Description Logic

Franz Baader

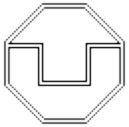
Literature:

- Franz Baader, Carsten Lutz, Ian Horrocks, and Uli Sattler: An Introduction to Description Logic. Cambridge University Press, 2017.

- F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (ed.): The Description Logic Handbook. Cambridge University Press, 2003.

- F. Baader, C. Lutz: Description Logics. In The Handbook of Modal Logic, Elsevier, 2006.

- F. Baader: Description Logics. In Reasoning Web: Semantic Technologies for Information Systems, 5th International Summer School 2009, Springer LNCS 5689, 2009.



Chapter 1: Introduction

Description Logic

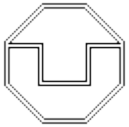
subfield of knowledge representation,

which is a subfield of Artificial Intelligence

Description comes from concept description, i.e., a formal expression that determines a set of individuals with common properties

Logics comes from the fact that the semantics of concept descriptions can be defined using logic;

in particular, most Description Logics can be seen as fragments of first-order logic.



Description Logic

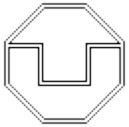
subfield of knowledge representation,
which is a subfield of Artificial Intelligence

Description Logic: name of a research field

Description Logics: a family of knowledge representation languages

Description Logic: a member of this family

DL(s)



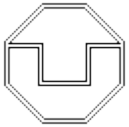
Knowledge Representation

general goal

"develop formalisms for providing high-level descriptions of the world
that can be effectively used to build intelligent applications"

[Brachman & Nardi, 2003]

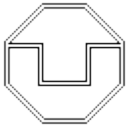
- formalism: well-defined syntax and formal, unambiguous semantics
- high-level description: only relevant aspects represented, others left out
- intelligent applications: must be able to reason about the knowledge, and infer implicit knowledge from the explicitly represented knowledge
- effectively used: need for practical reasoning tools and efficient implementations



Syntax

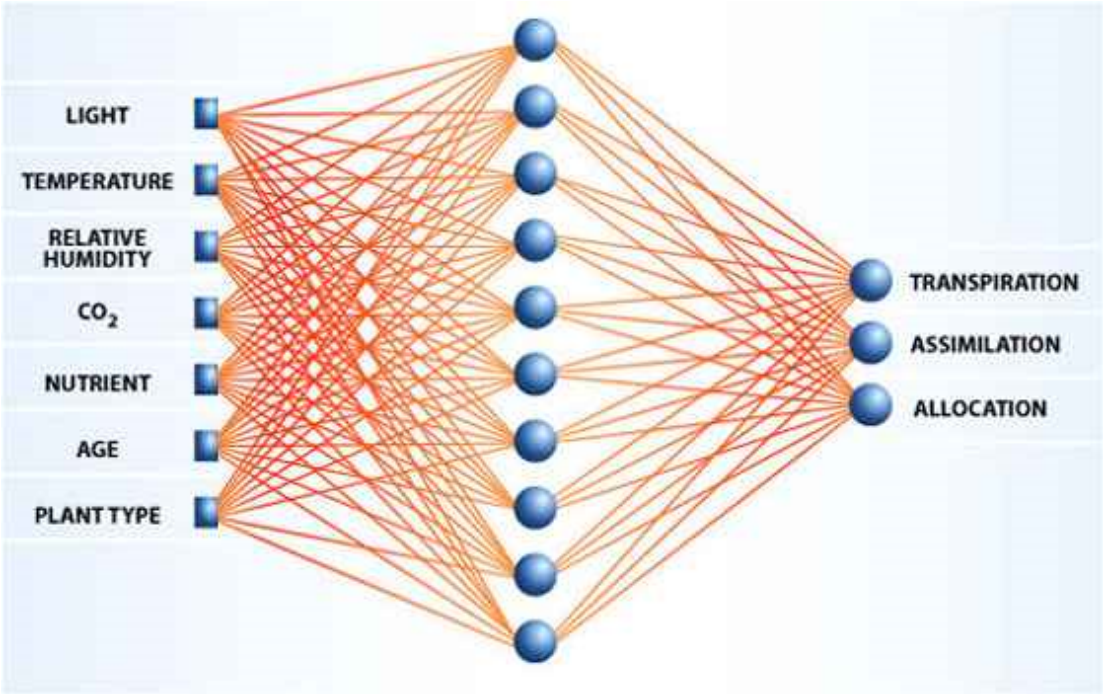
- provides an explicit symbolic representation of the knowledge
- not just implicit, as e.g. in neural networks

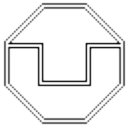
$\mathtt{Woman} \equiv \mathtt{Person} \sqcap \mathtt{Female}$		
$\mathtt{Man} \equiv \mathtt{Person} \sqcap \neg\mathtt{Female}$		
$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists \mathtt{hasChild}.\top$		
$\mathtt{Person} \equiv \mathtt{Man} \sqcup \mathtt{Woman}$	Male(JOHN)	
$\bot \equiv \mathtt{Male} \sqcap \mathtt{Female}$	Male(MARC)	
	Male(STEPHEN)	
hasChild(STEPHEN,MARC)	Male(JASON)	
hasChild(MARC,ANNA)	Female(MICHELLE)	
hasChild(JOHN,MARIA)	Female(ANNA)	
hasChild(ANNA,JASON)	Female(MARIA)	



Syntax

- provides an explicit symbolic representation of the knowledge
- not just implicit, as e.g. in neural networks

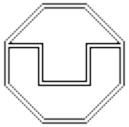




Semantics

describes the connection between the symbolic representation and the real-world entities it is supposed to represent

- no procedural semantics, i.e., should not just be defined by how certain programs using the symbolic representation behave
- instead declarative semantics:
  - mapping of the symbolic expressions to an abstraction of the "world" (interpretation)
  - notion of "truth" that allows us to determine whether a symbolic expression is true in the world under consideration (model)

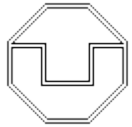


Syntax & Semantics

determine the expressive power of a formalism

Adequate expressive power:

- not too low: can all the knowledge relevant for solving the problem at hand be represented?
- not too high: are the available representational means really necessary in this application?



