Università di Parma

Dipartimento di Ingegneria e Architettura

Intelligenza Artificiale A.A. 2023/2024

Representation of Knowledge and Reasoning

The description logics

Description Logic

- The syntax of first-order logic is designed to make it easy to say things about objects.
- Description logics are notations designed to simplify the description of definitions and properties of categories.
- Description logic systems have evolved from semantic networks in response to the pressures to formalize the meaning of the networks maintaining the accent on the taxonomic structure as an organizational principle

Description Logic

- So if we organize the knowledge into categories (and sub-categories) it is sufficient to classify an object, through the perceived properties, to infer the properties of the category to which it belongs

- **Inheritance as a form of inference**

Description Logic

Taxonomy

- It captures the basic principle of organization of both semantic networks, frames systems and object systems

- It is used to store information at the most appropriate level of generality making it automatically available to more specific objects through the inheritance mechanism

Description Logic

- ***DL as formalization of semantic networks***

- Towards the 80s there was a turn towards the logic of semantic networks

- The process consists of reformulate constructs according to the canons of logic
  - And eliminate constructs that do not lend themselves to such reformulation (*default* and exceptions)

Description Logic

- In FOL one can express very articulate knowledge and, at least in principle, perform complex reasoning automatically.

- But there are two problems.
  - the deduction procedure (***test procedure or calculation***) is not a decision procedure, but only ***a semi-decision procedure***. Then:
    - If the conclusion is deducible of the premises, the procedure ends in a finite number of steps producing a proof;
    - if the conclusion is not deducible of the premises, the procedure can not terminate (an information system can "go on cycle")
  - The second problem is that the procedure, even in cases where it ends, is often very expensive in terms of computing resources.

Description Logic

- In description logic we have sought a language that is less expressive but able to:
  - however, be quite expressive for the applications;
  - the deduction is based on a decision procedure (which then ends in any case after a finite number of steps, both when the conclusion is deducible from the premises, and when it is not);
  - the deduction procedure has acceptable computational complexity (ie, it requires an acceptable amount of computing resources).

Description Logic

- Currently, the Description Logics are used in such sectors
  - software engineering,
  - medical diagnostics,
  - digital libraries, databases and information systems based on theweb
- Also the W3C standard given by the OWL language is nothing but a variant of the descriptive logics.

Description Logic

- A description language is simpler than a first-order language because it contains only
  - atomic concepts
    - (ex: a common name like 'father', 'wife'…)
  - roles
    - a binary relationship and possibly operators
  - object names
    - a single object
- In first-order languages they can be translated as
  - atomic concepts with unary predicative letters,
  - roles with binary predicative letters
  - the names of objects with individual constants.

Description Logic

- Compared to the language of the FOL, the DL do not have
  - variables
  - quantifiers
  - the logic connective
  - the predicate n-ary symbols, for n >= 3
  - the predicate of equality.
- … but they have other operators that can be seen as "fixed combinations" (*macro*) of them.

Descriptive Logic

- **Terms**: expressions that describe concepts
  - atomic: woman
    - They are called concepts or classes (they denote sets of objects of reality)
  - Complex: person $\cap$ female
  - Terminological equivalence: woman$\equiv$ person $\cap$ female
    - The case in which one of the two terms is complex and the other atomic is called *terminological definition*
- **Terminology or ontology** a finite set of terminological definitions.

Description Logic

- **Ontology**: assigns an unambiguous meaning to atomic terms *based on the meaning (in function) of other terms*, which in general will have a meaning thanks to further terminological definitions, and so on.

- Every ontology is finished: sooner or later we arrive at terms without a definition, which must be considered as *primitive*.
- **Primitive terms** : for ontology to have some practical utility these must be anchored to, and defined in, some domain (***problem of symbol grounding)***.

Description Logic

- **Semantics of terms**
- For the expressions of a DL it is possible to specify a *formal semantics*, which associates to each term an interpretation defined in a set-phrase.

