Knowledge Representation for the Semantic Web
Lecture 2: Description Logics I

Daria Stepanova

slides based on Reasoning Web 2011 tutorial "*Foundations of Description Logics and OWL*" by S. Rudolph

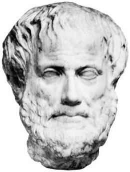
Max Planck Institute for Informatics
D5: Databases and Information Systems group

WS 2017/18

Unit Outline

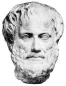
Logic-based Knowledge Representation

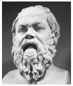
- 350 BC: roots of logic-based KR
- 17th century: idea to make knowledge explicit by logical computation
- 1930s: disillusion due to results about fundamental limits for the existence of generic algorithms
- adoption of computers and AI as a new area of research leads to intensified studies



Propositional and First-order Logic

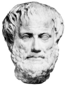
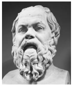
(1) Aristotel is a man. (2) Socrates is a man.





Propositional and First-order Logic

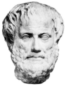
(1) Aristotel is a man. (2) Socrates is a man.

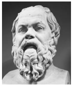



In which formalisms can we encode this knowledge?

Propositional and First-order Logic

(1) Aristotel is a man. (2) Socrates is a man.





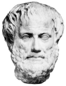
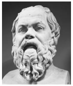
In which formalisms can we encode this knowledge?

- propositional logic (PL): propositional variables, $\neg$, $\vee$, $\wedge$, $\rightarrow$

(1) $AristotelIsAMan = true$; (2) $SocratesIsAMan = true$

Propositional and First-order Logic

(1) Aristotel is a man. (2) Socrates is a man.
(3) All men are mortal.

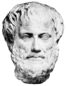
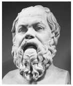
In which formalisms can we encode this knowledge?

- propositional logic (PL): propositional variables, $\neg$, $\vee$, $\wedge$, $\rightarrow$

(1) $AristotelIsAMan = true$; (2) $SocratesIsAMan = true$

Propositional and First-order Logic

(1) Aristotel is a man. (2) Socrates is a man.
(3) All men are mortal.

In which formalisms can we encode this knowledge?

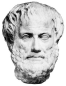
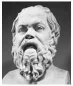
- propositional logic (PL): propositional variables, $\neg$, $\vee$, $\wedge$, $\rightarrow$

(1) $AristotelIsAMan = true$; (2) $SocratesIsAMan = true$
(3) $AristotelIsAMan \rightarrow AristotelIsMortal$
$SocratesIsAMan \rightarrow SocratesIsMortal$;

PL is **not expressive**..

Propositional and First-order Logic

(1) Aristotel is a man. (2) Socrates is a man.
(3) All men are mortal.

In which formalisms can we encode this knowledge?

- propositional logic (PL): propositional variables, $\neg$, $\vee$, $\wedge$, $\rightarrow$

(1) $AristotelIsAMan = true$; (2) $SocratesIsAMan = true$
(3) $AristotelIsAMan \rightarrow AristotelIsMortal$
$SocratesIsAMan \rightarrow SocratesIsMortal$;

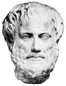
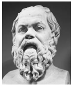
PL is **not expressive**..

- first order logic (FOL): predicates of arbitrary arity, constants, variables, function symbols, $\neg$, $\vee$, $\wedge$, $\forall$, $\exists$, $\rightarrow$

(1) $Man(socrates)$; (2) $Man(aristotel)$;
(3) $\forall X(Man(X) \rightarrow Mortal(X))$

Propositional and First-order Logic

(1) Aristotel is a man. (2) Socrates is a man.
(3) All men are mortal.

In which formalisms can we encode this knowledge?

- propositional logic (PL): propositional variables, $\neg$, $\vee$, $\wedge$, $\rightarrow$

(1) $AristotelIsAMan = true$; (2) $SocratesIsAMan = true$
(3) $AristotelIsAMan \rightarrow AristotelIsMortal$
$SocratesIsAMan \rightarrow SocratesIsMortal$;

PL is **not expressive**..

