

# 1 Description Logics

## 1.1 Formal Ontologies

### Formalizing Ontologies

- We heard about ontologies as “some shared knowledge structures often visualized through UML-like diagrams” ...
- How to express more complicated constructs like cardinalities, inverses, disjointness, etc.?
- How to check they are designed correctly? How to reason about the knowledge inside?
- We need a **formal language**.

### Logics for Ontologies

- propositional logic

#### Example

“John is clever.”  $\Rightarrow$   $\neg$ “John fails at exam.”

- first order predicate logic

#### Example

$(\forall x)(Clever(x) \Rightarrow \neg((\exists y)(Exam(y) \wedge Fails(x, y))))$ .

- modal logic

#### Example

$\Box((\forall x)(Clever(x) \Rightarrow \Diamond \neg((\exists y)(Exam(y) \wedge Fails(x, y))))$ .

- ... what is the meaning of these formulas ?

## 1 Description Logics

### Logics for Ontologies (2)

Logics are defined by their

- Syntax – to *represent* concepts (*defining symbols*)
- Semantics – to capture meaning of the syntactic constructs (*defining concepts*)
- Proof Theory – to enforce the semantics

### Logics trade-off

A logical calculus is always a trade-off between *expressiveness* and *tractability of reasoning*.

### Propositional Logic

#### Example

How to check satisfiability of the formula  $A \vee (\neg(B \wedge A) \vee B \wedge C)$  ?

**syntax** – atomic formulas and  $\neg, \wedge, \vee, \Rightarrow$

**semantics** ( $\models$ ) – an interpretation assigns true/false to each formula.

**proof theory** ( $\vdash$ ) – resolution, tableau

**complexity** – NP-Complete (Cook theorem)

### First Order Predicate Logic

#### Example

What is the meaning of this sentence ?

$$\begin{aligned} & (\forall x_1)((Student(x_1) \wedge (\exists x_2)(GraduateCourse(x_2) \wedge isEnrolledTo(x_1, x_2)))) \\ & \Rightarrow (\forall x_3)(isEnrolledTo(x_1, x_3) \Rightarrow GraduateCourse(x_3)) \end{aligned}$$

$$Student \sqcap \exists isEnrolledTo.GraduateCourse \sqsubseteq \forall isEnrolledTo.GraduateCourse$$

### First Order Predicate Logic – quick informal review

**syntax** – constructs involve

**term** (variable  $x$ , constant symbol  $JOHN$ , function symbol applied to terms  $fatherOf(JOHN)$ )

**axiom/formula** (predicate symbols applied to terms  $hasFather(x, JOHN)$ , possibly glued together with  $\neg, \wedge, \vee, \Rightarrow, \forall, \exists$ )

**universally closed formula** formula without free variable  $((\forall x)(\exists y)hasFather(x, y) \wedge Person(y))$

**semantics** – an interpretation (with valuation) assigns:

**domain element** to each term

**true/false** to each closed formula

**proof theory** – resolution; *Deduction Theorem, Soundness Theorem, Completeness Theorem*

**complexity** – undecidable (Goedel)

### Open World Assumption

#### OWA

FOPL accepts Open World Assumption, i.e. whatever is not known is not necessarily false.

As a result, FOPL is *monotonic*, i.e.

#### monotonicity

No conclusion can be invalidated by adding extra knowledge.

This is in contrary to relational databases, or Prolog that accept Closed World Assumption.

## 1.2 Towards Description Logics

### Languages sketched so far aren't enough ?

- Why not First Order Predicate Logic ?
  - ⊙ FOPL is undecidable – many logical consequences cannot be verified in finite time.
    - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ⊙ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.

