



Description Logics and Ontologies

Maurizio Lenzerini

DASI-lab, Data and Service Integration Laboratory
Dipartimento di Informatica e Sistemistica “A. Ruberti”
Università di Roma “La Sapienza”

*Seminars in Artificial Intelligence
A.Y. 2018-2019*

Many names for the same notion

- Semantic networks
- Conceptual model
- Domain model
- Type network
- Type hierarchy
- Class hierarchy
- Concept base
- Knowledge graph
- Database schema
- Conceptual graph
- RDF graph
-
- **Ontology**

- 1. Introduction to
(computational) ontologies**
- 2. Ontology languages**
- 3. Reasoning**
- 4. Conclusion**



The notion of ontology

- **Ontology** as “the metaphysical study of the nature of being and existence” is as old as the discipline of philosophy.
- More recently, ontologies have been studied in fields such as artificial intelligence, knowledge representation, because of the need to categorize and structure entities and concepts of interest.
- **Computational ontology**: a conceptualisation of a domain of interest, expressed in a computational format, i.e. in such a way that it can be manipulated by the computer to aid human and machine agents in their performance of tasks within that domain.

The structure of a computational ontology

- An ontology is specified at different levels:
 - **Meta-level**: specifies a set of modeling categories
 - **Intensional level**: specifies a set of elements (instances of categories) and constraints used to structure the description of the domain
 - **Extensional level**: specifies an actual world description (instances of elements) that is coherent with respect to the intensional level



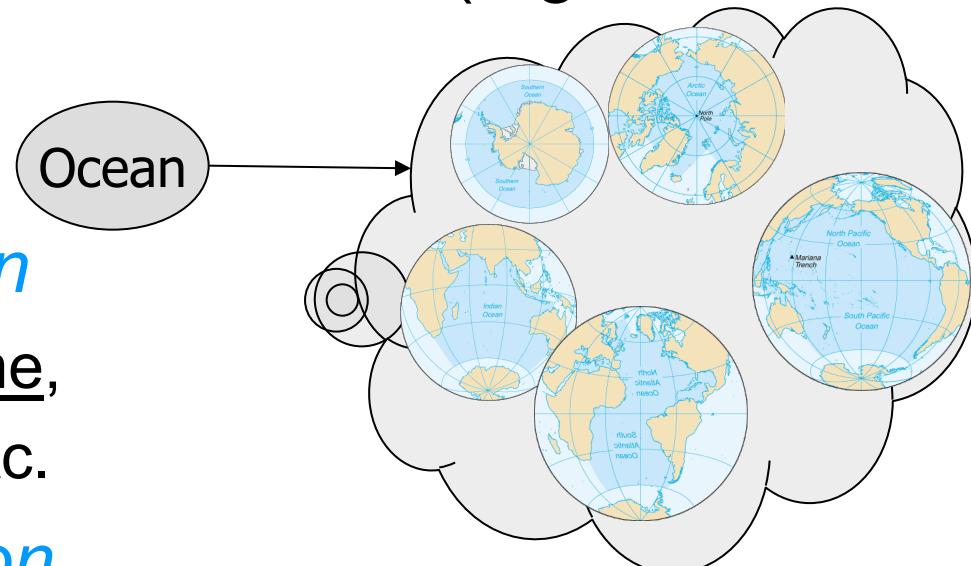
- 1. Introduction to ontologies**
- 2. Ontology languages**
- 3. Reasoning**
- 4. Conclusion**

Ontology languages

- An ontology language for expressing the intensional level usually includes constructs for:
 - Concepts
 - Properties of concepts
 - Relationships between concepts, and their properties
 - Axioms
 - Individuals and facts about individuals
 - Queries
- Ontologies are typically rendered as diagrams (e.g., **Semantic Networks**, Entity-Relationship schemas, **UML** class diagrams)

Concepts

- A *concept* is an element of the ontology that denotes a collection of instances (e.g., the set of “oceans”)
- *Intensional definition*
 - Specification of name, relations, axioms, etc.
- *Extensional definition*
 - Specification of the instances



Properties

- A *property* qualifies an element (e.g., a concept) of an ontology
- Property definition (intensional and extensional)
 - Name
 - Type
 - Atomic (integer, real, string, ...)
e.g., “eye-color” → {**blu**, **brown**, **green**, **grey**}
 - Structured (date,sets,lists...)
e.g., “date” → day/month/year
 - Default value

Relationships

- A *relationship* expresses an association among concepts
- *Intensional definition*
 - Specification of involved concepts (example: workFor is defined on Employee and Company)
- *Extensional definition*
 - Specification of the occurrences, called facts (worksFor(Fulvio,IASI))

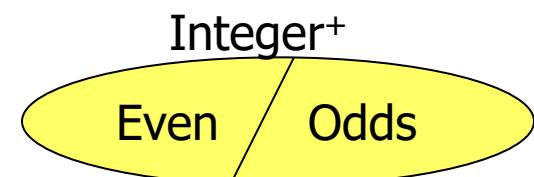
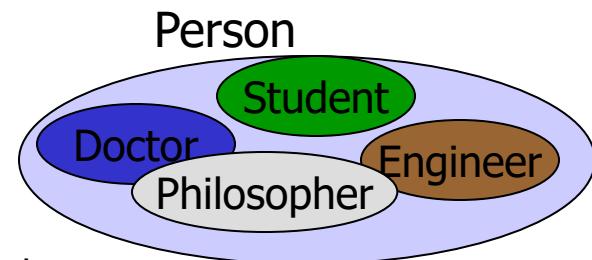
Axioms

