Description Logics

Ulrike Sattler
Univ. of Manchester, UK

many many slides used here are borrowed from Carsten Lutz, TU Dresden

Knowledge Representation

General goal of knowledge representation:
"develop formalisms for providing high-level descriptions of the world that can be effectively used to build intelligent applications."

- "formalisms": syntax + well-defined semantics + reasoning services
- "high-level descriptions": which aspects should be represented, which left out?
- "intelligent applications": are able to infer new knowledge from given knowledge
- "effectively used": reasoning techniques should allow "usable" implementation

Early Formalisms

How to represent terminological knowledge?
Early days of AI: KR through obscure pictures (semantic networks)

Questions: missing semantics (reasoning?), complex pictures
Remedy: Use a logical formalism for KR rather than pictures

Terminological Knowledge

DLS focus: representation of terminological knowledge or conceptual knowledge

Goal:
- formalize the basic terminology of modeled domain
- store it in an ontology / terminology / TBox for reasoning
- enable reasoning on this knowledge

Domain of Summerschool:
- concepts: classes of individuals
e.g. Course and Lecturer
- (binary) relations: links between individuals
e.g. gives-course and attends-course
Applications

- Medical informatics
  - e.g. SNOMED, the Systematized Nomenclature of Medicine
    - ~450,000 concepts about anatomy, diseases, etc.
- Bioinformatics
  - e.g. the GeneOntology (GO): controlled vocabulary of genes
    and gene products
    - ~17,000 concepts
- Semantic Web
  - goal: provide a semantic description of the content of web pages
  - realization: point to concepts defined in an ontology

Defining Concepts with DLs

The core part of any DL is the concept language

Person ⊑ Enrolled-at, University ⊑ Yattends, UnderGradCourse

- concept names assign a name to groups of objects
- role names assign a name to relations between objects
- constructors allow to relate concept names and role names

Different sets of constructors give rise to different concept languages

The Description Logic \( \mathcal{ALC} \): Syntax

Atomic types: concept names \( A, B, \ldots \) (unary predicates)
role names \( R, S, \ldots \) (binary predicates)

Constructors:
- \( \neg C \) (negation)
- \( C \land D \) (conjunction)
- \( C \lor D \) (disjunction)
- \( \exists R.C \) (existential restriction)
- \( \forall R.C \) (value restriction)

Abbreviations:
- \( C \rightarrow D = \neg C \lor D \) (implication)
- \( C \leftrightarrow D = C \rightarrow D \land D \rightarrow C \) (bi-implication)
- \( T = (A \lor \neg A) \) (top concept)
- \( \bot = A \land \neg A \) (bottom concept)

Examples

- Person ⊑ Female
- Person ⊑ Yattends, Course
- Person ⊑ Yattends, (Course → ~Easy)
- Person ⊑ Yteaches, (Course ⊑ Yattended-by, (Nice ⊑ Intelligent))
Semantics based on interpretations \((\Delta^T, \tau)\), where
- \(\Delta^T\) is a non-empty set (the domain)
- \(\tau\) is the interpretation function mapping:
  - each concept name \(A\) to a subset \(A^T\) of \(\Delta^T\)
  - each role name \(R\) to a binary relation \(R^T\) over \(\Delta^T\).

Intuition: interpretation is complete description of the world.

Technically: interpretation is first-order structure
  with only unary and binary predicates.

Semantics of Complex Concepts
\[\neg C^T = \Delta^T \setminus C^T\]
\[(C \cap D)^T = C^T \cap D^T\]
\[(C \cup D)^T = C^T \cup D^T\]
\[(\exists R.C)^T = \{d \mid \text{there is an } e \in \Delta^T \text{ with } (d, e) \in R^T \text{ and } e \in C^T\}\]
\[(\forall R.C)^T = \{d \mid \text{for all } e \in \Delta^T, (d, e) \in R^T \text{ implies } e \in C^T\}\]

TBoxes

TBoxes are used to hold background information:
- \(\exists \text{teaches.}\top \sqsubseteq \text{Lecturer} \sqcap \text{Person}\)

TBoxes are used to hold concept definitions
- \(\text{Lecturer} \equiv \text{Person} \sqcap \exists \text{teaches.}\text{Course}\)

Syntax:
A TBox is a finite set of general concept inclusion axioms \(C \sqsubseteq D\) with \(C \sqsubseteq D\) an abbreviation for \(C \sqsubseteq D\) and \(D \sqsubseteq C\).

