

The logic of Soft Computing IV  
Ostrava, Czech Republic, October 5-7, 2005

## Fuzzy Description Logics and the Semantic Web

**Umberto Straccia**

I.S.T.I. - C.N.R. Pisa, Italy

[straccia@isti.cnr.it](mailto:straccia@isti.cnr.it)



“Calla is a **very large,**  
**long** white flower on **thick**  
stalks”

# Outline

- Introduction to Description Logics (DLs)
- Semantic Web, Ontologies and DLs
- Fuzzy DLs
- Conclusions & Future Work

# Introduction to Description Logics

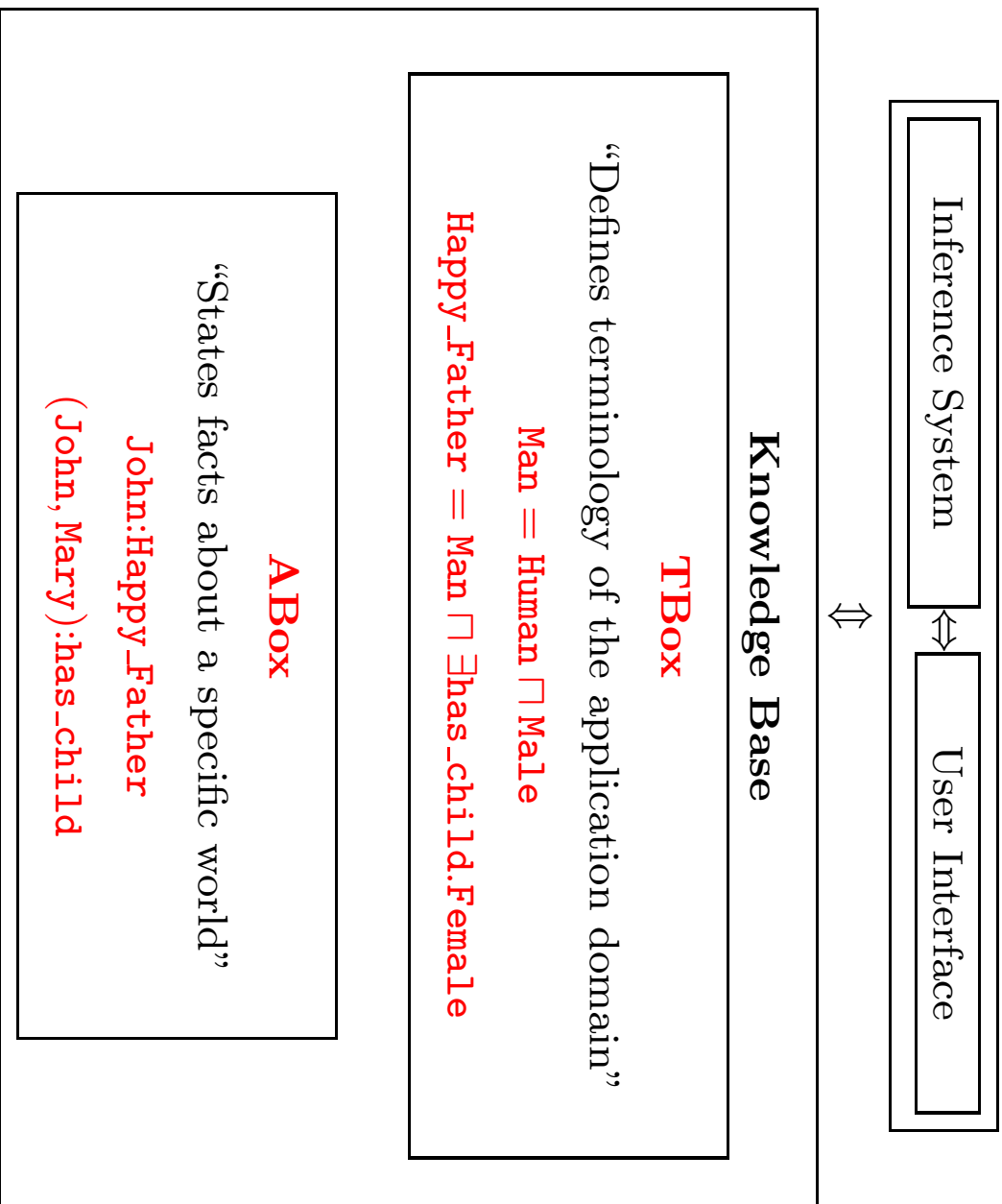
## What Are Description Logics? (<http://dl.kr.org/>)

- A family of logic-based knowledge representation formalisms
  - Descendants of semantic networks and KL-ONE
  - Describe domain in terms of **concepts** (classes), **roles** (properties, relationships) and **individuals**
- Distinguished by:
  - **Formal semantics** (typically model theoretic)
    - \* **Decidable fragments of FOL**
    - \* Closely related to Propositional Modal & Dynamic Logics
    - \* Closely related to Guarded Fragment of FOL, L2 and C2
  - Provision of **inference services**
    - \* Decision procedures for key problems (e.g., satisfiability, subsumption)
    - \* Implemented systems (highly optimised)

## DIs Basics

- **Concept** names are equivalent to unary predicates
  - In general, concepts equiv to formulae with one free variable
- **Role** names are equivalent to binary predicates
  - In general, roles equiv to formulae with two free variables
- **Individual** names are equivalent to constants
- **Operators** restricted so that:
  - Language is decidable and, if possible, of low complexity
  - No need for explicit use of variables
    - \* Restricted form of  $\exists$  and  $\forall$
  - Features such as counting can be succinctly expressed

# Description Logic System



# The DL Family

- A given DL is defined by set of concept and role forming operators
- Smallest propositionally closed DL is *ALC* (*A*ttributive *L*anguage with *C*omplement)
  - Concepts constructed using  $\sqcap, \sqcup, \neg, \exists$  and  $\forall$

	<i>Syntax</i>	<i>Example</i>
$C, D$	$\top$   (top concept)	
	$\perp$   (bottom concept)	
	$A$   (atomic concept)	Human
	$C \sqcap D$   (concept conjunction)	Human $\sqcap$ Male
	$C \sqcup D$   (concept disjunction)	Nice $\sqcap$ Rich
	$\neg C$   (concept negation)	$\neg$ Meat
	$\exists R.C$   (existential quantification)	$\exists$ has_child.Blond
	$\forall R.C$   (universal quantification)	$\forall$ has_child.Human

## DL Semantics

- Semantics is given in terms of an **interpretation**  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ , where
  - $\Delta^{\mathcal{I}}$  is the **domain** (a non-empty set)
  - $\cdot^{\mathcal{I}}$  is an **interpretation function** that maps:
    - \* **Concept** (class) name  $A$  into a subset  $A^{\mathcal{I}}$  of  $\Delta^{\mathcal{I}}$
    - \* **Role** (property) name  $R$  into a subset  $R^{\mathcal{I}}$  of  $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
    - \* **Individual** name  $a$  into an element of  $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
  - Interpretation function  $\cdot^{\mathcal{I}}$  is extended to concept expressions:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$\perp^{\mathcal{I}} = \emptyset$$

$$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$$

$$(C_1 \sqcup C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$$

$$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y. \langle x, y \rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$$

$$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y. \langle x, y \rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}$$