Description Logic

- Terms
  - Given a certain predefined portion of reality (which we will call *domain*), each term determines an *extension*, defined as the set of all the individuals in the domain to which the term applies.
    - Ex: the domain includes all the individuals present in a classroom during a lesson, the extension in that domain of the word WOMAN is made up of all the women present in the classroom

- Predefined Terms
  - T Universal concept

(totality of existing individuals)

  - ⊥ Empty concept (empty set of individuals)

The description logic allows to directly apply logical operations to the predicates

Description Logic

- *Operators*

- Intersection '$\cap$'
- Equivalence '$\equiv$' WOMAN $\equiv$ PERSON $\cap$ FEMALE
- Complement '$\neg$' *MAN* $\equiv$ *PERSON* $\cap$ $\neg$*FEMALE,*
- Union '$\cup$',

NATURAL$\equiv$ MINERAL $\cup$ (VEGETABLE $\cup$ ANIMAL)

Description Logic

- ***Role-based terms***
  - In addition to the terms corresponding to predicates with an argument (say concepts or classes), the DLs use terms corresponding to predicates with two arguments, which express binary relations between individuals of reality; these terms are roles, properties, attributes or relationships.
  - Ex: terminological definition that uses a role

    PARENT ≡ ∃Son, (in FOL -> ∃y Figlio(x,y))
  - that intuitively means "parent is who has a child".
  - The expression ∃Figlio is a complex term, formed by the *existential quantifier* ' '∃' and the *role* of Son.

Description Logic

Role-based terms

- Expressions such as ∃R can be combined as concepts:
- MOTHER ≡ WOMAN ∩ ∃Son
- MOTHER-F ≡ MOTHER ∩ ∀Son.FEMALE
  – Mother with only female daughters
- PARENT-A-M ≡ ∃Son.¬FEMALE
  – Parent who has at least one male son

Description Logic

- *Inversion of roles*

- For example, suppose we want to consider as an "Italian by adoption" every Italian citizen who has both foreign parents.

- Starting from the concept of "Italian citizen" and the role of "son" we can give the following definition:

$$\text{IT-ADOPTION} \equiv \text{CITIZEN-IT} \cap \forall \text{Son}^{-}.\neg\text{CITIZEN-IT}$$

Description Logic

- *Domain and codomain of a role*

- The sets of individuals that can appear as values of 'x' (domain) and as values of 'y' (codomain) in the expression R (x, y).
- Ex: we want to affirm that the role "age" always expresses a binary relationship between people and natural numbers.
- The following terminological equivalence says that the codomain of the "age" role is the class of natural numbers:

$$T \equiv \forall \text{Age.NATURAL.}$$

Description Logic

- ***Functional roles (role that express a function)***
- a *function* is a binary relationship with the following properties:
  - each element of the domain is related to at least one element of the codomain (the relation is total on the domain);
  - every element of the domain is in relation to at most one element of the codomain (the relation is unique in the codomain).
- From which we have that if a binary relation is functional every element of the domain (said *argument* of the function) is in relation with *exactly* one element of the codomain (said *value* of the function corresponding to the argument).
- Ex: the role "age" is functional, because each person has exactly one age

$$\text{PERSON} \equiv =1\text{Age},$$

Description Logic

- ***Transitive roles***
- If a property P is transitive and that property relates the individual **a** to with the individual **b** and also the individual **b** to **c** then it can be inferred that also **a** is related to **c** by the property P

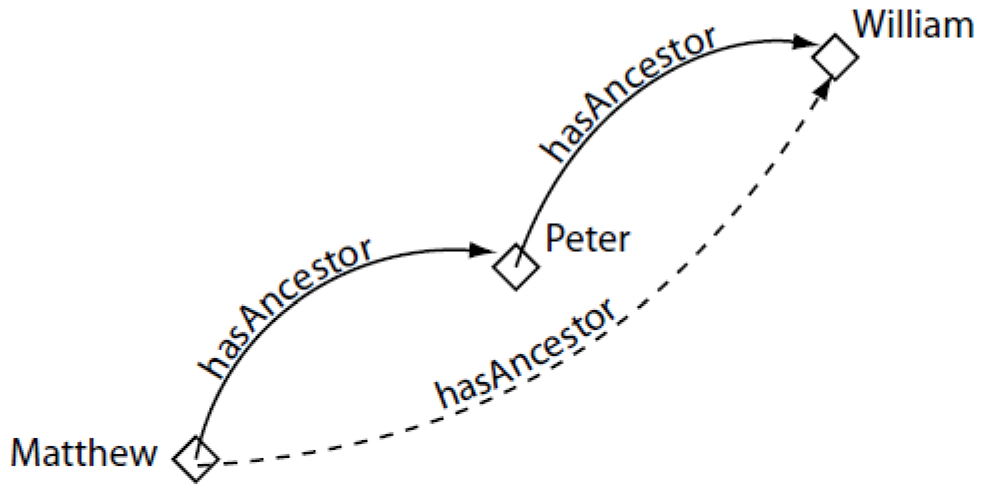


Figure 4.20: An Example Of A Transitive Property: hasAncestor

Description Logic

Main inferential methods

- *Subsumption*
  - It consists in checking if one category is a subset of another by comparing their definitions
- *Classification*
  - It consists in checking whether an object belongs to a category
- *Consistency of a category definitionn*
  - A definition is consistent whether the membership criteria are logically satisfiable
    (ie the category is not an empty set)

