Description Logic: A Formal Foundation for Languages and Tools

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

• Description Logic Basics
  – Syntax and semantics

• Description Logics and Ontology Languages
  – OWL ontology language
  – Ontology -v- Database

• Description Logic Reasoning
  – Reasoning services
  – Reasoning techniques

• Recent and Future work
DL Basics
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
    • Closely related to Guarded Fragment
  – Provision of inference services
    • Practical decision procedures (algorithms) for key problems (satisfiability, subsumption, etc)
    • Implemented systems (highly optimised)
and now:

A Word from our Sponsors
Crash Course in (simplified) FOL

• Syntax
  – Non-logical symbols (signature)
    • Constants: Felix, MyMat
    • Predicates(arity): Animal(1), Cat(1), has-color(2), sits-on(2)
  – Logical symbols:
    • Variables: x, y
    • Operators: ∧, ∨, →, ¬, ...
    • Quantifiers: ∃, ∀
    • Equality: =
  – Formulas:
    • Cat(Felix), Mat(MyMat), sits-on(Felix, MyMat)
    • Cat(x), Cat(x) ∨ Human(x), ∃y.Mat(y) ∧ sits-on(x, y)
    • ∀x.Cat(x) → Animal(x), ∀x.Cat(x) → (∃y.Mat(y) ∧ sits-on(x, y))
Crash Course in (simplified) FOL

• Semantics
Crash Course in (simplified) FOL

• Semantics

Why should I care about semantics? -- In fact I heard that a little goes a long way!
Crash Course in (simplified) FOL

• Semantics

Why should I care about semantics? -- In fact I heard that a little goes a long way!

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.
Crash Course in (simplified) FOL

• Semantics

Why should I care about semantics? -- In fact I heard that a little goes a long way!

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.

That’s OK, but I don’t get paid for philosophy.
Crash Course in (simplified) FOL

- Semantics

Why should I care about semantics? -- In fact I heard that a little goes a long way!

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.

That’s OK, but I don’t get paid for philosophy.

From a practical POV, we need to define relationships (like entailment) between logical statements -- without such a definition we can’t spec software such as a reasoner.
Crash Course in (simplified) FOL

- Semantics

In FOL we define the semantics in terms of models (a model theory). A model is supposed to be an analogue of (part of) the world being modeled. FOL uses a very simple kind of model, in which “objects” in the world (not necessarily physical objects) are modeled as elements of a set, and relationships between objects are modeled as sets of tuples.
Crash Course in (simplified) FOL

• Semantics

In FOL we define the semantics in terms of models (a model theory). A model is supposed to be an analogue of (part of) the world being modeled. FOL uses a very simple kind of model, in which “objects” in the world (not necessarily physical objects) are modeled as elements of a set, and relationships between objects are modeled as sets of tuples.

Note that this is exactly the same kind of model as used in a database: objects in the world are modeled as values (elements) and relationships as tables (sets of tuples).
Crash Course in (simplified) FOL

- Semantics
  - Model: a pair \( \langle D, \cdot^I \rangle \) with \( D \) a non-empty set and \( \cdot^I \) an interpretation
    - \( C^I \) is an element of \( D \) for \( C \) a constant
    - \( v^I \) is an element of \( D \) for \( v \) a variable
    - \( P^I \) is a subset of \( D^n \) for \( P \) a predicate of arity \( n \)
  - E.g., \( D = \{ a, b, c, d, e, f \} \), and
    - \( \text{Felix}^I = a \)
    - \( \text{MyMat}^I = b \)
    - \( \text{Cat}^I = \{ a, c \} \)
    - \( \text{Mat}^I = \{ b, e \} \)
    - \( \text{Animal}^I = \{ a, c, d \} \)
    - \( \text{sits-on}^I = \{ \langle a, b \rangle, \langle c, e \rangle \} \)
Crash Course in (simplified) FOL

• **Semantics**
  - **Evaluation:** truth value in a given model $M = \langle D, \cdot^I \rangle$
    - $P(t_1, \ldots, t_n)$ is *true* iff $\langle t_1^I, \ldots, t_n^I \rangle \in P^I$
    - $A \land B$ is *true* iff $A$ is *true* and $B$ is *true*
      - $\neg A$ is *true* iff $A$ is not *true*
  - **E.g.,**
    - $\text{Cat}(\text{Felix})$  $\text{true}$
    - $\text{Cat}(\text{MyMat})$  $\text{false}$
    - $\neg \text{Mat}(\text{Felix})$  $\text{true}$
    - $\text{sits-on}(\text{Felix, MyMat})$  $\text{true}$
    - $\text{Mat}(\text{Felix}) \lor \text{Cat}(\text{Felix})$  $\text{true}$