## DLS and FOI

- *ALC* mapping to FOI: introduce
  - a unary predicate  $A$  for an atomic concept  $A$
  - a binary predicate  $R$  for a role  $R$
- Translate concepts  $C$  and  $D$  as follows

$$\begin{aligned}t(\top, x) &= \text{true} \\t(\perp, x) &= \text{false} \\t(A, x) &\mapsto A(x) \\t(C_1 \sqcap C_2, x) &= t(C_1, x) \wedge t(C_2, x) \\t(C_1 \sqcup C_2, x) &\mapsto t(C_1, x) \vee t(C_2, x) \\t(\neg C, x) &= \neg t(C, x) \\t(\exists R.C, x) &= \exists y. R(x, y) \wedge t(C, y) \\t(\forall R.C, x) &= \forall y. R(x, y) \Rightarrow t(C, y)\end{aligned}$$

## DL Knowledge Base

- A DL **Knowledge Base** is a pair  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , where
  - $\mathcal{T}$  is a **TBox** containing **general inclusion axioms** of the form  $C \sqsubseteq D$  (“concept  $D$  subsumes concept  $C$ ”),

$$\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

- \* **concept definitions** are of the form  $A = C$  (equiv to  $A \sqsubseteq C, C \sqsubseteq A$ )
- \* **primitive concept definitions** are of the form  $A \sqsubseteq C$
- \* Sometimes, a TBox can contain primitive and concept definitions only, where no atom can be defined more than once and no recursion is allowed  $\mapsto$   
**Computational complexity changes dramatically**
- $\mathcal{A}$  is a **ABox** containing **assertions** of the form  $a:C$  (“individual  $a$  is an instance of concept  $C$ ) and  $(a, b):R$  (individual  $b$  is an  $R$ -filler of individual  $a$ )”)

$$\mathcal{I} \models a:C \quad \text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$$

$$\mathcal{I} \models (a, b):R \quad \text{iff} \quad \langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$$

## Note on DL naming

$\mathcal{AL}$ :  $C, D \longrightarrow \top \mid \perp \mid A \mid C \sqcap D \mid \neg A \mid \exists R. \top \mid \forall R. C$

$\mathcal{C}$ : Concept negation,  $\neg C$ . Thus,  $\mathcal{ALC} = \mathcal{AL} + \mathcal{C}$

$\mathcal{S}$ : Used for  $\mathcal{ALC}$  with transitive roles  $\mathcal{R}_+$

$\mathcal{U}$ : Concept disjunction,  $C_1 \sqcup C_2$

$\mathcal{E}$ : Existential quantification,  $\exists R. C$

$\mathcal{H}$ : Role inclusion axioms,  $R_1 \sqsubseteq R_2$

$\mathcal{N}$ : Number restrictions,  $(\geq n R)$  and  $(\leq n R)$ , e.g.  $(\geq 3 \text{ has\_Child})$  (has at least 3 children)

$\mathcal{Q}$ : Qualified number restrictions,  $(\geq n R.C)$  and  $(\leq n R.C)$ , e.g.  $(\leq 2 \text{ has\_Child.Adult})$  (has at most 2 adult children)

$\mathcal{O}$ : Nominals (singleton class),  $\{a\}$ , e.g.  $\text{Ehas\_child.}\{\text{mary}\}$ .

**Note:**  $a:C$  equiv to  $\{a\} \sqsubseteq C$  and  $(a,b):R$  equiv to  $\{a\} \sqsubseteq \exists R.\{b\}$

$\mathcal{I}$ : Inverse role,  $R^-$ , e.g.

$\mathcal{F}$ : Functional role,  $f$

For instance,

$$SHIF = S + \mathcal{H} + \mathcal{I} + \mathcal{F} = \mathcal{ALCR}_+ HIF$$

$$SHOIN = S + \mathcal{H} + \mathcal{O} + \mathcal{I} + \mathcal{N} = \mathcal{ALCR}_+ HOIN$$

# Short History of Description Logics

**Phase 1:** Mostly system development (early eighties)

- Incomplete systems (Back, Classic, Loom, Kandor, ...)
- Based on **structural algorithms**

**Phase 2:** first tableaux algorithms and complexity results (mid-eighties - mid-nineties)

- Development of **tableaux algorithms** and **complexity results**
- Tableaux-based systems (Kris, Crack)
- Investigation of optimization techniques

**Phase 3:** Optimized systems for very expressive DLs (mid-nineties - ...)

- Tableaux algorithms for **very expressive** DLs
- **Highly optimised** tableau systems (FaCT, DLP, Racer)
- Relationship to modal logic and decidable fragments of FOL

# Latest developments

## Phase 4:

- Mature **implementations**
- Mainstream **applications** and Tools
  - Databases
    - \* consistency of conceptual schema (EER, UML) using e.g. *DLR* (*n*-ary DL)
    - \* schema integration
    - \* Query subsumption (w.r.t. a conceptual schema)
  - Ontologies and the **Semantic Web** (and **Grid**)
    - \* Ontology engineering (design, maintenance, integration)
    - \* Reasoning with ontology-based markup (metadata)
    - \* Service description and discovery
  - **Commercial** implementations
    - \* Cerebra, Racer

# The Semantic Web

# The Semantic Web Vision and DIs

- The WWW as we know it now
  - 1st generation web mostly handwritten HTML pages
  - 2nd generation (current) web often machine generated/active
  - Both intended for direct human processing/interaction
- In next generation web, resources should be more accessible to automated processes
  - To be achieved via semantic markup
  - Metadata annotations that describe content/function

# Ontologies

- Semantic markup must be **meaningful** to automated processes
- Ontologies will play a key role
  - Source of **precisely defined** terms (vocabulary)
  - Can be **shared** across applications (and humans)
- Ontology typically consists of:
  - **Hierarchical** description of important **concepts** in domain
  - Descriptions of **properties** of instances of each concept
- Ontologies can be used, e.g.
  - To facilitate agent-agent communication in **e-commerce**
  - In semantic based **search**
  - To provide richer **service descriptions** that can be more flexibly interpreted by intelligent agents

## Example Ontology

- Vocabulary and meaning (definitions)
  - **Elephant** is a concept whose members are a kind of animal
  - **Herbivore** is a concept whose members are exactly those animals who eat only plants or parts of plants
  - **Adult\_Elephant** is a concept whose members are exactly those elephants whose age is greater than 20 years
- Background knowledge/constraints on the domain (general axioms)
  - **Adult\_Elephants** weigh at least 2,000 kg
  - All **Elephants** are either **African\_Elephants** or **Indian\_Elephants**
  - No individual can be both a **Herbivore** and a **Carnivore**

# Example Ontology (Protégé)

The screenshot displays the Protégé 3.0 interface for editing an ontology. The title bar indicates the file path: `file:/Users/horrocks/Software/OntEd/ontologies/elephants.pptj`. The menu bar includes **File**, **Edit**, **Project**, **OWL**, **Wizards**, **Code**, **Window**, and **Help**. The toolbar contains various icons for file operations and ontology editing.