Description Logic

- ***Subsumption***

- Suppose you want to specify that cats are pets.
- Equivalence can not be used, because naturally there are domestic animals that are not cats.
- The relationship between the two terms is then expressed in the following way:

$$\text{CAT} \subseteq \text{PET}.$$

- The operator '$\subseteq$' it is called an operator of *subsumption*;
  - The term CAT is *subsumed* by the (is a *hyponym* of the, is a *specialization* of the) term PET;
  - The term PET *subsumes* the (is a *hyperonym* of the, is a *generalization* of the) term CAT.

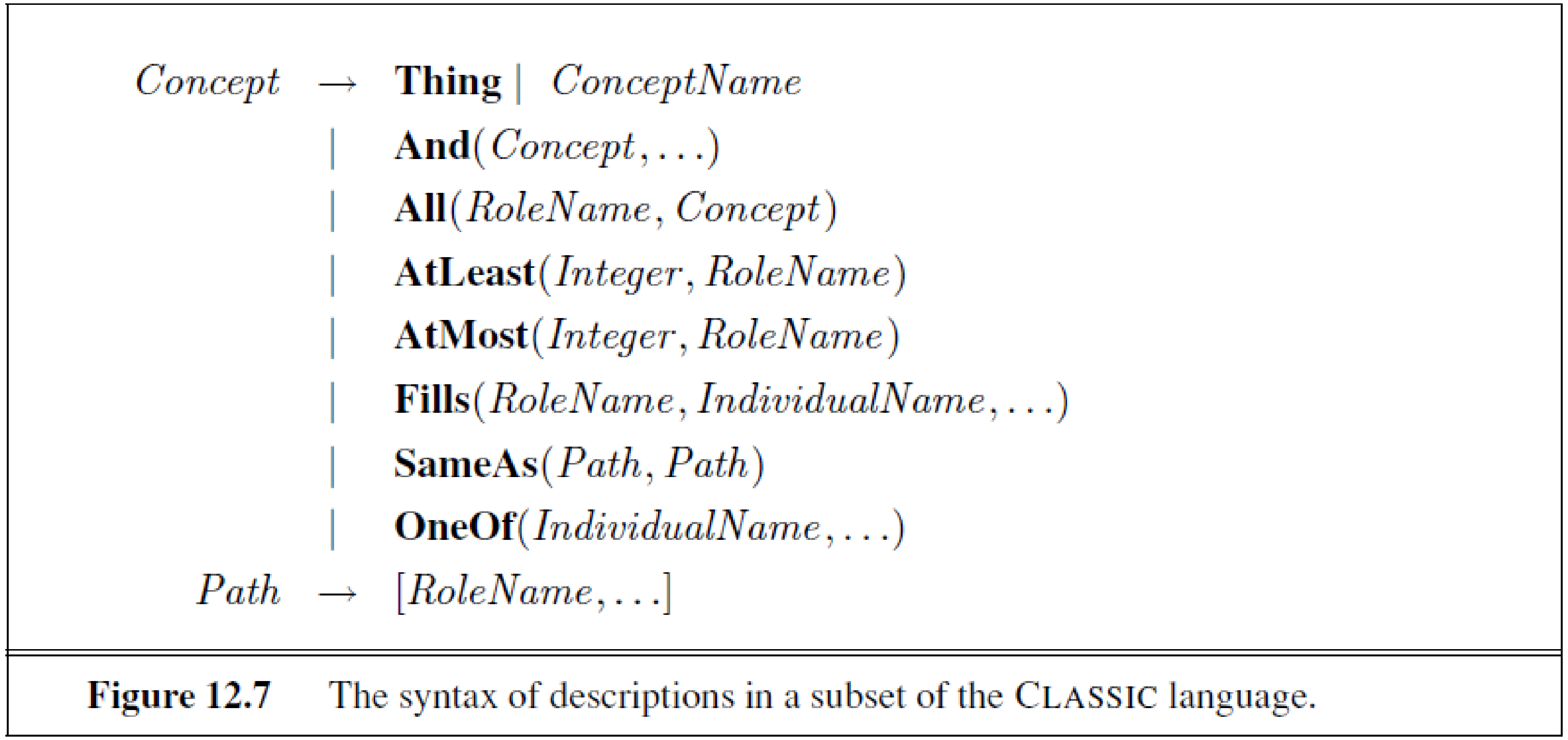
Description Logic

❑ *The factual knowledge in DL*

- ❑ In the DLs two types of factual knowledge can be expressed. Assertions of the type are in fact permitted
  - ***C(a),***
    - where C is an arbitrary term and *a* is a nominal (individual names)
  - ***R(a,b),***
    - where R is a role and *a*, *b* are nominal (not necessarily distinct).
- ❑ Examples:
  - MOTHER(anna)
  - WOMAN ∩ ∃Son(anna)(wherer *C* = WOMAN ∩ ∃Son)
  - Son(anna,bruno)