- An *axiom* is a logical formula that expresses at the intensional level a condition that must be satisfied by the elements at the extensional level

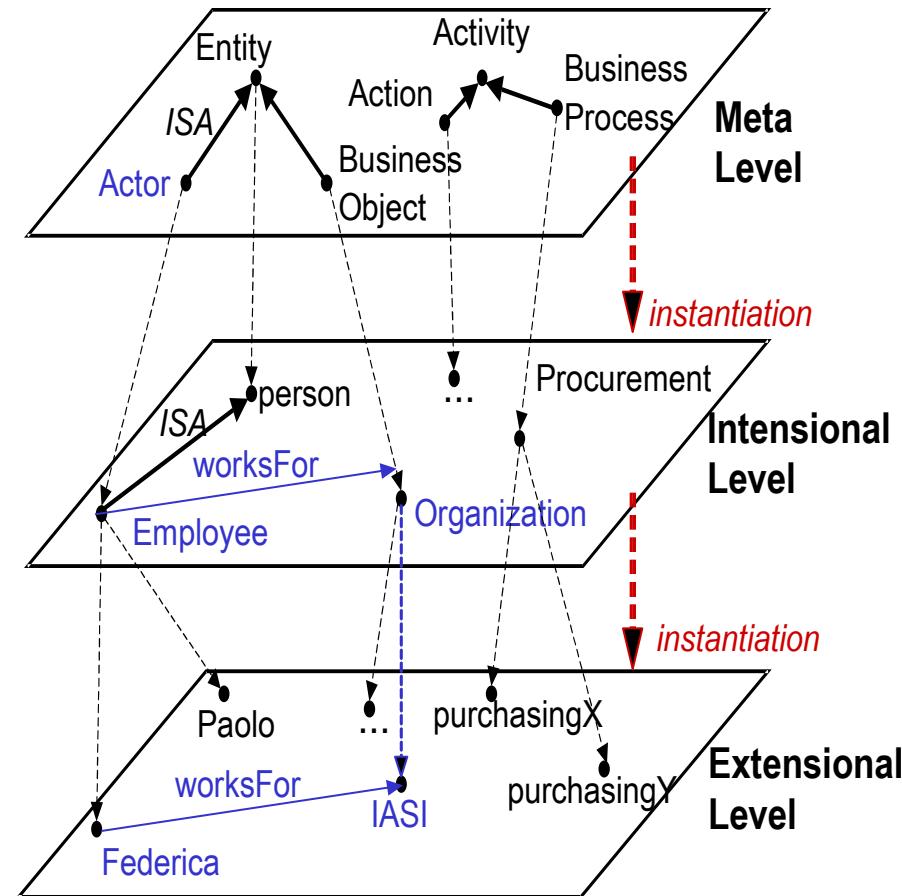
- Examples:

$\text{Person} \supseteq \text{Student} \cup \text{Doctor} \cup \text{Engineer} \cup \text{Philosopher}$

$\text{Integer}^+ = \text{Even} \cup \text{Odds}$, $\text{Even} \cap \text{Odds} = \emptyset$



