

# Chapter 7

## Ontologies and the Web Ontology Language – OWL

- *vocabularies* can be defined by RDFS
  - not so much stronger than the ER Model or UML (even weaker: no cardinalities)
  - not only a conceptual model, but a “real language” with a close connection to the data level (RDF)
  - *incremental* world-wide approach
  - “global” vocabulary can be defined by autonomous partners
- but: still restricted when *describing* the vocabulary.

283

Ontologies/ontology languages further extend the expressiveness:

- Description Logics
- Topic Maps (in SGML) since early 90s, XTM (XML Topic Maps)
- Ontolingua – non-XML approach from the Knowledge Representation area
- OIL (Ontology Inference Layer): initiative funded by the EU programme for Information Society Technologies (project: On-To-Knowledge, 1.2000-10.2002); based on RDF/RDFS
- DAML (Darpa Agent Markup Language; 2000) ... first ideas for a Semantic Web language
- DAML+OIL (Jan. 2001)
- developed into OWL (1st version March 02, finalized Feb. 04)

284

## THREE VARIANTS OF OWL

Several expressiveness/complexity/decidability levels:

- OWL Full: extension of RDF
  - classes can also be regarded as individuals (classes of classes ... higher-order reasoning)
- OWL DL
  - fragment of OWL that fits into the [Description Logics](#) Framework
  - decidable reasoning
- OWL Lite
  - subset of OWL DL
  - easier migration from frame-based tools (note: F-Logic was a frame-based framework)
  - easier reasoning

285

## 7.1 Description Logics

- Focus on the description of *concepts*, not of instances
- Terminological Reasoning
- Origin of DLs: Semantic Networks (graphical formalism)

### Notions

- Concepts (= classes),  
note: literal datatypes (string, integer etc.) are not classes in DL and OWL, but *data ranges*  
(cf. XML Schema: distinction between `simpleTypes` and `complexType`)
- Roles (= relationships),
- A Description Logic alphabet consists of a finite set of concept names (e.g. Person, Cat, LivingBeing, Male, Female, ...) and a finite set of role names (e.g., hasChild, marriedTo, ...),
- constructors for derived concepts and roles,
- axioms for asserting facts about concepts and roles.

286

## COMPARISON WITH OTHER LOGICS

Syntax and semantics defined different but similar from first-order logic

- formulas over an alphabet and a small set of additional symbols and combinators
- semantics defined via *interpretations* of the combinators
- set-oriented, no instance variables  
(FOL: instance-oriented with domain quantifiers)
- family of languages depending on what combinators are allowed.

The base:  $\mathcal{AL}$

The usual starting point is  $\mathcal{AL}$ :

- “attributive language”
- Manfred Schmidt-Schauss and Gert Smolka: *Attributive Concept Descriptions with Complements*. In *Artificial Intelligence* 48(1), 1991, pp. 1–26.
- extensions (see later:  $\mathcal{ALC}$ ,  $\mathcal{ALCQ}$ ,  $\mathcal{ALCQ}(D)$ ,  $\mathcal{ALCQI}$ ,  $\mathcal{ALCN}$  etc.)

287

## ATOMIC, NAMED CONCEPTS

- atomic concepts, e.g., Person, Male, Female
- the “universal concept”  $\top$  (often called “Thing” – everything is an instance of Thing)
- the empty concept  $\perp$  (“Nothing”). There is no thing that is an instance of  $\perp$ .

## SET OPERATIONS

- intersection of concepts:  $A \sqcap B$
- negation:  $\neg A$   
 $\mathcal{AL}$  allows only atomic negation.
- union:  $A \sqcup B$   
Union is not allowed in  $\mathcal{AL}$ .

288

## INTENSIONAL CONCEPTS

Concepts (as an intensional characterization of sets of instances) can be described implicitly by their properties (wrt. *roles*).