Description Logic

- The CLASSIC language (Borgida et al., 1989) is a typical description logic



$$
\begin{array}{rcl}
\mathit{Concept} & \rightarrow & \textbf{Thing} \mid \ \mathit{ConceptName} \\
& \mid & \textbf{And}(\mathit{Concept}, \ldots) \\
& \mid & \textbf{All}(\mathit{RoleName}, \mathit{Concept}) \\
& \mid & \textbf{AtLeast}(\mathit{Integer}, \mathit{RoleName}) \\
& \mid & \textbf{AtMost}(\mathit{Integer}, \mathit{RoleName}) \\
& \mid & \textbf{Fills}(\mathit{RoleName}, \mathit{IndividualName}, \ldots) \\
& \mid & \textbf{SameAs}(\mathit{Path}, \mathit{Path}) \\
& \mid & \textbf{OneOf}(\mathit{IndividualName}, \ldots) \\
\mathit{Path} & \rightarrow & [\mathit{RoleName}, \ldots]
\end{array}
$$

Figure 12.7 The syntax of descriptions in a subset of the CLASSIC language.

Description Logic

Examples

Bachelors are unmarried adult malesi

– *Bachelor=And(Unmarried, Adult, Male)*

➢ In predicate logic

– *Bachelor(x)* ⬄ *Unmarried(x) & Adult(x) & Male(x)*

➢ The descriptive logic allows to directly apply logical operations to the predicates

The set of men with at least three sons who are all unemployed and married to doctors, and at most two daughters who are all professors in physics or math departments, we would use

– *And(Man,AtLeast(3,Children),AtMost(2, Daughters), All(Children,And(Unemployed,Married,All(Spouse,Doctor))), All(Daughter,And(Professor,Fills(Department, Physics,Mathematics))))*

Description Logic

Aggregates

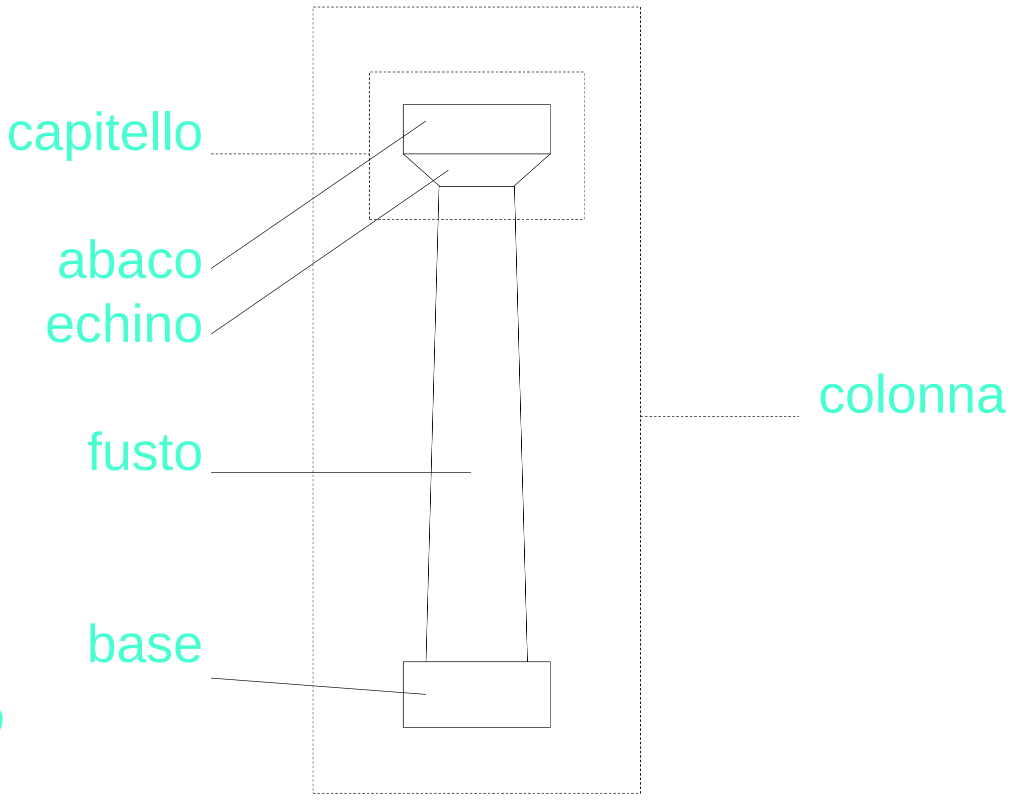
- In a DL there is generally no operator corresponding to the *aggregate* relationship (the concept of "being a part of”).
- The main reason for this lack is that it is not easy to assign a rigorous and univocal semantics to such an operator.
- *Roles* are used to talk about the parts of an object

