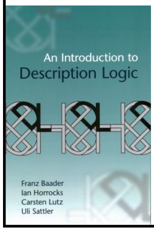
INFR11215 Knowledge Graphs

Description Logics

Jeff Z. Pan

http://knowledge-representation.org/j.z.pan/



[Reading: Baader et al., Chapters 1 and Sections 2.1 and 2.2]

Mind the Syntax

- All rich men love Jane
  - $\forall x[\mathrm{Rich}(x) \wedge \mathrm{Man}(x) \rightarrow \mathrm{love}(x,\mathrm{Jane})]$
  - $\forall x[\mathrm{Rich}(x) \wedge \mathrm{Man}(x) \supset \mathrm{love}(x,\mathrm{Jane})]$ //KR book
  - $\forall x[\mathrm{Rich}(x) \wedge \mathrm{Man}(x) \Rightarrow \mathrm{love}(x,\mathrm{Jane})]$ //DL book

  - $\mathrm{Rich} \sqcap \mathrm{Man} \sqsubseteq \exists \mathrm{love}.\{\mathrm{Jane}\}$
  - SubClassOf (

      intersectionOf(Rich, Man),

      restriction(love someValueFrom(oneOf(Jane)))

)

Basic or Complex Facts

- Jane loves both John and Jim
  - $\text{love(Jane,John)} \wedge \text{love(Jane,Jim)}$
  - Jane: $\exists\text{love.}(\{\text{John}\} \sqcup \{\text{Jim}\})$ //complex fact
  - or simply (Jane,John):love, (Jane,Jim):love //simple fact

  - Individual (Jane
    value(love, John) value(love,Jim)
) //simple fact
  - Individual (Jane
    type(restriction(love someValueFrom(oneOf(John,Jim))))
) //complex fact

Mind the Syntax

- Jane loves either John or Jim
  - $\text{love(Jane,John)} \vee \text{love(Jane,Jim)}$
  - Jane: $\exists\text{love.}(\{\text{John}\} \sqcup \{\text{Jim}\}) \sqcap \ {=}1\text{love}$

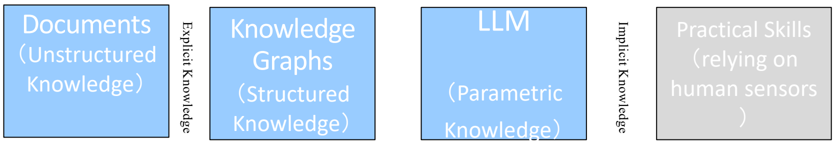
  - Individual (Jane
    type(intersectionOf
      restriction(love someValueFrom(oneOf(John,Jim)))
      restriction(love maxCardinality(1))
      restriction(love minCardinality(1))
)

Lecture Outline

- Motivation
- Overview of Description Logics (DLs)
- Semantics of DLs and Reasoning in DLs
- Practical

What is Knowledge How to Classify them

- **Knowledge: verified beliefs and practical skills** (e.g. operating an instrument)
- How to represent knowledge? How to classify knowledge?



Documents（Unstructured Knowledge）

Explicit Knowledge

Knowledge Graphs（Structured Knowledge）

LLM（Parametric Knowledge）

Implicit Knowledge

Practical Skills（relying on human sensors）

- Entity level knowledge (fact)
- Conceptual knowledge (schema)

KG vs Database

- Data in database can be seen as basic facts

Student ID	Name	take-course
p001	John	cs3019
p002	Tom	cs3023

  - [csd:p001 rdf:type csd:Student .]
  - [csd:p002 rdf:type csd:Student .]
  - [csd:p001 csd:name “John” .]
  - [csd:p002 csd:name “Tom” .]
  - [csd:p001 csd:take-course csd:cs3019 .]
  - [csd:p002 csd:take-course csd:cs3023 .]