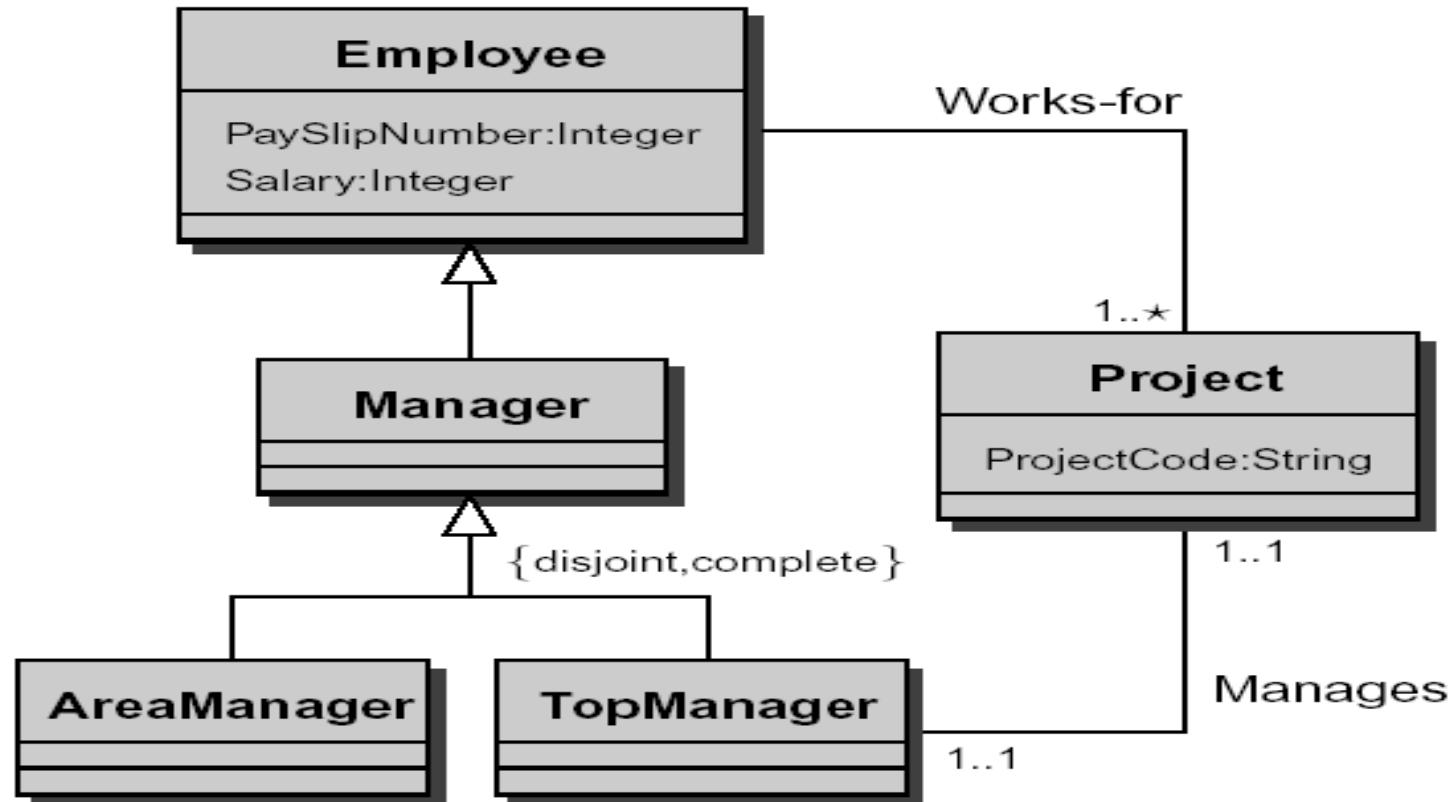
Instantiation



Queries

- An ontology language may also include constructs for expressing **queries**
 - **Queries**: expressions at the intensional level denoting collections of individuals satisfying a given condition
 - **Meta-queries**: expressions at the meta level denoting collections of elements satisfying a given condition
- The constructs for queries may be different from the constructs forming concepts and relationships

Example of query



$\{ (x.\text{Salary}, y.\text{ProjectCode}) \mid \text{Manages}(x,y) \wedge \neg \text{Works-for}(x,y) \}$

A family of ontology languages: Description Logics

We start with alphabets for concepts, roles, and individuals. Syntactically, concepts and roles are either atomic (i.e., denoted by a name), or non-atomic, i.e. built out using the constructors of a given **description language \mathcal{L}** .

An **interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$** consists of

- a nonempty set $\Delta^{\mathcal{I}}$, the domain of \mathcal{I}
- a function $\cdot^{\mathcal{I}}$, the interpretation function of \mathcal{I} , that maps
 - every individual to an element of $\Delta^{\mathcal{I}}$
 - every concept to a subset of $\Delta^{\mathcal{I}}$
 - every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

in such a way that suitable equations are satisfied.

Concept constructors

- atomic concept: $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ ($\perp^{\mathcal{I}} = \emptyset$, $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$)
- conjunction: $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- disjunction: $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- negation: $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- universal quantification: $(\forall R.C)^{\mathcal{I}} = \{a \mid \forall b. (a, b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$
- existential quantification: $(\exists R.C)^{\mathcal{I}} = \{a \mid \exists (a, b) \in R^{\mathcal{I}}. b \in C^{\mathcal{I}}\}$
- unqualified existential quantification: $\exists R$ equivalent to $\exists R.\top$
- qualified number restrictions
 $(\geq n R.C)^{\mathcal{I}} = \{a : |\{b \in C^{\mathcal{I}} : (a, b) \in R^{\mathcal{I}}\}| \geq n\}$
 $(\leq n R.C)^{\mathcal{I}} = \{a : |\{b \in C^{\mathcal{I}} : (a, b) \in R^{\mathcal{I}}\}| \leq n\}$
- unqualified number restrictions: $(\geq n R)$, $(\leq n R)$ eq. to $(\geq n R.\top)$, $(\leq n R.\top)$
- individual: $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

Examples

Atomic concepts: person, lawyer, doctor, male

Atomic roles: child, son, daughter, friend, colleague

person \sqcap (\exists child) \sqcap (\forall son.lawyer) \sqcap (\forall daughter.doctor)

person \sqcap (\exists child.male) \sqcap (≤ 2 child.(lawyer \sqcup doctor))

person \sqcap (≥ 5 friend) \sqcap (\forall colleague.male)

Role constructors

- **atomic roles:** $P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
- **atomic transitive roles:** $H^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
- **conjunction:** $(Q \sqcap R)^{\mathcal{I}} = Q^{\mathcal{I}} \cap R^{\mathcal{I}}$
- **disjunction:** $(Q \sqcup R)^{\mathcal{I}} = Q^{\mathcal{I}} \cup R^{\mathcal{I}}$
- **difference:** $(Q \setminus R)^{\mathcal{I}} = Q^{\mathcal{I}} \setminus R^{\mathcal{I}}$
- **inverse:** $(R^{-1})^{\mathcal{I}} = \{(a, b) \mid (b, a) \in R^{\mathcal{I}}\}$
- **chaining:** $(R \circ Q)^{\mathcal{I}} = \{(a, b) \mid \exists c. (a, c) \in R^{\mathcal{I}}, (c, b) \in Q^{\mathcal{I}}\}$
- **self:** $id(C)^{\mathcal{I}} = \{(a, a) \mid a \in C^{\mathcal{I}}\}$
- **reflexive-transitive closure:** $(R^*)^{\mathcal{I}} = (R^{\mathcal{I}})^*$

