ ☐ $\cup$ – role union: $R \cup S$ ☐ $\neg$ – role complement: full ☐ o – role chain (composition): $R \circ S$ ☐ * – reflexive-transitive closure[4]: R^* ☐ id – concept identity: $id(C)$ Forbid complex roles[5] in number restrictions[6]
TBox is *internalized* in extensions of $\mathcal{ALCIO}$, see [76, Lemma 4.12], [54, p.3] ● Empty TBox ○ Acyclic TBox ($A \equiv C$, A is a concept name; no cycles) ○ General TBox ($C \sqsubseteq D$ for arbitrary concepts C and D)	**Role axioms (RBox):** ☑ $\mathcal{S}$ – Role transitivity: Trans(R) ☑ $\mathcal{H}$ – Role hierarchy: $R \sqsubseteq S$ ☐ $\mathcal{R}$ – Complex role inclusions: $R \circ S \sqsubseteq R$, $R \circ S \sqsubseteq S$ ☐ $\mathcal{s}$ – some additional features

You have selected the Description Logic: $\mathcal{SHOIN}$

Complexity of reasoning problems[7]

Reasoning problem	Complexity[8]	Comments and references
Concept satisfiability	**NExpTime-complete**	• Hardness of even $\mathcal{ALCFIO}$ is proved in [76, Corollary 4.13]. In that paper, the result is formulated for $\mathcal{ALCQIO}$, but only number restrictions of the form $(\leq 1 R)$ are used in the proof. • A different proof of the NExpTime-hardness for $\mathcal{ALCFIO}$ is given in [54] (even with 1 nominal, and role inverses not used in number restrictions). • Upper bound for $\mathcal{SHOIQ}$ is proved in [77, Corollary 6.31] with numbers coded in unary (for binary coding, the upper bound remains an open problem for all logics in between $\mathcal{ALCNIO}$ and $\mathcal{SHOIQ}$. • **Important:** in number restrictions, only *simple* roles (i.e. which are neither transitive nor have a transitive subroles) are allowed; otherwise we gain undecidability even in $\mathcal{SHN}$; see [46]. • **Remark:** recently [47] it was observed that, in many cases, one can use transitive roles in number restrictions – and still have a decidable logic! So the above notion of a *simple* role could be substantially extended.
ABox consistency	**NExpTime-complete**	By reduction to concept satisfiability problem in presence of nominals shown in [69, Theorem 3.7].

Complexity Measures

- Taxonomic complexity
  - Measured w.r.t. total size of “schema” axioms
- Data complexity
  - Measured w.r.t. total size of “data” facts
- Query complexity
  - Measured w.r.t. size of query
- Combined complexity
  - Measured w.r.t. total size of KB (plus query if appropriate)

Complexity Classes

- LogSpace, PTime, NP, PSpace, ExpTime, etc
  - worst case for a given problem w.r.t. a given parameter
  - X-hard means at-least this hard (could be harder);
    in X means no harder than this (could be easier);
    X-complete means both hard and in, i.e., exactly this hard
    - e.g., $\mathcal{SROIQ}$ KB satisfiability is 2NExpTime-complete w.r.t. combined complexity and NP-hard w.r.t. data complexity
- Note that:
  - this is for the worst case, not a typical case
  - complexity of problem means we can never devise a more efficient (in the worst case) algorithm
  - complexity of algorithm may, however, be even higher (in the worst case)

DLs and Ontology Languages

DLs and Ontology Languages

- W3C's OWL 2 (like OWL, DAML+OIL & OIL) based on DL
  - OWL 2 based on $\mathcal{SROIQ}$, i.e., $\mathcal{ALC}$ extended with transitive roles, a role box nominals, inverse roles and qualified number restrictions
    - OWL 2 EL based on $\mathcal{EL}$
    - OWL 2 QL based on DL-Lite
    - OWL 2 EL based on $\mathcal{DLP}$
  - OWL was based on $\mathcal{SHOIN}$
    - only simple role hierarchy, and unqualified NRs

Class/Concept Constructors

OWL Constructor	DL Syntax	Example
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human $\sqcap$ Male
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer
complementOf	$\neg C$	¬Male
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}
allValuesFrom	$\forall P.C$	∀hasChild.Doctor
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer
maxCardinality	$\leqslant n P$	⩽1hasChild
minCardinality	$\geqslant n P$	⩾2hasChild

Ontology Axioms

OWL Syntax	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	$\mathsf{Human} \sqsubseteq \mathsf{Animal} \sqcap \mathsf{Biped}$
equivalentClass	$C_1 \equiv C_2$	$\mathsf{Man} \equiv \mathsf{Human} \sqcap \mathsf{Male}$
subPropertyOf	$P_1 \sqsubseteq P_2$	$\mathsf{hasDaughter} \sqsubseteq \mathsf{hasChild}$
equivalentProperty	$P_1 \equiv P_2$	$\mathsf{cost} \equiv \mathsf{price}$
transitiveProperty	$P^+ \sqsubseteq P$	$\mathsf{ancestor}^+ \sqsubseteq \mathsf{ancestor}$

OWL Syntax	DL Syntax	Example
type	$a : C$	$\mathsf{John} : \mathsf{Happy\text{-}Father}$
property	$\langle a, b \rangle : R$	$\langle \mathsf{John}, \mathsf{Mary} \rangle : \mathsf{has\text{-}child}$

- An Ontology is *usually* considered to be a TBox
  - but an OWL ontology is a mixed set of TBox and ABox axioms

Other OWL Features

- XSD datatypes and (in OWL 2) facets, e.g.,
  - integer, string and (in OWL 2) real, float, decimal, datetime, …
  - minExclusive, maxExclusive, length, …
  - PropertyAssertion(hasAge Meg "17"^^xsd:integer)
  - DatatypeRestriction(xsd:integer xsd:minInclusive "5"^^xsd:integer xsd:maxExclusive "10"^^xsd:integer)

  These are equivalent to (a limited form of) **DL concrete domains**

- Keys
  - E.g., HasKey(Vehicle Country LicensePlate)
    - Country + License Plate is a unique identifier for vehicles

  This is equivalent to (a limited form of) **DL safe rules**

OWL RDF/XML Exchange Syntax

E.g., $\text{Person} \sqcap \forall \text{hasChild}.(\text{Doctor} \sqcup \exists \text{hasChild}.\text{Doctor})$:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:someValuesFrom rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Complexity/Scalability

- From the complexity navigator we can see that:
  - OWL (aka $\mathcal{SHOIN}$) is NExpTime-complete
  - OWL Lite (aka $\mathcal{SHIF}$) is ExpTime-complete (oops!)
  - OWL 2 (aka $\mathcal{SROIQ}$) is 2NExpTime-complete
  - OWL 2 EL (aka $\mathcal{EL}$) is PTIME-complete (robustly scalable)
  - OWL 2 RL (aka $\mathcal{DLP}$) is PTIME-complete (robustly scalable)
    - And implementable using rule based technologies e.g., rule-extended DBs
  - OWL 2 QL (aka DL-Lite) is in AC^0 w.r.t. size of data
    - same as DB query answering -- nice!

Why (Description) Logic?

- OWL exploits results of 20+ years of DL research
  - Well defined (model theoretic) **semantics**

Constructor	DL Syntax	Example	FOL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human $\sqcap$ Male	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	¬Male	$\neg C(x)$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}	$x = x_1 \vee \ldots \vee x = x_n$
allValuesFrom	$\forall P.C$	∀hasChild.Doctor	$\forall y.P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer	$\exists y.P(x,y) \wedge C(y)$
maxCardinality	$\leqslant nP$	⩽1hasChild	$\exists^{\leqslant n} y.P(x,y)$
minCardinality	$\geqslant nP$	⩾2hasChild	$\exists^{\geqslant n} y.P(x,y)$

Why (Description) Logic?