Schema in a Database System

- A database system includes some schema constraints, such as the foreign key constraint

Student ID	Name	take-course
p001	John	~~cs3015~~
p002	Tom	cs3025

Course ID	Title	coordinator
cs3017	AIS	AS
cs3025	KBS	JP

Schema in a Knowledge Graph

1) Allow schema constraints, such as DisjointClasses (UndgStudent MastStudent)

UndgStudent ID	Name	take-course
csd:p001	John	csd:cs3014
~~csd:p002~~	Tom	csd:cs3025

MastStudent ID	Name	take-course
csd:p008	Yuan	csd:cs5010
~~csd:p002~~	Tom	csd:cs5017

Schema in a Knowledge Graph

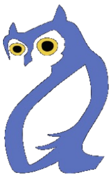
2) Allow some reasoning based on axioms (open world assumption), such as SubClassOf (MastStudent Student)

Student ID	Name	take-course
csd:p001	John	csd:cs3015
csd:p002	Tom	csd:cs3025

MastStudent ID	Name	take-course
csd:p008	Yuan	csd:cs5010
csd:p002	Tom	csd:cs5017

thus all the students include csd:p001, csd:p002, and csd:p008

Motivations



- Description Logics are the underpinning of the standard Web Ontology Language (OWL)
  - OWL v2 family
    - OWL 2 DL
    - OWL 2 EL, OWL 2 QL, OWL 2 RL
- OWL provides more expressive power than RDF (modern standard of semantic network) for
  - both terminological axioms (TBox)
  - and assertions (ABox)
- RDFa (HTML version of RDF) is e.g. used by schema.org

Google bing
schema.org
YAHOO!

```
<p>Christopher Froome was sponsored by Sky in the Tour de France.</p>
```

```
<p vocab="http://schema.org/" typeof="Person">
  <span property="name">Christopher Froome</span> was sponsored by
  <span property="sponsor" typeof="http://schema.org/Organization">
    <a property="url" href="http://www.skysports.com/">Sky</a></span> in the Tour de France.
</p>
```

```
<p itemscope itemprop="Person" itemtype="http://schema.org/Person">
  <span itemprop="name">Christopher Froome</span> was sponsored by
  <span itemprop="sponsor" itemtype="http://schema.org/Organization">
    <a itemprop="url" href="http://www.skysports.com/">Sky</a></span> in the Tour de France.
</p>
```

```
<script type="application/ld+json">
{
  "@context": "http://schema.org/",
  "@type": "Person",
  "name": "Christopher Froome",
  "sponsor":
  {
    "@type": "Organization",
    "name": "Sky",
    "url": "http://www.skysports.com/"
  }
}
</script>
```

Lecture Outline

- Motivation
- Overview of Description Logics (DLs)
- Semantics of DLs and Reasoning in DLs
- Practical

Description Logics (DLs)

- Description
  - comes from class description, a formal expression that determines a set of objects with common properties
- Logic
  - semantics of class descriptions can be defined using logic

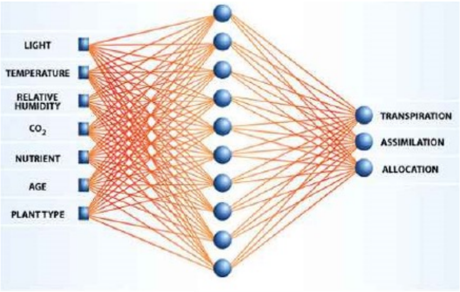
DLs as KR Languages

- Formalism: well defined syntax and formal semantics
- High-level description: only relevant aspect represented; others left out
- Adequate expressive power: trade-off between expressiveness and complexity
- Intelligent applications: must be able to provide reasoning services given requirements from applications
- Effetively used: need for scalable and efficient implementations

---

Syntax

- Provide an explicit symbolic representation of knowledge
- not just implicit, as e.g. nearual networks



[credit: F Baader]

Semantics and Reasoning

- Declarative semantics
  - mapping of the symbolic expressions to an abstraction of the “world” (interpretation)
  - allow ones to determine whether a symbolic expression is true in the given world (model)
- Reasoning result should depend only on the semantics and not on the syntactive representation
- Not prededual semantics
  - Should not be defined by how certain programs using the symbolic representation behave

Reasoning Procedures

- The procedure should be a decision procedure for reasoning problems
  - soundness: positive answers are correct
  - compleness: negative answers are correct
  - termination: always give an answer in finite time
- First Order Logic (FOL)
  - Satisfiability does not have a decision procedure
  - Thus FOL is not an appropriate KR formalism
- Propositional Logic (0th order Logic)
  - Satisfibility is NP-complete; however, there are highly optimised SAR slovers
  - Expressive power is not sufficient