Description Logic

Aggregates

- EX:

  a doric column (colonna dorica), is composed of a *base*, a *fusto* and a *capitello*, which in turn is composed of a *abaco* and a *echino*



Description Logic

Aggregates

- General parts

- Given a term PHYSICAL-OBJECT, we can define a transitive role 'Part' that relates a physical object with a *generic part* of it:
  - T ≡ ∀Parte. PHYSICAL-OBJECT,
  - *Tr*(Parte).

- From which we can define the structure of the Doric columns:
  - COLONNA ≡ ∃PartOf.BASE ∩ ∃Part.FUSTO ∩ ∃Part.CAPITELLO,
  - CAPITELLO ≡ ∃Part.ABACO ∩ ∃Part.ECHINO.,
  - *Disj*(COLONNA,BASE,FUSTO,CAPITELLO,ABACO,ECHINO).

Description Logic

Aggregates

- General parts

- From these definitions we can deduce, for example, that:

  - COLONNA $\subseteq$ PHYSICAL-OBJECT,
  - BASE $\subseteq$ PHYSICAL-OBJECT,
  - FUSTO $\subseteq$ PHYSICAL-OBJECT
  - CAPITELLO $\subseteq$ OGGETTO-FISICO,
  - ABACO $\subseteq$ PHYSICAL-OBJECT,
  - ECHINO $\subseteq$ PHYSICAL-OBJECT,
  - COLONNA $\subseteq$ $\exists$Part.ABACO,
  - COLONNA $\subseteq$ $\exists$Part.ECHINO.

Description Logic: *discussion*

Emphasis on the tractability of inference

- *A problem instance is solved by describing it and then asking the system if it is subsumed among many possible categories of solutions*

- In systems based on FOL, give a prediction about the required time is almost always impossible

- In DL we try to ensure that the subsumption task is carried out in a time that increases with a polynomial of the dimension of the descriptions

Description Logic: *discussion*

Consequences

- Difficult problems can not be expressed
- Or their descriptions will have to be exponentially large
- However, the results help the user to recognize the constructs that cause problems
- Description logics are usually lack negation and disjunction
  - With disjunctive descriptions, nested definitions can lead easily to an exponential number of alternative routes by which one category can subsume another

Description Logic: *discussion*

- *The description logic allows to directly apply logical operations to the predicates*
- The idea is: given an atomic term A, the semantics of the term must enable us to identify the extension of A in the domain; this extension is represented by all the individuals in the domain that make the formula A (x) true when taken as x values.

Description Logic: examples

$$
\begin{aligned}
\text{Woman} &\equiv \text{Person} \sqcap \text{Female} \\
\text{Man} &\equiv \text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female}) \\
\text{Mother} &\equiv (\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person} \\
\text{Father} &\equiv (\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \sqcap \exists \text{hasChild}.\text{Person} \\
\text{Parent} &\equiv ((\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \sqcap \exists \text{hasChild}.\text{Person}) \\
&\quad \sqcup ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
\text{Grandmother} &\equiv ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
&\quad \sqcap \exists \text{hasChild}.(((\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \\
&\qquad\qquad\qquad \sqcap \exists \text{hasChild}.\text{Person}) \\
&\qquad\qquad\quad \sqcup ((\text{Person} \sqcap \text{Female}) \\
&\qquad\qquad\qquad\quad \sqcap \exists \text{hasChild}.\text{Person})) \\
\text{MotherWithManyChildren} &\equiv ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \sqcap \geqslant 3\, \text{hasChild} \\
\text{MotherWithoutDaughter} &\equiv ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
&\quad \sqcap \forall \text{hasChild}.(\neg(\text{Person} \sqcap \text{Female})) \\
\text{Wife} &\equiv (\text{Person} \sqcap \text{Female}) \\
&\quad \sqcap \exists \text{hasHusband}.(\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female}))
\end{aligned}
$$