

IN3070/4070 – Logic – Autumn 2020

Lecture 12: Description Logics and Termination

Egor V. Kostylev

5th November 2019



DEPARTMENT OF
INFORMATICS



UNIVERSITY OF
OSLO

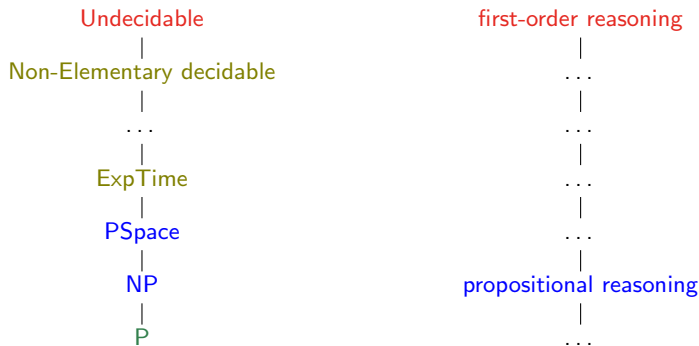
Today's Plan

- ▶ Motivation and Examples
- ▶ \mathcal{ALC} Syntax and Semantics
- ▶ Calculus for \mathcal{ALC} Terminological Reasoning
- ▶ Discussion

Outline

- ▶ Motivation and Examples
- ▶ *ALC* Syntax and Semantics
- ▶ Calculus for *ALC* Terminological Reasoning
- ▶ Discussion

Motivation



What are **Description (and Modal) Logics**?

- ▶ Fragments of first-order with **decidable** reasoning
- ▶ Typically 'in-between' propositional and first-order
 - ▶ Complexity typically between PSpace and 2NExpTime
 - ▶ Some **lightweight** DLs have polynomial reasoning

Motivation

Many applications (e.g., in Knowledge Representation, the Semantic Web) **do not require full power of first-order**

What can we leave out?

- ▶ Key reasoning problems should become **decidable**
- ▶ Sufficient expressive power to model application domain

Description Logics are a family of first-order fragments that meet these requirements for many applications:

- ▶ Underlying formalisms of modern ontology languages
- ▶ Widely-used in information systems (bio-medical, oil and gas, etc.)
- ▶ Core component of the Semantic Web

Motivation

Consider an example from the bio-medical domain:

- ▶ A juvenile disease affects only children or teens
- ▶ Children and teens are not adults
- ▶ A person is a child, a teen, or an adult
- ▶ Juvenile arthritis is a kind of arthritis and a juvenile disease
- ▶ Every kind of arthritis damages some joint

The types of objects given by unary first-order predicates:
juvenile disease, child, teen, adult, ...

The types of relationships given by binary first-order predicates:
affects, damages, ...

Motivation

The **vocabulary of a Description Logic** is composed of

- ▶ Unary first-order predicates
Arthritis, Child, ...
- ▶ Binary first-order predicates
Affects, Damages, ...
- ▶ first-order constants
JohnSmith, MaryJones, JRA, ...

We are already **restricting the expressive power of first-order logic**

- ▶ No function symbols
- ▶ No predicates of arity greater than 2

Motivation

Now, let's take a look at the first-order formulas for our example:

$$\begin{aligned} &\forall x.(\text{JuvDis}(x) \rightarrow \forall y.(\text{Affects}(x, y) \rightarrow \text{Child}(y) \vee \text{Teen}(y))) \\ &\quad \forall x.(\text{Child}(x) \vee \text{Teen}(x) \rightarrow \neg \text{Adult}(x)) \\ &\quad \forall x.(\text{Person}(x) \rightarrow \text{Child}(x) \vee \text{Teen}(x) \vee \text{Adult}(x)) \\ &\quad \forall x.(\text{JuvArthritis}(x) \rightarrow \text{Arthritis}(x) \wedge \text{JuvDis}(x)) \\ &\quad \forall x.(\text{Arthritis}(x) \rightarrow \exists y.(\text{Damages}(x, y) \wedge \text{Joint}(y))) \end{aligned}$$

We can find several **regularities** in these formulas:

- ▶ There is an outermost universal quantifier on a single variable x
- ▶ They can be split into two parts by the implication symbol

Each part is a formula with one free variable

- ▶ Atomic formulas involving a binary predicate occur only quantified in a syntactically restricted way

Motivation

Consider as an example one of our formulas:

$$\forall x. (\textit{Child}(x) \vee \textit{Teen}(x) \rightarrow \neg \textit{Adult}(x))$$

Let's look at all its sub-formulas at each side of the implication

| | |
|---|---|
| $\textit{Child}(x)$ | Set of all children |
| $\textit{Teen}(x)$ | Set of all teens |
| $\textit{Child}(x) \vee \textit{Teen}(x)$ | Set of all people that are either children or teens |
| $\textit{Adult}(x)$ | Set of all adults |
| $\neg \textit{Adult}(x)$ | Set of all objects that are not adult people |

Important observations concerning **formulas with one free variable**:

- ▶ Some are **atomic** (e.g., $\textit{Child}(x)$)
 - do not contain other formulas as subformulas
- ▶ Others are **complex** (e.g., $\textit{Child}(x) \vee \textit{Teen}(x)$)
- ▶ Variables are redundant!