- **D = \{a, b, c, d, e, f\}**
- $\text{Felix}^I = a$
- $\text{MyMat}^I = b$
- $\text{Cat}^I = \{a, c\}$
- $\text{Mat}^I = \{b, e\}$
- $\text{Animal}^I = \{a, c, d\}$
- $\text{sits-on}^I = \{(a, b), (c, e)\}$
Crash Course in (simplified) FOL

• Semantics
  – Evaluation: truth value in a given model $M = \langle D, \cdot^I \rangle$
    • $\exists x. A$ is true iff exists $\cdot^I'$ s.t. $\cdot^I$ and $\cdot^I'$ differ only w.r.t. $x$, and $A$ is true w.r.t. $\langle D, \cdot^I' \rangle$
    • $\forall x. A$ is true iff for all $\cdot^I'$ s.t. $\cdot^I$ and $\cdot^I'$ differ only w.r.t. $x$, $A$ is true w.r.t. $\langle D, \cdot^I' \rangle$

E.g.,

$\exists x. \text{Cat}(x)$
$\forall x. \text{Cat}(x)$
$\exists x. \text{Cat}(x) \land \text{Mat}(x)$
$\forall x. \text{Cat}(x) \rightarrow \text{Animal}(x)$
$\forall x. \text{Cat}(x) \rightarrow (\exists y. \text{Mat}(y) \land \text{sits-on}(x, y))$

$D = \{a, b, c, d, e, f\}$
$\text{Felix}^I = a$
$\text{MyMat}^I = b$
$\text{Cat}^I = \{a, c\}$
$\text{Mat}^I = \{b, e\}$
$\text{Animal}^I = \{a, c, d\}$
$s\text{its-on}^I = \{\langle a, b \rangle, \langle c, e \rangle\}$
Crash Course in (simplified) FOL

• Semantics
  – Given a model $M$ and a formula $F$, $M$ is a model of $F$ (written $M \models F$) iff $F$ evaluates to true in $M$
  – A formula $F$ is **satisfiable** iff there exists a model $M$ s.t. $M \models F$
  – A formula $F$ **entails** another formula $G$ (written $F \models G$) iff every model of $F$ is also a model of $G$ (i.e., $M \models F$ implies $M \models G$)

E.g.,

\[
M \models \exists x.\text{Cat}(x) \\
M \not\models \forall x.\text{Cat}(x) \\
M \not\models \exists x.\text{Cat}(x) \land \text{Mat}(x) \\
M \models \forall x.\text{Cat}(x) \rightarrow \text{Animal}(x) \\
M \models \forall x.\text{Cat}(x) \rightarrow (\exists y.\text{Mat}(y) \land \text{sits-on}(x, y))
\]

\[
D = \{a, b, c, d, e, f\} \\
\text{Felix}^I = a \\
\text{MyMat}^I = b \\
\text{Cat}^I = \{a, c\} \\
\text{Mat}^I = \{b, e\} \\
\text{Animal}^I = \{a, c, d\} \\
\text{sits-on}^I = \{(a, b), (c, e)\}
\]
Crash Course in (simplified) FOL

• Semantics
  – Given a model $M$ and a formula $F$, $M$ is a model of $F$ (written $M \models F$) iff $F$ evaluates to true in $M$
  – A formula $F$ is **satisfiable** iff there exists a model $M$ s.t. $M \models F$
  – A formula $F$ **entails** another formula $G$ (written $F \models G$) iff every model of $F$ is also a model of $G$ (i.e., $M \models F$ implies $M \models G$)

E.g.,

- $\checkmark$ Cat(Felix) $\models \exists x.\text{Cat}(x)$ (Cat(Felix) $\land \neg \exists x.\text{Cat}(x)$ is not satisfiable)
- $\checkmark$ $(\forall x.\text{Cat}(x) \rightarrow \text{Animal}(x)) \land \text{Cat}(\text{Felix}) \models \text{Animal}(\text{Felix})$
- $\checkmark$ $(\forall x.\text{Cat}(x) \rightarrow \text{Animal}(x)) \land \neg \text{Animal}(\text{Felix}) \models \neg \text{Cat}(\text{Felix})$
- $\times$ Cat(Felix) $\models \forall x.\text{Cat}(x)$
- $\times$ sits-on(Felix, Mat1) $\land$ sits-on(Tiddles, Mat2) $\models \neg$ sits-on(Felix, Mat2)
- $\times$ sits-on(Felix, Mat1) $\land$ sits-on(Tiddles, Mat1) $\models \exists \geq 2 x.\text{sits-on}(x, \text{Mat1})$
Decidable Fragments