Let  $R$  be a role,  $C$  a concept. Then, the expressions  $\exists R.C$  and  $\forall R.C$  also describe concepts (intensionally defined concepts) by constraining the roles:

- Existential quantification:  $\exists R.C$  – all things that have a *filler* for the role  $R$  that is in  $C$ .  
 $\exists \text{hasChild.Male}$  describes all things that have a male child.
- $\mathcal{AL}$ : only as restricted existential quantification:  $\exists R.\top$   
 $\exists \text{hasChild.}\top$  describes all things that have a child (formally: that belongs to the concept “Thing”).
- Range constraints:  $\forall R.C$   
 $\forall \text{hasChild.Male}$  describes all things that have only male children (including those that have no children at all).
- Note that  $\perp$  can be used to express non-existence:  $\forall R.\perp$  describes all things where all fillers of role  $R$  are of the concept  $\perp$  (= Nothing) – i.e., all things that do not have a filler for the role  $R$ .  
 $\forall \text{hasChild.}\perp$  describes the things that have no children.

289

## SEMANTICS OF CONCEPT CONSTRUCTORS

As usual: by interpretations.

An interpretation  $\mathcal{I}$  consists of the following:

- a domain  $\mathcal{D}$ ,
- for every concept name  $C$ :  $C^{\mathcal{I}} \subseteq \mathcal{D}$  is a subset of the domain,
- for every role name  $R$ :  $R^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$  is a binary relation over the domain.

### Structural Induction

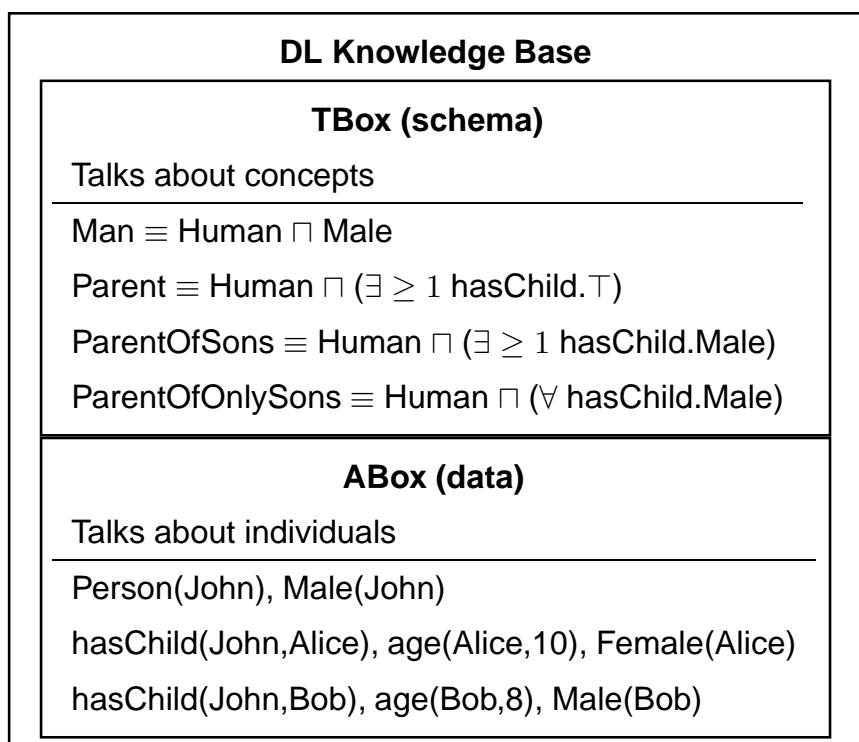
- $(A \sqcup B)^{\mathcal{I}} = A^{\mathcal{I}} \cup B^{\mathcal{I}}$
- $(A \sqcap B)^{\mathcal{I}} = A^{\mathcal{I}} \cap B^{\mathcal{I}}$
- $(\neg A)^{\mathcal{I}} = \mathcal{D} \setminus A^{\mathcal{I}}$
- $(\exists R.C)^{\mathcal{I}} = \{x \mid \text{there is an } y \text{ such that } (x, y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$
- $(\forall R.C)^{\mathcal{I}} = \{x \mid \text{for all } y \text{ such that } (x, y) \in R^{\mathcal{I}}, y \in C^{\mathcal{I}}\}$

### Example

$\text{Male} \sqcap \forall \text{hasChild.Male}$  is the set of all men who have only sons.