Examples

Atomic concepts: person, doctor, lawyer, male

Atomic roles: child, son, daughter, friend, colleague

$$\text{person} \sqcap (\exists(\text{colleague} \setminus \text{friend})) \sqcap (\forall \text{colleague}.\text{male})$$
$$(\geq 2 (\text{son} \sqcup \text{daughter})) \sqcap (\forall \text{son}.\text{lawyer}) \sqcap (\forall \text{daughter}.\text{doctor})$$
$$(\exists(\text{son} \sqcup \text{daughter})^*.\text{doctor}) \sqcap \forall((\text{son} \sqcup \text{daughter}) \circ \text{son}).(\text{lawyer} \sqcup \text{doctor})$$

TBox e ABox

An **L-Tbox T** is a set of statements (**inclusion assertions**) of the form:

$$\boxed{C \sqsubseteq D} \quad \boxed{R \sqsubseteq Q}$$

An **L-ABox Σ** is a set of statements (**membership assertions**) of the forms (a, b are individuals, and we have $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$):

$$\boxed{C(a)} \quad \boxed{R(a, b)}$$

- $C \sqsubseteq D$ is satisfied by \mathcal{I} if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $R \sqsubseteq Q$ is satisfied by \mathcal{I} if $R^{\mathcal{I}} \subseteq Q^{\mathcal{I}}$
- $C(a)$ is satisfied by \mathcal{I} if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $R(a, b)$ is satisfied by \mathcal{I} if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Knowledge base (Ontology)

An \mathcal{L} -knowledge base is a pair $\langle T, \Sigma \rangle$, where T is an \mathcal{L} -Tbox, and Σ is an \mathcal{L} -ABox.

An interpretation \mathcal{I} is a **model** of $K = \langle T, \Sigma \rangle$ if it satisfies all assertions of T and all assertions of Σ . K is said to be **satisfiable** if it admits a model.

K **logically implies** an assertion α (written $K \models \alpha$) if α is satisfied by every model of K . C is **subsumed** by D in K , if $K \models C \sqsubseteq D$.

open world assumption

Example

Note: $\{ C \sqsubseteq D, D \sqsubseteq C \}$ is written simply as $C = D$

TBox T :

$$\exists(\text{child}^-)^*.\exists\text{live}.\text{SouthOfPo} \sqsubseteq \neg \text{RealPadano}$$
$$\begin{aligned} \text{RealPadano} = & \text{ Italian} \sqcap (\forall \text{child}^*.\text{RealPadano}) \sqcap \\ & (\forall \text{friend}^*.\text{RealPadano}) \end{aligned}$$

ABox Σ :

RealPadano(Umberto),

child(Umberto, Aldo),

\neg RealPadano(Gianfranco)

OWL Ontology Web Language

OWL concept constructors:

Constructor	DL Syntax	Example	Modal Syntax
intersectionOf	$C_1 \sqcap \dots \sqcap C_n$	Human \sqcap Male	$C_1 \wedge \dots \wedge C_n$
unionOf	$C_1 \sqcup \dots \sqcup C_n$	Doctor \sqcup Lawyer	$C_1 \vee \dots \vee C_n$
complementOf	$\neg C$	\neg Male	$\neg C$
oneOf	$\{x_1\} \sqcup \dots \sqcup \{x_n\}$	{john} \sqcup {mary}	$x_1 \vee \dots \vee x_n$
allValuesFrom	$\forall P.C$	\forall hasChild.Doctor	$[P]C$
someValuesFrom	$\exists P.C$	\exists hasChild.Lawyer	$\langle P \rangle C$
maxCardinality	$\leq n P$	≤ 1 hasChild	$[P]_{n+1}$
minCardinality	$\geq n P$	≥ 2 hasChild	$\langle P \rangle_n$

OWL Ontology Web Language

Types of axioms:

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human \sqsubseteq Animal \sqcap Biped
equivalentClass	$C_1 \equiv C_2$	Man \equiv Human \sqcap Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$ Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} \equiv {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg \{x_2\}$	{john} $\sqsubseteq \neg$ {peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter \sqsubseteq hasChild
equivalentProperty	$P_1 \equiv P_2$	cost \equiv price
inverseOf	$P_1 \equiv P_2^-$	hasChild \equiv hasParent $^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor $^+$ \sqsubseteq ancestor
functionalProperty	$T \sqsubseteq \leqslant 1P$	T $\sqsubseteq \leqslant 1$ hasMother
inverseFunctionalProperty	$T \sqsubseteq \leqslant 1P^-$	T $\sqsubseteq \leqslant 1$ hasSSN $^-$



- 1. Introduction to ontologies**
- 2. Ontology languages**
- 3. Reasoning**
- 4. Conclusion**

Reasoning over ontologies

- Given an ontology, it is possible that *additional* properties can be inferred, by
 - Meta-querying
 - Logical reasoning
- Different goals of reasoning
 - Verification
 - Validation
 - Analysis
 - Synthesis

Logical reasoning

- Based on logic
- No logical reasoning without formal semantics: **soundness and completeness**
- Great interest in **automated** logical reasoning
- Feasibility/complexity of automated reasoning

Types of logical reasoning

- Based on semantic property
 - Classical
 - Non-classical (e.g., non-monotonic reasoning, common-sense reasoning, etc.)
- Based on the type of desired conclusions
 - Deduction
 - Induction
 - Abduction