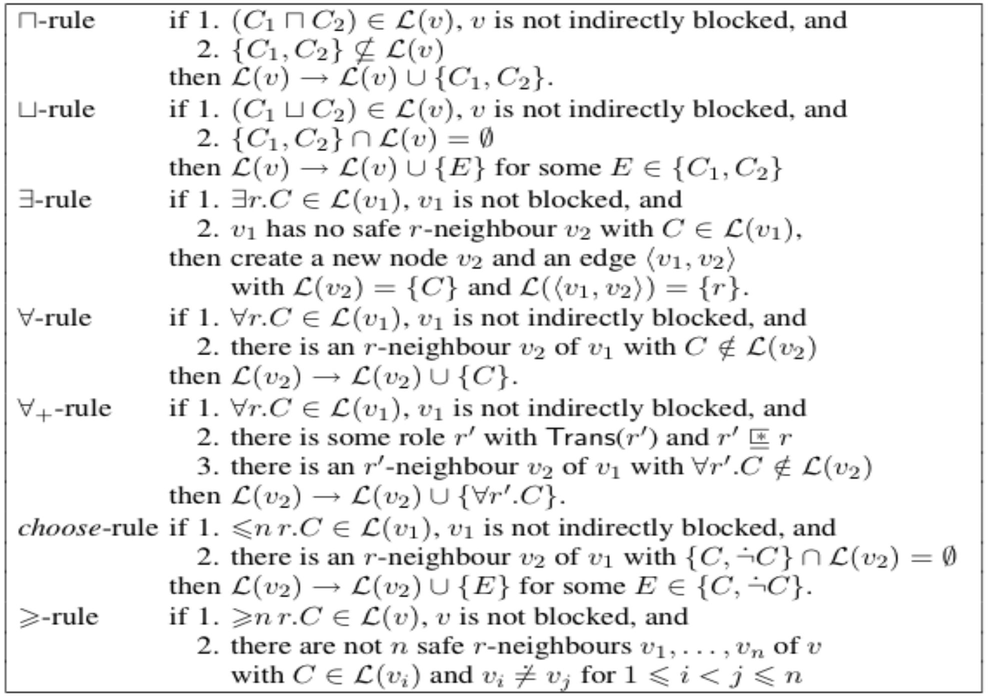
- OWL exploits results of 20+ years of DL research
  - Well defined (model theoretic) **semantics**
  - **Formal properties** well understood (complexity, decidability)

I can't find an efficient algorithm, but neither can all these famous people.

[Garey & Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, 1979.]

Why (Description) Logic?

- OWL exploits results of 20+ years of DL research
  - Well defined (model theoretic) **semantics**
  - **Formal properties** well understood (complexity, decidability)
  - Known **reasoning algorithms**



$\sqcap$-rule	if 1. $(C_1 \sqcap C_2) \in \mathcal{L}(v)$, v is not indirectly blocked, and 2. $\{C_1, C_2\} \not\subseteq \mathcal{L}(v)$ then $\mathcal{L}(v) \longrightarrow \mathcal{L}(v) \cup \{C_1, C_2\}$.
$\sqcup$-rule	if 1. $(C_1 \sqcup C_2) \in \mathcal{L}(v)$, v is not indirectly blocked, and 2. $\{C_1, C_2\} \cap \mathcal{L}(v) = \emptyset$ then $\mathcal{L}(v) \longrightarrow \mathcal{L}(v) \cup \{E\}$ for some $E \in \{C_1, C_2\}$
$\exists$-rule	if 1. $\exists r.C \in \mathcal{L}(v_1)$, v_1 is not blocked, and 2. v_1 has no safe r-neighbour v_2 with $C \in \mathcal{L}(v_1)$, then create a new node v_2 and an edge $\langle v_1, v_2 \rangle$ with $\mathcal{L}(v_2) = \{C\}$ and $\mathcal{L}(\langle v_1, v_2 \rangle) = \{r\}$.
$\forall$-rule	if 1. $\forall r.C \in \mathcal{L}(v_1)$, v_1 is not indirectly blocked, and 2. there is an r-neighbour v_2 of v_1 with $C \notin \mathcal{L}(v_2)$ then $\mathcal{L}(v_2) \longrightarrow \mathcal{L}(v_2) \cup \{C\}$.
$\forall_+$-rule	if 1. $\forall r.C \in \mathcal{L}(v_1)$, v_1 is not indirectly blocked, and 2. there is some role r' with Trans(r') and $r' \mathrel{\underline{\boxed{*}}} r$ 3. there is an r'-neighbour v_2 of v_1 with $\forall r'.C \notin \mathcal{L}(v_2)$ then $\mathcal{L}(v_2) \longrightarrow \mathcal{L}(v_2) \cup \{\forall r'.C\}$.
choose-rule	if 1. $\leqslant n\, r.C \in \mathcal{L}(v_1)$, v_1 is not indirectly blocked, and 2. there is an r-neighbour v_2 of v_1 with $\{C, \dot{\neg} C\} \cap \mathcal{L}(v_2) = \emptyset$ then $\mathcal{L}(v_2) \longrightarrow \mathcal{L}(v_2) \cup \{E\}$ for some $E \in \{C, \dot{\neg} C\}$.
$\geqslant$-rule	if 1. $\geqslant n\, r.C \in \mathcal{L}(v)$, v is not blocked, and 2. there are not n safe r-neighbours $v_1, \ldots, v_n$ of v with $C \in \mathcal{L}(v_i)$ and $v_i \neq v_j$ for $1 \leqslant i < j \leqslant n$

Why (Description) Logic?

- OWL exploits results of 20+ years of DL research
  - Well defined (model theoretic) **semantics**
  - **Formal properties** well understood (complexity, decidability)
  - Known **reasoning algorithms**
  - **Scalability** demonstrated by **implemented systems**

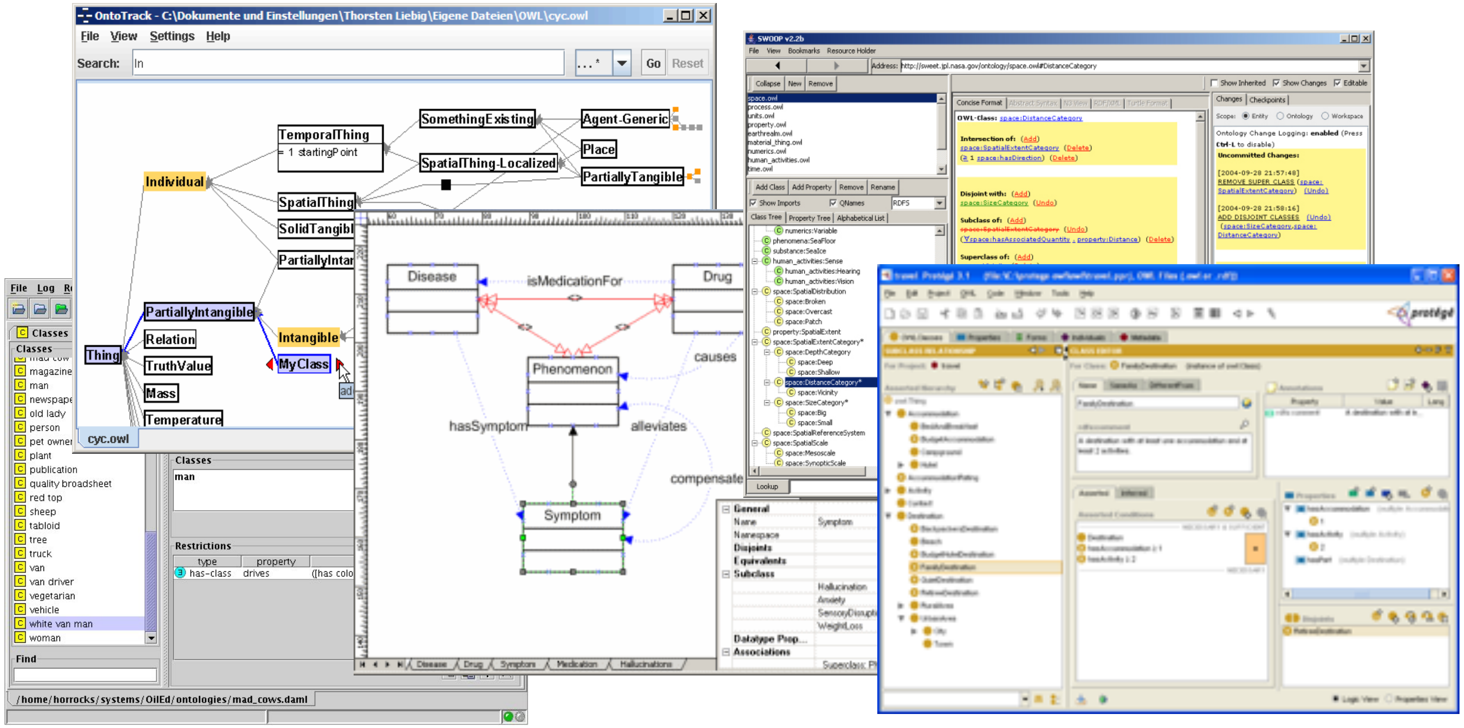
Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