The main workspace is divided into several panes:

- Left Pane:** Shows the class hierarchy. The selected class is `ns0:giraffe`. The hierarchy includes `owl:Thing`, `ns0:animal`, `ns0:african_animal`, `ns0:asian_animal`, `ns0:carnivore`, `ns0:elephant`, `ns0:adult_elephant`, `ns0:african_elephant`, `ns0:indian_elephant`, `ns0:kenyan_elephant`, `ns0:giraffe`, `ns0:herbivore`, `ns0:large_animal`, `ns0:lion`, `ns0:branch`, and `ns0:continent`.
- CLASS EDITOR:** Shows the selected class `ns0:giraffe` (instance of `owl:Class`). It includes a **Name** field with `ns0:giraffe`, a **SameAs** field, and a **DifferentFrom** field. A text area contains the comment: "Funny looking things with long necks".
- Right Pane:** Contains **Annotations** and **Property** sections. The **Property** section shows `rdfs:comment` with the value "Funny looking...".
- Asserted Conditions:** A list of conditions for the class, including `ns0:animal` and `ns0:eats ns0:leaf`. It includes checkboxes for **NECESSARY & SUFFICIENT** and **NECESSARY**.
- Disjoints:** A list of disjoint properties, including `ns0:eats`, `ns0:leaf`, and `ns0:gnaaws`.

The bottom status bar shows the current view: **Logic View** (selected) and **Properties View**.

## Ontology Description Languages

- Should be **sufficiently expressive** to capture most useful aspects of domain knowledge representation
- Reasoning in it should be **decidable** and **efficient**
- Many different languages has been proposed: RDF, RDFS, OIL, DAML+OIL
- OWL (**O**ntology **W**eb **L**anguage) is the current emerging language: it's a standard

# OWL Language

- Three species of OWL
  - OWL full is union of OWL syntax and RDF (but, undecidable)
  - OWL DL restricted to FOL fragment (reasoning problem in NEXPTIME)
  - OWL Lite is easier to implement subset of OWL DL (reasoning problem in EXPTIME)
- Semantic layering
  - OWL DL  $\equiv$  OWL full within **Description Logic fragment**
  - DL semantics officially **definitive**
- OWL DL based on **SHIQ Description Logic** ( $ALC\mathcal{H}\mathcal{I}\mathcal{Q}\mathcal{R}_+$ )
  - In fact it is equivalent to  $SH\mathcal{O}\mathcal{I}\mathcal{N}(\mathcal{D})$
- OWL Lite based on **SHITF Description Logic** ( $ALC\mathcal{H}\mathcal{I}\mathcal{T}\mathcal{F}\mathcal{R}_+$ )
- Benefits from many years of DL research
  - **Formal properties** well understood (complexity, decidability)
  - **Known reasoning algorithms**
  - **Implemented systems** (highly optimised)

Abstract Syntax	DL Syntax	Example
Descriptions ( $C$ )		
$A$ (URI reference)	$A$	Conference
<code>owl:Thing</code>	$\top$	
<code>owl:Nothing</code>	$\perp$	
<code>intersectionOf(<math>C_1</math> <math>C_2</math>...)</code>	$C_1 \sqcap C_2$	Reference $\sqcap$ Journal
<code>unionOf(<math>C_1</math> <math>C_2</math>...)</code>	$C_1 \sqcup C_2$	Organization $\sqcup$ Institution
<code>complementOf(<math>C</math>)</code>	$\neg C$	$\neg$ MasterThesis
<code>oneOf(<math>o_1</math> ...)</code>	$\{o_1, \dots\}$	$\{\text{"WISE"}, \text{"ISWC"}, \dots\}$
<code>restriction(<math>R</math> someValuesFrom(<math>C</math>))</code>	$\exists R.C$	$\exists$ parts.InCollection
<code>restriction(<math>R</math> allValuesFrom(<math>C</math>))</code>	$\forall R.C$	$\forall$ date.Date
<code>restriction(<math>R</math> hasValue(<math>o</math>))</code>	$R : o$	date : 2005
<code>restriction(<math>R</math> minCardinality(<math>n</math>))</code>	$(\geq n R)$	$\geq 1$ location
<code>restriction(<math>R</math> maxCardinality(<math>n</math>))</code>	$(\leq n R)$	$\leq 1$ publisher
<code>restriction(<math>U</math> someValuesFrom(<math>D</math>))</code>	$\exists U.D$	$\exists$ issue.integer
<code>restriction(<math>U</math> allValuesFrom(<math>D</math>))</code>	$\forall U.D$	$\forall$ name.string
<code>restriction(<math>U</math> hasValue(<math>v</math>))</code>	$U : v$	series : "LNCS"
<code>restriction(<math>U</math> minCardinality(<math>n</math>))</code>	$(\geq n U)$	$\geq 1$ title
<code>restriction(<math>U</math> maxCardinality(<math>n</math>))</code>	$(\leq n U)$	$\leq 1$ author

Abstract Syntax	DL Syntax	Example
<b>Axioms</b>		
Class( $A$ partial $C_1 \dots C_n$ )	$A \sqsubseteq C_1 \sqcap \dots \sqcap C_n$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
Class( $A$ complete $C_1 \dots C_n$ )	$A = C_1 \sqcap \dots \sqcap C_n$	Man = Human $\sqcap$ Male
EnumeratedClass( $A$ $o_1 \dots o_n$ )	$A = \{o_1\} \sqcup \dots \sqcup \{o_n\}$	RGB = {r} $\sqcup$ {g} $\sqcup$ {b}
SubClassOf( $C_1 C_2$ )	$C_1 \sqsubseteq C_2$	
EquivalentClasses( $C_1 \dots C_n$ )	$C_1 = \dots = C_n$	
DisjointClasses( $C_1 \dots C_n$ )	$C_i \sqcap C_j = \perp, i \neq j$	Male $\sqsubseteq$ $\neg$ Female
ObjectProperty( $R$ super ( $R_1 \dots$ super ( $R_n$ )))	$R \sqsubseteq R_i$	HasDaughter $\sqsubseteq$ hasChild
domain( $C_1 \dots$ domain( $C_n$ ))	$(\geq 1 R) \sqsubseteq C_i$	$(\geq 1$ hasChild) $\sqsubseteq$ Human
range( $C_1 \dots$ range( $C_n$ ))	$T \sqsubseteq \forall R.D_i$	$T \sqsubseteq \forall$ hasChild.Human
[inverseof( $R_0$ )]	$R = R_0^-$	hasChild = hasParent $^-$
[symmetric]	$R = R^-$	similar = similar $^-$
[functional]	$T \sqsubseteq (\leq 1 R)$	$T \sqsubseteq (\leq 1$ hasMother)
[Inversefunctional]	$T \sqsubseteq (\leq 1 R^-)$	
[Transitive]	$T^r(R)$	$T^r(\text{ancestor})$
SubPropertyOf( $R_1 R_2$ )	$R_1 \sqsubseteq R_2$	
EquivalentProperties( $R_1 \dots R_n$ )	$R_1 = \dots = R_n$	cost = price
AnnotationProperty( $S$ )		