Semantics:
- interpretation \(\mathcal{I}\) satisfies \(C \sqsubseteq D\) if \(C^T \subseteq D^T\)
- \(\mathcal{I}\) is model of \(\mathcal{T}\) if it satisfies all GCI s in \(\mathcal{T}\)
**TBox: Example**

TBoxes are used as ontologies:

\[\exists\text{Attends.}: \top \sqsubseteq \text{Student} \sqcap \text{Person}\]

\[\exists\text{teaches.}: \top \sqsubseteq \text{Lecturer} \sqcap \text{Person}\]

\[\text{Woman} \equiv \text{Person} \sqcap \neg \text{Female}\]

\[\text{Man} \equiv \text{Person} \sqcap \neg \text{Woman}\]

\[\text{Lecturer} \equiv \text{Person} \sqcap \exists\text{teaches.}\text{Course}\]

\[\text{Student} \equiv \text{Person} \sqcap \exists\text{Attends.}\text{Course}\]

\[\text{BadLecturer} \equiv \text{Person} \sqcap \exists\text{teaches.}(\text{Course} \to \text{Boring})\]

---

**TBox: Example II**

A TBox restricts the set of admissible interpretations:

\[\text{Lecturer} \equiv \text{Person} \sqcap \exists\text{teaches.}\text{Course}\]

\[\text{Student} \equiv \text{Person} \sqcap \exists\text{Attends.}\text{Course}\]

---

**Reasoning Tasks — Subsumption**

*C* subsumed by *D* w.r.t. *T* (written *C* \(\sqsubseteq_T D\))

if

\[C^T \sqsubseteq D^T\] holds for all models *I* of *T*

**Intuition:** If *C* \(\sqsubseteq_T D\), then *D* is more general than *C*

**Example:**

For *T* = \{Lecturer \equiv Person \sqcap \exists\text{teaches.}\text{Course},

Student \equiv Person \sqcap \exists\text{Attends.}\text{Course}\}

we have that

\[\text{Lecturer} \sqcap \exists\text{Attends.}\text{Course} \sqsubseteq_T \text{Student}\]

---

**Reasoning Tasks — Classification**

**Classification:** arrange all concept names from a TBox in a hierarchy w.r.t. generality

\[\text{Woman} \equiv \text{Person} \sqcap \text{Female}\]

\[\text{Man} \equiv \text{Person} \sqcap \neg \text{Woman}\]

\[\text{MaleLecturer} \equiv \text{Man} \sqcap \exists\text{teaches.}\text{Course}\]

Can be computed using multiple subsumption tests

Provides a principled view on ontology for browsing, maintaining, etc.
Reasoning Tasks — Satisfiability

$C$ is satisfiable w.r.t. $\mathcal{T}$ if $\mathcal{T}$ has a model with $C^\mathcal{E} \neq \emptyset$

Intuition: If unsatisfiable, the concept contains a contradiction.

Example: $\text{Woman} \sqsubseteq \text{Person} \cap \text{Female}$
$\text{Man} \sqsubseteq \text{Person} \cap \neg \text{Woman}$

Then $\text{Sibling} \cap \text{Man} \cap \text{Sibling} \cap \text{Woman}$ is unsatisfiable w.r.t. $\mathcal{T}$

Subsumption can be reduced to (un)satisfiability and vice versa.

- $C \subseteq_T D$ if $C \cap \neg D$ is not satisfiable w.r.t. $\mathcal{T}$
- $C$ is satisfiable w.r.t. $\mathcal{T}$ if not $C \subseteq_T \bot$

Many reasoners decide satisfiability rather than subsumption.
**Definitional TBoxes**

A concept name $A$ is defined in $T$ if $T$ contains exactly 1 GCI of the form $A \equiv C$. All other concept names are primitive in $T$.

A primitive interpretation for TBox $T$ interprets the primitive concept names in $T$ and all role names.

A TBox is called definitional if every primitive interpretation for $T$ can be uniquely extended to a model of $T$.

i.e.: primitive concepts (and roles) uniquely determine defined concepts.

Not all TBoxes are definitional:

```
Person \equiv \exists parent . Person
```

Non-definitional TBoxes describe constraints, e.g. from background knowledge.

---

**Acyclic TBoxes**

TBox $T$ is acyclic if there are no definitional cycles.