- Editors/development environments



Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

- Editors/development environments
- Reasoners

Hermit FaCT++ ORACLE

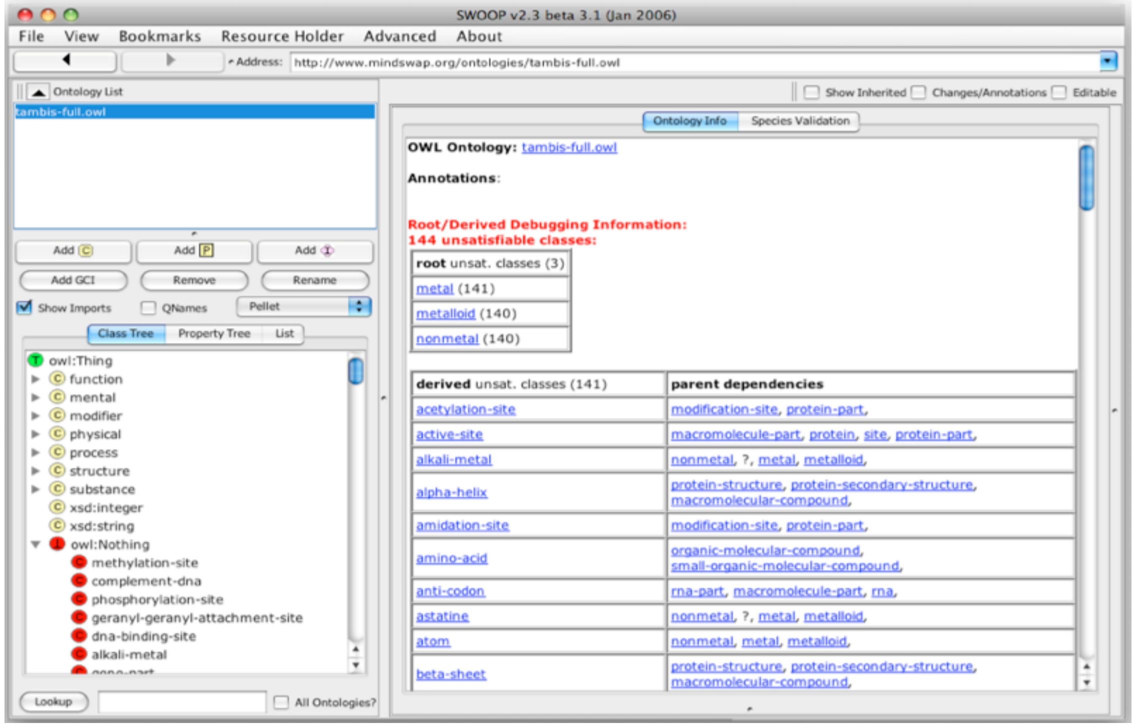
Racer pellet QuOnto QUerying ONTOlogies

KAON2 CEL semantic web framework Jena TrOWL www.trowl.eu

Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

- Editors/development environments
- Reasoners
- Explanation, justification and pinpointing



Address: http://www.mindswap.org/ontologies/tambis-full.owl

tambis-full.owl

☑ Show Imports ☐ QNames Pellet

- owl:Thing
  - function
  - mental
  - modifier
  - physical
  - process
  - structure
  - substance
  - xsd:integer
  - xsd:string
- owl:Nothing
  - methylation-site
  - complement-dna
  - phosphorylation-site
  - geranyl-geranyl-attachment-site
  - dna-binding-site
  - alkali-metal

OWL Ontology: tambis-full.owl

Annotations:

Root/Derived Debugging Information:
144 unsatisfiable classes:

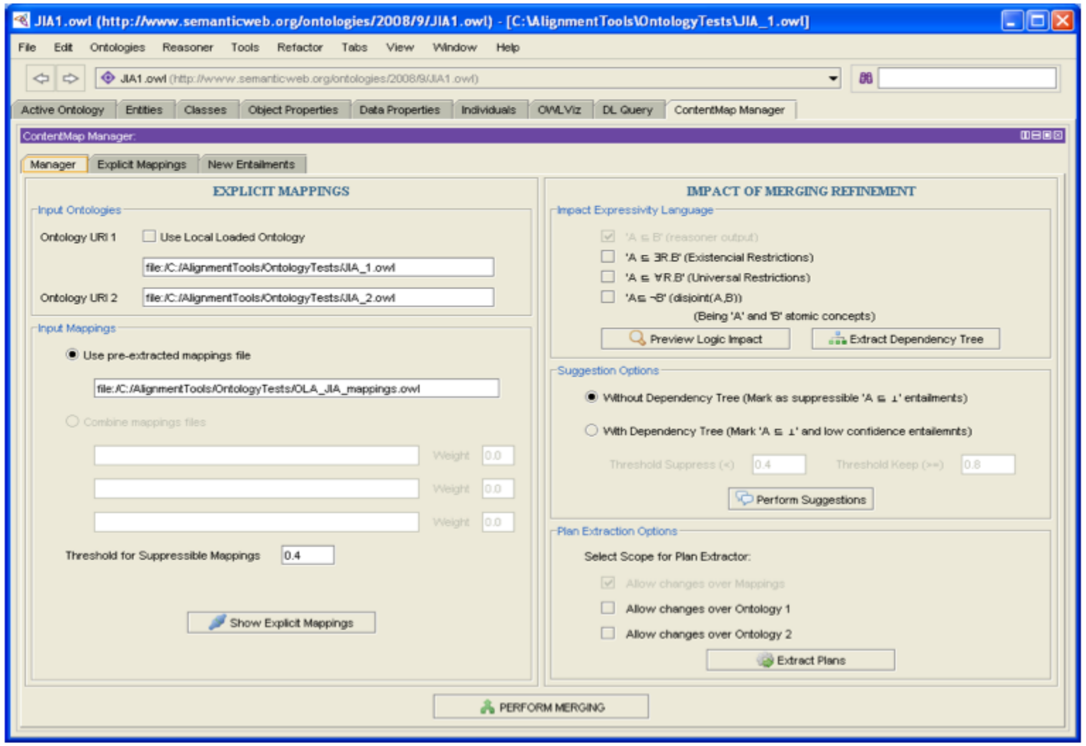
root unsat. classes (3)
metal (141)
metalloid (140)
nonmetal (140)

derived unsat. classes (141)	**parent dependencies**
acetylation-site	modification-site, protein-part,
active-site	macromolecule-part, protein, site, protein-part,
alkali-metal	nonmetal, ?, metal, metalloid,
alpha-helix	protein-structure, protein-secondary-structure, macromolecular-compound,
amidation-site	modification-site, protein-part,
amino-acid	organic-molecular-compound, small-organic-molecular-compound,
anti-codon	rna-part, macromolecule-part, rna,
astatine	nonmetal, ?, metal, metalloid,
atom	nonmetal, metal, metalloid,
beta-sheet	protein-structure, protein-secondary-structure, macromolecular-compound,

Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

- Editors/development environments
- Reasoners
- Explanation, justification and pinpointing
- Integration and modularisation



EXPLICIT MAPPINGS

Input Ontologies

Ontology URI 1 ☐ Use Local Loaded Ontology
file:/C:/AlignmentTools/OntologyTests/JIA_1.owl

Ontology URI 2 file:/C:/AlignmentTools/OntologyTests/JIA_2.owl

Input Mappings

● Use pre-extracted mappings file
file:/C:/AlignmentTools/OntologyTests/OLA_JIA_mappings.owl

○ Combine mappings files
Weight 0.0
Weight 0.0
Weight 0.0

Threshold for Suppressible Mappings 0.4

IMPACT OF MERGING REFINEMENT

Impact Expressivity Language

☑ 'A ⊑ B' (reasoner output)
☐ 'A ⊑ ∃R.B' (Existencial Restrictions)
☐ 'A ⊑ ∀R.B' (Universal Restrictions)
☐ 'A⊑ ¬B' (disjoint(A,B))
(Being 'A' and 'B' atomic concepts)

Suggestion Options

● Without Dependency Tree (Mark as suppressible 'A ⊑ ⊥' entailments)
○ With Dependency Tree (Mark 'A ⊑ ⊥' and low confidence entailemnts)
Threshold Suppress (<) 0.4 Threshold Keep (>=) 0.8

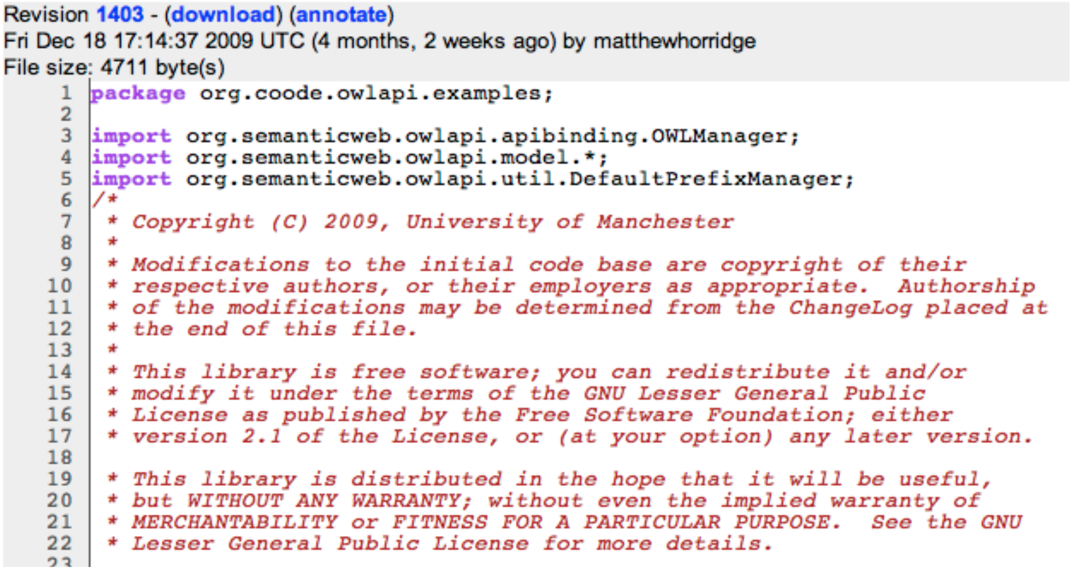
Plan Extraction Options

Select Scope for Plan Extractor:
☑ Allow changes over Mappings
☐ Allow changes over Ontology 1
☐ Allow changes over Ontology 2

Tools, Tools, Tools

Major benefit of OWL has been huge increase in range and sophistication of tools and infrastructure:

- Editors/development environments
- Reasoners
- Explanation, justification and pinpointing
- Integration and modularisation
- APIs, in particular the **OWL API**



Revision 1403 - (download) (annotate)
Fri Dec 18 17:14:37 2009 UTC (4 months, 2 weeks ago) by matthewhorridge
File size: 4711 byte(s)

```
package org.coode.owlapi.examples;

import org.semanticweb.owlapi.apibinding.OWLManager;
import org.semanticweb.owlapi.model.*;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
/*
 * Copyright (C) 2009, University of Manchester
 *
 * Modifications to the initial code base are copyright of their
 * respective authors, or their employers as appropriate.  Authorship
 * of the modifications may be determined from the ChangeLog placed at
 * the end of this file.
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.

 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
```

OWL 2 Profiles and Reasoning

OWL 2 "DL" (full language)

- Standard technique is refutation via model construction:

  $$\mathcal{O} \models Q(x) \text{ iff } \mathcal{O} \cup \{\neg Q(x)\} \models \bot$$

  - Try to refute by constructing model of $\mathcal{O} \cup \{\neg Q(x)\}$
  - Model construction very similar to DB CHASE techniques
- E.g., HermiT, FaCT++, Pellet, ...
- Scalability issues for query answering (number and size of models)
  - but many optimisations are possible

OWL 2 Profiles and Reasoning

OWL 2 EL

- A (near maximal) fragment of OWL 2 such that
  - Satisfiability checking is in PTime (**PTime-Complete**)
  - Data complexity of query answering also PTime-Complete
- Based on $\mathcal{EL}$ family of description logics
- Can exploit **“saturation”** reasoning techniques
  - Deductive inference rules used to materialise all relevant schema axioms (e.g., atomic subsumption axioms)
- E.g., CB, CEL, Snorocket, ...

OWL 2 Profiles and Reasoning

OWL 2 QL

- A (near maximal) fragment of OWL 2 such that
  - Data complexity of conjunctive query answering in **AC^0**
- Based on **DL-Lite** family of description logics
- Can exploit **query rewriting** based reasoning technique
  - Ontology axioms treated as backward chaining rules and used to expand query
  - Data storage and query evaluation can be delegated to standard RDBMS
- E.g., QuOnto, **Oracle**

OWL 2 Profiles and Reasoning

OWL 2 RL

- A (near maximal) fragment of OWL 2 such that
  - Reasoning can be implemented via forward chaining rule engines
- Can exploit **materialisation** based reasoning technique
  - Ontology plus standard set of forward chaining inference rules used to materialise all relevant facts (data)
  - Can be implemented on top of standard RDBMS with rule engine
- E.g., Jena, Sesame, Owlim, **Oracle**

OWL 2 Profiles and Reasoning

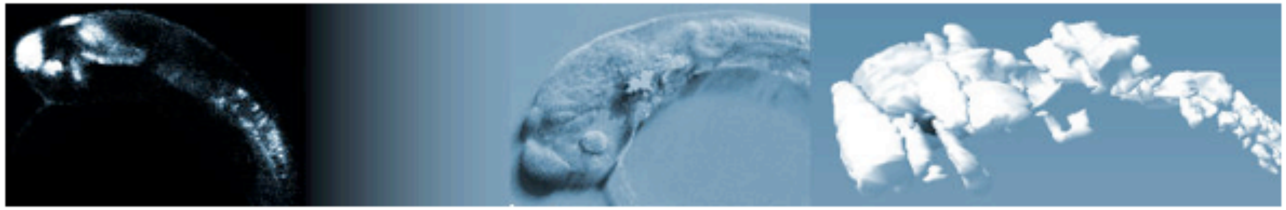
Oracle Database Semantic Technologies

- Scalable, secure, and standard-compliant platform for storage, inference, and querying of semantic data
  - RDF/RDFS/OWL/SKOS/SPARQL
  - OWL RL and EL (SNOMED support)
  - semantic document indexing framework that works with 3rd party entity extraction engines
  - set of easy to use Java programming APIs (Jena Adapter/ Sesame Adapter)