• FOL (satisfiability) well known to be undecidable
  - A sound, complete and terminating algorithm is impossible

• Interesting decidable fragments include, e.g.,
  - C2: FOL with 2 variables and Counting quantifiers ($\exists^{\geq n}, \exists^{\leq n}$)
    • Counting quantifiers abbreviate pairwise (in-) equalities, e.g.:
      - $\exists^{\geq 3}x.\text{Cat}(x)$ equivalent to
        - $\exists x, y, z.\text{Cat}(x) \land \text{Cat}(y) \land \text{Cat}(z) \land x \neq y \land x \neq z \land y \neq z$
      - $\exists^{\leq 2}x.\text{Cat}(x)$ equivalent to
        - $\forall x, y, z.\text{Cat}(x) \land \text{Cat}(y) \land \text{Cat}(z) \rightarrow x = y \lor x = z \lor y = z$
  - Propositional modal and description logics
  - Guarded fragment
Back to our Scheduled Program
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 (\(\cap, \cup, \neg\))
    • Restricted form of quantifiers (\(\exists, \forall\))
    • Counting (\(\geq, \leq, =\))
    • …
DL Syntax

• Concept expressions, e.g.,
  – Doctor ⊔ Lawyer
  – Rich ⊓ Happy
  – Cat ⊓ ∃sits-on.Mat

• Equivalent to FOL formulae with one free variable
  – Doctor(x) ∨ Lawyer(x)
  – Rich(x) ∧ Happy(x)
  – ∃y.(Cat(x) ∧ 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.,
  - $\text{loves}^-$
  - $\text{hasParent} \circ \text{hasBrother}$

• Equivalent to FOL formulae with two free variables
  - $\text{loves}(y, x)$
  - $\exists z. (\text{hasParent}(x, z) \land \text{hasBrother}(z, y))$
DL Syntax

• “Schema” Axioms, e.g.,
  - Rich $\subseteq \neg$Poor  
    (concept inclusion)
  - Cat $\cap \exists$sits-on.Mat $\subseteq$ Happy
    (concept inclusion)
  - BlackCat $\equiv$ Cat $\cap \exists$hasColour.Black
    (concept equivalence)
  - sits-on $\subseteq$ touches
    (role inclusion)
  - Trans(part-of)
    (transitivity)

• Equivalent to (particular form of) FOL sentence, e.g.,
  - $\forall x.(\text{Rich}(x) \rightarrow \neg\text{Poor}(x))$
  - $\forall x.(\text{Cat}(x) \land \exists y.(\text{sits-on}(x,y) \land \text{Mat}(y)) \rightarrow \text{Happy}(x))$
  - $\forall x.(\text{BlackCat}(x) \leftrightarrow (\text{Cat}(x) \land \exists y.(\text{hasColour}(x,y) \land \text{Black}(y))))$
  - $\forall x,y.(\text{sits-on}(x,y) \rightarrow \text{touches}(x,y))$
  - $\forall x,y,z.((\text{sits-on}(x,y) \land \text{sits-on}(y,z)) \rightarrow \text{sits-on}(x,z))$
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.:

{Doctor ⊆ Person,
 Parent ≡ Person ⊓ ∃hasChild.Paren
 HappyParent ≡ Parent ⊓ ∀hasChild.

• A set of facts is called an ABox:

{HappyParent(John),
 hasChild(John,Mary)}

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

Note
Facts sometimes written
John:HappyParent,
John hasChild Mary,
⟨John,Mary⟩:hasChild
The DL Family

• Many different DLs, often with “strange” names
  – E.g., \textit{EL}, \textit{ALC}, \textit{SHIQ}

• Particular DL defined by:
  – Concept operators (\textit{\Pi}, \textit{\sqcup}, \textit{\neg}, \textit{\exists}, \textit{\forall}, etc.)
  – Role operators (\textit{\cdot}, \textit{o}, etc.)
  – Concept axioms (\textit{\sqsubseteq}, \textit{\equiv}, etc.)
  – Role axioms (\textit{\sqsubseteq}, \textit{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.:

  \[ \text{Parent} \equiv \text{Person} \sqcap \exists \text{hasChild.P} \text{erson} \]
The DL Family

- **ALC** is the smallest propositionally closed DL
  - Concept operators: $\cap$, $\cup$, $\neg$, $\exists$, $\forall$
  - No role operators (only atomic roles)
  - Concept axioms: $\equiv$, $\equiv$
  - No role axioms