Description Logic History

- Phase 1: incomplete structural subsumption algorithms
  - Systems: Back, K-Rep, Loom, Meson
- Phase 2: tableau algorithms (with complexity results) and optimisations
  - Systems: Kris, Crack
- Phase 3: : tableau algorithms for very expressive DLs
  - Systems: FaCT, Racer, HermiT, Konclude, ...
- Phase 4: OWL standard (on top of RDF), lightweight language and approximate reasoning
  - Systems: CEL, TrOWL, Ontop, Mastro,...

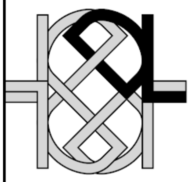
RDF in Description Logics

- Class assertions C(e)
  - [e rdf:type C .]
- Property assertion r(e1,e2)
  - [e1 r e2 .]
- SubClassOf axiom: C1 $\sqsubseteq$ C2
  - [C1 rdfs:subClassOf C2 .]
- SubPropertyOf Axiom r1 $\sqsubseteq$ r2
  - [r1 rdfs:SubPropertyOf r2 .]
- Property Domain (Range) axioms $\exists p \sqsubseteq D$ ($\exists p^- \sqsubseteq R$)
  - [p rdfs:domain D .] ([p rdfs:range R .])

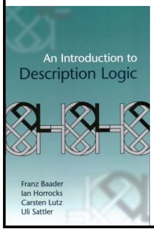
Lecture Outline

- Motivation
- Overview of Description Logics (DLs)
- Semantics of DLs and Reasoning in DLs
- Practical

ALC: A Basic yet Expressive DL



- ALC: Attributed Language with Complement [Schmidt-Schauss&Smolka, 1991]
- It is a basic language
  - Reasoning complexity is EXPTIME-complete
- Naming scheme:
  - foundation language AL
  - can be further extended with constructors whose “letter” can be added aftter AL
  - C for complement (¬)
  - H for property subsumption ($r1 \sqsubseteq r2$)
  - I for inverse property (r^-)
  - O for one of ({e})
  - Q for number restrictions (≥r.C, ≤r.C, =r.C)



ALC Syntax

Let **C** and **R** be disjoint sets of concept names and role names, respectively.

$\mathcal{ALC}$-concept descriptions are defined by induction:

- If $A \in \mathbf{C}$, then A is an $\mathcal{ALC}$-concept description.
- If C, D are $\mathcal{ALC}$-concept descriptions, and $r \in \mathbf{R}$, then the following are $\mathcal{ALC}$-concept descriptions:
  - $C \sqcap D$ (conjunction)
  - $C \sqcup D$ (disjunction)
  - $\neg C$ (negation)
  - $\forall r.C$ (value restriction)
  - $\exists r.C$ (existential restriction)

Abbreviations:
- $\top := A \sqcup \neg A$ (top)
- $\bot := A \sqcap \neg A$ (bottom)
- $C \Rightarrow D := \neg C \sqcup D$ (implication)

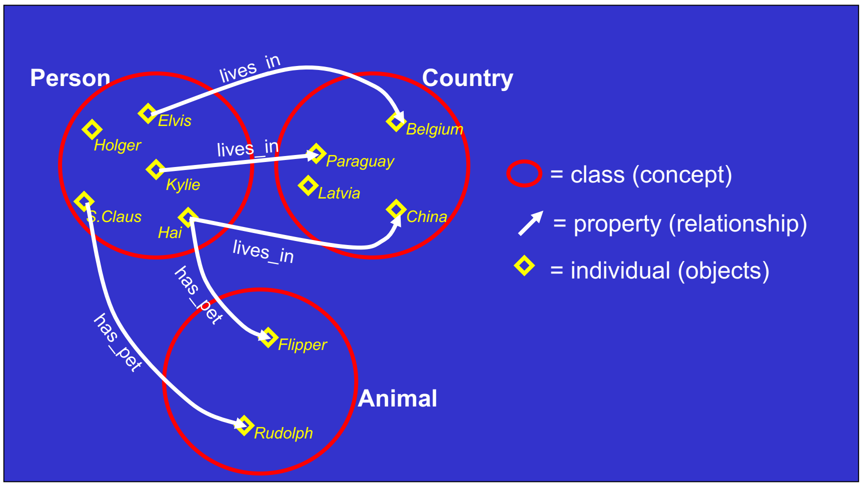
[credit: F Baader]

