Non-classical logics

Lecture 18: Description Logics (Part 2)

Viorica Sofronie-Stokkermans

sofronie@uni-koblenz.de

Until now

Description logics

$\mathcal{ALC}$: Syntax, Semantics

Knowledge Base (KB): TBOX, ABOX

Reasoning problems; reduction to concept satisfiability/satisfiability of KB

Decidability $\mapsto$ express $\mathcal{ALC}$ as multi-modal logic.

$\mathcal{ALC}$ as a multi-modal logic

Lemma $C_1 \sqsubseteq C_2$ iff $F_{C_1 \sqcap \neg C_2}$ is unsatisfiable in the multi-modal logic.

Proof. $C_1 \sqsubseteq C_2$ iff for all $\mathcal{I}$ and all $d \in \Delta^{\mathcal{I}}$ we have: $d \notin (C_1 \sqcap \neg C_2)^{\mathcal{I}}$

From the first lemma, this happens iff $(\mathcal{K}, d) \not\models F_{C_1} \wedge \neg F_{C_2}$ for all $\mathcal{I}$ and all $d \in \Delta^{\mathcal{I}}$.

This is the same as saying that $F_{C_1 \sqcap \neg C_2}$ is unsatisfiable.

Reasoning procedures

- Terminating, efficient and complete algorithms for deciding **satisfiability** – and all the other reasoning services – are available.
- Algorithms are based on tableaux-calculi techniques or resolution.

Description logics

Two directions of research:

- Extensions in order to increase expressivity

- Restrict language in order to identify "tractable" description logics

Description logics

Two directions of research:

- Extensions in order to increase expressivity

  SHIQ

- Restrict language in order to identify “tractable” description logics

  $\mathcal{EL}$

Some extensions of ALC

SHIQ:

Syntax:

N_C primitive concept symbols

N_R^0 set of atomic role symbols

$N_t^0 \subseteq N_R^0$ set of transitive role symbols

The set N_R of role symbols contains all atomic roles and for every role $R \in N_R^0$ also its inverse role R^-.

Some extensions of ALC

SHIQ:

Role hierarchy:

A role hierarchy is a finite set $\mathcal{H}$ of formulae of the form

$$R_1 \sqsubseteq R_2$$

for $R_1, R_2 \in N_R$.

All following definitions assume that a role hierarchy is given (and fixed)

SHIQ concept descriptions: Syntax

$$
\begin{array}{lcll}
C & := & A & \text{if } A \text{ is a primitive concept} \\
 & & |\top & \\
 & & |\neg C & \\
 & & |C_1 \sqcap C_2 & \\
 & & |C_2 \sqcup C_2 & \\
 & & |\exists R.C & \\
 & & |\forall R.C & \\
 & & | \leq nR.C & \text{where } n \in \mathbb{N}, R \text{ simple role} \\
 & & | \geq nR.C & \text{where } n \in \mathbb{N}, R \text{ simple role}
\end{array}
$$

R is a simple role if $R \notin N_t^0$ and R does not contain any transitive sub-role.

SHIQ concept descriptions: Syntax

$$
\begin{array}{lll}
C & := & A \quad\quad\quad\quad \text{if } A \text{ is a primitive concept} \\
& & |\top \\
& & |\neg C \\
& & |C_1 \sqcap C_2 \\
& & |C_2 \sqcup C_2 \\
& & |\exists R.C \\
& & |\forall R.C \\
& & | \leq nR.C \quad \text{where } n \in \mathbb{N}, R \text{ simple role} \\
& & | \geq nR.C \quad \text{where } n \in \mathbb{N}, R \text{ simple role}
\end{array}
$$

R is a simple role if $R \notin N_t^0$ and R does not contain any transitive sub-role.

Abbreviations: $\geq nR := \geq nR.\top \quad \leq nR := \geq nR.\top$

Cardinality Restriction

Role quantification cannot express that a woman has *at least 3* (or *at most 5*) children.

Cardinality restrictions can express conditions on the number of fillers:

- $\mathtt{Busy{-}Woman} \doteq \mathtt{Woman} \sqcap (\geq 3\mathtt{CHILD})$
- $\mathtt{Woman{-}with{-}at{-}most5children} \doteq \mathtt{Woman} \sqcap (\leq 5\mathtt{CHILD})$

$$\boxed{(\geq 1 R) \Longleftrightarrow (\exists R)}$$

Interpretations for SHIQ

Interpretations: $\mathcal{I} = (D^{\mathcal{I}}, \cdot^{\mathcal{I}})$

- $C \in N_C \mapsto C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $R \in N_R \mapsto R^{\mathcal{I}} \subseteq D^{\mathcal{I}} \times D^{\mathcal{I}}$

such that:

- for all $R \in N_t^0$, $R^{\mathcal{I}}$ is a transitive relation
- for all $R \in N_R^0$, $(R^{-1})^{\mathcal{I}}$ is the inverse of $R^{\mathcal{I}}$
- for all $R_1 \sqsubseteq R_2 \in \mathcal{H}$ we have $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$

SHIQ constructors: Semantics

Constructor	Syntax	Semantics
concept name	A	$A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
top	$\top$	$D^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
negation	$\neg C$	$D^{\mathcal{I}} \setminus C^{\mathcal{I}}$
universal	$\forall R.C$	$\{x \mid \forall y(R^{\mathcal{I}}(x,y) \rightarrow y \in C^{\mathcal{I}})\}$
existential	$\exists R.C$	$\{x \mid \exists y(R^{\mathcal{I}}(x,y) \wedge y \in C^{\mathcal{I}}\}$
cardinality	$\geq nR$	$\{x \mid \#\{y \mid R^{\mathcal{I}}(x,y)\} \geq n\}$
	$\leq nR$	$\{x \mid \#\{y \mid R^{\mathcal{I}}(x,y)\} \leq n\}$
qual. cardinality	$\geq nR.C$	$\{x \mid \#\{y \mid R^{\mathcal{I}}(x,y) \wedge y \in C^{\mathcal{I}}\} \geq n\}$
	$\leq nR.C$	$\{x \mid \#\{y \mid R^{\mathcal{I}}(x,y) \wedge y \in C^{\mathcal{I}}\} \leq n\}$

Decidability

Theorem. The satisfiability and subsumption problem for $\mathcal{SHIQ}$ are decidable

Proof: cf. Horrocks et al.

Undecidability

Theorem. If in the definition of SHIQ we do not impose the restriction about simple roles, the satisfiability problem becomes undecidable

(even if we only allow for cardinality restrictions of the form $\leq nR.\top$ and $\geq nR.\top$).

Proof: cf. Horrocks et al.

Reasoning procedures

- For decidable description logic it is important to have efficient reasoning procedures which are sound, complete and termination.
- Literature: tableau calculi

Goals:

- Completeness is important for the usability of description logics in real applications.
- Efficiency: Algorithms need to be efficient for both average and real knowledge bases, even if the problem in the corresponding logic is in PSPACE or EXPTIME.

A tractable DL

Tractable description logic: $\mathcal{EL}, \mathcal{EL}^+$ and extensions [Baader'03–]
used e.g. in medical ontologies (SNOMED)

$\mathcal{EL}$: Generalities

Concepts:
- primitive concepts N_C
- complex concepts (built using concept constructors $\sqcap, \exists r$)

Roles: N_R

Interpretations: $\mathcal{I} = (D^{\mathcal{I}}, \cdot^{\mathcal{I}})$
- $C \in N_C \mapsto C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $r \in N_R \mapsto r^{\mathcal{I}} \subseteq D^{\mathcal{I}} \times D^{\mathcal{I}}$

Constructor name	Syntax	Semantics
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \mid \exists y((x, y) \in r^{\mathcal{I}}$ and $y \in C^{\mathcal{I}})\}$

$\mathcal{EL}$: Generalities

Concepts: • primitive concepts N_C

• complex concepts (built using concept constructors $\sqcap, \exists r$)

Roles: N_R

Interpretations: $\mathcal{I} = (D^{\mathcal{I}}, \cdot^{\mathcal{I}})$

• $C \in N_C \mapsto C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

• $r \in N_R \mapsto r^{\mathcal{I}} \subseteq D^{\mathcal{I}} \times D^{\mathcal{I}}$

Problem:

Given: TBox (set $\mathcal{T}$ of concept inclusions $C_i \sqsubseteq D_i$)

concepts C, D

Task: test whether $C \sqsubseteq_{\mathcal{T}} D$, i.e. whether for all $\mathcal{I} = (D^{\mathcal{I}}, \cdot^{\mathcal{I}})$

if $C_i^{\mathcal{I}} \subseteq D_i^{\mathcal{I}} \;\; \forall C_i \sqsubseteq D_i \in \mathcal{T}$ then $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

$\mathcal{EL}$: Example

Primitive concepts:	protein, process, substance
Roles:	catalyzes, produces
Terminology:	enzyme $=$ protein $\sqcap$ $\exists$catalyzes.reaction
(TBox)	catalyzer $=$ $\exists$catalyzes.process
	reaction $=$ process $\sqcap$ $\exists$produces.substance
Query:	enzyme $\sqsubseteq$ catalyzer?