- first order logic (FOL): predicates of arbitrary arity, constants, variables, function symbols, $\neg$, $\vee$, $\wedge$, $\forall$, $\exists$, $\rightarrow$

(1) $Man(socrates)$; (2) $Man(aristotel)$;
(3) $\forall X(Man(X) \rightarrow Mortal(X))$

FOL is expressive but **undecidable** in general...

Brief Note on Decidability

Decidability

A class of problems is called decidable, if there is an algorithm that given any problem instance from this class as input can output a "yes" or "no" answer to it after finite time.

Decidable logics

In logic context, the following generic problem is normally studied:
Given: a set of statements T and a statement ϕ,
Output: "yes", iff T logically entails ϕ and "no" otherwise.
In case there is no danger of confusion about the type of problem considered, sometimes the logic itself is called decidable or undecidable.

Brief Note on Decidability (cont'd)

Decidability of propositional logic

Consider propositional logic (PL) and the following statements T and ϕ:

$$\underbrace{(SocrIsAMan \rightarrow SocrIsMortal) \wedge SocrIsAMan}_{T} \underbrace{\models}_{entails} \underbrace{SocrIsMortal}_{\phi}$$

The following questions in PL are equivalent:

- $T \models \phi$?
- $T \rightarrow \phi$ for every valuation of $socrIsAMan, socrIsMortal$?
- $T \wedge \neg\phi$ is unsatisfiable, i.e., false for every valuation?

The (un)satisfiability problem in PL is called (UN)SAT.
Propositional logic is decidable, since (UN)SAT is decidable (consider 2^n truth assignments of n variables in $T \wedge \neg\phi$).

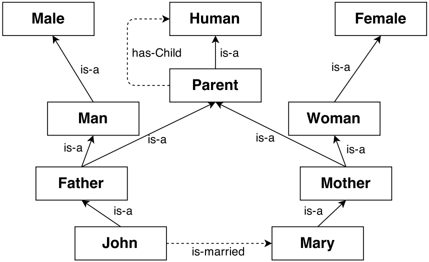
Description Logics

- 1930's: First order logic for KR (undecidable)

Description Logics

- 1930's: First order logic for KR (undecidable)

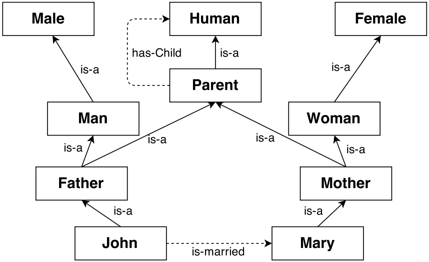

- 1970's: Network-shaped structures for KR
  - Semantic networks [Quillian, 1968], conceptual graphs, SNePs, NETL
  - Frames [Minsky, 1974]



Description Logics

- 1930's: First order logic for KR (undecidable)

- 1970's: Network-shaped structures for KR (no formal semantics)
  - Semantic networks [Quillian, 1968], conceptual graphs, SNePs, NETL
  - Frames [Minsky, 1974]



Description Logics

- 1930's: First order logic for KR (undecidable)


- 1970's: Network-shaped structures for KR (no formal semantics)
  - Semantic networks [Quillian, 1968], conceptual graphs, SNePs, NETL
  - Frames [Minsky, 1974]

- 1979: Encoding of frames into FOL [Hayes, 1979]

Description Logics

- 1930's: First order logic for KR (undecidable)

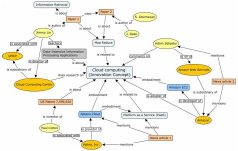
- 1970's: Network-shaped structures for KR (no formal semantics)
  - Semantic networks [Quillian, 1968], conceptual graphs, SNePs, NETL
  - Frames [Minsky, 1974]
- 1979: Encoding of frames into FOL [Hayes, 1979]
- 1980's: Description logics (DL) for KR
  - Decidable fragments of FOL
  - Theories encoded in DLs are called ontologies
  - Many DLs with different expressiveness and computational features

Description Logics

- 1930's: First order logic for KR (undecidable)

- 1970's: Network-shaped structures for KR (no formal semantics)
  - Semantic networks [Quillian, 1968], conceptual graphs, SNePs, NETL
  - Frames [Minsky, 1974]