DL Interpretations

- An interpretation **I** is written as $(\Delta^I, \bullet^I)$
  - Δ^I is the non-empty domain (similar to universal set)
  - $\bullet^I$ is the interpretation function
    - all individuals (inc. unnamed ones) are members of the domain: $o^I \in \Delta^I$
    - all classes are subsets of the domain $A^I \subseteq \Delta^I$
      - e.g., Employee^I= {E1, E2, E3, E4}
    - all properties are subsets $R^I \subseteq \Delta^I \times \Delta^I$

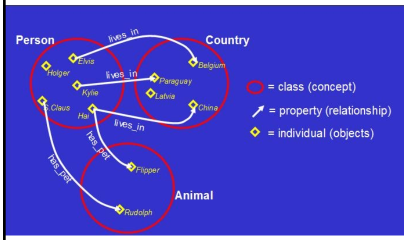
      e.g., Works-for^I= {<E1,P1>, <E2,P1>, <E2,P2>, <E3,P1>, <E3,P2>, <E4,P2>}
- Interpretation function allows us to consider all possible assignment of class and property memberships
  - all possible databases for the given schema

Excercise: DL Interpretations



[Picture Credit: Protégé Team]

Excercise: DL Interpretations (II)



- Δ^I = {Elvis, Holger, …}
- Named objects
  - Elvis^I = Elvis
  - Holger^I = Holger
  - …
- Named classes
  - Animal^I = {Flipper, Rudolph}
  - Person^I = {Elvis,Holger,Kylie,Hai,S.Claus}
  - Country^I = {Belgium,Paraguar,Latvia, China}
- Named properties
  - has_pet^I = {<Hai,Plipper>, <S.Claus,Rudolph>}
  - lives_in^I = {<Elvis,Brlgium>, <Kylie,Paraguar>, <Hai, China> }

ALC Semantics

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and an extension mapping $\cdot^{\mathcal{I}}$:

- $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for all $A \in \mathbf{C}$, concepts interpreted as sets
- $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for all $r \in \mathbf{R}$. roles interpreted as binary relations

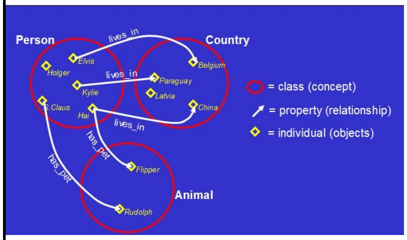
The extension mapping is extended to complex $\mathcal{ALC}$-concept descriptions as follows:

- $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} := C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\forall r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \text{for all } e \in \Delta^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}}\}$
- $(\exists r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \text{there is } e \in \Delta^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\}$

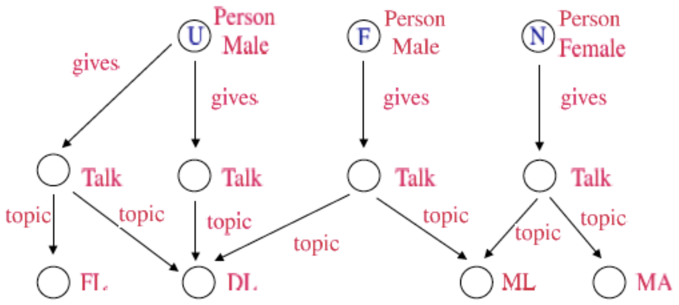
[credit: F Baader]

Excercise: DL Interpretations (III)

- Suppose we extend the vocabulary with
  - Young
- Given the following interpretation of Young:
  - Young^{I} ={Holger, Hai, Kylie, Flipper}
  - How about the interepretation of the OWL class description?
    - Young ⊓ Person = **{Holger, Hai, Kylie}**
    - ∃has_pet.Young = **{Hai}**