Basic Definitions

Idea: Define **operators** for constructing complex formulas with one free variable out of simple **building blocks**

Atomic concept: Represents an atomic formula with one free variable

$$Child \rightsquigarrow Child(x)$$

Complex concepts (part 1):

- ▶ Concept Union (\sqcup): applies to two concepts

$$Child \sqcup Teen \rightsquigarrow Child(x) \vee Teen(x)$$

- ▶ Concept Intersection (\sqcap): applies to two concepts

$$Arthritis \sqcap JuvDis \rightsquigarrow Arthritis(x) \wedge JuvDis(x)$$

- ▶ Concept Negation (\neg): applies to one concept

$$\neg Adult \rightsquigarrow \neg Adult(x)$$

Motivation

Consider examples with binary predicates:

$$\forall x.(\textit{Arthritis}(x) \rightarrow \exists y.(\textit{Damages}(x, y) \wedge \textit{Joint}(y)))$$
$$\forall x.(\textit{JuvDis}(x) \rightarrow \forall y.(\textit{Affects}(x, y) \rightarrow \textit{Child}(y) \vee \textit{Teen}(y)))$$

- ▶ We have a **concept** and a binary predicate (called **role**) mentioning concept's free variable
- ▶ The role and the concept are connected via conjunction (existential quantification) or implication (universal quantification)

Basic Definitions

Atomic role: Represents an atom with two free variables

$$\mathit{Affects} \rightsquigarrow \mathit{Affects}(x, y)$$

Complex concepts (part 2): apply to an **atomic role** and a **concept**

- ▶ Existential Restriction:

$$\exists \mathit{Damages}.\mathit{Joint} \rightsquigarrow \exists y.(\mathit{Damages}(x, y) \wedge \mathit{Joint}(y))$$

- ▶ Universal Restriction:

$$\forall \mathit{Affects}.\mathit{Child} \sqcup \mathit{Teen} \rightsquigarrow \forall y.(\mathit{Affects}(x, y) \rightarrow \mathit{Child}(y) \vee \mathit{Teen}(y))$$

Outline

- ▶ Motivation and Examples
- ▶ *ALC* Syntax and Semantics
- ▶ Calculus for *ALC* Terminological Reasoning
- ▶ Discussion

ALC Concepts

ALC is the basic description logic (*Attributive Language with Complements*)

ALC concepts inductively defined from atomic concepts and roles:

- ▶ Every atomic concept is a concept
- ▶ \top and \perp are concepts
- ▶ If C is a concept, then $\neg C$ is a concept
- ▶ If C and D are concepts, then so are $C \sqcap D$ and $C \sqcup D$
- ▶ If C a concept and R a role, then $\forall R.C$ and $\exists R.C$ are concepts

Concepts describe sets of objects with certain common features:

$Woman \sqcap \exists hasChild.(\exists hasChild.Person)$

Women with a grandchild

$Disease \sqcap \forall Affects.Child$

Diseases affecting only children

$Person \sqcap \neg \exists owns.DetHouse$

People not owning a detached house

$Man \sqcap \exists hasChild.\top \sqcap \forall hasChild.Man$

Fathers having only sons

Very useful idea for Knowledge Representation!

General Concept Inclusion Axioms

Recall our example formulas:

$$\forall x.(\text{JuvDis}(x) \rightarrow \forall y.(\text{Affects}(x, y) \rightarrow \text{Child}(y) \vee \text{Teen}(y)))$$

$$\forall x.(\text{Child}(x) \vee \text{Teen}(x) \rightarrow \neg \text{Adult}(x))$$

$$\forall x.(\text{Person}(x) \rightarrow \text{Child}(x) \vee \text{Teen}(x) \vee \text{Adult}(x))$$

$$\forall x.(\text{JuvArthritis}(x) \rightarrow \text{Arthritis}(x) \wedge \text{JuvDis}(x))$$

$$\forall x.(\text{Arthritis}(x) \rightarrow \exists y.(\text{Damages}(x, y) \wedge \text{Joint}(y)))$$

They are of the following form, with $\alpha_C(x)$ and $\alpha_D(x)$ corresponding to \mathcal{ALC} concepts C and D

$$\forall x.(\alpha_C(x) \rightarrow \alpha_D(x))$$

Such closed formulas (sentences) are \mathcal{ALC} **General Concept Inclusions (GCIs)**

$$C \sqsubseteq D$$

Where C and D are \mathcal{ALC} -concepts

General Concept Inclusion Axioms

$$\begin{aligned}
 \forall x.(\text{JuvDis}(x) \rightarrow \forall y.(\text{Affects}(x, y) \rightarrow \\
 \quad \text{Child}(y) \vee \text{Teen}(y))) &\rightsquigarrow \text{JuvDis} \sqsubseteq \forall \text{Affects}.(\text{Child} \sqcup \text{Teen}) \\
 \forall x.(\text{Child}(x) \vee \text{Teen}(x) \rightarrow \neg \text{Adult}(x)) &\rightsquigarrow \text{Child} \sqcup \text{Teen} \sqsubseteq \neg \text{Adult} \\
 \forall x.(\text{Person}(x) \rightarrow \text{Child}(x) \vee \\
 \quad \text{Teen}(x) \vee \text{Adult}(x)) &\rightsquigarrow \text{Person} \sqsubseteq \text{Child} \sqcup \text{Teen} \sqcup \text{Adult} \\
 \forall x.(\text{JuvArth}(x) \rightarrow \text{Arth}(x) \wedge \text{JuvDis}(x)) &\rightsquigarrow \text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis} \\
 \forall x.(\text{Arth}(x) \rightarrow \exists y.(\text{Damages}(x, y) \wedge \\
 \quad \text{Joint}(y))) &\rightsquigarrow \text{Arth} \sqsubseteq \exists \text{Damages}. \text{Joint}
 \end{aligned}$$