- 1979: Encoding of frames into FOL [Hayes, 1979]



- 1980's: Description logics (DL) for KR
  - Decidable fragments of FOL
  - Theories encoded in DLs are called ontologies
  - Many DLs with different expressiveness and computational features

Description Logics (cont'd)

- Goal: ensure decidable reasoning and formal logic-based semantics
- Description logics cater for this goal
- They can be seen as decidable fragments of first-order logic, closely related to modal logics
- A significant portion of DL-related research devoted to clarifying the computational effort of reasoning tasks in terms of their worst-case complexity
- Despite high worst-case complexity, even for expressive DLs optimized reasoning algorithms exist with good behaviour in practical relevant settings
  - cf. SAT Solving: NP-complete in general but works well in practice

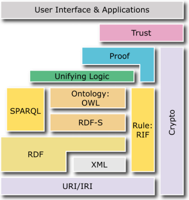
Description Logics (cont'd)

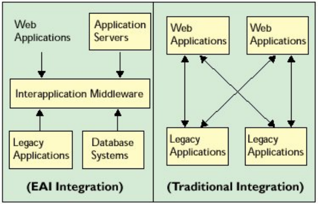
- Description logics one of today's main KR paradigms
- influenced standardization of Semantic Web languages, in particular the web ontology language OWL
- comprehensive tool support available

Fact++ **Pellet** **HermiT** **ELK**

Applications

- Semantic Web (OWL)
- Enterprise Application Integration (EAI)
- Data Modelling (UML)
- Knowledge Representation for life sciences: SNOMED Clinical Terms, Gene ontology, UniProtKB/Swiss-Prot protein sequence database, GALEN medical concepts for e-healthcare
- Ontology-Based Data Access (OBDA)
- ...





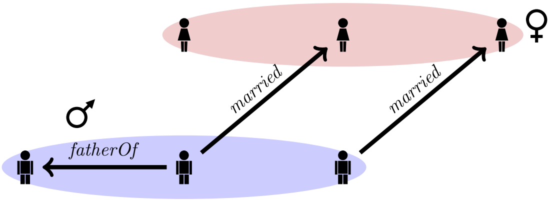
UNIFIED MODELING LANGUAGE ™

Syntax of Description Logics

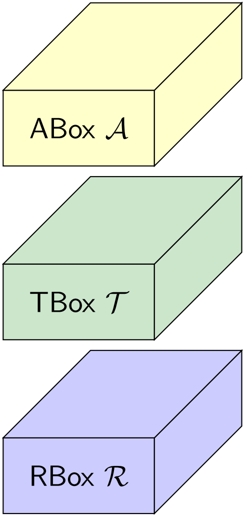
DL Building Blocks

- Individual names: $john, mary, sun, lalaland$
  aka: constants (FOL), resources (RDF)
- Concept names: $Male, Planet, Film, Country$
  aka: unary predicates (FOL), classes (RDFS)
- Role names: $married, fatherOf, actedIn$
  aka: binary predicates (FOL), properties (RDFS)

The set of all individual, concept and role names is commonly referred to as signature or vocabulary.



Constituents of a DL Knowledge Base



- information about individuals and their concept and role memberships
- information about concepts and their taxonomic dependencies
- information about roles and their dependencies

Constituents of a DL

A DL is characterized by:

- A description language: how to form concept/role expressions
  $Human \sqcap Male \sqcap \exists hasChild \sqcap \forall hasChild.(Doctor \sqcup Lawyer)$

- A mechanism to specify knowledge about concepts (i.e., TBox $\mathcal{T}$) and roles (i.e., RBox $\mathcal{R}$)
  $\mathcal{T} = \{Father \equiv Human \sqcap Male \sqcap \exists hasChild,$
  $\quad HappyFather \sqsubseteq Father \sqcap \forall hasChild.(Doctor \sqcup Lawyer)\}$
  $\mathcal{R} = \{hasFather \sqsubseteq hasParent\}$

- A mechanism to specify properties of objects (i.e., an ABox)
  $\mathcal{A} = \{HappyFather(john), hasChild(john, mary)\}$