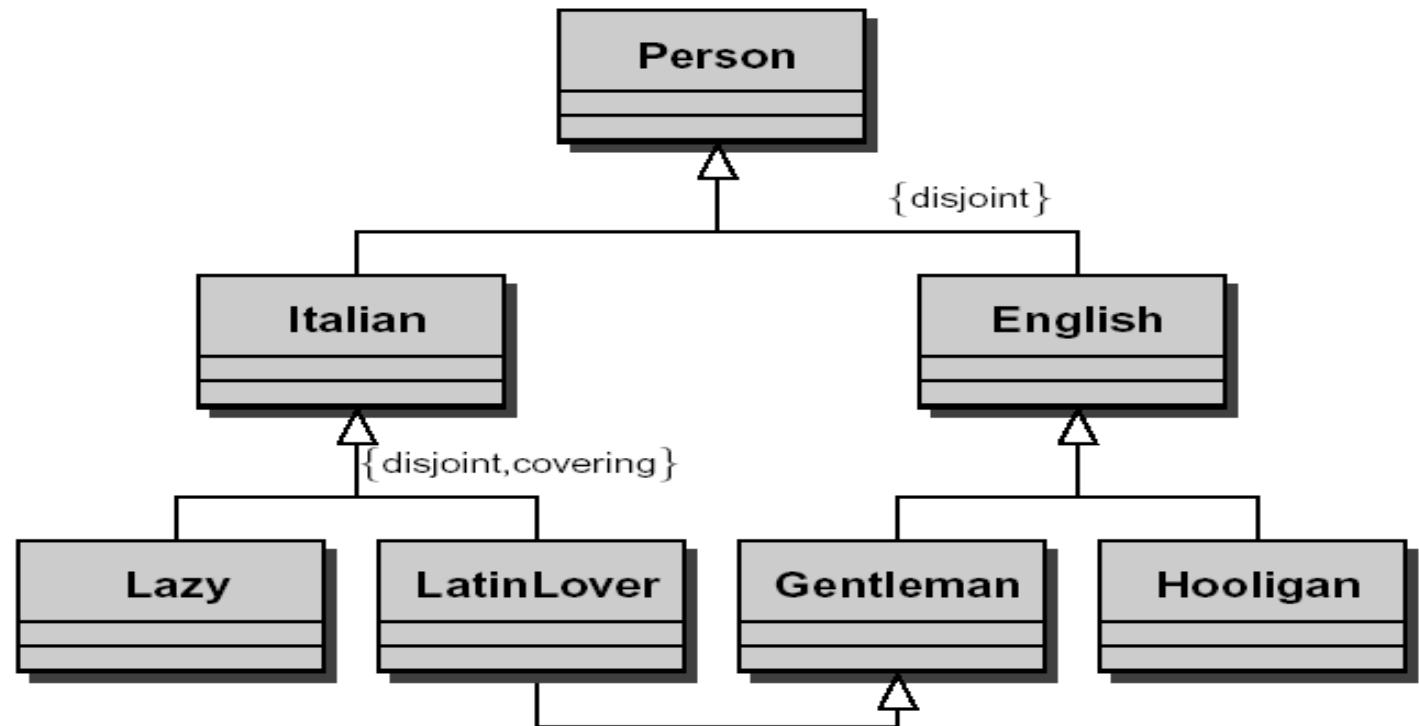
Classical reasoning: deduction

Let Ω and σ be the intensional level and the extensional level of an ontology, respectively.