Abstract Syntax	DL Syntax	Example
DatatypeProperty( $U$ super ( $U_1$ )... super ( $U_n$ )) domain( $C_1$ ) ... domain( $C_n$ ) range( $D_1$ ) ... range( $D_n$ ) [functional] SubPropertyOf( $U_1 U_2$ ) EquivalentProperties( $U_1$ ... $U_n$ )	$U \sqsubseteq U_i$ $(\geq 1 U) \sqsubseteq C_i$ $T \sqsubseteq \forall U.D_i$ $T \sqsubseteq (\leq 1 U)$ $U_1 \sqsubseteq U_2$ $U_1 = \dots = U_n$	$(\geq 1 \text{ hasAge}) \sqsubseteq \text{Human}$ $T \sqsubseteq \forall \text{hasAge.posInteger}$ $T \sqsubseteq (\leq 1 \text{ hasAge})$ $\text{hasName} \sqsubseteq \text{hasFirstName}$
Individuals		
Individual( $o$ type ( $C_1$ )... type ( $C_n$ )) value( $R_1 o_1$ ) ... value( $R_n o_n$ ) value( $U_1 v_1$ ) ... value( $U_n v_n$ ) SameIndividual( $o_1$ ... $o_n$ ) DifferentIndividuals( $o_1$ ... $o_n$ )	$o:C_i$ $(o, o_i):R_i$ $(o, v_1):U_i$ $o_1 = \dots = o_n$ $o_i \neq o_j, i \neq j$	$\text{tim:Human}$ $(\text{tim, mary}): \text{hasChild}$ $(\text{tim, 14}): \text{hasAge}$ $\text{president\_Bush} = \text{G.W.Bush}$ $\text{john} \neq \text{peter}$

## XML representation of OWL statements

E.g.,  $\text{Person} \sqcap \text{VhasChild}.$  ( $\text{Doctor} \sqcup \text{EhasChild}.\text{Doctor}$ ):

```
<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>
```

## Concrete domains in OWL

- OWL supports **concrete domains**: integers, strings, ...
- Clean separation between object classes and concrete domains
  - Disjoint interpretation domain:  $d^{\mathcal{I}} \subseteq \Delta_D$ , and  $\Delta_D \cap \Delta^{\mathcal{I}} = \emptyset$
  - Disjoint concrete properties:  $UI \subseteq \Delta^{\mathcal{I}} \times \Delta_D$ , e.g.,  $(\text{tim}, 14):\text{hasAge}$ , (**sf**, “SoftComputing”):hasAcronym
- Philosophical reasons:
  - Concrete domains structured by **built-in predicates**
- Practical reasons:
  - Ontology language remains **simple and compact**
  - **Semantic integrity** of ontology language not compromised
    - **Implementability** not compromised can use hybrid reasoner
      - \* Only need sound and complete decision procedure for  $d_1^{\mathcal{I}} \cap \dots \cap d_n^{\mathcal{I}}$ , where  $d_i$  is a (possibly negated) datatype
- In the DL literature, these are called **Concrete Domains**
- Notation:  $(D)$ . E.g.,  $ALC(D)$  is  $ALC +$  concrete domains, OWL DL =  $SHOIN(D)$

# Reasoning with OWL DL

# Reasoning

- What can we do with it?
  - **Design and maintenance** of ontologies
    - \* Check class consistency and compute class hierarchy
    - \* Particularly important with large ontologies/multiple authors ( $\geq 2^{10}$  defined concepts)
  - **Integration** of ontologies
    - \* Assert inter-ontology relationships
    - \* Reasoner computes integrated class hierarchy/consistency
  - **Querying** class and instance data w.r.t. ontologies
    - \* Determine if set of facts are consistent w.r.t. ontologies
    - \* Determine if individuals are instances of ontology classes
    - \* Retrieve individuals/tuples satisfying a query expression
    - \* Check if one class subsumes (is more general than) another w.r.t. ontology
- How do we do it?
  - Use DLs reasoner (OWL DL = *SHOIN(D)*)

# Basic Inference Problems

**Consistency:** Check if knowledge is meaningful

- Is  $\mathcal{K}$  consistent?  $\mapsto$  There exists some model  $\mathcal{I}$  of  $\mathcal{K}$
- Is  $C$  consistent?  $\mapsto C^{\mathcal{I}} \neq \emptyset$  for some some model  $\mathcal{I}$  of  $\mathcal{K}$

**Subsumption:** structure knowledge, compute taxonomy

- $\mathcal{K} \models C \sqsubseteq D$  ?  $\mapsto C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$  for all models  $\mathcal{I}$  of  $\mathcal{K}$

**Equivalence:** check if two classes denote same set of instances

- $\mathcal{K} \models C = D$  ?  $\mapsto C^{\mathcal{I}} = D^{\mathcal{I}}$  for all models  $\mathcal{I}$  of  $\mathcal{K}$

**Instantiation:** check if individual  $a$  instance of class  $C$

- $\mathcal{K} \models a:C$  ?  $\mapsto a^{\mathcal{I}} \in C^{\mathcal{I}}$  for all models  $\mathcal{I}$  of  $\mathcal{K}$

**Retrieval:** retrieve set of individuals that instantiate  $C$

- Compute the set  $\{a : \mathcal{K} \models a:C\}$

Problems are all **reducible** to consistency, e.g.

- $\mathcal{K} \models C \sqsubseteq D$  iff  $\langle \mathcal{T}, \mathcal{A} \cup \{a:C \sqcap \neg D\} \rangle$  not consistent
- $\mathcal{K} \models a:C$  iff  $\langle \mathcal{T}, \mathcal{A} \cup \{a:\neg C\} \rangle$  not consistent

## Reasoning in DLs: Basics

- Tableau algorithm deciding concept consistency
- Try to build a **tree-like model**  $\mathcal{I}$  of the input concept  $C$
- Decompose  $C$  syntactically
  - Apply tableau **expansion rules**
  - Infer constraints on elements of model
- Tableau rules correspond to constructors in logic ( $\top, \sqcup, \dots$ )
  - Some rules are **nondeterministic** (e.g.,  $\sqcup, \leq$ )
  - In practice, this means **search**
- Stop when no more rules applicable or **clash** occurs
  - Clash is an obvious contradiction, e.g.,  $A(x), \neg A(x)$
- Cycle check (**blocking**) may be needed for termination
- $C$  satisfiable iff rules can be applied such that a fully expanded clash free tree is constructed

## Tableaux checking consistency of an *ALC* concept

- Works on a tree (semantics through viewing tree as an ABox)
  - Nodes represent elements of  $\Delta^{\mathcal{T}}$ , labelled with sub-concepts of  $C$
  - Edges represent role-successorships between elements of  $\Delta^{\mathcal{T}}$
- Works on concepts in **negation normal form**: push negation inside using de Morgan' laws and

$$\begin{aligned}\neg(\exists R.C) &\mapsto \forall R.\neg C \\ \neg(\forall R.C) &\mapsto \exists R.\neg C\end{aligned}$$