Why call $C \sqsubseteq D$ a concept inclusion axiom?

- ▶ Intuitively, every object belonging to C should belong also to D
- ▶ States that C is **more specific** than D

Terminological Statements

GCI's allow us to represent a **surprising variety of terminological statements**

- ▶ Sub-type statements

$$\forall x. (JuvArth(x) \rightarrow Arth(x)) \rightsquigarrow JuvArth \sqsubseteq Arth$$

- ▶ Full definitions:

$$\forall x. (JuvArth(x) \leftrightarrow Arth(x) \wedge JuvDis(x)) \rightsquigarrow JuvArth \sqsubseteq Arth \sqcap JuvDis \\ Arth \sqcap JuvDis \sqsubseteq JuvArth$$

- ▶ Disjointness statements:

$$\forall x. (Child(x) \rightarrow \neg Adult(x)) \rightsquigarrow Child \sqsubseteq \neg Adult$$

- ▶ Covering statements:

$$\forall x. (Person(x) \rightarrow Adult(x) \vee Child(x)) \rightsquigarrow Person \sqsubseteq Adult \sqcup Child$$

- ▶ Type restrictions:

$$\forall x. (\forall y. (Affects(x, y) \rightarrow Arth(x) \wedge Person(y))) \rightsquigarrow \exists Affects. \top \sqsubseteq Arth \\ \top \sqsubseteq \forall Affects. Person$$

Data Assertions

In description logics, we can also represent data:

| | |
|---------------------------|-------------------------------|
| $Child(JohnSmith)$ | John Smith is a child |
| $JuvenileArthritis(JRA)$ | JRA is a juvenile arthritis |
| $Affects(JRA, MaryJones)$ | Mary Jones is affected by JRA |

Usually **data assertions** correspond to first-order ground (variable-free) atoms

In ALC , we have two types of data assertions, for a, b constants:

| | | |
|-----------|--------------------|-------------------------|
| $C(a)$ | \rightsquigarrow | C is an ALC concept |
| $R(a, b)$ | \rightsquigarrow | R is an atomic role |

Examples of acceptable data assertions in ALC :

| | | |
|-------------------------------------|--------------------|--|
| $\exists hasChild. Teacher(John)$ | \rightsquigarrow | $\exists y. (hasChild(John, y) \wedge Teacher(y))$ |
| $HistorySt \sqcup ClassicsSt(John)$ | \rightsquigarrow | $HistorySt(John) \vee ClassicsSt(John)$ |

DL Knowledge Base: TBox + ABox

An *ALC* knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is composed of

- ▶ a **TBox** \mathcal{T} (Terminological component):
Finite set of GCIs
- ▶ an **ABox** \mathcal{A} (Assertional component):
Finite set of assertions

TBox:

$$\begin{aligned} JuvArthritis &\sqsubseteq Arthritis \sqcap JuvDisease \\ Arthritis \sqcap JuvDisease &\sqsubseteq JuvArthritis \\ Arthritis &\sqsubseteq \exists Damages.Joint \\ JuvDisease &\sqsubseteq \forall Affects.(Child \sqcup Teen) \\ Child \sqcup Teen &\sqsubseteq \neg Adult \end{aligned}$$

ABox:

$$\begin{aligned} &Child(JohnSmith) \\ &JuvArthritis(JRA) \\ &Affects(JRA, MaryJones) \\ &Child \sqcup Teen(MaryJones) \end{aligned}$$

Semantics via First-Order Translation

Semantics of \mathcal{ALC} can be defined **via translation into first-order logic**:

- ▶ Concepts translated as formulas with one free variable

$$\begin{array}{ll}
 \pi_x(A) = A(x) & \pi_y(A) = A(y) \\
 \pi_x(\neg C) = \neg \pi_x(C) & \pi_y(\neg C) = \neg \pi_y(C) \\
 \pi_x(C \sqcap D) = \pi_x(C) \wedge \pi_x(D) & \pi_y(C \sqcap D) = \pi_y(C) \wedge \pi_y(D) \\
 \pi_x(C \sqcup D) = \pi_x(C) \vee \pi_x(D) & \pi_y(C \sqcup D) = \pi_y(C) \vee \pi_y(D) \\
 \pi_x(\exists R.C) = \exists y.(R(x, y) \wedge \pi_y(C)) & \pi_y(\exists R.C) = \exists x.(R(y, x) \wedge \pi_x(C)) \\
 \pi_x(\forall R.C) = \forall y.(R(x, y) \rightarrow \pi_y(C)) & \pi_y(\forall R.C) = \forall x.(R(y, x) \rightarrow \pi_x(C))
 \end{array}$$