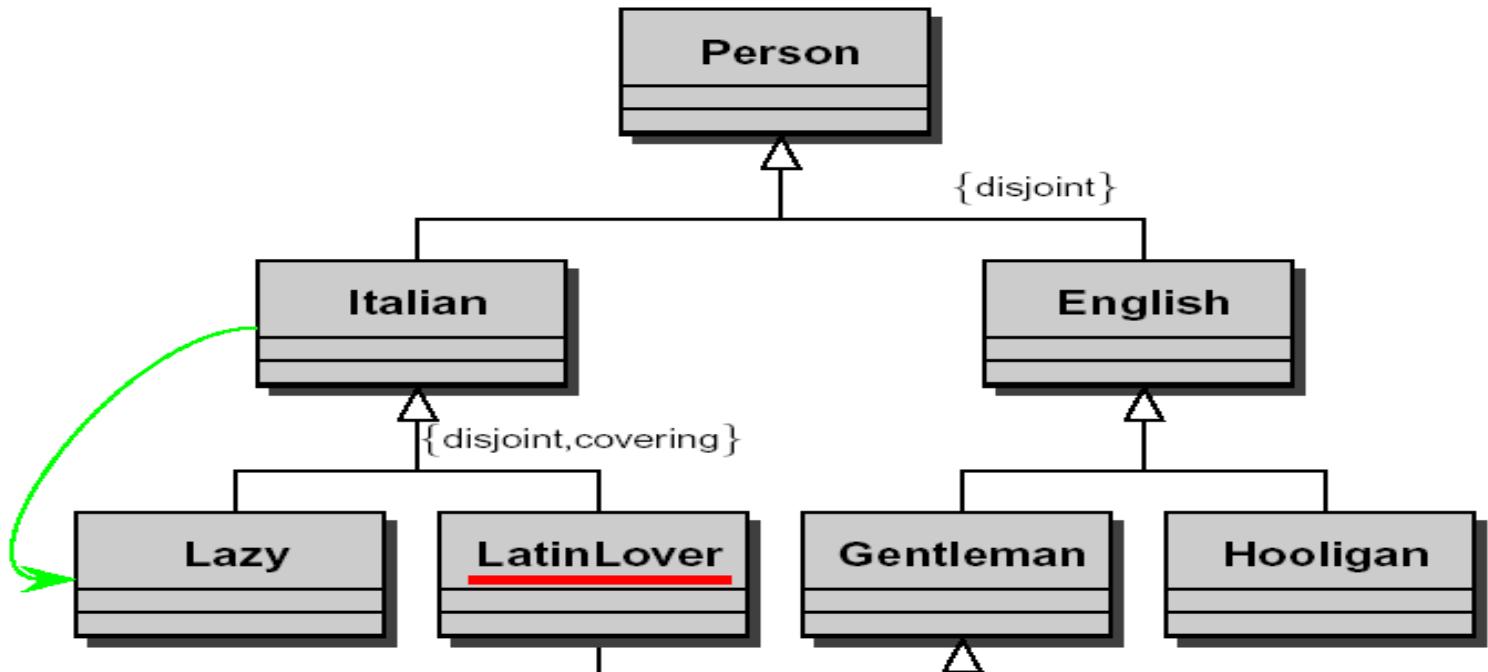
Deduction

P is a deductive conclusion from Ω ($\Omega, \sigma \models P$) if P holds in every situation coherent with Ω and σ , i.e., if P is true in every (logical) model of Ω and σ

Example of deduction



Example of deduction



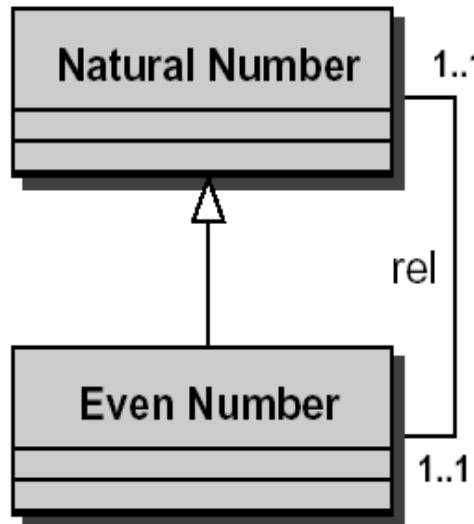
implies

$$\text{LatinLover} = \emptyset$$

$$\text{Italian} \subseteq \text{Lazy}$$

$$\text{Italian} \equiv \text{Lazy}$$

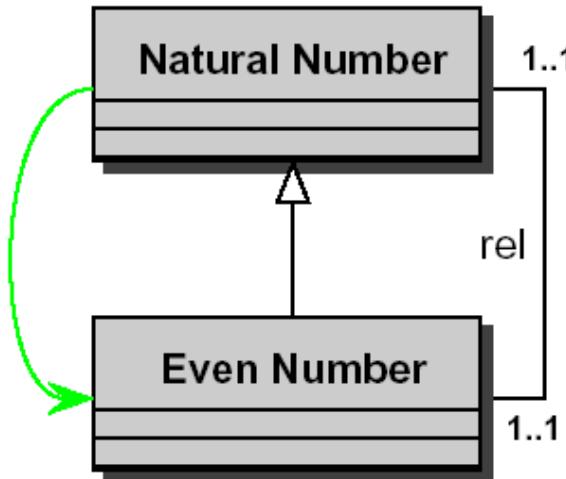
Example of deduction



implies

“the classes ‘Natural Number’ and ‘Even Number’ contain the same number of instances”.

Example of logical reasoning



implies

“the classes '*Natural Number*' and '*Even Number*' contain the same number of instances”.

If the domain is finite: $\text{Natural Number} \equiv \text{Even Number}$

Logical reasoning: induction

Let T and A be the intensional level and the extensional level of an ontology, respectively.

Let α be a set of **observations** at the extensional level.

Induction

P is an inductive conclusion wrt T, A and α if P is an **intensional level** property such that

- $T, A \not\models \alpha$ (T, A do not already imply α)
- $T, \{A, \alpha\} \not\models \neg P$ (P is consistent with $T, \{A, \alpha\}$)
- $\{T, P\}, A \models \alpha$ ($\{T, P\}, A$ imply α)

Logical reasoning: abduction

Let T and A be the intensional level and the extensional level of an ontology, respectively.

Let α be a set of **observations** (facts at the extensional level).