$\mathcal{EL}^+$: generalities

Concepts:
- primitive concepts N_C
- complex concepts (built using concept constructors $\sqcap, \exists r$)

Roles: N_R

Interpretations: $\mathcal{I} = (D^\mathcal{I}, \cdot^\mathcal{I})$
- $C \in N_C \mapsto C^\mathcal{I} \subseteq D^\mathcal{I}$
- $r \in N_R \mapsto r^\mathcal{I} \subseteq D^\mathcal{I} \times D^\mathcal{I}$

Problem:

Given: CBox $\mathcal{C} = (\mathcal{T}, RI)$, where $\mathcal{T}$ set of concept inclusions $C_i \sqsubseteq D_i$;
RI set of role inclusions $r \circ s \sqsubseteq t$ or $r \sqsubseteq t$
concepts C, D

Task: test whether $C \sqsubseteq_\mathcal{C} D$, i.e. whether for all $\mathcal{I} = (D^\mathcal{I}, \cdot^\mathcal{I})$

if $C_i^\mathcal{I} \subseteq D_i^\mathcal{I} \quad \forall C_i \sqsubseteq D_i \in \mathcal{T}$ and
$r^\mathcal{I} \circ s^\mathcal{I} \subseteq t^\mathcal{I} \quad \forall r \circ s \sqsubseteq t \in RI$ then $C^\mathcal{I} \subseteq D^\mathcal{I}$

$\mathcal{EL}^+$: Example

Primitive concepts:	protein, process, substance
Roles:	catalyzes, produces, helps-producing
Terminology:	enzyme $=$ protein $\sqcap$ $\exists$catalyzes.reaction
(TBox)	reaction $=$ process $\sqcap$ $\exists$produces.substance
Role inclusions:	catalyzes $\circ$ produces $\sqsubseteq$ helps-producing
Query:	enzyme $\sqsubseteq$ protein $\sqcap$ $\exists$helps-producing.substance ?

Complexity

T-Box subsumption for $\mathcal{EL}$ decidable in PTIME

C-Box subsumption for $\mathcal{EL}^+$ decidable in PTIME

Methods:

Reductions to checking satisfiability of clauses in propositional logic.

$\mathcal{EL}$: Hierarchical reasoning

Primitive concepts:	protein, process, substance
Roles:	catalyzes, produces
Terminology: (TBox)	enzyme $=$ protein $\sqcap$ $\exists$catalyzes.reaction catalyzer $=$ $\exists$catalyzes.process reaction $=$ process $\sqcap$ $\exists$produces.substance
Query:	enzyme $\sqsubseteq$ catalyzer?

$$\begin{array}{rl} \text{SLat} \cup \text{Mon} \models & \text{enzyme} = \text{protein} \sqcap \text{catalyzes-some}(\text{reaction}) \quad \wedge \\ & \text{catalyzer} = \text{catalyze-some}(\text{process}) \quad \wedge \\ & \text{reaction} = \text{process} \sqcap \text{produces-some}(\text{substance}) \\ & \Rightarrow \text{enzyme} \sqsubseteq \text{catalyzer} \end{array}$$

$$\begin{array}{rl} \text{Mon}: & \forall C, D (C \sqsubseteq D \rightarrow \text{catalyze-some}(C) \sqsubseteq \text{catalyze-some}(D)) \\ & \forall C, D (C \sqsubseteq D \rightarrow \text{produces-some}(C) \sqsubseteq \text{produces-some}(D)) \end{array}$$

$\mathcal{EL}$: Hierarchical reasoning

$$\mathsf{SLat} \cup \mathsf{Mon} \wedge \underbrace{\begin{array}{l} \mathsf{enzyme} = \mathsf{protein} \sqcap \mathsf{catalyzes\text{-}some}(\mathsf{reaction}) \wedge \\ \mathsf{catalyzer} = \mathsf{catalyze\text{-}some}(\mathsf{process}) \wedge \\ \mathsf{reaction} = \mathsf{process} \sqcap \mathsf{produces\text{-}some}(\mathsf{substance}) \wedge \\ \mathsf{enzyme} \not\sqsubseteq \mathsf{catalyzer} \end{array}}_{G} \models \perp$$

$G \wedge \mathsf{Mon}$
$\mathsf{enzyme} = \mathsf{protein} \sqcap \mathsf{catalyzes\text{-}some}(\mathsf{reaction}) \wedge$
$\mathsf{catalyzer} = \mathsf{catalyze\text{-}some}(\mathsf{process}) \wedge$
$\mathsf{reaction} = \mathsf{process} \sqcap \mathsf{produces\text{-}some}(\mathsf{substance}) \wedge$
$\mathsf{enzyme} \not\sqsubseteq \mathsf{catalyzer}$
$\forall C, D(C \sqsubseteq D \rightarrow \mathsf{catalyze\text{-}some}(C) \sqsubseteq \mathsf{catalyze\text{-}some}(D))$
$\forall C, D(C \sqsubseteq D \rightarrow \mathsf{produces\text{-}some}(C) \sqsubseteq \mathsf{produces\text{-}some}(D))$