- ▶ GCIs and assertions translated as closed formulas

$$\begin{array}{l}
 \pi(C \sqsubseteq D) = \forall x.(\pi_x(C) \rightarrow \pi_x(D)) \\
 \pi(R(a, b)) = R(a, b) \\
 \pi(C(a)) = \pi_{x/a}(C)
 \end{array}$$

- ▶ TBoxes, ABoxes and KBs are translated in the obvious way

Semantics via First-Order Translation

Note that concept-forming operators are **not independent**:

$$\begin{aligned} \perp &\rightsquigarrow \neg \top \\ C \sqcup D &\rightsquigarrow \neg(\neg C \sqcap \neg D) \\ \forall R.C &\rightsquigarrow \neg(\exists R.\neg C) \end{aligned}$$

These equivalences can be proved using first-order semantics:

$$\begin{aligned} \pi_x(\neg \exists R.\neg C) &= \neg \exists y.(R(x, y) \wedge \neg \pi_y(C)) \\ &\equiv \forall y.(\neg(R(x, y) \wedge \neg \pi_y(C))) \\ &\equiv \forall y.(\neg R(x, y) \vee \pi_y(C)) \\ &\equiv \forall y.(R(x, y) \rightarrow \pi_y(C)) \\ &= \pi_x(\forall R.C) \end{aligned}$$

We can define syntax of \mathcal{ALC} using **only conjunction and negation operators and the existential role operator**

Direct (Model-Theoretic) Semantics

Direct semantics: An alternative (and convenient) way of specifying semantics

DL interpretation $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ is a first-order interpretation over the DL vocabulary:

- ▶ each constant a interpreted as an object $a^{\mathcal{I}} \in D$
- ▶ each atomic concept A interpreted as a set $A^{\mathcal{I}} \subseteq D$
- ▶ each atomic role R interpreted as a binary relation $R^{\mathcal{I}} \subseteq D \times D$

We specify a mechanism for interpreting concepts:

$$\begin{aligned}
 \top^{\mathcal{I}} &= D \\
 \perp^{\mathcal{I}} &= \emptyset \\
 (\neg C)^{\mathcal{I}} &= D \setminus C^{\mathcal{I}} \\
 (C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
 (C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
 (\exists R.C)^{\mathcal{I}} &= \{u \in D \mid \exists w \in D \text{ s.t. } \langle u, w \rangle \in R^{\mathcal{I}} \text{ and } w \in C^{\mathcal{I}}\} \\
 (\forall R.C)^{\mathcal{I}} &= \{u \in D \mid \forall w \in D, \langle u, w \rangle \in R^{\mathcal{I}} \text{ implies } w \in C^{\mathcal{I}}\}
 \end{aligned}$$

Direct (Model-Theoretic) Semantics

Consider the interpretation $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$

$$\begin{aligned} D &= \{u, v, w\} \\ \text{JuvDis}^{\mathcal{I}} &= \{u\} \\ \text{Child}^{\mathcal{I}} &= \{w\} \\ \text{Teen}^{\mathcal{I}} &= \emptyset \\ \text{Affects}^{\mathcal{I}} &= \{\langle u, w \rangle\} \end{aligned}$$

We can then interpret any concept as a subset of D:

$$\begin{aligned} (\text{JuvDis} \sqcap \text{Child})^{\mathcal{I}} &= \emptyset \\ (\text{Child} \sqcup \text{Teen})^{\mathcal{I}} &= \{w\} \\ (\exists \text{Affects} . (\text{Child} \sqcup \text{Teen}))^{\mathcal{I}} &= \{u\} \\ (\neg \text{Child})^{\mathcal{I}} &= \{u, v\} \\ (\forall \text{Affects} . \text{Teen})^{\mathcal{I}} &= \{v, w\} \end{aligned}$$

Direct (Model-Theoretic) Semantics

We can now determine whether \mathcal{I} is a **model of** ...

- ▶ A General Concept Inclusion Axiom $C \sqsubseteq D$:

$$\mathcal{I} \models (C \sqsubseteq D) \quad \text{iff} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

- ▶ An assertion $C(a)$:

$$\mathcal{I} \models C(a) \quad \text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$$

- ▶ An assertion $R(a, b)$:

$$\mathcal{I} \models R(a, b) \quad \text{iff} \quad \langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$$

- ▶ A TBox \mathcal{T} , ABox \mathcal{A} , and knowledge base:

$$\mathcal{I} \models \mathcal{T} \quad \text{iff} \quad \mathcal{I} \models \alpha \text{ for each } \alpha \in \mathcal{T}$$

$$\mathcal{I} \models \mathcal{A} \quad \text{iff} \quad \mathcal{I} \models \alpha \text{ for each } \alpha \in \mathcal{A}$$

$$\mathcal{I} \models \mathcal{K} \quad \text{iff} \quad \mathcal{I} \models \mathcal{T} \text{ and } \mathcal{I} \models \mathcal{A}$$