Abduction

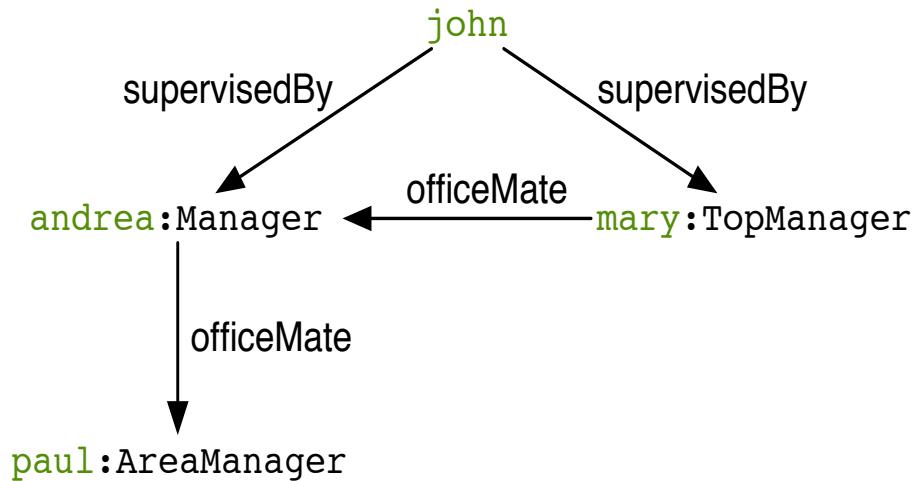
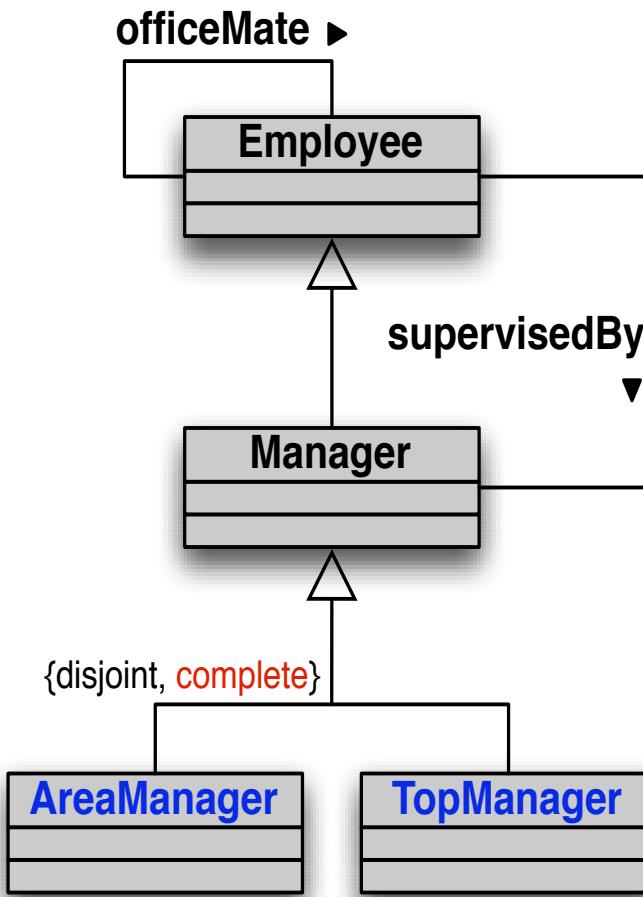
E is an abductive conclusion wrt T , A and α if E is an **extensional level** property such that

- $T, A \not\models \alpha$ (T, A do not already imply α)
- $T, \{A, \alpha\} \not\models \neg E$ (E is consistent with $T, \{A, \alpha\}$)
- $T, \{A, E\} \models \alpha$ ($T, \{A, E\}$ imply α)

Query answering

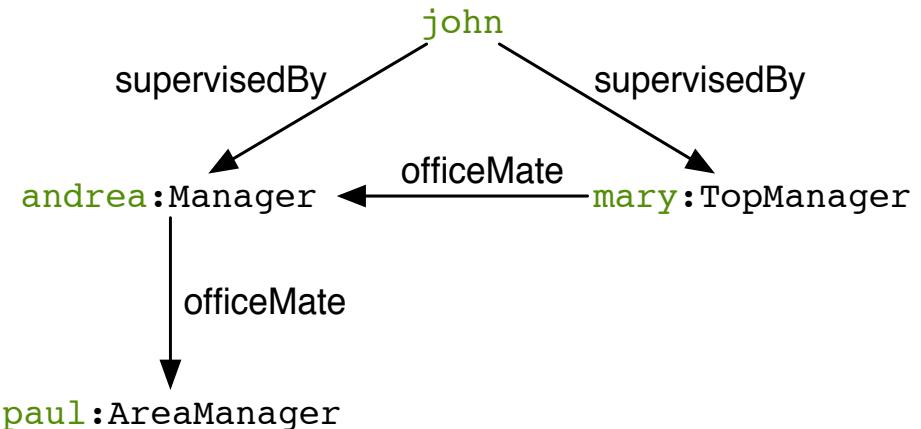
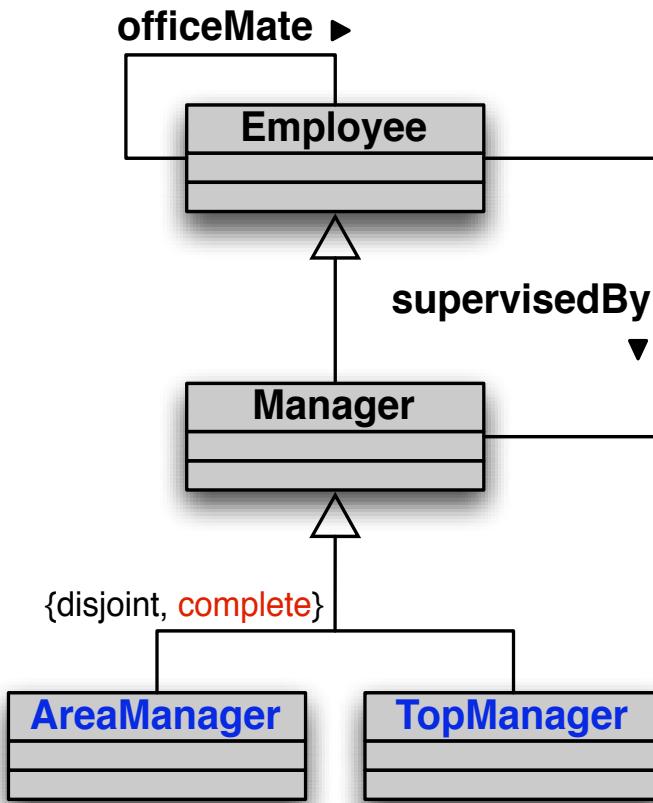
- Query answering is a kind of deductive reasoning of special importance
- In general, query answering over ontologies is very different from and much more complex than query answering in databases, because an ontologies can be seen as an **abstraction for a set of models (i.e., databases)**

Example of query answering


$$q(x) \leftarrow \exists y, z. \text{supervisedBy}(x, y), \text{TopManager}(y), \text{officeMate}(y, z), \text{AreaManager}(z)$$

Answer: ???