Lecturer \equiv Person \land \exists teaches . Course

Course \equiv \exists has title . Title \land \exists taught by. Lecturer

Expansion of acyclic TBox $T$:

- exhaustively replace defined concept names with their definition (terminates due to acyclicity)

Acyclic TBoxes are always definitional:

- first expand, then set $A^T := C^T$ for all $A \equiv C \in T$

**Acyclic TBoxes II**

For reasoning, acyclic TBox can be eliminated:

- to decide $C \subseteq T D$ with $T$ acyclic,
  - expand $T$
  - replace defined concept names in $C, D$ with their definition
  - decide $C \subseteq D$
- analogously for satisfiability

May yield an exponential blow-up

```
A_0 \equiv \forall r. A_1 \land \forall s . A_1
A_1 \equiv \forall r . A_2 \land \forall s . A_2
\ldots
A_{n-1} \equiv \forall r . A_n \land \forall s . A_n
```

**General Concept Inclusions**

Recall: our TBoxes are general: finite set of general concept inclusions (GCIs)

$C \sqsubseteq D$

with both $C$ and $D$ allowed to be complex

E.g. Course \land \exists attended by. Sleeping \sqsubseteq Boring

E.g. Student \land \exists has favourite. FootballTeam \sqsubseteq Student \land \exists has favourite. Beer

Recall $C \sqsubseteq D$ is an abbreviation for $C \sqsubseteq D, D \sqsubseteq C$

Note: $C \sqsubseteq D$ is equivalent to $T \sqsubseteq C \rightarrow D$ is equivalent to $T \sqsubseteq \neg C \lor D$
ABoxes

ABoxes describe a snapshot of the world.

An ABox is a finite set of assertions:

\[ a : C \text{ if } a^2 \in C^2 \]
\[ (a, b) : R \text{ if } (a^2, b^2) \in R^2 \]

E.g. \{peter : Student, (uli, di-course) : teaches\}

Interpretations \( I \) map each individual name \( a \) to an element of \( \Delta^2 \).

\( I \) satisfies an assertion:

\[ a : C \]
\[ (a, b) : R \]

\( I \) is a model for an ABox \( \mathcal{A} \) if \( I \) satisfies all assertions in \( \mathcal{A} \).

Reasoning with ABoxes

ABox consistency

Given an ABox \( \mathcal{A} \) and a TBox \( \mathcal{T} \), do they have a common model?

Instance checking

Given an ABox \( \mathcal{A} \), a TBox \( \mathcal{T} \), an individual name \( a \), and a concept \( C \), does \( a^2 \in C^2 \) hold in all models of \( \mathcal{A} \) and \( \mathcal{T} \)?

[written \( \mathcal{A}, \mathcal{T} \models a : C \)]

The two tasks are interreducible:

\[ \mathcal{A}, \mathcal{T} \models a : C \text{ if } \mathcal{A} \not\models a : \bot \]
\[ \mathcal{A}, \mathcal{T} \models a : C \text{ if } \mathcal{A} \cup \{a : \neg C\} \text{ is not consistent} \]

Example for ABox Reasoning

ABox:
\[
\text{dumbo} : \text{Mammal} \\
\text{t14} : \text{Trunk} \\
\text{g23} : \text{Darkgrey} \\
\text{(dumbo, g23)} : \text{color} \\
\text{(dumbo, t14)} : \text{bodypart} \\
\text{dumbo} : \text{vcolor.Lightgrey} \\
\]

TBox:
\[
\text{Elephant} \equiv \text{Mammal} \land \exists \text{bodypart}. \text{Trunk} \land \forall \text{vcolor}. \text{Grey} \\
\text{Grey} \equiv \text{Lightgrey} \lor \text{Darkgrey} \\
\text{dumbo} \equiv \text{Darkgrey} \\
\]

1. ABox is inconsistent w.r.t. TBox.
2. dumbo is an instance of Elephant: TBox, ABox \models \text{dumbo} : \text{Elephant}
Good Morning!

Yesterday:
- ALC, syntax, semantics
- TBox and ABox
- reasoning problems:
  - subsumption
  - classification
  - instance
  - consistency

Next:
- DLs, FOL and modal logic
- DLs and OWL
- reasoning algorithms:
  - tableau-based
  - automata-based
- computational complexity

### ABox Reasoning vs. Concept Reasoning

Concept reasoning can be reduced to ABox reasoning:
- $C$ satisfiable w.r.t. $T$ iff $a : C$ is consistent
- $C \subseteq_D D$ iff $\{a : C\}, T \models a : D$

In ALC, ABox reasoning can also be reduced to concept reasoning:

To decide whether $A$ is consistent:
1. Precomputation: explicate knowledge in $A$ by applying rules such as:
   \[
   a : (C \cap D) \implies a : C \text{ and } a : D
   \]
   \[
   a : \forall x.C \text{ and } (a, b) : r \implies b : C
   \]
2. For each of the resulting ABoxes $A_1, \ldots, A_n$, and each individual $a$,
   check whether the conjunction of $\{C \mid a : C \in A_i\}$ is satisfiable