ALC Semantics: Excecise



$\text{Person} \sqcap \exists \text{gives}.(\text{Talk} \sqcap \forall \text{topic}.\text{DL})$

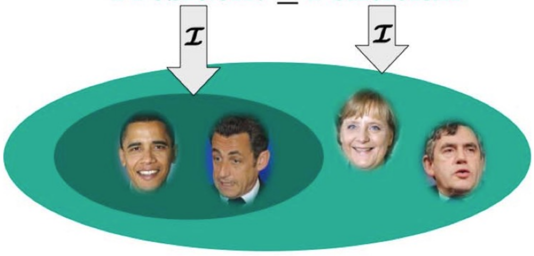
$\text{Person} \sqcap \forall \text{gives}.(\text{Talk} \sqcap \exists \text{topic}.\text{DL})$

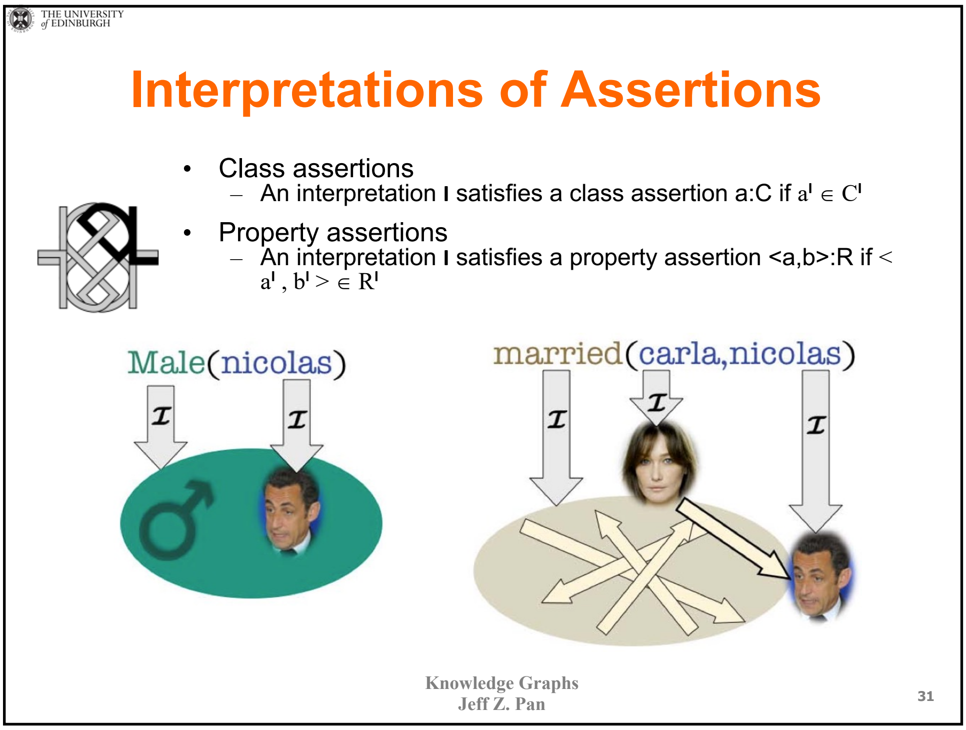
[credit: F Baader]

Interpretations Axioms

- Axioms are used to “filter out” invalid interpretations from valid ones
  - An interpretation I is a model for an ontology O if it satisfies all its axioms
  - An ontology O is consistent if it has some model (valid interpretation).