Example of query answering


$$q(x) \leftarrow \exists y, z. \text{supervisedBy}(x, y), \text{TopManager}(y), \text{officeMate}(y, z), \text{AreaManager}(z)$$

Answer: { john }

To determine this answer, we need to resort to **reasoning by cases**.

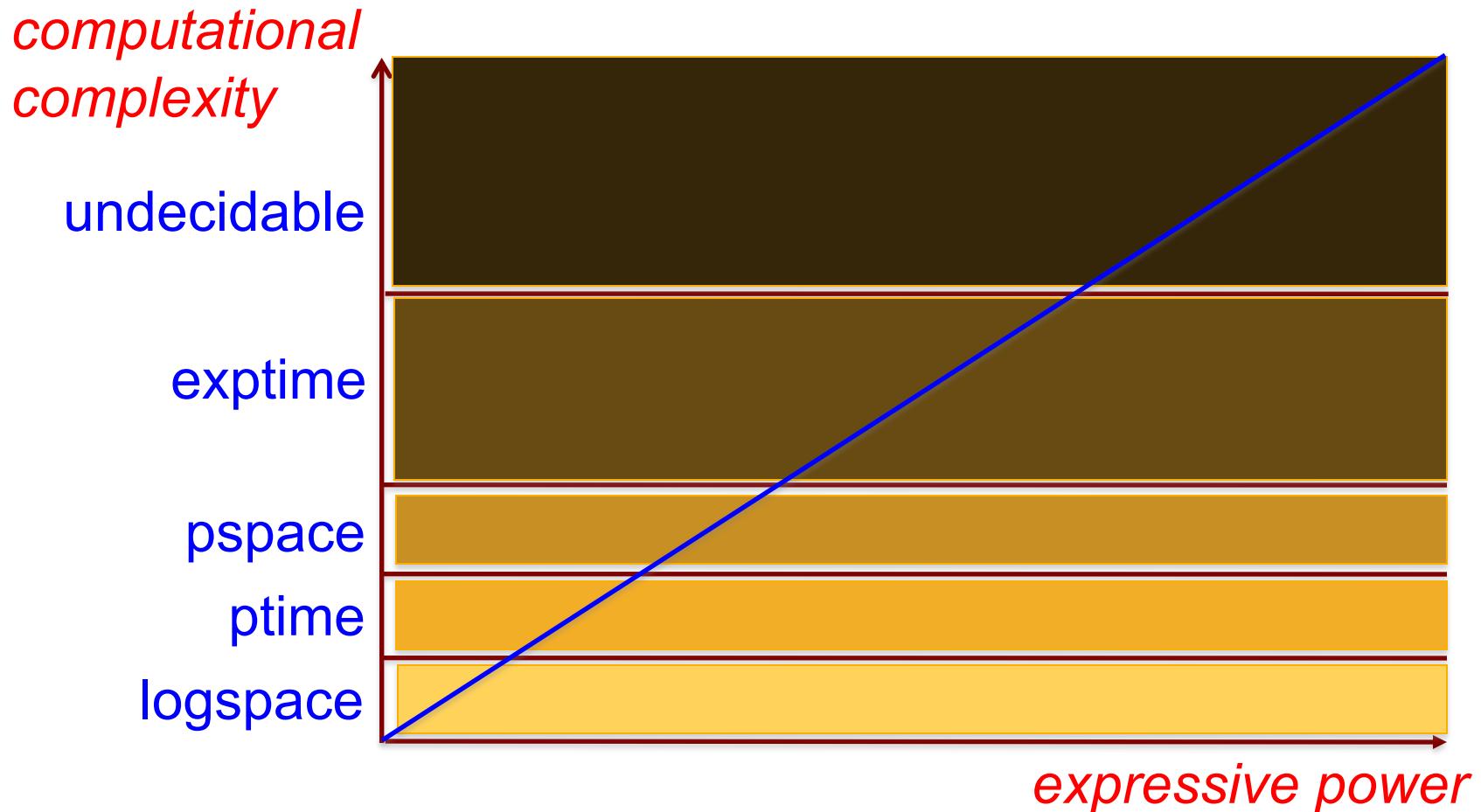
Example of complexity analysis

\mathcal{AL} +	P	CoNP	NP	PSPACE								
$C \sqcup D$		\times	\times				\times	\times		\times	\times	\times
$(\geq nR)$		\times		\times					\times	\times		
$(\leq nR)$											\times	\times
$\exists R.C$				\times		\times	\times		\times	\times		\times
$R \sqcap R'$					\times	\times		\times	\times		\times	\times

polynomial time

exponential time

A fundamental trade-off





1. Introduction to ontologies
2. Ontology languages
3. Reasoning
4. Conclusion



(Some) Use of ontologies

- To make domain assumptions explicit and to share common understanding of a domain (bioinformatics, medicine, finance, ...) among people or software agents
- To enable interoperability of different systems and data exchange
- To enable reuse of domain knowledge (Natural Language processing, Robotics, ...)
- To separate domain knowledge from the operational knowledge
- To analyze domain knowledge
- Ontology-based information retrieval
- **Ontology-based data management (See later)**



Conclusion

- The notion of computational ontology is gaining attention in several fields
- Automated reasoning is one crucial aspects of computational ontologies
- We will investigate one particular aspect of computational ontologies in what follows