### Description Logics and First-order Logic

| concept names $A$ | $\iff$ | unary predicates $P_A$ |
| concept names $R$ | $\iff$ | binary predicates $P_R$ |
| concepts | $\iff$ | formulas with one free variable |

\[
\varphi^A(A) = P_A(x)
\]
\[
\varphi^A(\neg C) = \neg \varphi^A(C)
\]
\[
\varphi^A(C \land D) = \varphi^A(C) \land \varphi^A(D)
\]
\[
\varphi^A(C \lor D) = \varphi^A(C) \lor \varphi^A(D)
\]
\[
\varphi^A(\exists R.C) = \exists y.P_R(x, y) \land \varphi^A(C)
\]
\[
\varphi^A(\forall R.C) = \forall y.P_R(x, y) \rightarrow \varphi^A(C)
\]

Note: not all DLs are purely first-order (transitive closure, etc.)
So far, we have seen

- Syntax and semantics of ALC
- TBoxes (also acyclic and general ones)
- ABoxes
- Reasoning services and their relationship
  - Subsumption and satisfiability of possibly w.r.t. a Tbox
  - ABox consistency and instance checking
  - Relationship between ALC and FOL and Modal Logic

Today, we will see

- Some more expressive Description Logics
- Relationship between DLs and OWL
- Tableau algorithms for ALC and its extension
  - With general Tboxes
  - With inverse roles
- And discuss optimisation techniques for these algorithms

Wednesday: automata-based algorithms
Thursday: computational complexity
Friday: sub-Boolean DLs and rules

---

TBoxes:

Let $\mathcal{T}$ a general TBox und $U$ the universal modal parameter.

\[ \varphi(\mathcal{T}) = [U] \bigwedge_{\varphi(D) \in \mathcal{T}} \varphi(D) \rightarrow \varphi(E) \]

ABoxes:

- Individual names $\alpha$ \leftrightarrow nominals $\alpha$

\[ \varphi(\alpha : C) = \{ \alpha \varphi(C) \} \]

\[ \varphi((\alpha, \beta) : R) = \{ \alpha, \beta \varphi(R) \} \]

\[ \varphi(A) = \bigwedge_{\beta \in A} \varphi(\beta) \]

---

Extensions of ALC:

Lecturer $\equiv$ Person $\cap$ Teaches, Course

Course $\equiv$ $\exists$ has-title, Title $\cap$ $\exists$ taught-by, Lecturer

We should want that:

\[ \{ (x, y) \in \text{teaches} \text{ iff } (y, x) \in \text{taught-by} \} \]

e.g., so that Lecturer $\cap$ Blond $\cap$ $\forall$ teaches.$\forall$taught-by.$\neg$Blond is unsatisfiable!

Extension of ALC to ALCI

- Allow inverse roles $r^-$ in place of role names
- Semantics ensures that $(r^-)^T$ is converse of $r^T$

Lecturer $\equiv$ Person $\cap$ teaches, Course

Course $\equiv$ $\exists$has-title, Title $\cap$ $\exists$ teaches$^{-}$, Lecturer

Now Lecturer $\cap$ Blond $\cap$ $\forall$ teaches.$\forall$taught-by.$\neg$Blond is unsatisfiable!
In $\mathcal{ALC}$, how to say that:
- a small course has at most 10 students
- a shared course is taught by at least two lecturers
- every person has exactly two hands

Extension of $\mathcal{ALC}$ to $\mathcal{ALC}Q$ and $\mathcal{ALC}I$ to $\mathcal{ALC}QI$:
- new concept constructors
  - $(\leq n R C)$ and $(\geq n R C)$ (qualified number restrictions)
- e.g. SmallCourse $\equiv$ Course $\cap$ (\leq 10 attended by Student)
- sometimes only available in "unqualified", indicated by $\mathcal{N}$
  - $(\leq n R T)$ and $(\geq n R T)$ (number restrictions)

Description Logics and Computational Complexity

Numerous complexity results have been established for DLs. In general:
- we are interested in the decidability/worst-case complexity of determining the
  - subsumption between two concepts (w.r.t. a general/restricted TBox)
  - satisfiability of a concept (w.r.t. a general/restricted TBox)
  - consistency of an ABox w.r.t. a TBox
  - retrieving instances of a concept from an ABox and TBox
  - etc.
- for DLs, these problems are decidable and anywhere from
  - LogSpace, P, PSpace, ExpTime, and NExpTime-complete
- but people are strongly interested in implementations of decision procedures for these
  reasoning problems: so "practicable" is important

Extensions of $\mathcal{ALC}$ III

in $\mathcal{ALC}$, how to model the interaction between:
- the relation between has-daughter and has-child
- the part-whole relation, i.e., the one between parts and wholes
- the relation between has-child and has-descendant

Extension of $\mathcal{ALC}$ with transitive roles is called $\mathcal{ALC}$,
transitive roles and role hierarchies is called $\mathcal{ALC}_h$
- new inclusions $R \subseteq S$ in TBox (or RBox)
- new statements $\text{Tr}(R)$ in TBox (or RBox)

in $\mathcal{ALC}$, how to model persons who have seen Mona Lisa?