Reasoning

deduce implicit knowledge from the
explicitly represented knowledge

$$\forall x.\forall y.\,(male(y) \wedge \exists z.(has\_child(x,z) \wedge has\_child(z,y)) \rightarrow has\_grandson(x,y)$$

$has\_child(John, Mary)$

$has\_child(Mary, Paul)$

$male(Paul)$

$grandson\_of\,(John, Paul)$

implicit knowledge

Knowledge representation systems

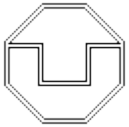
should provide their users with inference tools

that can deduce (certain) implicit consequences

results should depend only on the semantics of the representation language,

and not on the syntactic representation:

semantically equivalent knowledge should lead to the same result



Reasoning

deduce implicit knowledge from the
explicitly represented knowledge

$$\forall x.\forall y.\forall z.\,(has\_child(x,z) \wedge has\_child(z,y) \wedge male(y)) \rightarrow has\_grandson\_of(x,y)$$

$has\_child(John, Mary)$

$has\_child(Mary, Paul)$

$male(Paul)$

$grandson\_of(John, Paul)$

implicit knowledge

Knowledge representation systems

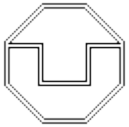
should provide their users with inference tools

that can deduce (certain) implicit consequences

results should depend only on the semantics of the representation language,

and not on the syntactic representation:

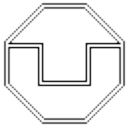
semantically equivalent knowledge should lead to the same result



Reasoning procedures

requirements in knowledge representation

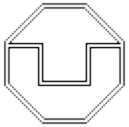
- The procedure should be a decision procedure for the problem:
  - soundness: positive answers are correct
  - completeness: negative answers are correct
  - termination: always gives an answer in finite time
- The procedure should be as efficient as possible:
  preferably optimal w.r.t. the (worst-case) complexity of the problem
- The procedure should be practical:
  easy to implement and optimize, and behave well in applications



Reasoning procedures

example

- Satisfiability in first-order logic does not have a decision procedure.
  - full first-order logic is thus not an appropriate knowledge representation formalism
- Satisfiability in propositional logic has a decision procedure, but the problem is NP-complete.
  - there are, however, highly optimized SAT solvers that behave well in practice
  - expressive power is, however, often not sufficient to express the relevant knowledge



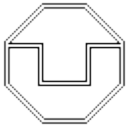
Terminological knowledge

formalize the terminology of the application domain:

- define important notions (classes, relations, objects) of the domain
- state constraints on the way these notions can be interpreted
- deduce consequences of definitions and constraints:
  subclass relationships, instance relationships

Example: domain conference

- classes (concepts) like Person, Speaker, Author, Talk, Participant, Workshop, ...
- relations (roles) like gives, attends, likes, ...
- objects (individuals) like Richard, Frank, Paper_176, ...
- constraints like: every talk is given by a speaker, every speaker is an author, every workshop must have at least 10 participants, ...

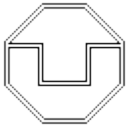


Ontologies

terminological knowledge bases are nowadays often called ontologies

Ontologies are, for example, used in:

- the Semantic Web to enable a common understanding of important notions, which can be used in the semantic labeling of Web pages
- Information Retrieval to support the automatic extraction of information from text documents
- Medicine to provide formal definitions for important notions that can be used by medical doctors to describe findings and procedures, insurance companies to determine payment, . . . (SNOMED CT, GALEN, . . .)
- Biology to enable semantic access to gene databases (Gene Ontology)
- . . .



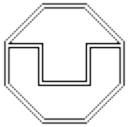
Description Logics

class of logic-based knowledge representation formalisms
tailored towards representing terminological knowledge

Prehistory:

- Descended from early approaches for representing terminological knowledge
  - semantic networks (Quillian, 1968)
  - frames (Minsky, 1975)

- problems with missing semantics lead to
  - structured inheritance networks (Brachman, 1978)
  - the first DL system KL-ONE (Brachman&Schmolze, 1985)



Description Logic

history

Phase 1:

- implementation of systems (Back, K-Rep, Loom, Meson, . . .)
- based on incomplete structural subsumption algorithms

Phase 2:

- development of tableau-based algorithms and complexity results
- first implementation of tableau-based systems (Kris, Crack)
- first formal investigation of optimization methods

Phase 3:

- tableau-based algorithms for very expressive DLs
- highly optimized tableau-based systems (FaCT, Racer, HermiT, Konclude, ...)
- relationship to modal logic and decidable fragments of FOL

Phase 4:

- Web Ontology Language (OWL-DL) based on very expressive DL
- industrial-strength reasoners and ontology editors for OWL-DL
- investigation of light-weight DLs with tractable reasoning problems
- query answering w.r.t. ontologies for large data sets