- It is initialised with a tree consisting of a single (root) node  $x_0$  with  $\mathcal{L}(x_0) = \{C\}$
- A tree  $T$  contains a **clash** if, for a node  $x$  in  $T$ ,  $\{A, \neg A\} \subseteq \mathcal{L}(x)$
- Returns “ $C$  is consistent” if rules can be applied s.t. they yield a clash-free, complete (no more rules apply) tree

## AFC Tableau rules

$x \bullet \{C_1 \sqcap C_2, \dots\}$	$\longrightarrow \rightarrow \sqcap$	$x \bullet \{C_1 \sqcap C_2, C_1, C_2, \dots\}$
$x \bullet \{C_1 \sqcup C_2, \dots\}$	$\longrightarrow \rightarrow \sqcup$	$x \bullet \{C_1 \sqcup C_2, C, \dots\}$ for $C \in \{C_1, C_2\}$
$x \bullet \{\exists R.C, \dots\}$	$\longrightarrow \rightarrow \exists$	$x \bullet \{\exists R.C, \dots\}$ $R \downarrow$ $y \bullet \{C\}$
$x \bullet \{\forall R.C, \dots\}$	$\longrightarrow \rightarrow \forall$	$x \bullet \{\forall R.C, \dots\}$ $R \downarrow$ $y \bullet \{\dots, C\}$

## Soundness and Completeness

### Theorem 1

1. *The tableau algorithm is a PSPACE (using depth-first search) decision procedure for consistency (and subsumption) of ALC concepts.*
2. *ALC has the tree-model property*

The tableau can be modified to a PSPACE decision procedure for

- e.g. *ALCN* and *ALCI*
- TBox with acyclic concept definitions using lazy unfolding (unfolding on demand)
- Note: *ALC* with general inclusion axioms,  $C \sqsubseteq D$ , jumps to EXPTIME

## Extensions: general inclusion axioms

- TBoxes with general inclusion axioms  $C \sqsubseteq D$ 
  - Each node must be labeled with  $\neg C \sqcup D$  (recall,  $C \sqsubseteq D \equiv_{FOL} \forall x. \neg t(C, x) \vee t(D, x)$ )
  - However, termination not guaranteed

Given,  $a:A$  and  $A \sqsubseteq \exists R.A$  then

$$a \bullet \{A, \neg A \sqcup \exists R.A\}$$

$$a \bullet \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$

$R \downarrow$

$$y_1 \bullet \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$

$R \downarrow$

$$y_2 \bullet \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$

$\vdots$

Note:  $y_1, y_2$  share same properties

## Blocking

- When creating new node, check ancestors for equal (superset) label
- If such a node is found, new node is **blocked**

Given,  $a:A$  and  $A \sqsubseteq \exists R.A$  then

$$a \bullet \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$

$R \downarrow$

$$y_1 \bullet \{A, \neg A \sqcup \exists R.A, \exists R.A\} \text{ Node is blocked}$$

**Note:**

- Blocking may not work with more complex DLs (e.g., using  $\mathcal{R}^-$ )
- With number restrictions ( $\mathcal{N}$ ), some satisfiable concepts have only **non-finite** models: e.g., testing  $\neg C$  with  $\mathcal{T} = \{\top \sqsubseteq \exists R.C, \top \sqsubseteq (\leq 1 R^-)\}$
- Sophisticated methods has been developed to detect  $\infty$  repetition of substructures

## Focus of DL Research

- decidability/complexity of reasoning
- requires restricted description language
- system and complexity results available for various combinations of constructors

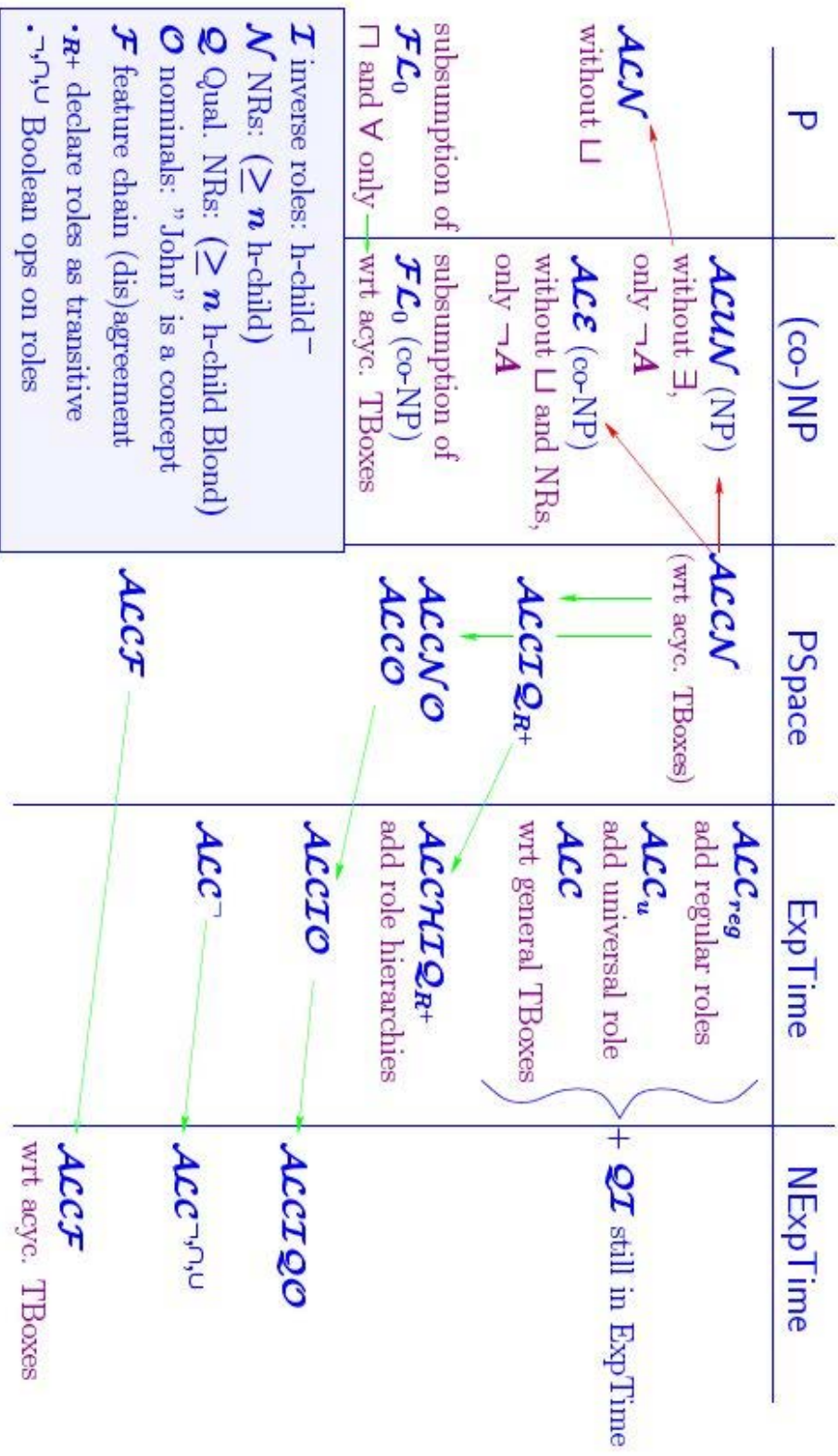
Reasoning feasible

*versus*

- application relevant concepts must be definable
- some applications domains require very expressive DLs
- efficient algorithms in practice for very expressive DLs