But: Increasing expressivity may increase computational complexity

$\implies$ Tradeoff between expressivity and computational complexity!
DLs and OWL

- Originally, DLs were designed to represent terminological knowledge (TBox) and partial descriptions of the world (ABox).
- They turned out to be useful as ontology languages, and thus they form the logical basis of:
  - OIL, DAML+OIL
  - OWL-light is based in SHIQ
  - OWL-DL is based on SROIQ, and not SHOIQ.
- Hence ontology designers/users can make use of DL reasoners to check ontologies for consistency/answer queries, etc.
- And ontology editors such as Protégé are now connected to DL reasoners.

Description Logics and OWL-DL

<table>
<thead>
<tr>
<th>Axiom</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>subClassOf</td>
<td>$C_1 \sqsubseteq C_2$</td>
<td>Human $\sqsubseteq$ Animal $\sqcap$ Biped</td>
</tr>
<tr>
<td>equivalentClass</td>
<td>$C_1 \equiv C_2$</td>
<td>Man $\equiv$ Human $\sqcap$ Male</td>
</tr>
<tr>
<td>subPropertyOf</td>
<td>$P_1 \sqsubseteq P_2$</td>
<td>hasDaughter $\sqsubseteq$ hasChild</td>
</tr>
<tr>
<td>equivalentProperty</td>
<td>$P_1 \equiv P_2$</td>
<td>cost $\equiv$ price</td>
</tr>
<tr>
<td>disjointWith</td>
<td>$C_1 \sqcap \neg C_2$</td>
<td>Male $\sqcap$ ~Female</td>
</tr>
<tr>
<td>sameAs</td>
<td>${x_1} \sqsubseteq {x_2}$</td>
<td>{Pres. Bush} $\equiv$ {G.W.Bush}</td>
</tr>
<tr>
<td>differentFrom</td>
<td>${x_1} \sqsubseteq \neg {x_2}$</td>
<td>{john} $\equiv$ ~{peter}</td>
</tr>
<tr>
<td>TransitiveProperty</td>
<td>$\text{Tr}(P)$</td>
<td>$\text{Tr}(\text{hasAncestor})$</td>
</tr>
<tr>
<td>FunctionalProperty</td>
<td>$\exists \leq 1 P$</td>
<td>$\exists \leq 1 \text{hasMother}$</td>
</tr>
<tr>
<td>InverseFunctionalProperty</td>
<td>$\exists \leq 1 P^-$</td>
<td>$\exists \leq 1 \text{isMotherOf}^-$</td>
</tr>
<tr>
<td>SymmetricProperty</td>
<td>$P \equiv P^-$</td>
<td>$\text{isSiblingOf} \equiv \text{isSiblingOf}^-$</td>
</tr>
</tbody>
</table>

Description Logics and OWL-DL

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \sqcap \ldots \sqcap C_n$</td>
<td>Human $\sqcap$ Male</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \sqcup \ldots \sqcup C_n$</td>
<td>Doctor $\sqcup$ Lawyer</td>
</tr>
<tr>
<td>complementOf</td>
<td>$C \setminus$</td>
<td>Male $\neg$</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1, \ldots, x_n}$</td>
<td>{john, mary}</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>$\exists P C$</td>
<td>\text{hasChildDoctor}</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>$\exists r {x}$</td>
<td>\text{hasChild, Lawyer}</td>
</tr>
<tr>
<td>hasValue</td>
<td>\text{CitizenOf}{USA}</td>
<td></td>
</tr>
<tr>
<td>minCardinality</td>
<td>$\geq n r$</td>
<td>$\geq 2 \text{hasChild}$</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>$\leq n r$</td>
<td>$\leq 1 \text{hasChild}$</td>
</tr>
<tr>
<td>inverseOf</td>
<td>$r^-$</td>
<td>$\text{hasChild}^-$</td>
</tr>
</tbody>
</table>

Overview of the Course

- Tableau algorithms for Description Logics
- Automata-based decision procedures for Description Logics
- Computational complexity of selected Description Logics
- Sub-Boolean Description Logics and Non-Standard Reasoning