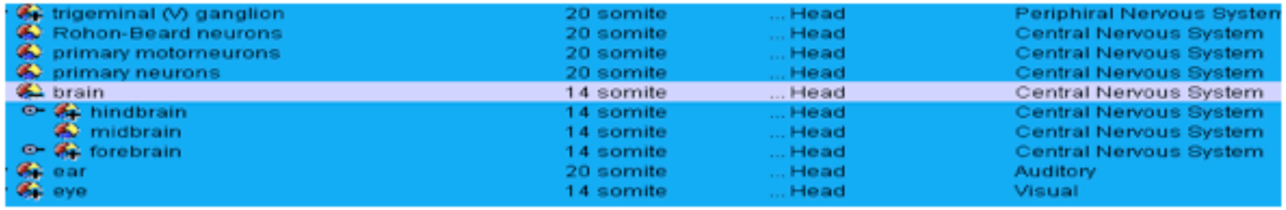
Motivating Applications

- OWL playing **key role** in increasing number & range of applications
  - eScience

3D Analysis of Patterns of Gene Expression

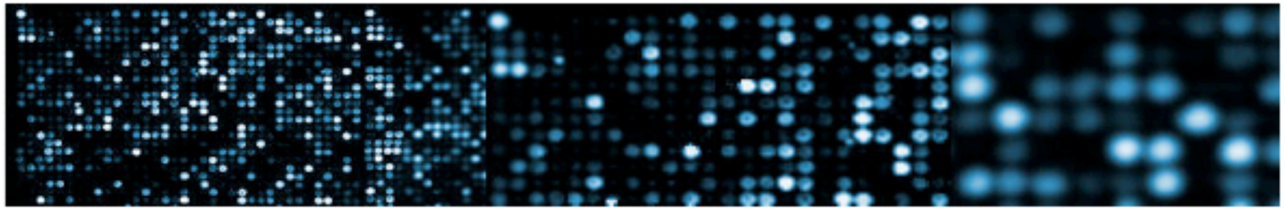


Ontology of Zebrafish Developmental Anatomy



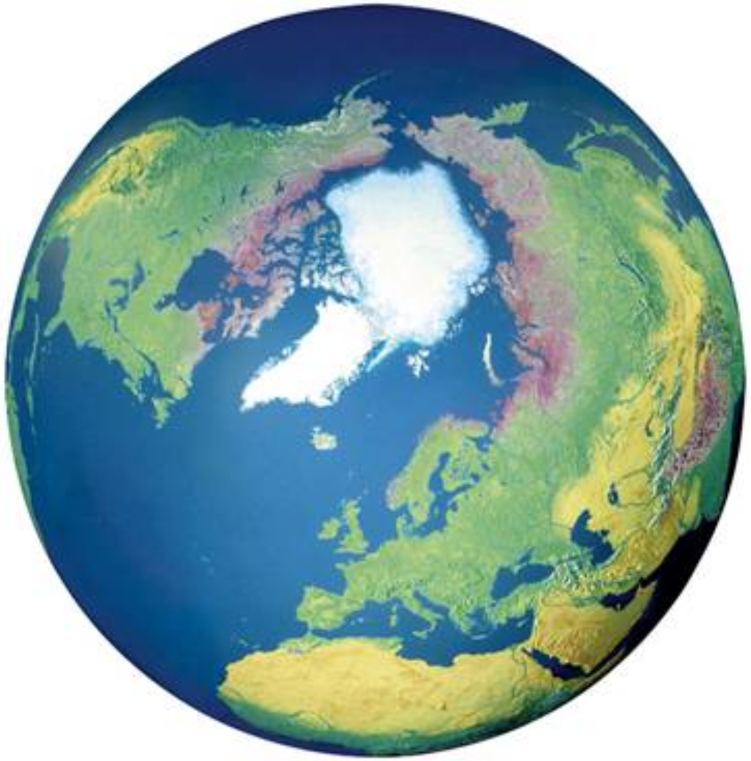
trigeminal (V) ganglion	20 somite	... Head	Periphiral Nervous Systen
Rohon-Beard neurons	20 somite	... Head	Central Nervous System
primary motorneurons	20 somite	... Head	Central Nervous System
primary neurons	20 somite	... Head	Central Nervous System
brain	14 somite	... Head	Central Nervous System
hindbrain	14 somite	... Head	Central Nervous System
midbrain	14 somite	... Head	Central Nervous System
forebrain	14 somite	... Head	Central Nervous System
ear	20 somite	... Head	Auditory
eye	14 somite	... Head	Visual

Integration of Heterogeneous gene expression data



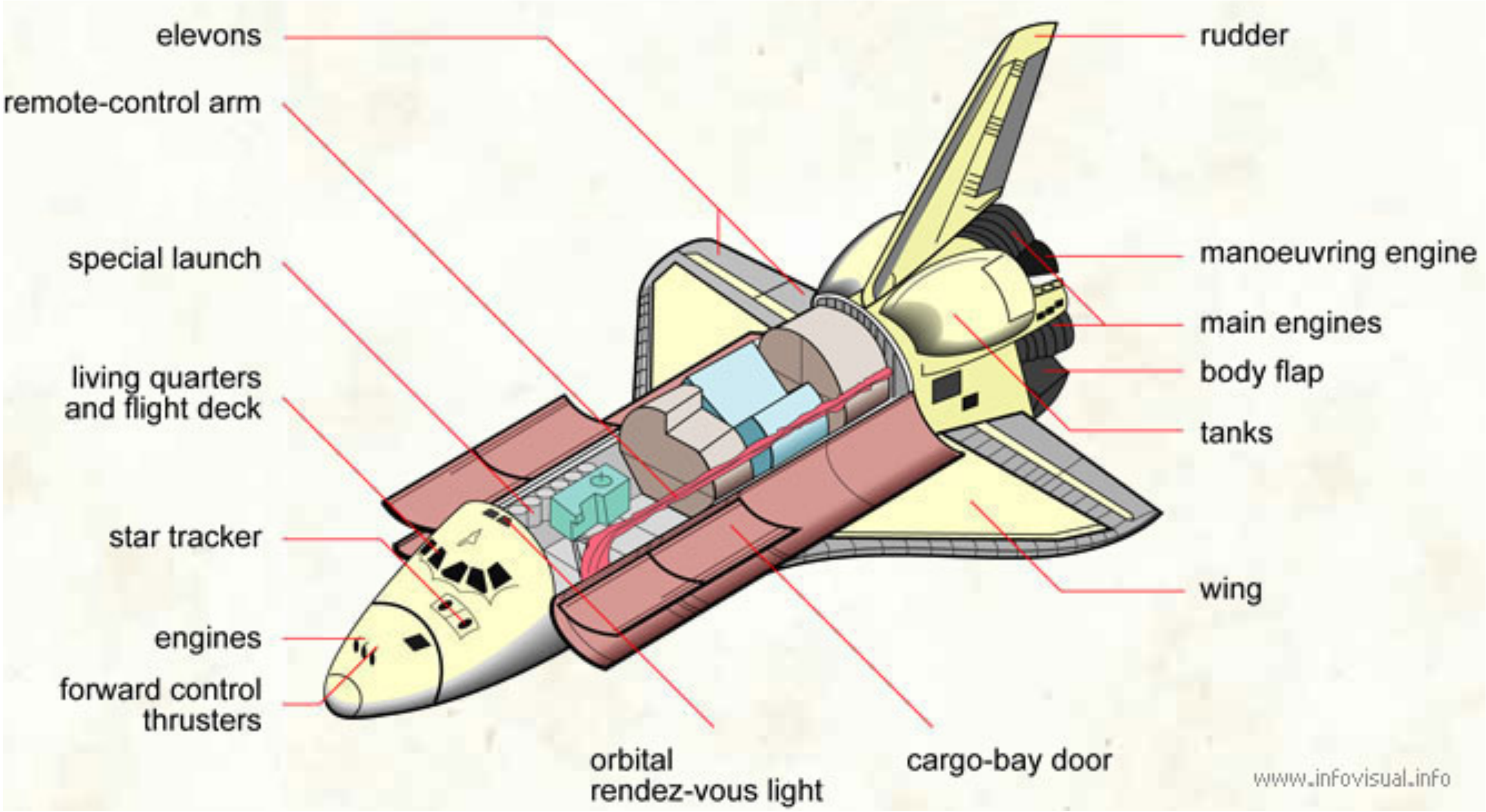
Motivating Applications

- OWL playing **key role** in increasing number & range of applications
  - eScience, geography