Expressivity sufficient

# Some Complexity Results



## Fuzzy Description Logics



“Calla is a **very large**,  
**long** white flower on **thick**  
stalks”

# Objective

- To extend classical DLs towards the representation of and reasoning with *vague concepts*
- Application: Semantic Web
- Development of practical reasoning algorithms
- System implementations

# Example (fuzzy DL-Lite, Current work)

$\text{Hotel} \sqsubseteq \exists \text{hasLocation}$   
 $\text{Conference} \sqsubseteq \exists \text{hasLocation}$   
 $\text{Hotel} \sqsubseteq \neg \text{Conference}$   
 $\text{Location}^T \subseteq \text{GISCoordinates}$   
 $\text{distance}^T : \text{GISCoord} \times \text{GISCoord} \rightarrow \mathbb{N}$   
 $\text{close}^T : \mathbb{N} \rightarrow [0, 1]$   
 $\text{distance}(x, y) = \dots$   
 $\text{close}(x) = \max(0, 1 - \frac{x}{1000})$

hasLocation	hasLocation	distance
h11	c11	300
h11	c12	500
h12	c11	750
h12	c12	750
⋮	⋮	

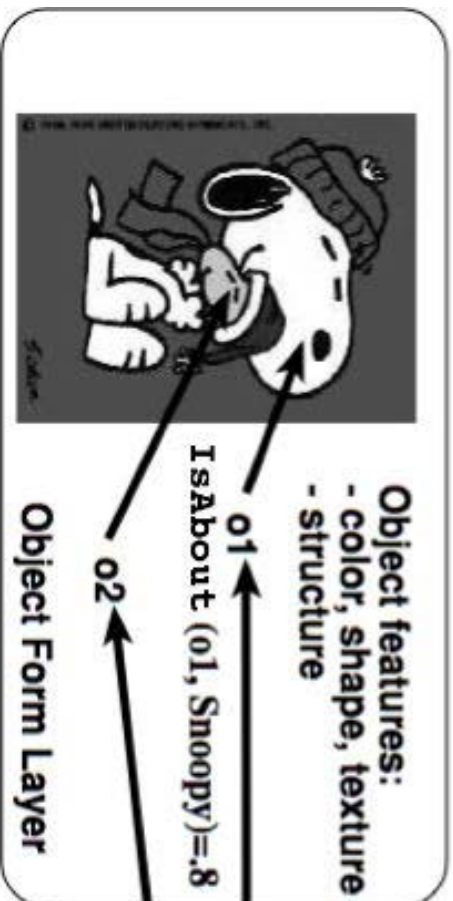
HotelID	hasLocation	ConferenceID	hasLocation
h1	h11	c1	c11
h2	h12	c2	c12
⋮	⋮	⋮	⋮

HotelID	closeness degree
h1	0.7
h2	0.25
⋮	⋮

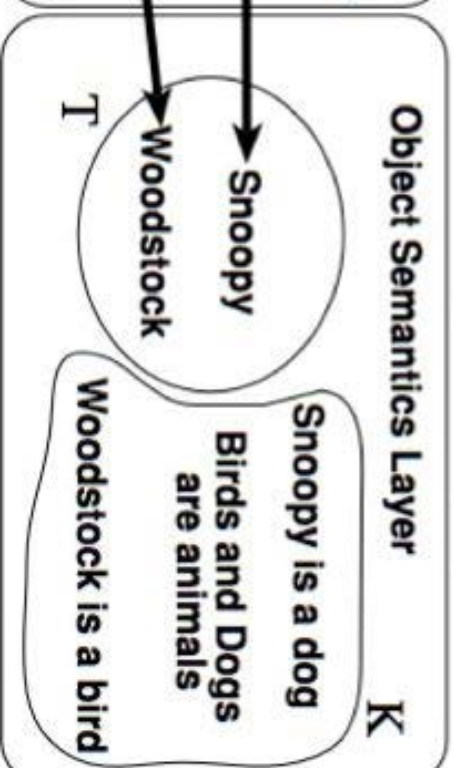
“Find hotel close to conference c1”:  $\text{Query}(c1, h) \leftarrow$   
 $\text{Hotel}(h), \text{hasLocation}(h, hl), \text{Conference}(c1), \text{hasLocation}(c1, cl), \text{distance}(hl, cl, d), \text{close}(d)$

# Example (Logic-based information retrieval model)

media dependent properties



media independent properties



Bird  $\sqsubseteq$  Animal  
Dog  $\sqsubseteq$  Animal  
snoopy : Dog  
woodstock : Bird

ImageRegion	Object ID	isAbout degree
o1	snoopy	0.8
o2	woodstock	0.7
:	:	:

$Query = ImageRegion \sqcap \exists isAbout.Animal$

$Query(ir) \leftarrow ImageRegion(ir), isAbout(ir, x), Animal(x)$

# Example (Graded Entailment)

		
audi_tt	mg	ferrari_enzo

Car	speed
audi_tt	243
mg	$\leq 170$
ferrari_enzo	$\geq 350$

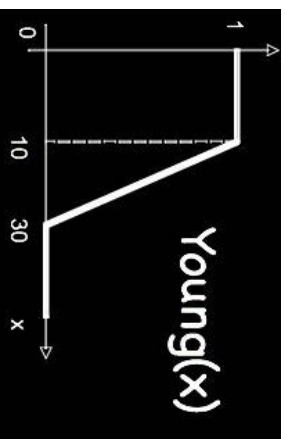
SportsCar = Car  $\sqcap$  HasSpeed.very(High)

$\mathcal{K} \models \langle \text{ferrari\_enzo}:\text{SportsCar}, 1 \rangle$

$\mathcal{K} \models \langle \text{audi\_tt}:\text{SportsCar}, 0.92 \rangle$

$\mathcal{K} \models \langle \text{audi\_tt}:\neg\text{SportsCar}, 0.72 \rangle$

## Example (Graded Subsumption)



Minor = Person  $\sqcap$   $\exists$ hasAge.  $\leq_{18}$

YoungPerson = Person  $\sqcap$   $\exists$ hasAge.Young

$\mathcal{K} \models \langle \text{Minor} \sqsubseteq \text{YoungPerson}, 0.2 \rangle$

Note: without explicit membership function of Young, **inference cannot be drawn**

## Basic principles of Fuzzy DLs

- In classical DLs, a concept  $C$  is interpreted by an interpretation  $\mathcal{I}$  as a set of individuals
- In fuzzy DLs, a concept  $C$  is interpreted by  $\mathcal{I}$  as a fuzzy set of individuals
- Each individual is instance of a concept to a degree in  $[0, 1]$
- Each pair of individuals is instance of a role to a degree in  $[0, 1]$

