TECHNISCHE UNIVERSITÄT DRESDEN

Computational Logic Group

Hannes Strass (based on slides by Bernardo Cuenca Grau, Ian Horrocks, Przemysław Wałęga)

Faculty of Computer Science, Institute of Artificial Intelligence, Computational Logic Group

Description Logics – Syntax and Semantics II

Lecture 5, 14th Nov 2022 // Foundations of Knowledge Representation, WS 2022/23

$\mathcal{ALC}$ Concepts

$\mathcal{ALC}$ is the basic description logic

$\mathcal{ALC}$ concepts C are inductively defined from atomic concepts A and roles R:

$$C ::= \top \mid \bot \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

$\mathcal{ALC}$ Concepts

$\mathcal{ALC}$ is the basic description logic

$\mathcal{ALC}$ concepts C are inductively defined from atomic concepts A and roles R:

$$C ::= \top \mid \bot \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

Semantics given through DL interpretations $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ with

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

$\mathcal{ALC}$ Fragments

What happens to $\mathcal{ALC}$ if we disallow negation? That is, if we define "$\mathcal{ALC}^+$" via

$$C ::= \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

$\mathcal{ALC}$ Fragments

What happens to $\mathcal{ALC}$ if we disallow negation? That is, if we define "$\mathcal{ALC}^{+}$" via

$$C ::= \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

Nothing:
Instead of $\neg C$, we can use A_C for a new concept name A_C and add the GCIs

$$\top \sqsubseteq C \sqcup A_C$$
$$C \sqcap A_C \sqsubseteq \bot$$

$\mathcal{ALC}$ Fragments

What happens to $\mathcal{ALC}$ if we disallow negation? That is, if we define "$\mathcal{ALC}^+$" via

$$C ::= \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

Nothing:
Instead of $\neg C$, we can use A_C for a new concept name A_C and add the GCIs

$$\top \sqsubseteq C \sqcup A_C$$
$$C \sqcap A_C \sqsubseteq \bot$$

What happens if we disallow negation, disjunction, and value restriction?

$\mathcal{ALC}$ Fragments

What happens to $\mathcal{ALC}$ if we disallow negation? That is, if we define "$\mathcal{ALC}^{+}$" via

$$C ::= \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

Nothing:
Instead of $\neg C$, we can use A_C for a new concept name A_C and add the GCIs

$$\top \sqsubseteq C \sqcup A_C$$
$$C \sqcap A_C \sqsubseteq \bot$$

What happens if we disallow negation, disjunction, and value restriction?
A lot – complexity (of concept satisfiability) drops from PSpace to PTime.

It is an important objective of DL (indeed KR) research to identify fragments that are "computationally well-behaved".

Basic Reasoning Problems and Services

What kinds of reasoning problems and services might be interesting?

Scenario: Ontology design

- We are building a conceptual model (a TBox) for our domain
- At this design stage we haven't yet 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

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

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 arthitis
  - 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 $\mathit{JuvArthritis}^{\mathcal{I}} = \emptyset$
  - 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 FOL setting, C is satisfiable w.r.t. $\mathcal{T}$ if and only if

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

Concept Subsumption

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

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

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 arthitis

Each arthritis damages some joint

Each juvenile arthritis damages 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 $\mathit{JuvArthritis}^{\mathcal{I}} \subseteq \mathit{JointDisease}^{\mathcal{I}}$
  - Juvenile arthritis is a sub-type of joint disease
    - All instances of juvenile arthitis are also joint diseases

Such implicit information is 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 $\mathcal{T} \models C \sqsubseteq D$).

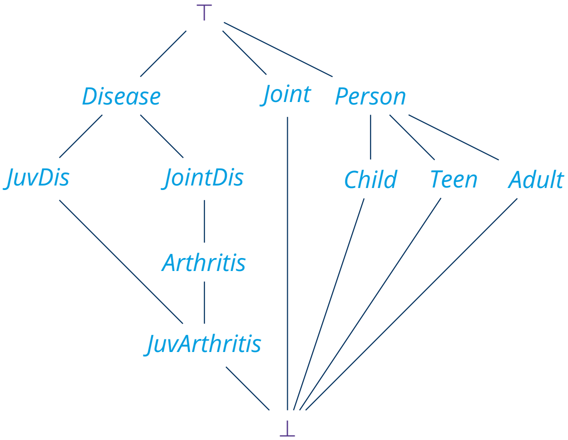
In a FOL setting, C is subsumed by D w.r.t. $\mathcal{T}$ if and only if

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

TBox Classification

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

Allows us to organise atomic concepts in a subsumption hierarchy:



Knowledge Base Reasoning

TBox:

$$JuvArthritis \sqsubseteq Arthritis \sqcap JuvDisease$$
$$JuvDisease \sqsubseteq Disease$$
$$Arthritis \sqsubseteq \exists Damages.Joint \sqcap \forall Damages.Joint$$
$$JuvDisease \sqsubseteq \forall Affects.(Child \sqcup Teen)$$
$$Child \sqcup Teen \sqsubseteq \neg Adult$$
$$Disease \sqcap \exists Damages.Joint \sqsubseteq JointDisease$$

ABox:

$$JuvArthritis(JRA)$$
$$Affects(JRA, MaryJones)$$
$$Disease(D)$$
$$Joint(J)$$
$$Damages(D, J)$$
$$\neg Teen(MaryJones)$$

May want to answer questions about individuals and/or KB as a whole:

- Is KB (TBox + ABox) consistent, i.e., does there exist a model?
  - What if we add $\neg JointDisease(JRA)$?
- Can we infer additional information about individuals?
  - Is D an instance of any class other than $Disease$?
  - Do we know if $MaryJones$ is an $Adult$ or a $Child$?