- A set of inference services: how to reason on a given KB
  $\mathcal{T} \models HappyFather \sqcap \exists hasChild.(Doctor \sqcap Lawyer)$
  $\mathcal{T} \cup \mathcal{A} \models (Doctor \sqcap Lawyer)(mary)$

Concept Expressions

- Concept expressions are defined inductively as follows:
  - every concept name is a concept expression,
  - $\top$ and $\bot$ are concept expressions,
  - for $a_1, \ldots, a_n$ individual names, $\{a_1, \ldots, a_n\}$ is a concept expression,
  - for C and D concept expressions, $\neg C$ and $C \sqcap D$ and $C \sqcup D$ are concept expressions,
  - for r a role and C a concept expression, $\exists r.C$ and $\forall r.C$ are concept expressions,
  - for s a *simple* role, C a concept expression and n a natural number, $\exists s.\mathsf{Self}$ and $\leq n\ s.C$ and $\geq n\ s.C$ are concept expressions.
- Note: we formally define roles and simple roles later (for the moment, we use role names)

Examples of Concept Expressions

- Conjunction: $Singer \sqcap Actor$
- Disjunction: $\forall hasChild.(Doctor \sqcup Lawyer)$
- Qualified existential restriction: $\exists hasChild.Doctor$
- Full negation: $\neg(Doctor \sqcup Lawyer)$
- Number restrictions: $(\geq 2 hasChild) \sqcap (\leq 1 sibling)$
- Qualified number restrictions: $(\geq 2 hasChild.Doctor)$
- Inverse role: $\forall hasChild^{-}.Doctor$

TBox

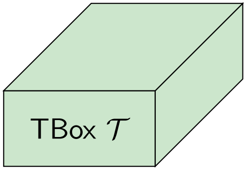
- A general concept inclusion (GCI) has the form

$$C \sqsubseteq D$$

where C and D are concept expressions.

- A TBox consists of a set of GCIs.

N.B.: Definition of TBox presumes already known RBox due to role simplicity constraints.



Example Knowledge Base

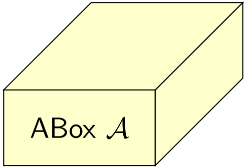
$TBox\ \mathcal{T}$

$$
\begin{array}{rcl}
Healthy & \sqsubseteq & \neg Dead \\
 & & \text{"Healthy beings are not dead."} \\
Cat & \sqsubseteq & Dead \sqcup Alive \\
 & & \text{"Every cat is dead or alive."} \\
HappyCatOwner & \sqsubseteq & \exists owns.Cat \sqcap \forall caresFor.Healthy \\
 & & \text{"A happy cat owner owns a cat and} \\
 & & \text{all beings he cares for are healthy."}
\end{array}
$$

ABox

- An individual assertion can have any of the following forms
  - $C(a)$, called concept assertion
  - $r(a, b)$, called role assertion
  - $\neg r(a, b)$, called negated role assertion
  - $a \approx b$, called equality statement, or
  - $a \not\approx b$, called inequality statement.

- An ABox consists of a set of individual assertions.



Example Knowledge Base

$TBox\ \mathcal{T}$

$$Healthy \sqsubseteq \neg Dead$$

"Healthy beings are not dead."

$$Cat \sqsubseteq Dead \sqcup Alive$$

"Every cat is dead or alive."

$$HappyCatOwner \sqsubseteq \exists owns.Cat \sqcap \forall caresFor.Healthy$$

"A happy cat owner owns a cat and all beings he cares for are healthy."

$ABox\ \mathcal{A}$

$$HappyCatOwner(schroedinger)$$

"Schrödinger is a happy cat owner."

Role Incusion Axioms

- A role can be
    - a role name r or
    - an inverted role name r^- (intuitively, reversed participants) or
    - the universal role u.
- A role inclusion axiom (RIA) is a statement of the form

$$r_1 \circ \cdots \circ r_n \sqsubseteq r$$

where $r_1, \ldots, r_n, r$ are roles.