# Fuzzy $ALC$ concepts

$\mathcal{I}$	$=$	$\Delta^{\mathcal{I}}$	$t$	$=$	t-norm
$C^{\mathcal{I}}$	:	$\Delta^{\mathcal{I}} \rightarrow [0, 1]$	$s$	$=$	s-norm
$R^{\mathcal{I}}$	:	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow [0, 1]$	$n$	$=$	negation
			$i$	$=$	implication

Syntax	Semantics
$C, D \longrightarrow$	$T^{\mathcal{I}}(x)$
$\perp$	$\perp^{\mathcal{I}}(x)$
$A$	$A^{\mathcal{I}}(x)$
$C \sqcap D$	$(C_1 \sqcap C_2)^{\mathcal{I}}(x)$
$C \sqcup D$	$(C_1 \sqcup C_2)^{\mathcal{I}}(x)$
$\neg C$	$(\neg C)^{\mathcal{I}}(x)$
$\exists R.C$	$(\exists R.C)^{\mathcal{I}}(x)$
$\forall R.C$	$(\forall R.C)^{\mathcal{I}}(u)$

**Assertions:**  $\langle a:C, n \rangle, \mathcal{I} \models \langle a:C, n \rangle$  iff  $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq n$  (similarly for roles)

- individual  $a$  is instance of concept  $C$  at least to degree  $n$ ,  $n \in [0, 1] \cap \mathbb{Q}$

**Inclusion axioms:**  $C \sqsubseteq D$ ,

- $\mathcal{I} \models C \sqsubseteq D$  iff  $\forall x \in \Delta^{\mathcal{I}}. C^{\mathcal{I}}(x) \leq D^{\mathcal{I}}(x)$ , (alternative,  $\forall x \in \Delta^{\mathcal{I}}. i(C^{\mathcal{I}}(x), D^{\mathcal{I}}(x)) = 1$ )

# Basic Inference Problems

**Consistency:** Check if knowledge is meaningful

- Is  $\mathcal{K}$  consistent?

**Subsumption:** structure knowledge, compute taxonomy

- $\mathcal{K} \models C \sqsubseteq D$  ?

**Equivalence:** check if two fuzzy concepts are the same

- $\mathcal{K} \models C = D$  ?

**Graded instantiation:** Check if individual  $a$  instance of class  $C$  to degree at least  $n$

- $\mathcal{K} \models \langle a:C, n \rangle$  ?

**BTVB:** Best Truth Value Bound problem

- $glb(\mathcal{K}, a:C) = \sup\{n \mid \mathcal{K} \models \langle a:C, n \rangle\}$  ?

**Retrieval:** Rank set of individuals that instantiate  $C$  w.r.t. best truth value bound

- Rank the set  $\mathcal{R}(\mathcal{K}, C) = \{\langle a, glb(\mathcal{K}, a:C) \rangle\}$

## Some Notes on . . .

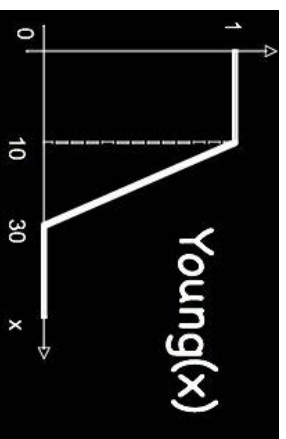
- Value restrictions:
  - In classical DLs,  $VR.C \equiv \neg \exists R. \neg C$
  - The same is not true, in general, in fuzzy DLs (depends on the operators' semantics, not true in Gödel logic).  
 $\text{VhasParent.Human} \not\equiv \neg \exists \text{hasParent.} \neg \text{Human} ??$
- Models:
  - In classical DLs  $\top \sqsubseteq \neg(\text{VR.A}) \sqcap (\neg \exists R. \neg A)$  has no classical model
  - In Gödel logic it has no finite model, but has an **infinite** model
- The **choice** of the appropriate semantics of the logical connectives is **important**.
  - Should have reasonable logical properties
  - **Certainly it must have efficient algorithms solving basic inference problems**

## Towards fuzzy OWL Lite and OWL DL

- Recall that OWL Lite and OWL DL relate to  $SHL\mathcal{F}(\mathcal{D})$  and  $SHOIN(\mathcal{D})$ , respectively
- We need to extend the semantics of fuzzy  $ALC$  to fuzzy  $SHOIN(\mathcal{D}) = ALCC\mathcal{R}_+HOIN\mathcal{R}(\mathcal{D})$
- Additionally, we add **modifiers** (e.g., **very**)
- Additionally, we add **concrete fuzzy concepts** (e.g., **Young**)

## Concrete fuzzy concepts

- E.g., Small, Young, High, etc. with explicit membership function
- Use the idea of concrete domains:
  - $D = \langle \Delta_D, \Phi_D \rangle$
  - $\Delta_D$  is an interpretation domain
  - $\Phi_D$  is the set of concrete fuzzy domain predicates  $d$  with a predefined arity  $n$  and **fixed** interpretation  $d^D: \Delta_D^n \rightarrow [0, 1]$
  - For instance,



$$\begin{aligned} \text{Minor} &= \text{Person} \sqcap \exists \text{hasAge} \leq 18 \\ \text{YoungPerson} &= \text{Person} \sqcap \exists \text{hasAge} \cdot \text{Young} \end{aligned}$$

## Modifiers

- Very, moreOrLess, slightly, etc.
- Apply to fuzzy sets to change their membership function
  - $\text{very}(x) = x^2$
  - $\text{slightly}(x) = \sqrt{x}$
- For instance,

`SportsCar = Car  $\sqcap$   $\exists$ speed.very(High)`

## Number Restrictions

- **May be a problem computationally**
- The semantics of the concept  $(\geq n S)$

$$(\geq n R)^T(x) = \sup_{\{y_1, \dots, y_n\} \subseteq \Delta^T} \bigwedge_{i=1}^n R^T(x, y_i)$$

- Is the result of viewing  $(\geq n R)$  as the open first order formula

$$\exists y_1, \dots, y_n. \bigwedge_{i=1}^n R(x, y_i) \wedge \bigwedge_{1 \leq i < j \leq n} y_i \neq y_j .$$

- The semantics of the concept  $(\leq n R)$

$$(\leq n R)^T(x) = \neg(\geq n + 1 R)^T(x)$$

- Note:  $(\geq 1 R) \equiv \exists R.T$

## Reasoning

- For full fuzzy *SHOIN*(D) or *SHTF*(D): **does not exist yet!!**
- **Exists** for fuzzy *ALC*(D) + modifiers + fuzzy concrete concepts (under Lukasiewicz semantics, also under “Zadeh semantics”)
- Usual fuzzy tableaux calculus **does not work anymore** (problems with modifiers and concrete fuzzy concepts)
- Usual fuzzy tableaux calculus does not solve the BTVB problem
- New algorithm uses **bounded Mixed Integer Programming oracle**, as for Many Valued Logics
  - Recall: the *general MILP problem* is to find