290

## STRUCTURE OF A DL KNOWLEDGE BASE



291

## THE TBOX: TERMINOLOGICAL AXIOMS

Definitions and assertions (not to be understood as constraints) about concepts:

- concept subsumption:  $C \sqsubseteq D$ ; defining a concept hierarchy.  $\mathcal{I} \models C \sqsubseteq D \Leftrightarrow C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ .
- concept equivalence:  $C \equiv D$ ; often used for defining the left-hand side concept.  
Semantics:  $\mathcal{I} \models C \equiv D \Leftrightarrow C \sqsubseteq D$  and  $D \sqsubseteq C$ .
- analogous for role subsumption and role equivalence.

### TBox Reasoning

- is a concept  $C$  satisfiable?
- is  $C \sqsubseteq D$  implied by a TBox
- given the definition of a new concept  $D$ , classify it wrt. the given concept hierarchy.

292

## THE ABOX: ASSERTIONAL AXIOMS

- contains the facts about instances (using names for the instances) in terms of the basic concepts and roles:  
`Person(John), Male(John), hasChild(John,Alice)`
- contains also knowledge in terms of intensional concepts, e.g., `∃hasChild.Male(John)`

### TBox + ABox Reasoning

- check consistency between ABox and a given TBox
- ask whether a given instance satisfies a concept  $C$
- ask for all instances that have a given property
- ask for the most specific concepts that an instance satisfies

Note: instances are allowed only in the ABox, not in the TBox.

If instances should be used in the definition of concepts (e.g., “European Country” or “Italian City”), *Nominals* must be used (see later).

293

## EXTENSIONS TO $\mathcal{AL}$

- $\mathcal{U}$ : “union”; e.g. `Parent  $\equiv$  Father  $\sqcup$  Mother`.
- $\mathcal{C}$ : negation (“complement”) of non-atomic concepts.  
`Person  $\sqcap$   $\neg$ ∃hasChild.⊤` characterizes the set of persons who have no children (note: open-world semantics of negation!)

Note: the FOL equivalent would be expressed via variables:

$$\forall x(\text{Childless}(x) \leftrightarrow (\text{Person}(x) \wedge \neg \exists y(\text{hasChild}(x, y))))$$

- $\mathcal{E}$ : unrestricted existential quantification of the form `∃R.C`.  
`∃hasChild.Male`

Note: the FOL equivalent uses variables:

$$p(x) \leftrightarrow \exists y(\text{hasChild}(x, y) \wedge \text{male}(y)),$$

or `∃hasChild.hasChild.⊤` for grandparents.

- $\mathcal{N}$ : (unqualified) cardinalities of roles (“number restrictions”).  
`( $\geq 3$  hasChild.⊤)` for persons who have at least 3 children.
- $\mathcal{Q}$ : qualified role restrictions like `( $\leq 2$  hasChild.Male)`. A weaker form,  $\mathcal{F}$ , is restricted to cardinalities 0, 1 and “arbitrary”.

294

## THE EXTENDED LANGUAGES

- $\mathcal{AL}$  has no “branching” (no union, or any kind of disjunction; so tableau proofs in  $\mathcal{AL}$  are linear.  
Exercise: show why unrestricted existential quantification  $\exists R.C$  in contrast to  $\exists R.\top$  leads to branching.
- The logics are named by the letters, e.g.  $\mathcal{ALUN}$  for  $\mathcal{AL}$  with union and unqualified  $n$ -cardinalities.
- $\mathcal{U}$  and  $\mathcal{E}$  can be expressed by  $\mathcal{C}$ .  
Thus,  $\mathcal{ALC}$  is frequently used.
- $\mathcal{ALC}$  is the “smallest” Description Logic that is closed wrt. the set operations.
- A frequently used restriction of  $\mathcal{AL}$  is called  $\mathcal{FL}^-$  (for “Frame-Language”), which is obtained by disallowing negation completely (i.e., having only positive knowledge).