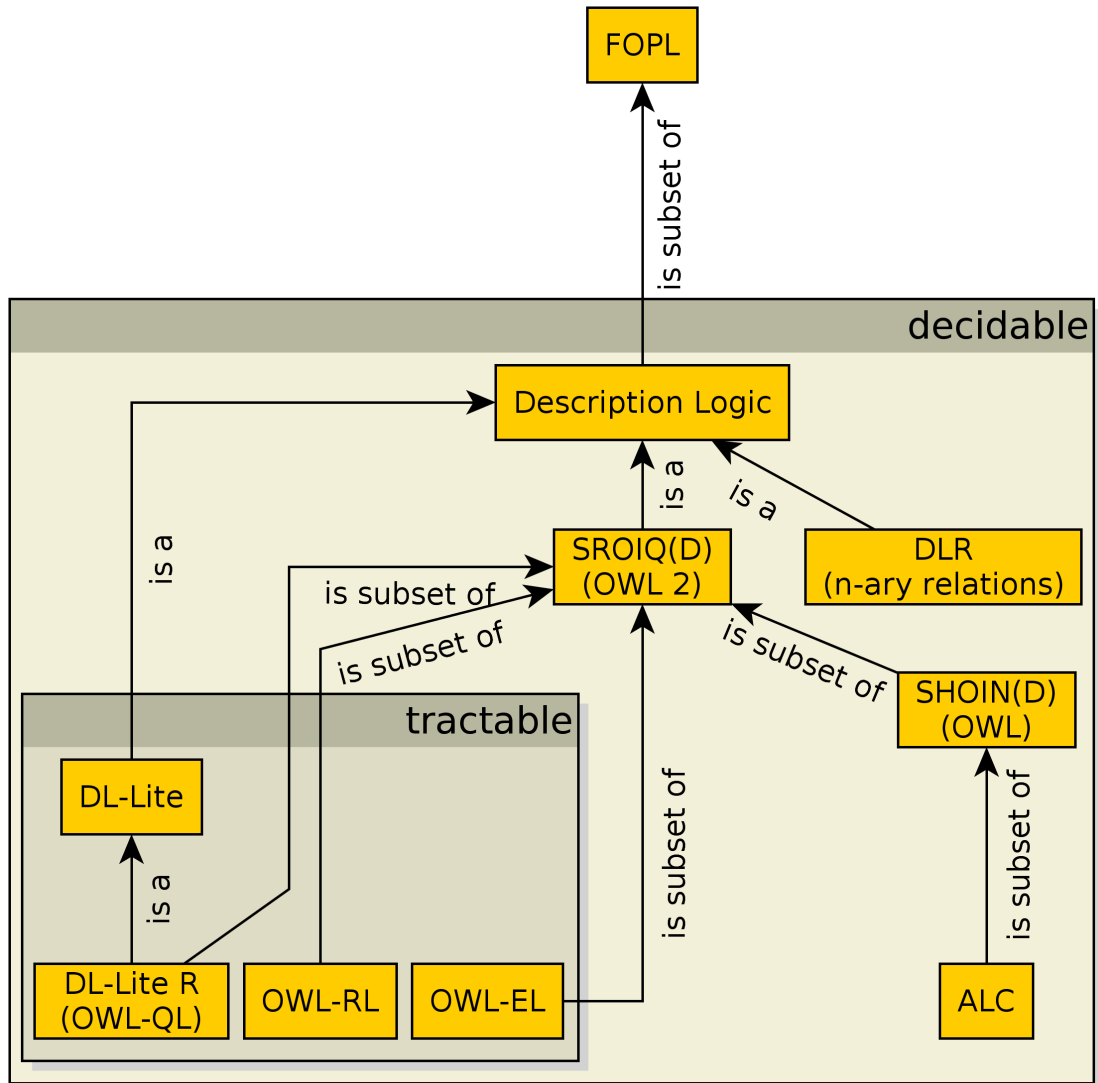
### What are Description Logics ?

Description logics (DLs) are (almost exclusively) decidable subsets of FOPL aimed at modeling *terminological incomplete knowledge*.

- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.

## 1 Description Logics

- 90's *ALC*
- 2004 *SHOIN(D)* – OWL
- 2009 *SROIQ(D)* – OWL 2



### 1.3 *ALC* Language

#### Concepts and Roles

- Basic building blocks of DLs are :

**(atomic) concepts** - representing (named) *unary predicates* / classes, e.g. *Parent*, or  $Person \sqcap \exists hasChild \cdot Person$ .

**(atomic) roles** - represent (named) *binary predicates* / relations, e.g. *hasChild*

**individuals** - represent ground terms / individuals, e.g. *JOHN*

- Theory  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$  (in OWL referred as Ontology) consists of a

**TBOX**  $\mathcal{T}$  - representing axioms generally valid in the domain, e.g.  $\mathcal{T} = \{Man \sqsubseteq Person\}$

**ABOX**  $\mathcal{A}$  - representing a particular relational structure (data), e.g.  $\mathcal{A} = \{Man(JOHN), loves(JOHN, MARY)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

### Semantics, Interpretation

- as  $\mathcal{ALC}$  is a subset of FOPL, let's define semantics analogously (and restrict interpretation function where applicable):
- **Interpretation** is a pair  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ , where  $\Delta^{\mathcal{I}}$  is an interpretation domain and  $\cdot^{\mathcal{I}}$  is an interpretation function.
- Having *atomic* concept  $A$ , *atomic* role  $R$  and individual  $a$ , then

$$\begin{aligned} A^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \\ R^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\ a^{\mathcal{I}} &\in \Delta^{\mathcal{I}} \end{aligned}$$