Direct (Model-Theoretic) Semantics

Consider our previous example interpretation:

$$\begin{aligned}
 D &= \{u, v, w\} & \text{Affects}^{\mathcal{I}} &= \{\langle u, w \rangle\} \\
 \text{JuvDis}^{\mathcal{I}} &= \{u\} & \text{Child}^{\mathcal{I}} &= \{w\} & \text{Teen}^{\mathcal{I}} &= \emptyset
 \end{aligned}$$

\mathcal{I} is a model of the following axioms:

$$\begin{aligned}
 \text{JuvDis} \sqsubseteq \exists \text{Affects}. \text{Child} &\rightsquigarrow \{u\} \subseteq \{u\} \\
 \text{Child} \sqsubseteq \neg \text{Teen} &\rightsquigarrow \{w\} \subseteq \{u, v, w\} \\
 \text{JuvDisease} \sqsubseteq \forall \text{Affects}. \text{Child} &\rightsquigarrow \{u\} \subseteq \{u, v, w\}
 \end{aligned}$$

However \mathcal{I} is not a model of the following axioms:

$$\begin{aligned}
 \text{JuvDis} \sqsubseteq \exists \text{Affects}. (\text{Child} \sqcap \text{Teen}) &\rightsquigarrow \{u\} \not\subseteq \emptyset \\
 \neg \text{Teen} \sqsubseteq \text{Child} &\rightsquigarrow \{u, v, w\} \not\subseteq \{w\} \\
 \exists \text{Affects}. \top \sqsubseteq \text{Teen} &\rightsquigarrow \{u\} \not\subseteq \emptyset
 \end{aligned}$$

Observations

- ▶ The 'square' syntax of DLs looks odd at first, but it is **less verbose** than that of first-order logic (and may even be **more intuitive** for engineers not biased towards theoretical CS)
- ▶ *ALC* (and other DLs) underlies **OWL ontology language** with its own 'serialisation' syntax using RDF triples
- ▶ **Modal Logic K** formulas are essentially *ALC* concepts with a **single role** *R*:

$$\diamond A \rightsquigarrow \exists R.A$$

$$\square A \rightsquigarrow \forall R.A$$

- ▶ **Other Modal Logics** also have corresponding DLs

Outline

- ▶ Motivation and Examples
- ▶ \mathcal{ALC} Syntax and Semantics
- ▶ Calculus for \mathcal{ALC} Terminological Reasoning
- ▶ Discussion

Ontology Design

Scenario: Ontology design

- ▶ We are building a **conceptual model** (a TBox) for our domain
- ▶ At this design stage we have not included the data (no ABox)

Our TBox should be

- ▶ **Error-free:**

No unintended logical consequences

- ▶ **Sufficiently detailed:**

Contain all relevant knowledge for our application

Ontology Design

$$\begin{aligned}
 JuvArthritis &\sqsubseteq Arthritis \sqcap JuvDisease \\
 JuvDisease &\sqsubseteq Disease \\
 Arthritis &\sqsubseteq \exists Damages.Joint \sqcap \forall Damages.Joint \\
 JuvDisease &\sqsubseteq \forall Affects.(Child \sqcup Teen) \\
 Child \sqcup Teen &\sqsubseteq \neg Adult \\
 Arthritis &\sqsubseteq \exists Affects.Adult \\
 Disease \sqcap \exists Damages.Joint &\sqsubseteq JointDisease
 \end{aligned}$$

This TBox contains modeling errors:

Juvenile arthritis is a kind of juvenile disease

Juvenile disease affects only children or teens, which are not adults

A juvenile arthritis cannot affect any adult

Juvenile arthritis is a kind of arthritis

Each arthritis affects some adult

Each juvenile arthritis affects some adult

Concept Satisfiability

What is the **impact of the error**?

All models \mathcal{I} of \mathcal{T} must be such that $JuvArthritis^{\mathcal{I}} = \emptyset$

A juvenile arthritis cannot exist!

We cannot add data concerning juvenile arthritis

Such errors can be detected by solving the following problem:

Concept satisfiability w.r.t. a TBox:

An instance is a pair $\langle C, \mathcal{T} \rangle$ with C a concept and \mathcal{T} a TBox.

The answer is **true** iff a model $\mathcal{I} \models \mathcal{T}$ exists such that $C^{\mathcal{I}} \neq \emptyset$.

In a first-order setting, C is satisfiable w.r.t. \mathcal{T} if and only if

$$\pi(\mathcal{T}) \wedge \exists x. (\pi_x(C)) \quad \text{is satisfiable}$$

Concept Subsumption

Parts of our arthritis TBox, however, **do conform to our intuitions**

$$JuvArthritis \sqsubseteq Arthritis \sqcap JuvDisease$$

$$JuvDisease \sqsubseteq Disease$$

$$Arthritis \sqsubseteq \exists Damages.Joint \sqcap \forall Damages.Joint$$

$$JuvDisease \sqsubseteq \forall Affects.(Child \sqcup Teen)$$

$$Child \sqcup Teen \sqsubseteq \neg Adult$$

$$Arthritis \sqsubseteq \exists Affects.Adult$$

$$Disease \sqcap \exists Damages.Joint \sqsubseteq JointDisease$$

Juvenile arthritis is a kind of juvenile disease

Juvenile disease is a kind of disease

Juvenile arthritis is a kind of disease

Juvenile arthritis is a kind of arthritis

Each arthritis damages some joint

Each juvenile arthritis affects some joint

Juvenile arthritis is a joint disease.