Role Simplicity

- Given RIAs, roles are divided into simple and non-simple roles.
- Roughly, roles are non-simple if they may occur on the rhs of a complex RIA.
- More precisely,
  - for any RIA $r_1 \circ r_2 \circ \ldots \circ r_n \sqsubseteq r$ with $n > 1$, r is non-simple,
  - for any RIA $s \sqsubseteq r$ with s non-simple, r is non-simple, and
  - all other properties are simple.

Example

$$q \circ p \sqsubseteq r \qquad r \circ p \sqsubseteq r \qquad r \sqsubseteq s \qquad p \sqsubseteq r \qquad q \sqsubseteq s$$

Role Simplicity

- Given RIAs, roles are divided into simple and non-simple roles.
- Roughly, roles are non-simple if they may occur on the rhs of a complex RIA.
- More precisely,
  - for any RIA $r_1 \circ r_2 \circ \ldots \circ r_n \sqsubseteq r$ with $n > 1$, r is non-simple,
  - for any RIA $s \sqsubseteq r$ with s non-simple, r is non-simple, and
  - all other properties are simple.

Example

$$q \circ p \sqsubseteq r \qquad r \circ p \sqsubseteq r \qquad r \sqsubseteq s \qquad p \sqsubseteq r \qquad q \sqsubseteq s$$

non-simple: r, s

Role Simplicity

- Given RIAs, roles are divided into simple and non-simple roles.
- Roughly, roles are non-simple if they may occur on the rhs of a complex RIA.
- More precisely,
  - for any RIA $r_1 \circ r_2 \circ \ldots \circ r_n \sqsubseteq r$ with $n > 1$, r is non-simple,
  - for any RIA $s \sqsubseteq r$ with s non-simple, r is non-simple, and
  - all other properties are simple.

Example

$q \circ p \sqsubseteq r \qquad r \circ p \sqsubseteq r \qquad r \sqsubseteq s \qquad p \sqsubseteq r \qquad q \sqsubseteq s$

non-simple: r, s simple: p, q

RBox

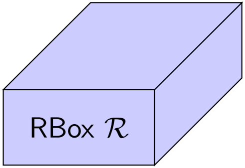
- A role disjointness statement has the form

$$Dis(s_1, s_2)$$

where s_1 and s_2 are simple roles.

- An RBox consists of regular[1] set of RIAs and a set of role disjointness statements.

- In expressive Description Logics, $\mathcal{R}$ might contain further axioms, such as $Asym(r)$ (asymmetry) and $Ref(r)$ (reflexivity).



[1] Syntactic conditions put on the usage of non-simple roles (see [Rudolph, 2011])

Example Knowledge Base

RBox $\mathcal{R}$

$$owns \sqsubseteq caresFor$$

"If somebody owns something, s/he cares for it."

TBox $\mathcal{T}$

$$Healthy \sqsubseteq \neg Dead$$

"Healthy beings are not dead."

$$Cat \sqsubseteq Dead \sqcup Alive$$

"Every cat is dead or alive."

$$HappyCatOwner \sqsubseteq \exists owns.Cat \sqcap \forall caresFor.Healthy$$

"A happy cat owner owns a cat and all beings he cares for are healthy."

ABox $\mathcal{A}$

$$HappyCatOwner(schroedinger)$$

"Schrödinger is a happy cat owner."

Exercise: try to compute all facts that follow from the KB yourself!

Summary

1. Introduction and background
    - Brief recap on propositional and first order logic
    - Decidability of logics
    - History of DLs
2. Syntax of DLs
    - DL building blocks
    - Concept expressions
    - TBox
    - ABox
    - RBox

References I

- Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors.
  The Description Logic Handbook: Theory, Implementation and Applications.
  Cambridge University Press, 2007.

- Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph.
  Foundations of Semantic Web Technologies.
  Chapman and Hall, 2010.

- Sebastian Rudolph.
  Foundations of description logics.
  In Axel Polleres, Claudia d'Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter Patel-Schneider, editors, *Reasoning Web. Semantic Technologies for the Web of Data*, volume 6848 of *Lecture Notes in Computer Science*, pages 76–136. Springer Berlin / Heidelberg, 2011.