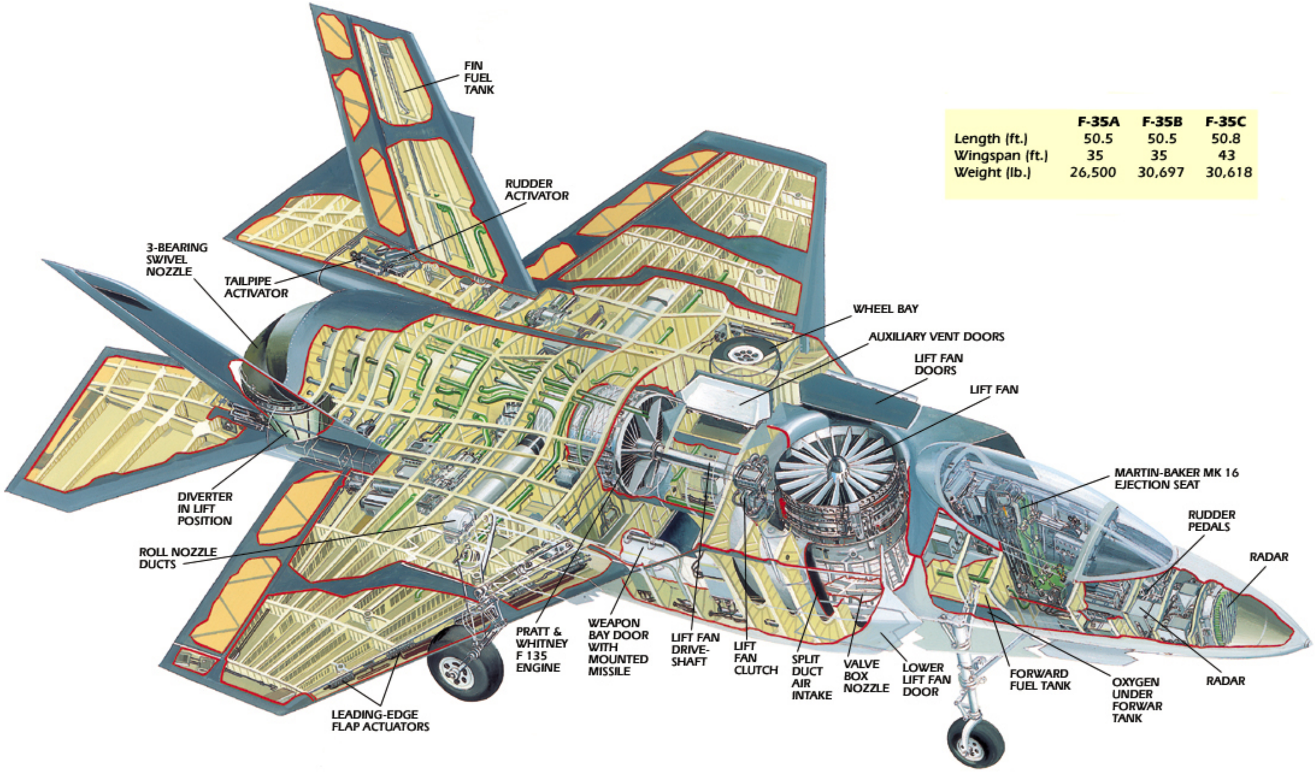
Motivating Applications

- OWL playing **key role** in increasing number & range of applications
  - eScience, geography, engineering,



Motivating Applications

- OWL playing **key role** in increasing number & range of applications
  - eScience, geography, engineering, defence, …



	F-35A	F-35B	F-35C
Length (ft.)	50.5	50.5	50.8
Wingspan (ft.)	35	35	43
Weight (lb.)	26,500	30,697	30,618

Motivating Applications: HCLS

- **OBO foundry** includes more than 100 biological and biomedical ontologies
- **Siemens** “actively building OWL based clinical solutions”
- OWL tools used to find and repair critical errors in ontology used at **Columbia Presbyterian**
- **SNOMED-CT** (Clinical Terms) ontology
  - used in healthcare systems of more than 15 countries, including Australia, Canada, Denmark, Spain, Sweden and the UK
  - also used by major US providers, e.g., Kaiser Permanente
  - ontology provides common vocabulary for recording clinical data

Motivating Applications: BBC

Latest matches

NED 2-1 BRA
Highlights & report

URU 1-1 GHA
Highlights & report

ARG 0-4 GER
Highlights & report

PAR 0-1 ESP

England

England 1-1 United States — Match report
Saturday, 12 June

England 0-0 Algeria — Match report
Friday, 18 June

Slovenia 0-1 England — Match report
Wednesday, 23 June

Germany 4-1 England — Match report
Sunday, 27 June

A | B | C | D | E | F | G | H

Group C Teams	W	D	L	GD	PTS
USA	1	2	0	1	5
England	1	2	0	1	5
Slovenia	1	1	1	0	4
Algeria	0	1	2	-2	1

Latest stories

Gerrard commits future to England NEW

Pressure got to Rooney - Ferguson

- England sponsorship likely to end
- Capello to remain England manager
- Mueller blames England imbalance
- Capello receives Gartside backing
- FA unfit for purpose says Caborn
- England's fear of crossing borders
- England duo bypass London event
- Barwick baffled by dismal England