Concept Subsumption

We have discovered **new interesting information**

All models \mathcal{I} of \mathcal{T} must be such that $JuvArthritis^{\mathcal{I}} \subseteq JointDisease^{\mathcal{I}}$

Juvenile arthritis is a sub-type of joint disease

All instances of juvenile arthritis are also joint diseases

Such **implicit information** detectable by solving the following problem:

Concept subsumption w.r.t. a TBox:

An instance is a triple $\langle C, D, \mathcal{T} \rangle$ with C, D concepts, \mathcal{T} a TBox.

The answer is **true** iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for each $\mathcal{I} \models \mathcal{T}$ (written $\mathcal{T} \models C \sqsubseteq D$).

In the first-order setting, C is subsumed by D w.r.t. \mathcal{T} if and only if

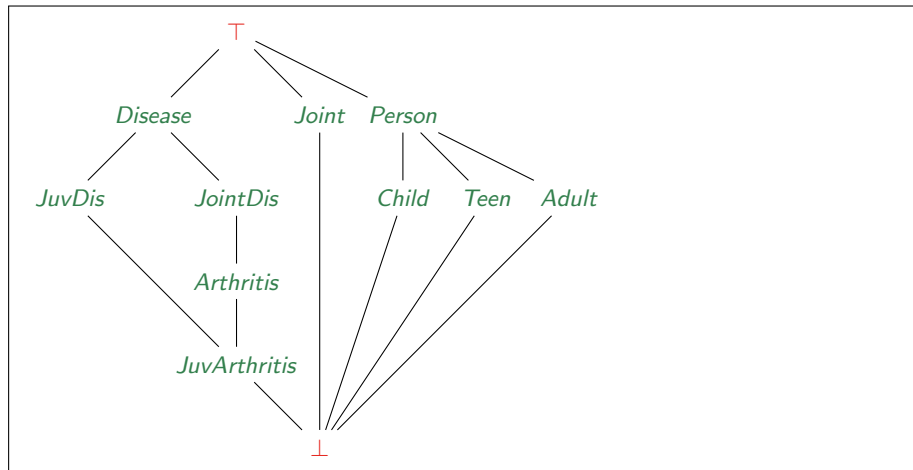
$$\pi(\mathcal{T}) \models \forall x. (\pi_x(C) \rightarrow \pi_x(D))$$

In the Modal Logic setting, subsumption is **local logical consequence**

TBox Classification

Problem of finding all subsumptions between atomic concepts in \mathcal{T}

Allows us to organise atomic concepts in a **subsumption hierarchy**



Reductions and Special Cases

In \mathcal{ALC} , concept subsumption **reducible** to concept satisfiability:

$$\mathcal{T} \models C \sqsubseteq D \quad \text{iff} \quad (C \sqcap \neg D) \text{ is unsatisfiable w.r.t. } \mathcal{T}$$

In \mathcal{ALC} , concept satisfiability is **reducible** to subsumption:

$$C \text{ satisfiable w.r.t. } \mathcal{T} \quad \text{iff} \quad \mathcal{T} \not\models (C \sqsubseteq \perp)$$

Interesting particular cases:

- ▶ $C \sqsubseteq \perp$ with $\mathcal{T} = \emptyset$: Can a concept be instantiated at all?
- ▶ $\top \sqsubseteq \perp$: Does \mathcal{T} have a model?

Validity, etc. can be defined and reduced in a similar way

We focus on algorithms for \mathcal{ALC} concept **subsumption** w.r.t. TBox

Sequent Calculus for \mathcal{ALC} Subsumption (Empty TBox)

We start with concept subsumption w.r.t. **empty TBox** ($\mathcal{T} = \emptyset$)

We can reuse the calculus for consequence for K (generalised to several roles)

- ▶ A **labelled formula** is a pair $u : A$ where u is a label and A a concept, an **accessibility formula** is uRv for two labels u, v and R a role
- ▶ Propositional rules for labelled formulas ('square' version): e.g.

$$\frac{\Gamma \Rightarrow u : A, \Delta \quad \Gamma \Rightarrow u : B, \Delta}{\Gamma \Rightarrow u : A \sqcap B, \Delta} \wedge\text{-right}$$

- ▶ The $\exists R$ -left rule, for each role R , creates a new label:

$$\frac{\Gamma, uRv, v : A \Rightarrow \Delta}{\Gamma, u : \exists R.A \Rightarrow \Delta} \exists R\text{-left} \quad \text{for a fresh label } v$$

- ▶ The $\forall R$ -left rule, for each role R , transfers info to other labels:

$$\frac{\Gamma, uRv, v : A, u : \forall R.A \Rightarrow \Delta}{\Gamma, uRv, u : \forall R.A \Rightarrow \Delta} \forall R\text{-left}$$

- ▶ Axioms for \top and \perp (or get rid of them using $A \sqcup \neg A$ for \top , etc.):

$$\frac{}{\Gamma, u : \perp \Rightarrow \Delta} \text{axiom} \qquad \frac{}{\Gamma \Rightarrow u : \top, \Delta} \text{axiom}$$