$\mathcal{EL}$: Hierarchical reasoning

$$\mathsf{SLat} \cup \mathsf{Mon} \wedge \underbrace{\begin{array}{l} \mathsf{enzyme} = \mathsf{protein} \sqcap \mathsf{catalyzes\text{-}some}(\mathsf{reaction}) \wedge \\ \mathsf{catalyzer} = \mathsf{catalyze\text{-}some}(\mathsf{process}) \wedge \\ \mathsf{reaction} = \mathsf{process} \sqcap \mathsf{produces\text{-}some}(\mathsf{substance}) \wedge \\ \mathsf{enzyme} \not\leq \mathsf{catalyzer} \end{array}}_{G} \models \perp$$

Solution 1: Use $DPLL(\mathsf{SLat} + UIF)$

$G \wedge \mathsf{Mon}[G]$

$\mathsf{enzyme} = \mathsf{protein} \sqcap \mathsf{catalyzes\text{-}some}(\mathsf{reaction})$
$\mathsf{catalyzer} = \mathsf{catalyzes\text{-}some}(\mathsf{process})$
$\mathsf{reaction} = \mathsf{process} \sqcap \mathsf{produces\text{-}some}(\mathsf{substance})$
$\mathsf{enzyme} \not\leq \mathsf{catalyzer}$
$\mathsf{reaction} \rhd \mathsf{process} \rightarrow \mathsf{catalyzes\text{-}some}(\mathsf{reaction}) \rhd \mathsf{catalyzes\text{-}some}(\mathsf{process}), \ \rhd \in \{\leq, \geq, =\}$

$\mathcal{EL}$: Hierarchical reasoning

$$\text{SLat} \cup \text{Mon} \wedge \underbrace{\begin{array}{l} \text{enzyme} = \text{protein} \sqcap \text{catalyzes-some}(\text{reaction}) \wedge \\ \text{catalyzer} = \text{catalyze-some}(\text{process}) \wedge \\ \text{reaction} = \text{process} \sqcap \text{produces-some}(\text{substance}) \wedge \\ \text{enzyme} \not\leq \text{catalyzer} \end{array}}_{G} \models \perp$$

Solution 2: Hierarchical reasoning

Base theory (SLat)	Extension
enzyme $=$ protein $\sqcap c_1$	$c_1 =$ catalyzes-some(reaction)
catalyzer $= c_2$	$c_2 =$ catalyzes-some(process)
reaction $=$ process $\sqcap c_3$	$c_3 =$ produces-some(substance)
enzyme $\not\leq$ catalyzer	
reaction $\rhd$ process $\rightarrow c_1 \rhd c_2$ $\quad \rhd \in \{\leq, \geq, =\}$	

Test satisfiability using any prover for SLat (e.g. reduction to SAT)

$\mathcal{EL}$: Hierarchical reasoning

Idea in the translation to SAT:

Base theory $\mapsto$	SAT (FOL)
enzyme $=$ protein $\sqcap$ c_1	$\forall x$ enzyme$(x) \leftrightarrow$ protein$(x) \wedge c_1(x)$
catalyzer $=$ c_2	$\forall x$ catalyzer$(x) \leftrightarrow c_2(c)$
reaction $=$ process $\sqcap$ c_3	$\forall x$ reaction$(x) \leftrightarrow$ process$(x) \wedge c_3(x)$
enzyme $\not\sqsubseteq$ catalyzer	enzyme$(c) \wedge \neg$catalyzer(c)
reaction $\sqsubseteq$ process $\rightarrow c_1 \sqsubseteq c_2$	$(\forall x($reaction$(x) \rightarrow$ process$(x))) \rightarrow (\forall x(c_1(x) \rightarrow c_2(x)))$
...	

$$\Downarrow$$

$$(\text{reaction}(d) \rightarrow \text{process}(d)) \rightarrow (\forall x(c_1(x) \rightarrow c_2(x)))$$

$$\Downarrow$$

Clause normal form: no function symbols of arity ≥ 1; Horn except for last class of clauses (a small amount of case distinction $\mapsto$ no increase in compl.)

By Herbrand's theorem the set of clauses is satisfiable iff its set of instances is.
Size of instantiated set: polynomial. Satisfiability of Horn clauses: in PTIME.