295

## COMPLEXITY AND DECIDABILITY: OVERVIEW

- Logic  $\mathcal{L}^2$ , i.e., FOL with only two (reusable) variable symbols is decidable.
- Full FOL is undecidable.
- DLs: incremental, modular set of semantical notions.
- only part of FOL is required for concept reasoning.
- $\mathcal{ALC}$  can be *expressed* by FOL, but then, the inherent semantics is lost  $\rightarrow$  full FOL reasoner required.
- Actually,  $\mathcal{ALC}$  can be encoded in FOL by only using two variables  $\rightarrow$   $\mathcal{ALC}$  is decidable.
- Consistency checking of  $\mathcal{ALC}$ -TBoxes and -ABoxes is PSPACE-complete (proof by reduction to *Propositional Dynamic Logic* which is in turn a special case of propositional multimodal logics).  
There are algorithms that are efficient in the average case.
- $\mathcal{ALCN}$  goes beyond  $\mathcal{L}^2$  and PSPACE. Reduction to  $\mathcal{C}^2$  (including “counting” quantifiers) yields decidability, but now in NEXPTIME). There are algorithms for  $\mathcal{ALCN}$  and even  $\mathcal{ALCQ}$  in PSPACE.

296

## FURTHER EXTENSIONS

- *Role Constructors*, i.e., derived roles as union or intersection  
( $\text{hasChild} \equiv \text{hasSon} \cup \text{hasDaughter}$ ), concatenation  
( $\text{hasGrandchild} \equiv \text{hasChild} \circ \text{hasChild}$ ), transitive closure ( $\text{hasDescendant} \equiv \text{hasChild}^+$ )  
(indicated by e.g.  $\mathcal{ALC}_{reg}$ ), and inverse ( $\text{isChildOf} \equiv \text{hasChild}^-$ ) ( $\mathcal{I}$ ).
- *Data types* (indicated by "(D)"), e.g. integers.  
 $\text{Adult} \equiv \text{Person} \sqcap \exists \text{age.} \geq 18$ .
- *Nominals* ( $\mathcal{O}$ ) allow to use individuals from the ABox also in the TBox.  
 $\text{GermanCity} \equiv \forall \text{inCountry. Germany}$   
They are used in a class constructor like  $\text{one-of}\{o_1, \dots, o_n\}$  (for defining enumeration concepts) or in  $\text{has-value}\{x\}$  for value constraints of properties.
- *Role-Value-Maps*:  
Equality Role-Value-Map:  $(R_1 = R_2) \equiv \{x \mid R_1(x, y) \leftrightarrow R_2(x, y)\}$ .  
Containment Role-Value-Map:  $(R_1 \subseteq R_2) \equiv \{x \mid R_1(x, y) \rightarrow R_2(x, y)\}$ .  
 $\text{knows} \subseteq \text{likes}$  for people who like all people they know.