Features

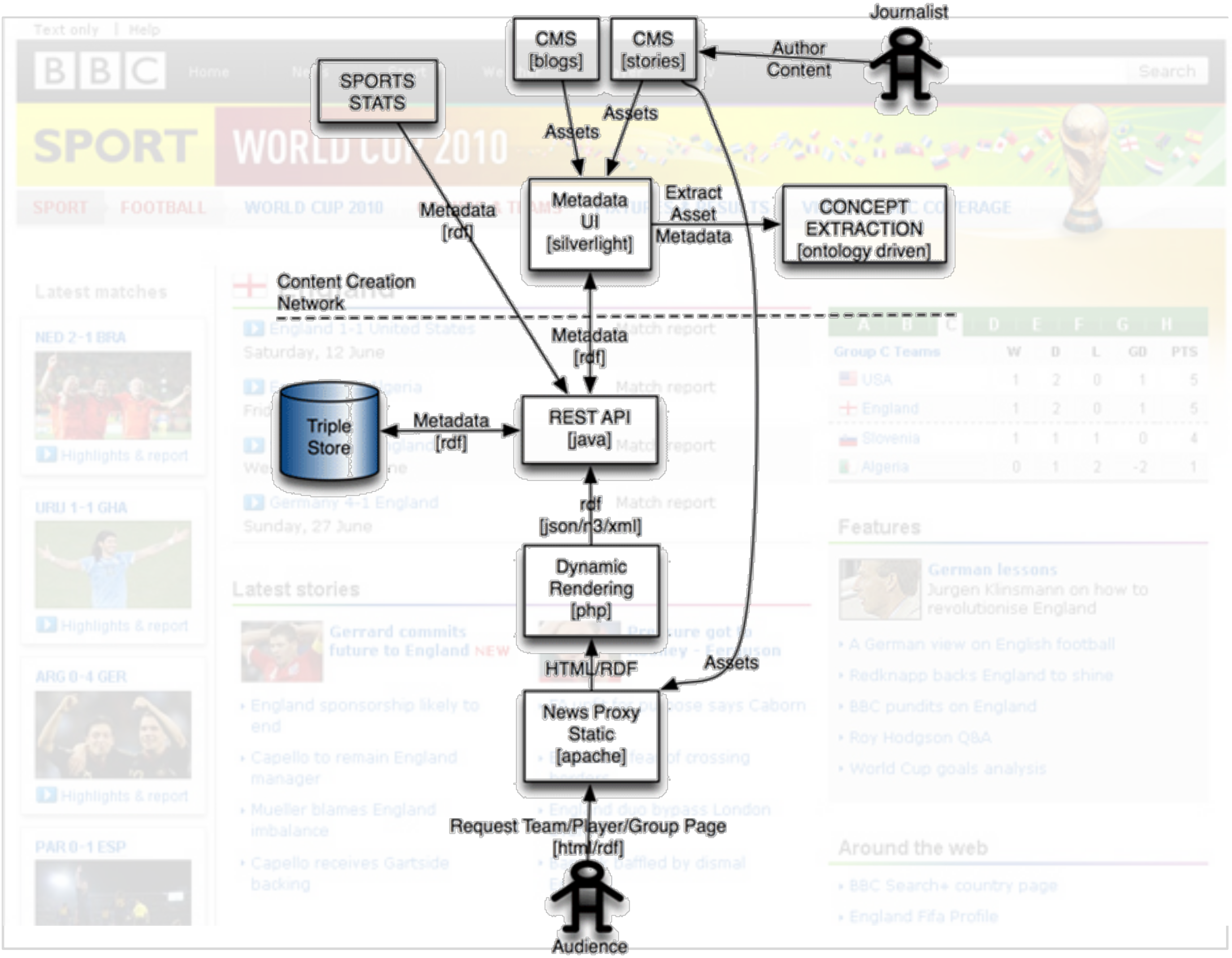
German lessons
Jurgen Klinsmann on how to revolutionise England

- A German view on English football
- Redknapp backs England to shine
- BBC pundits on England
- Roy Hodgson Q&A
- World Cup goals analysis

Around the web

- BBC Search+ country page
- England Fifa Profile

Motivating Applications: BBC



Motivating Applications: BBC

Ontology -v- Database

Obvious Database Analogy

- Ontology axioms analogous to DB **schema**
  - Schema describes structure of and constraints on data
- Ontology facts analogous to DB **data**
  - Instantiates schema
  - Consistent with schema constraints
- But there are also important differences…

Obvious Database Analogy

Database:

- Closed world assumption (**CWA**)
  - Missing information treated as false
- Unique name assumption (**UNA**)
  - Each individual has a single, unique name
- Schema behaves as **constraints** on structure of data
  - Define legal database states

Ontology:

- Open world assumption (**OWA**)
  - Missing information treated as unknown
- **No UNA**
  - Individuals may have more than one name
- Ontology axioms behave like **implications** (inference rules)
  - Entail implicit information

Database -v- Ontology

E.g., given the following **ontology/schema**:

$\text{HogwartsStudent} \equiv \text{Student} \sqcap \exists\, \text{attendsSchool.Hogwarts}$

$\text{HogwartsStudent} \sqsubseteq \forall \text{hasPet.(Owl or Cat or Toad)}$

$\text{hasPet} \equiv \text{isPetOf}^{-}$ (i.e., hasPet inverse of isPetOf)

$\exists \text{hasPet}.\top \sqsubseteq \text{Human}$ (i.e., domain of hasPet is Human)

$\text{Phoenix} \sqsubseteq \forall \text{isPetOf.Wizard}$ (i.e., only Wizards have Phoenix pets)

$\text{Muggle} \sqsubseteq \neg \text{Wizard}$ (i.e., Muggles and Wizards are disjoint)

Database -v- Ontology

And the following **facts/data**:

HarryPotter: Wizard
DracoMalfoy: Wizard
HarryPotter hasFriend RonWeasley
HarryPotter hasFriend HermioneGranger
HarryPotter hasPet Hedwig

Query: Is Draco Malfoy a friend of HarryPotter?

- DB: No
- Ontology: Don’t Know

  OWA (didn’t say Draco was not Harry’s friend)

Database -v- Ontology

And the following **facts/data**:

HarryPotter: Wizard
DracoMalfoy: Wizard
HarryPotter hasFriend RonWeasley
HarryPotter hasFriend HermioneGranger
HarryPotter hasPet Hedwig

Query: How many friends does Harry Potter have?

- DB: 2
- Ontology: at least 1

  No UNA (Ron and Hermione may be 2 names for same person)

Database -v- Ontology

And the following **facts/data**:

HarryPotter: Wizard
DracoMalfoy: Wizard
HarryPotter hasFriend RonWeasley
HarryPotter hasFriend HermioneGranger
HarryPotter hasPet Hedwig

➔ **RonWeasley ≠ HermioneGranger**

Query: How many friends does Harry Potter have?

- DB: 2
- Ontology: at least 2

    OWA (Harry may have more friends we didn’t mention yet)

Database -v- Ontology

And the following **facts/data**:

HarryPotter: Wizard
DracoMalfoy: Wizard
HarryPotter hasFriend RonWeasley
HarryPotter hasFriend HermioneGranger
HarryPotter hasPet Hedwig

RonWeasley $\neq$ HermioneGranger

➔ **$\text{HarryPotter}: \forall \text{hasFriend}.\{\text{RonWeasley}\} \sqcup \{\text{HermioneGranger}\}$**

Query: How many friends does Harry Potter have?

- DB: 2
- Ontology: 2!

Database -v- Ontology

$\exists hasPet.\top \sqsubseteq Human$
$Phoenix \sqsubseteq \forall isPetOf.Wizard$

Inserting new facts/data:

Dumbledore: Wizard
Fawkes: Phoenix
Fawkes isPetOf Dumbledore

What is the response from DBMS?

- Update rejected: **constraint violation**
  - Domain of hasPet is Human; Dumbledore is not Human (CWA)

What is the response from Ontology reasoner?

- **Infer** that Dumbledore is Human (domain restriction)
- Also infer that Dumbledore is a Wizard (only a Wizard can have a pheonix as a pet)

DB Query Answering

- Schema plays **no role**
  - Data must explicitly satisfy schema constraints
- Query answering amounts to **model checking**
  - I.e., a “look-up” against the data
- Can be very **efficiently implemented**
  - Worst case complexity is low (logspace) w.r.t. size of data