- E.g.:

  \[
  \text{ProudParent} \equiv \text{Person} \cap \forall \text{hasChild.}(\text{Doctor} \sqcup \exists \text{hasChild.}\text{Doctor})
  \]
The DL Family

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

\[(C \cap D)^I = C^I \cap D^I\]
\[(C \sqcup D)^I = C^I \cup D^I\]
\[(\neg C)^I = \Delta^I \setminus C^I\]
\[\{x\}^I = \{x^I\}\]
\[(\exists R.C)^I = \{x \mid \exists y.\langle x, y \rangle \in R^I \land y \in C^I\}\]
\[(\forall R.C)^I = \{x \mid \forall y.\langle x, y \rangle \in R^I \Rightarrow y \in C^I\}\]
\[(\leq nR)^I = \{x \mid \#\{y \mid \langle x, y \rangle \in R^I\} \leq n\}\]
\[(\geq nR)^I = \{x \mid \#\{y \mid \langle x, y \rangle \in R^I\} \geq n\}\]
DL Semantics

• Given a model $M = \langle D, I \rangle$
  
  - $M \models C \subseteq D$ 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 $\langle a^I, b^I \rangle \in R^I$
  - $M \models \langle T, A \rangle$ iff for every axiom $ax \in T \cup A$, $M \models ax$
DL Semantics

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

E.g.,

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

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

✓ - \(\mathcal{K} \models \text{John:Person}？\)
✓ - \(\mathcal{K} \models \text{Peter:Doctor}？\)
✓ - \(\mathcal{K} \models \text{Mary:HappyParent}？\)

- What if we add “Mary hasChild Jane”？
  \(\mathcal{K} \models \text{Peter = Jane}\)
- What if we add “HappyPerson \equiv \text{Person} \sqcap \exists \text{hasChild}.\text{Doctor}”？
  \(\mathcal{K} \not\models \text{HappyPerson} \sqsubseteq \text{Parent}\)
DL and FOL

• Most DLs are subsets of C2
  – But reduction to C2 may be (highly) non-trivial
    • Trans(R) naively reduces to $\forall x, y, z. R(x, y) \land R(y, z) \rightarrow R(x, z)$

• 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

Concept constructors:
- $f$ - functionality, $\leq 1$ (R)
- $\forall$ - (unqualified) number restrictions, $\geq n$ (R), $\leq n$ (R)
- $Q$ - qualified number restrictions, $\geq n$ (R,C), $\leq n$ (R,C)
- $O$ - nominals, $a$ or $\{a_1, \ldots, a_n\}$ ("one-of" constructor)
- $\mu$ - least fixpoint operator, $\mu X.C$
- $R\subseteq S$ - role-value-maps
- $f = g$ - agreement of functional role chains ("same-as")

Role constructors:
- $f^{-1}$ - role inverses, $R^{-}$
- $\cap$ - role intersection, $R \cap S$
- $\cup$ - role union, $R \cup S$
- $\cap$ - role complement, $\text{Ref} \cap f$
- $o$ - role chain (composition), $R \circ S$
- $*$ - reflexive-transitive closure, $R^{*}$
- $id$ - concept identity, $id(C)$

TBox is internalized in extensions of $\mathcal{ALCQI}$, see [76, Lemma 4.12], [54, p.3]
- Empty TBox
- Acyclic TBox (A=C, A is a concept name; no cycles)
- General TBox (C\D for arbitrary concepts C and D)

You have selected the Description Logic: SHOIN

Complexity of reasoning problems

<table>
<thead>
<tr>
<th>Reasoning problem</th>
<th>Complexity</th>
<th>Comments and references</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept satisfiability</td>
<td>NExpTime-complete</td>
<td>Hardness of even $\mathcal{ALCQI}$ is proved in [76, Corollary 4.13]. In that paper, the result is formulated for $\mathcal{ALCQI}$, but only number restrictions of the form $\leq 1$ (R) are used in the proof. A different proof of the NExpTime-hardness for $\mathcal{ALCQI}$ is given in [54] (even with 1 nominal, and role inverses not used in number restrictions). Upper bound for SHOIN is proved in [22, Corollary 6.31] with numbers coded in unary (for binary coding, the upper bound remains an open problem for all logics in between $\mathcal{ALCQI}$ and SHOIN). 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 SHOIN 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.</td>
</tr>
<tr>
<td>ABox consistency</td>
<td>NExpTime-complete</td>
<td>By reduction to concept satisfiability problem in presence of nominals shown in [69, Theorem 3.7].</td>
</tr>
</tbody>
</table>
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)