### $\mathcal{ALC}$ (= attributive language with complements)

Having concepts  $C, D$ , atomic concept  $A$  and atomic role  $R$ , then for interpretation  $\mathcal{I}$  :

concept	concept <sup><math>\mathcal{I}</math></sup>	description
$\top$	$\Delta^{\mathcal{I}}$	(universal concept)
$\perp$	$\emptyset$	(unsatisfiable concept)
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	(negation)
$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$	(intersection)
$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$	(union)
$\forall R \cdot C$	$\{a \mid \forall b((a, b) \in R^{\mathcal{I}} \implies b \in C^{\mathcal{I}})\}$	(universal restriction)
$\exists R \cdot C$	$\{a \mid \exists b((a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}})\}$	(existential restriction)

axiom	$\mathcal{I} \models$ axiom iff	description
$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$	(inclusion)
$C_1 \equiv C_2$	$C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$	(equivalence)

axiom	$\mathcal{I} \models$ axiom iff	description
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	(concept assertion)
$R(a_1, a_2)$	$(a_1^{\mathcal{I}}, a_2^{\mathcal{I}}) \in R^{\mathcal{I}}$	(role assertion)

<sup>1</sup>two different individuals denote two different domain elements

***ALC* – Example**

**Example**

Consider an information system for genealogical data integrating multiple genealogical databases. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

- Set of persons that have just men as their descendants (if any)
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ? (specify an *axiom*)
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x (GrandParent(x) \equiv (Person(x) \wedge \exists y (hasChild(x, y) \wedge \exists z (hasChild(y, z))))))$$

***ALC* Example –  $\mathcal{T}$**

**Example**

$$\begin{aligned} Woman &\equiv Person \sqcap Female \\ Man &\equiv Person \sqcap \neg Woman \\ Mother &\equiv Woman \sqcap \exists hasChild \cdot Person \\ Father &\equiv Man \sqcap \exists hasChild \cdot Person \\ Parent &\equiv Father \sqcup Mother \\ Grandmother &\equiv Mother \sqcap \exists hasChild \cdot Parent \\ MotherWithoutDaughter &\equiv Mother \sqcap \forall hasChild \cdot \neg Woman \\ Wife &\equiv Woman \sqcap \exists hasHusband \cdot Man \end{aligned}$$

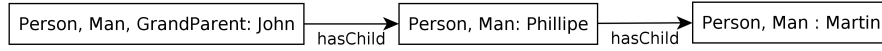
**Interpretation – Example**

**Example**

- Consider a theory  $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent\})$ . Find some model.
- a model of  $\mathcal{K}_1$  can be interpretation  $\mathcal{I}_1$  :
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillippe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillippe), (Phillippe, Martin)\}$

- $GrandParent^{\mathcal{I}_1} = \{John\}$
- $JOHN^{\mathcal{I}_1} = \{John\}$

- this model is finite and has the form of a tree with the root in the node John :



### Shape of DL Models

The last example revealed several important properties of DL models:

#### Tree model property (TMP)

Every consistent  $\mathcal{K} = (\{\}, \{C(I)\})$  has a model in the shape of a *rooted tree*.

#### Finite model property (FMP)

Every consistent  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$  has a *finite model*.

Both properties represent important characteristics of  $\mathcal{ALC}$  that significantly speed-up reasoning.

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

### Example – CWA $\times$ OWA

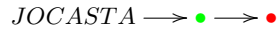
#### Example

**ABOX**  $\begin{matrix} hasChild(JOCASTA, OEDIPUS) & hasChild(JOCASTA, POLYNEIKES) \\ hasChild(OEDIPUS, POLYNEIKES) & hasChild(POLYNEIKES, THERSANDROS) \\ Patricide(OEDIPUS) & \neg Patricide(THERSANDROS) \end{matrix}$

Edges represent role assertions of *hasChild*; red/green colors distinguish concepts instances – *Patricide* a  $\neg Patricide$



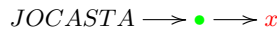
**Q1**  $(\exists hasChild \cdot (Patricide \sqcap \exists hasChild \cdot \neg Patricide))(JOCASTA),$



**Q2** Find individuals  $x$  such that  $\mathcal{K} \models C(x)$ , where  $C$  is

$\neg Patricide \sqcap \exists hasChild^- \cdot (Patricide \sqcap \exists hasChild^- \cdot \{JOCASTA\})$

What is the difference, when considering CWA ?



### References





## Bibliography

- [1] \* Vladimír Mařík, Olga Štěpánková, and Jiří Lažanský. *Umělá inteligence 6 [in czech], Chapters 2-4*. Academia, 2013.
- [2] \* Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook, Theory, Implementation and Applications, Chapters 2-4*. Cambridge, 2003.
- [3] \* Enrico Franconi. *Course on Description Logics*. <http://www.inf.unibz.it/franconi/dl/course/>, cit. 22.9.2013.