Ontology Query Answering

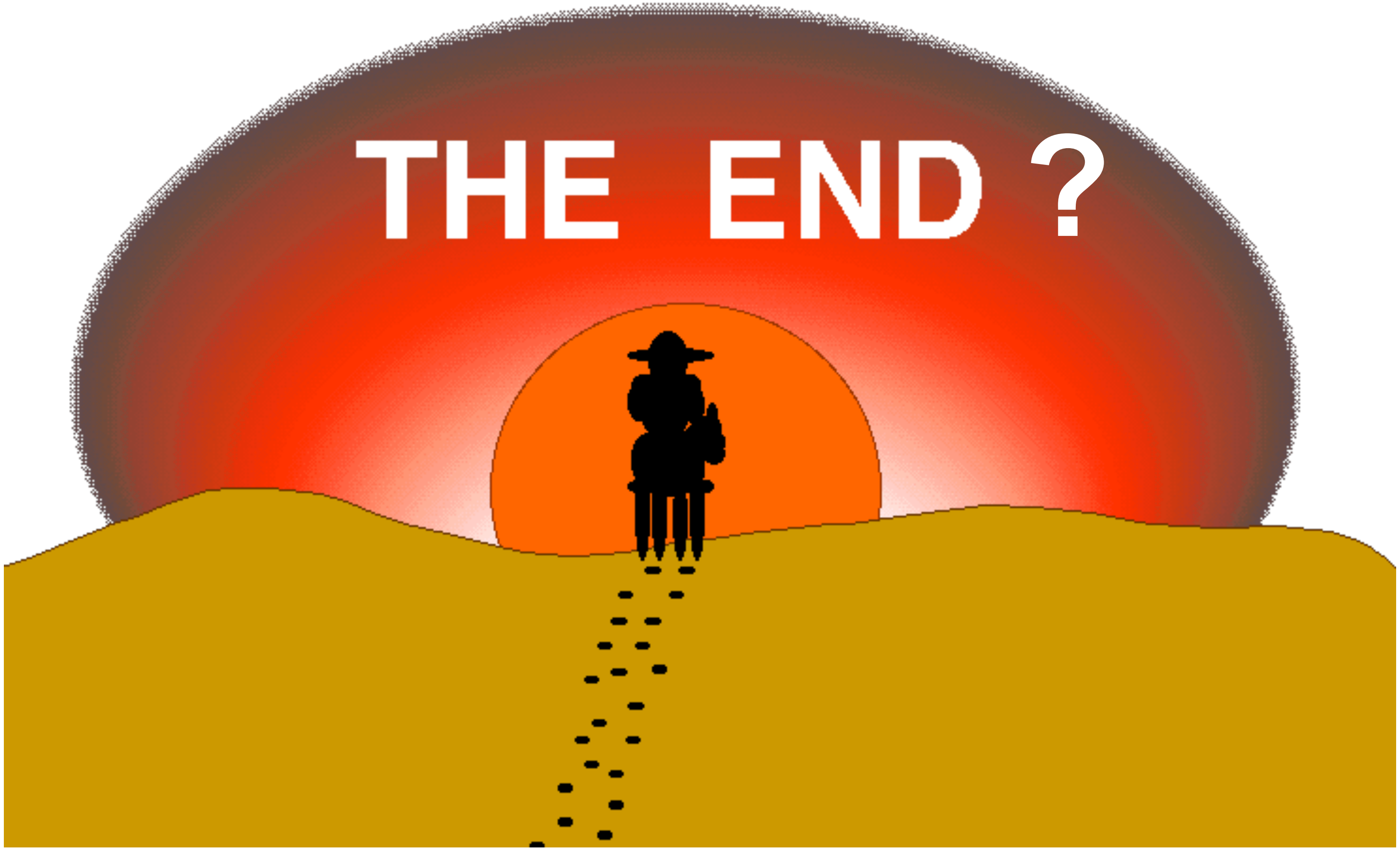
- Ontology axioms play a powerful and **crucial role**
  - Answer may include implicitly derived facts
  - Can answer conceptual as well as extensional queries
    - E.g., Can a Muggle have a Phoenix for a pet?
- Query answering amounts to **theorem proving**
  - I.e., logical entailment
- May have very **high worst case complexity**
  - E.g., for OWL, NP-hard w.r.t. size of data (upper bound is an open problem)
  - Implementations may still behave well in typical cases
  - Fragments/profiles may have much better complexity

Ontology Based Information Systems

- Analogous to **relational database management systems**
  - Ontology ≈ schema; instances ≈ data
- Some important (**dis**)**advantages**
  - \+ (Relatively) easy to maintain and update schema
    - Schema plus data are integrated in a logical theory
  - \+ Query answers reflect both schema and data
  - \+ Can deal with incomplete information
  - \+ Able to answer both intensional and extensional queries
  - – Semantics can seem counter-intuitive, particularly w.r.t. data
    - Open -v- closed world; axioms -v- constraints
  - – Query answering (logical entailment) may be much more difficult
    - Can lead to scalability problems with expressive logics

Ontology Based Information Systems

- Analogous to **relational d** **anagement systems**
  – Ontology ≈ schem
- Some important
  + (Relatively)
    - Schema
  + Query ans
  + Can deal
  + Able to an eries
  – Semantics w.r.t. data
    - Open -v- c
  – Query answering nuch more difficult
    - Can lead to scalab ssive logics


THE END ?

Ongoing Research

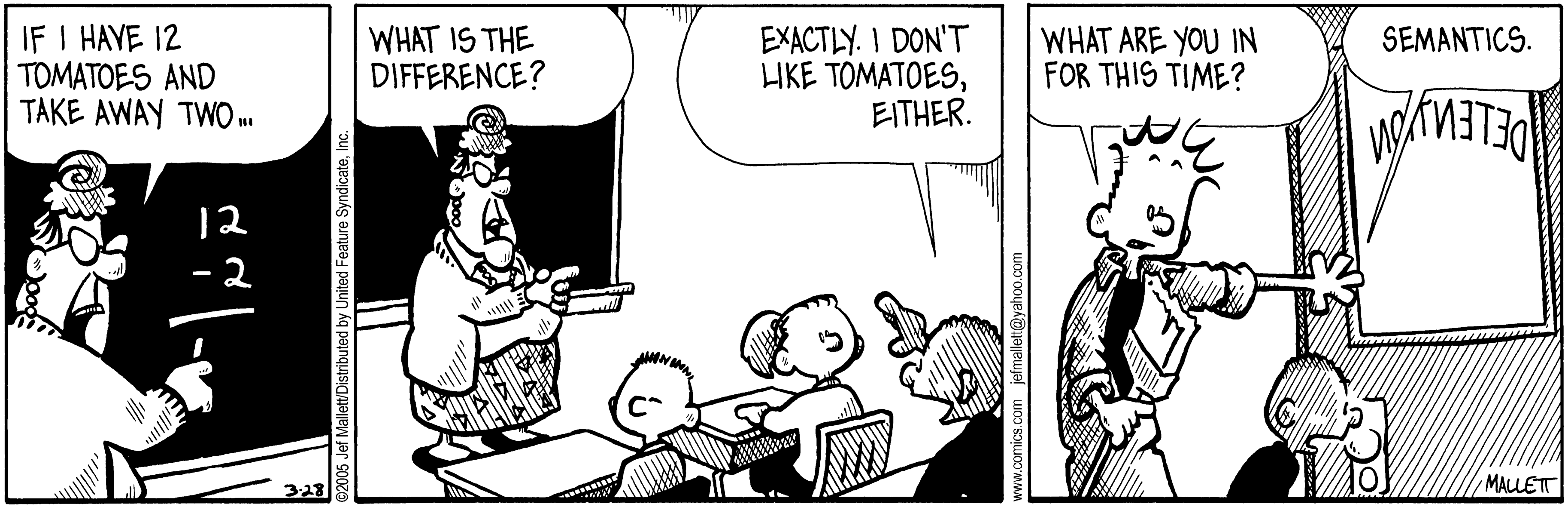
- Query answering
  - [Kontchakov et al], [Konev et al], [Baader et al]
- Diagnosis and repair
  - [Horridge et al], [Peñaloza et al]
- Extensions
  - [Motik et al], [Artale et al]
- Optimisation/Profiles
  - [Kazakov], [Glimm et al], [Faddoul et al], [Savo et al]
- ...

Ongoing Standardisation Efforts

- Standardised query language
  - SPARQL standard for RDF
  - Currently being extended for OWL, see http://www.w3.org/TR/sparql11-entailment/
- RDF
  - Revision currently being considered, see http://www.w3.org/2009/12/rdf-ws/

Thank you for listening

Thank you for listening



FRAZZ: © Jeff Mallett/Dist. by United Feature Syndicate, Inc.

Any questions?