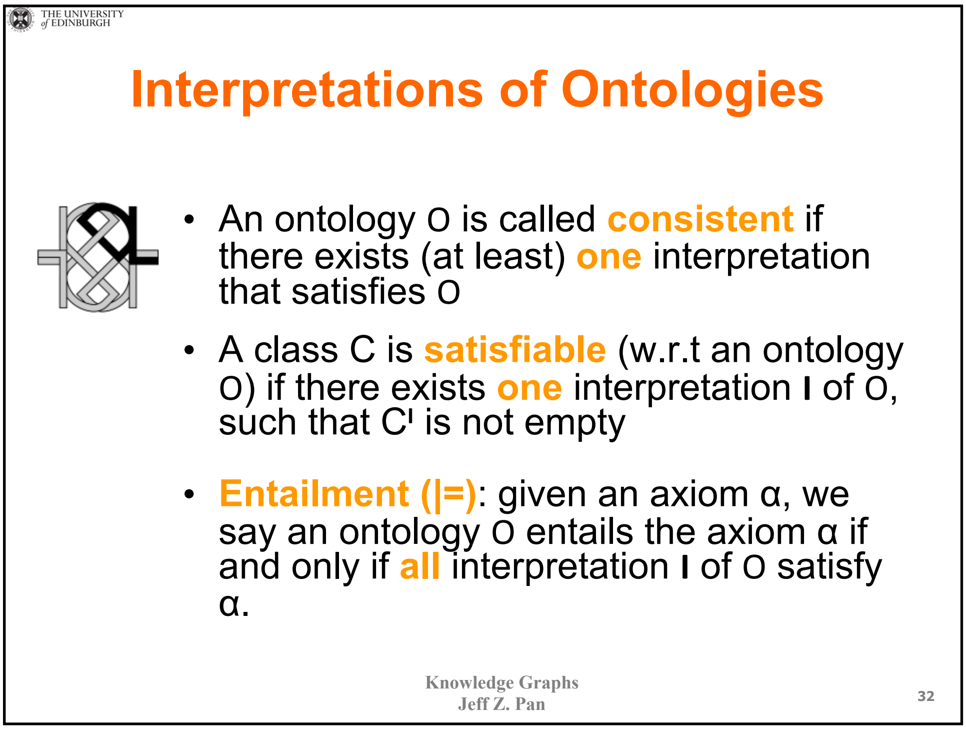
$\text{President} \sqsubseteq \text{Politician}$





Interpretations of Assertions

- Class assertions
  - An interpretation I satisfies a class assertion a:C if $a^I \in C^I$
- Property assertions
  - An interpretation I satisfies a property assertion <a,b>:R if $< a^I , b^I > \in R^I$

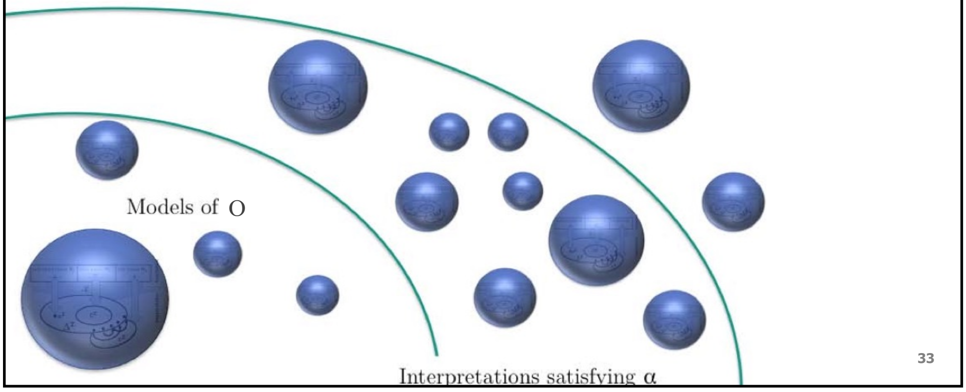


Interpretations of Ontologies

- An ontology O is called **consistent** if there exists (at least) **one** interpretation that satisfies O
- A class C is **satisfiable** (w.r.t an ontology O) if there exists **one** interpretation I of O, such that C^I is not empty
- **Entailment (|=)**: given an axiom α, we say an ontology O entails the axiom α if and only if **all** interpretation I of O satisfy α.

Entailments of Axioms

- **Entailment (|=)**: given an axiom α, we say an ontology O entails the axiom α if and only if **all** interpretation **I** of O satisfy α.



Lecture Outline

- Motivation: DLs are the underpinning of the standard KG schema language OWL
- Introduction: Syntax and semantics of DL
- Focus: The ALC DL
- Exercises (Next time we introduce reasoning in Description Logics)
  - Formulate ALC concepts:
    - Young pet owner
    - Pet owner only have cats
  - Subsumption checking: are the following statements correct?
    - Young ⊓ Person is subsumed by Person
    - Young ⊔ Person is subsumed by Person