297

## SEMANTICS OF EXTENSIONS

- $(\geq nR.C)^{\mathcal{I}} = \{x \mid \#\{y \mid (x, y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \geq n\}$ ,
- $(\leq nR.C)^{\mathcal{I}} = \{x \mid \#\{y \mid (x, y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \leq n\}$ ,
- $(nR.C)^{\mathcal{I}} = \{x \mid \#\{y \mid (x, y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} = n\}$ ,
- $(R \circ S)^{\mathcal{I}} = \{(x, z) \mid \exists y : (x, y) \in R^{\mathcal{I}} \text{ and } (y, z) \in S^{\mathcal{I}}\}$ ,
- $(R^-)^{\mathcal{I}} = \{(y, x) \mid (x, y) \in R^{\mathcal{I}}\}$ ,
- $(R^+)^{\mathcal{I}} = (R^{\mathcal{I}})^+$ .
- If Nominals are used,  $\mathcal{I}$  also assigns an element of  $\mathcal{D}$  to each nominal symbol  $x$ .  
 $\{i_1, \dots, i_n\}^{\mathcal{I}} = \{i_1^{\mathcal{I}}, \dots, i_n^{\mathcal{I}}\}$ , and  
 $R.y = \{x \mid \{z \mid (x, z) \in R^{\mathcal{I}}\} = \{y\}\}$ .

298



## COMPLEXITY OF EXTENSIONS

- Role constructors:  $\mathcal{ALC}_{reg}$ , including transitivity, composition and union is EXPTIME-complete; this stays the same when inverse roles and even cardinalities for *atomic* roles are added ( $\mathcal{ALCQI}_{reg}$ ).  
Recall that inverse and transitive closure are important for ontologies.
- Combining such *composite* roles with cardinalities becomes undecidable (encoding in FOL requires 3 variables).
- Encoding of Role-Value Maps with composite roles in FOL is undecidable (encoding in FOL requires 3 variables; the logic loses the *tree model property*).
- $\mathcal{ALCQI}_{reg}$  with role-value maps restricted to boolean compositions of *basic* roles remains decidable. Decidability is also preserved when role-value-maps are restricted to functional roles.

299

## DESCRIPTION LOGIC MODEL THEORY

The definition is the same as in FOL:

- an interpretation is a model of an ABox  $A$  if
  - for every atomic concept  $C$  and individual  $x$  such that  $C(x) \in A$ ,  $x^{\mathcal{I}} \in C^{\mathcal{I}}$ , and
  - for every atomic role  $R$  and individuals  $x, y$  such that  $R(x, y) \in A$ ,  $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in R^{\mathcal{I}}$ .
- note: the interpretation of the non-atomic concepts and roles is given as before,
- all axioms  $\phi$  of the TBox are satisfied, i.e.,  $\mathcal{I} \models \phi$ .

Based on this, DL entailment is also defined as before:

- a set  $\Phi$  of formulas entails another formula  $\Psi$  (denoted by  $\Phi \models \Psi$ ), if  $\Psi^{\mathcal{I}} = \text{true}$  in all models of  $\Phi$ .

300

## DECIDABILITY, COMPLEXITY, AND ALGORITHMS

Many DLs are decidable, but in high complexity classes.

- decidability is due to the fact that often *local* properties are considered, and the verification proceeds tree-like through the graph without connections between the branches.
- This locality does not hold for cardinalities over composite roles, and for role-value maps – these lead to undecidability.
- Reasoning algorithms for  $\mathcal{ALC}$  and many extensions are based on tableau algorithms, some use model checking (finite models), others use tree automata.

### Three types of Algorithms

- restricted (to polynomial languages) and complete
- expressive logics with complete, worst-case EXPTIME algorithms that solve realistic problems in “reasonable” time. (Fact, Racer, Pellet)
- more expressive logics with incomplete reasoning.

301

## EXAMPLE

- Given facts:  $\text{Person} \equiv \text{Male} \sqcup \text{Female}$  and  $\text{Person}(\text{unknownPerson})$ .
- Query  $?-\text{Male}(X)$  yields an empty answer
- Query  $?-\text{Female}(X)$  yields an empty answer
- Query  $?-(\text{Male} \sqcup \text{Female})(X)$  yields  $\text{unknownPerson}$  as an answer
- for query answering, *all* models of the TBox+ABox are considered.
- in some models, the  $\text{unknownPerson}$  is Male, in the others it is female.
- in all models it is in  $(\text{Male} \sqcup \text{Female})$ .

302