- ▶ The $\exists R$ - and $\forall R$ -right rules, other axioms: the same as for K

Sequent Calculus for \mathcal{ALC} Subsumption (Empty TBox)

- ▶ The calculi are **sound and complete**
- ▶ Termination **is** guaranteed
 - ▶ Proof by structural induction: along each branch, the formulas become simpler and simpler
 - ▶ May take quite long time (exponential, in fact PSpace-complete)
- ▶ A non-closed branch can be used for extracting counter-model
 - ▶ the domain is the set of labels, labelled formulas $u : A$ define concept interpretations, accessibility formulas uRv define role interpretations
 - ▶ this counter-model is always **finite** and **tree-shaped**
- ▶ **What about the general case with non-empty TBox?**

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

A TBox contains GCIs of the form $C \sqsubseteq D$

Each GCI equivalent to $\top \sqsubseteq \neg C \sqcup D$

We can 'compile' the whole TBox

$$\mathcal{T} = \{C_i \sqsubseteq D_i \mid 1 \leq i \leq n\}$$

into a single, equivalent GCI:

$$\top \sqsubseteq \prod_{1 \leq i \leq n} \neg C_i \sqcup D_i$$

Let's call $C_{\mathcal{T}}$ the concept on the right-hand side of this GCI

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

- ▶ Check concept subsumption $C \sqsubseteq D$ w.r.t. \mathcal{T}
- ▶ Intuitively, $C_{\mathcal{T}}$ should hold in all labels, so add $C_{\mathcal{T}}$ to Γ when creating new v
- ▶ The $\exists R$ -left rule w.r.t. \mathcal{T} , for each role R :

$$\frac{\Gamma, uRv, v : A, v : C_{\mathcal{T}} \Rightarrow \Delta}{\Gamma, u : \exists R.A \Rightarrow \Delta} \exists R\text{-left} \quad \text{for a fresh label } v$$

- ▶ The \forall -right rule w.r.t. \mathcal{T} , for each role R :

$$\frac{\Gamma, uRv, v : C_{\mathcal{T}} \Rightarrow v : A, \Delta}{\Gamma \Rightarrow u : \forall R.A, \Delta} \forall R\text{-right} \quad \text{for a fresh label } v$$

- ▶ Start with $1 : C_{\mathcal{T}} \Rightarrow 1 : \neg C \sqcup D$
- ▶ The rest as in the $\mathcal{T} = \emptyset$ case

- ▶ Soundness and completeness as before, but **termination is not guaranteed:**
no decrease in the formula size along branches

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

Example: $A \sqsubseteq \perp$ w.r.t. $\mathcal{T} = \{A \sqsubseteq \exists R.A\}$

Essentially, (un)satisfiability of concept A w.r.t. \mathcal{T}

$$\begin{array}{c}
 \frac{1 : A \Rightarrow 1 : A, 1 : \perp}{1 : \neg A, 1 : A \Rightarrow 1 : \perp} \quad \frac{\dots \quad \frac{1 : \exists R.A, 1 : A, 1R2, 2 : \exists R.A, 2 : A \Rightarrow 1 : \perp}{1 : \exists R.A, 1 : A, 1R2, 2 : \neg A \sqcup \exists R.A, 2 : A \Rightarrow 1 : \perp}}{1 : \exists R.A, 1 : A \Rightarrow 1 : \perp} \quad \dots}{1 : \neg A \sqcup \exists R.A, 1 : A \Rightarrow 1 : \perp} \quad \sqcup\text{-left} \\
 \frac{1 : \neg A \sqcup \exists R.A, 1 : A \Rightarrow 1 : \perp}{1 : \neg A \sqcup \exists R.A \Rightarrow 1 : \neg A, 1 : \perp} \quad \neg\text{-right} \\
 \frac{1 : \neg A \sqcup \exists R.A \Rightarrow 1 : \neg A, 1 : \perp}{1 : \neg A \sqcup \exists R.A \Rightarrow 1 : \neg A \sqcup \perp} \quad \sqcup\text{-right}
 \end{array}$$

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

Solution: Regain termination with **cycle detection**

Definition 3.1.

Label v' is **reachable** from label v in $\Gamma \Rightarrow \Delta$ if there are $v_0 R_1 v_1, \dots, v_{n-1} R_n v_n$ in Γ with $v' = v_0$ and $v = v_n$.

A label v' is **directly blocked** by a label v (in Γ and Δ) if

- ▶ v' is reachable from v
- ▶ $v : C \in \Gamma$ if and only if $v' : C \in \Gamma$, and $v : C \in \Delta$ if and only if $v' : C \in \Delta$ for every concept C .

A label v' is **blocked** if either

- ▶ it is directly blocked by some v or
- ▶ there exists a directly blocked v such that v' is reachable from v .