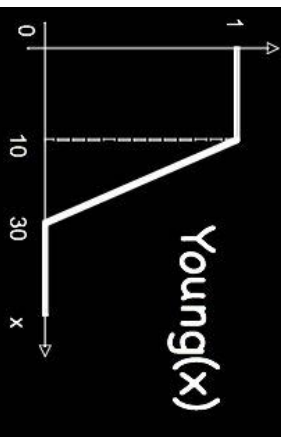
$$\bar{\mathbf{x}} \in \mathbb{Q}^k, \bar{\mathbf{y}} \in \mathbb{Z}^m$$

$$f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \min \{ f(\mathbf{x}, \mathbf{y}) : A\mathbf{x} + B\mathbf{y} \geq \mathbf{h} \}$$

$A, B$  integer matrixes

## Requirements

- Works for usual fuzzy DL semantics (Zadeh semantics) and Lukasiewicz logic
- Modifiers are definable as linear in-equations over  $\mathbb{Q}, \mathbb{Z}$  (e.g., linear hedges)
- Fuzzy concrete concepts are definable as linear in-equations over  $\mathbb{Q}, \mathbb{Z}$  (e.g., crisp, triangular, trapezoidal, left shoulder and right shoulder membership functions)



$$\text{Minor} = \text{Person} \sqcap \exists \text{hasAge.} \leq_{18}$$

$$\text{YoungPerson} = \text{Person} \sqcap \exists \text{hasAge.Young}$$

$$\text{Young} = \text{Is}(10, 30)$$

- Then

$$glb(\mathcal{K}, a:C) = \min\{x \mid \mathcal{K} \cup \{\langle a:C \leq x \rangle\} \text{ satisfiable}\}$$

$$glb(\mathcal{K}, C \sqsubseteq D) = \min\{x \mid \mathcal{K} \cup \{\langle a:C \sqcap \neg D \geq 1 - x \rangle\} \text{ satisfiable}\}$$

- Apply tableaux calculus, then use bounded Mixed Integer Programming oracle

## AFC Tableau rules (excerpt)

$x \bullet \{\langle C_1 \sqcap C_2, \geq, l \rangle, \dots\}$	$\longrightarrow \sqcap$	$x \bullet \{\langle C_1 \sqcap C_2, \geq, l \rangle, \langle C_1, \geq, l \rangle, \langle C_2, \geq, l \rangle, \dots\}$
$x \bullet \{\langle C_1 \sqcup C_2, \geq, l \rangle, \dots\}$	$\longrightarrow \sqcup$	$x \bullet \{\langle C_1 \sqcup C_2, \geq, l \rangle, \langle C_1, \geq, x_1 \rangle, \langle C_2, \geq, x_2 \rangle, x_1 + x_2 = l, x_1 \leq y, x_2 \leq 1 - y, x_i \in [0, 1], y \in \{0, 1\}, \dots\}$
$x \bullet \{\langle \exists R.C, \geq, l \rangle, \dots\}$	$\longrightarrow \exists$	$x \bullet \{\langle \exists R.C, \geq, l \rangle, \dots\}$ $\langle R, \geq, l \rangle \downarrow$ $y \bullet \{\langle C, \geq, l \rangle\}$
$x \bullet \{\langle \forall R.C, \geq, l_1 \rangle, \dots\}$	$\longrightarrow \forall$	$x \bullet \{\langle \forall R.C, \geq, l_1 \rangle, \dots\}$ $\langle R, \geq, l_2 \rangle \downarrow$ $y \bullet \{\dots, \langle C, \geq, x \rangle\}$ $x + y \geq l_1, x \leq y, l_1 + l_2 \leq 2 - y,$ $x \in [0, 1], y \in \{0, 1\}\}$
$\vdots$	$\vdots$	$\vdots$
$x \bullet \{A \sqsubseteq C, \langle A, \geq, l \rangle, \dots\}$	$\longrightarrow \sqsubseteq_1$	$x \bullet \{A \sqsubseteq C, \langle C, \geq, l \rangle, \dots\}$
$x \bullet \{C \sqsubseteq A, \langle A, \leq, l \rangle, \dots\}$	$\longrightarrow \sqsubseteq_2$	$x \bullet \{C \sqsubseteq A, \langle C, \leq, l \rangle, \dots\}$
$\vdots$	$\vdots$	$\vdots$

## Example

$$\bullet \text{ Suppose } \mathcal{K} = \begin{cases} A \sqcap B \sqsubseteq C \\ \langle a:A \geq 0.3 \rangle \\ \langle a:B \geq 0.4 \rangle \end{cases}$$

*Query* :=  $glb(\mathcal{K}, a:C) = \min\{x \mid \mathcal{K} \cup \{\langle a:C \leq x \rangle\} \text{ satisfiable}\}$

Step	Tree	
1.	$a \bullet \{\langle A, \geq, 0.3 \rangle, \langle B, \geq, 0.4 \rangle, \langle C, \leq, x \rangle\}$	(Hypothesis)
2.	$\cup \{\langle A \sqcap B, \leq, x \rangle\}$	$(\rightarrow \sqsubseteq_2)$
3.	$\cup \{\langle A, \leq, x_1 \rangle, \langle B, \leq, x_2 \rangle\}$ $\cup \{x = x_1 + x_2 - 1, 1 - y \leq x_1, y \leq x_2\}$ $\cup \{x_i \in [0, 1], y \in \{0, 1\}\}$	$(\rightarrow \sqcap_{\leq})$
4.	find $\min\{x \mid \langle a:A \geq 0.3 \rangle, \langle a:B \geq 0.4 \rangle,$ $\langle a:C \leq x \rangle, \langle a:A \leq x_1 \rangle, \langle a:B \leq x_2 \rangle,$ $x = x_1 + x_2 - 1, 1 - y \leq x_1, y \leq x_2,$ $x_i \in [0, 1], y \in \{0, 1\}\}$	(MILP Oracle)
5.	MILP oracle: $\mathbf{x} = \mathbf{0.3}$	

## Implementation issues

- Several options exists:
  - Try to map fuzzy DLs to classical DLs
  - Try to map fuzzy DLs to some fuzzy/annotated logic programming framework
  - Build an ad-hoc theorem prover for fuzzy DLs, using e.g., MILLP
- A theorem prover for fuzzy *A<sub>CC</sub>* + linear hedges + concrete fuzzy concepts, using MILLP, has been implemented
  - Looking for volunteers to catch-up to the expressive power of fuzzy OWL Lite or fuzzy OWL DL (EXPTIME, NEXPTIME class)

## Conclusions

- Classical DLs are the core of state of the art ontology description languages, as OWL DL and OWL Lite
- Efficient implementations exists, which take advantage of research on DLs decision algorithms and computational complexity analysis
- Fuzzy DLs aim at enhancing the expressive power of DLs towards the representation of vague concepts
- Research is still in it's infancy

## Future Work

- To get rid with the subtleties of both Description Logics and Fuzzy Logics, experts from both areas are needed
- Research directions:
  - Computational complexity of the fuzzy DLs family
  - Design of efficient reasoning algorithms
  - Combining fuzzy DLs with Logic Programming
  - Language extensions: e.g. fuzzy quantifiers
    - TopCustomer = Customer  $\sqcap$  (Usually)buys.ExpensiveItem
    - ExpensiveItem = Item  $\sqcap$   $\exists$ price.High
  - Developing a system
  - ...