Restrict application of $\exists R$ -left and $\forall R$ -right rules to labels that are **not blocked**

Intuitively, a branch where everything is blocked is a **finite representation** of an **infinite** branch

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

Example: $A \sqsubseteq \perp$ w.r.t. $\mathcal{T} = \{A \sqsubseteq \exists R.A\}$

Essentially, (un)satisfiability of concept A w.r.t. \mathcal{T}

$$\begin{array}{c}
 \frac{}{1 : A \Rightarrow 1 : A, 1 : \perp} \\
 \frac{}{1 : \neg A, 1 : A \Rightarrow 1 : \perp} \\
 \hline
 \frac{}{1 : \neg A \sqcup \exists R.A, 1 : A \Rightarrow 1 : \perp} \quad \neg\text{-right} \\
 \frac{}{1 : \neg A \sqcup \exists R.A \Rightarrow 1 : \neg A, 1 : \perp} \quad \sqcup\text{-right} \\
 \hline
 \frac{}{1 : \neg A \sqcup \exists R.A \Rightarrow 1 : \neg A \sqcup \perp} \\
 \hline
 \frac{}{1 : \neg A \sqcup \exists R.A, 1 : A \Rightarrow 1 : \perp} \quad \sqcup\text{-left} \\
 \frac{}{1 : \exists R.A, 1 : A, 1R2, 2 : \neg A \sqcup \exists R.A, 2 : A \Rightarrow 1 : \perp} \\
 \frac{}{1 : \exists R.A, 1 : A, 1R2, 2 : \exists R.A, 2 : A \Rightarrow 1 : \perp} \\
 \hline
 \dots \quad \frac{}{1 : \exists R.A, 1 : A, 1R2, 2 : \exists R.A, 2 : A \Rightarrow 1 : \perp}
 \end{array}$$

Label 2 is directly blocked by label 1

Label 2 is blocked

$\exists R$ -left does not apply to 2

Other rules can apply, and even can 'unblock' $\exists R$ -left for 2!

Sequent Calculus for \mathcal{ALC} Subsumption (with TBox)

Theorem 3.1.

Calculus for \mathcal{ALC} subsumption with blocking is sound, complete and *terminating*.

Proof idea.

- ▶ Soundness as before
- ▶ Completeness since every block can be ‘infinitely unrolled’ to a counter-model
- ▶ Termination is guaranteed since there are finite number of (sets of) labelled formulae □

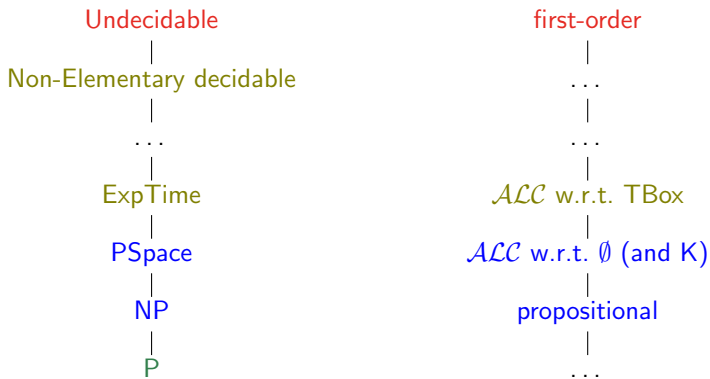
Corollary: reasoning in \mathcal{ALC} is *decidable* (in fact ExpTime-complete)

Observation: The ‘unrolled’ counter-model is *tree-shaped* (but may be infinite)

A general reason for decidability

Comment: adding ABox (assertions as $A(a)$, $R(a, b)$) does not change anything conceptually

The Picture



Outline

- ▶ Motivation and Examples
- ▶ \mathcal{ALC} Syntax and Semantics
- ▶ Calculus for \mathcal{ALC} Terminological Reasoning
- ▶ Discussion

Other Description Logics

- ▶ This was \mathcal{ALC} , the *Attributive Language with Complements*.
- ▶ The \mathcal{C} actually denotes an extension of a more restrictive language \mathcal{AL} .
- ▶ In a similar way, we have the following possible extensions of our logic:
 - ▶ \mathcal{H} : Role hierarchies;
 - ▶ \mathcal{R} : Complex role hierarchies;
 - ▶ \mathcal{N} : Cardinality restrictions;
 - ▶ \mathcal{Q} : Qualified cardinality restrictions;
 - ▶ \mathcal{O} : Closed classes;
 - ▶ \mathcal{I} : Inverse roles;
 - ▶ ...
- ▶ We name the languages by adding the letters of the features to \mathcal{ALC} . So e.g. \mathcal{ALCN} is \mathcal{ALC} extended with cardinality restrictions and \mathcal{ALCHI} is \mathcal{ALC} extended with role hierarchies and inverse roles.
- ▶ It is common to shorten \mathcal{ALC} (extended with transitive roles) to just \mathcal{S} for more advanced languages, so e.g. \mathcal{SHOIN} is $\mathcal{ALC} + \mathcal{H} + \mathcal{O} + \mathcal{I} + \mathcal{N}$.

Description Logic Applications

- ▶ Description logics are decidable
- ▶ Can be used to describe large vocabularies ($>100\,000$ concepts)
- ▶ E.g., in medicine, engineering, ...
- ▶ Reasoning helps to find mistakes when authoring
- ▶ Can be used in domain modelling, data integration, etc.

- ▶ Interested? Take IN3060/IN4060 – Semantic Technologies next semester!