Summary of Basic Reasoning Problems

Definition

Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an $\mathcal{ALC}$ knowledge base, C, D possibly compound $\mathcal{ALC}$ concepts, and b an individual name. We say that

1. C is satisfiable with respect to $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ and some $d \in \Delta^{\mathcal{I}}$ with $d \in C^{\mathcal{I}}$;
2. C is subsumed by D with respect to $\mathcal{T}$, written $\mathcal{T} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$;
3. C and D are equivalent with respect to $\mathcal{T}$, written $\mathcal{T} \models C \equiv D$, if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$;
4. $\mathcal{K}$ is consistent if there exists a model of $\mathcal{K}$;
5. b is an instance of C with respect to $\mathcal{K}$, written $\mathcal{K} \models b : C$, if $b^{\mathcal{I}} \in C^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{K}$.

We write $C \sqsubseteq_{\mathcal{T}} D$ for $\mathcal{T} \models C \sqsubseteq D$ and $C \equiv_{\mathcal{T}} D$ for $\mathcal{T} \models C \equiv D$.

Important Properties of Subsumption

Lemma

Let C, D and E be concepts, b an individual name, and $(\mathcal{T}, \mathcal{A})$, $(\mathcal{T}', \mathcal{A}')$ knowledge bases with $\mathcal{T} \subseteq \mathcal{T}'$ and $\mathcal{A} \subseteq \mathcal{A}'$.

1. $C \sqsubseteq_{\mathcal{T}} C$.
2. If $C \sqsubseteq_{\mathcal{T}} D$ and $D \sqsubseteq_{\mathcal{T}} E$, then $C \sqsubseteq_{\mathcal{T}} E$.
3. If b is an instance of C with respect to $(\mathcal{T}, \mathcal{A})$ and $C \sqsubseteq_{\mathcal{T}} D$, then b is an instance of D with respect to $(\mathcal{T}, \mathcal{A})$.
4. If $\mathcal{T} \models C \sqsubseteq D$ then $\mathcal{T}' \models C \sqsubseteq D$.
5. If $\mathcal{T} \models C \equiv D$ then $\mathcal{T}' \models C \equiv D$.
6. If $(\mathcal{T}, \mathcal{A}) \models b : E$ then $(\mathcal{T}', \mathcal{A}') \models b : E$.

Proofs follow easily from semantics

Reasoning Problem Reductions

Theorem

Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an $\mathcal{ALC}$ knowledge base, C, D possibly compound $\mathcal{ALC}$ concepts and b an individual name.

1. $C \equiv_{\mathcal{T}} D$ if and only if $C \sqsubseteq_{\mathcal{T}} D$ and $D \sqsubseteq_{\mathcal{T}} C$.
2. $C \sqsubseteq_{\mathcal{T}} D$ if and only if $C \sqcap \neg D$ is not satisfiable with respect to $\mathcal{T}$.
3. C is satisfiable with respect to $\mathcal{T}$ if and only if $C \not\sqsubseteq_{\mathcal{T}} \bot$.
4. C is satisfiable with respect to $\mathcal{T}$ if and only if $(\mathcal{T}, \{b : C\})$ is consistent.
5. $(\mathcal{T}, \mathcal{A}) \models b : C$ if and only if $(\mathcal{T}, \mathcal{A} \cup \{b : \neg C\})$ is *not* consistent.

Consequently, all the previously mentioned reasoning problems can be reduced to KB (in)consistency.

Basic Reasoning Services

Correspond one-to-one with basic reasoning problems:

1. Given a TBox $\mathcal{T}$ and a concept C, check whether C is *satisfiable* with respect to $\mathcal{T}$.
2. Given a TBox $\mathcal{T}$ and two concepts C and D, check whether C is *subsumed by* D with respect to $\mathcal{T}$.
3. Given a TBox $\mathcal{T}$ and two concepts C and D, check whether C and D are *equivalent* with respect to $\mathcal{T}$.
4. Given a knowledge base $(\mathcal{T}, \mathcal{A})$, check whether $(\mathcal{T}, \mathcal{A})$ is *consistent*.
5. Given a knowledge base $(\mathcal{T}, \mathcal{A})$, an individual name a, and a concept C, check whether a is an *instance of* C w.r.t. $(\mathcal{T}, \mathcal{A})$.

All can be realised via KB consistency checks, e.g.:

$$(\mathcal{T}, \mathcal{A}) \models C \sqsubseteq D \quad \text{iff} \qquad\qquad \text{is not consistent}$$

for a an individual name not occurring in $\mathcal{A}$.

Basic Reasoning Services

Correspond one-to-one with basic reasoning problems:

1. Given a TBox $\mathcal{T}$ and a concept C, check whether C is *satisfiable* with respect to $\mathcal{T}$.
2. Given a TBox $\mathcal{T}$ and two concepts C and D, check whether C is *subsumed by* D with respect to $\mathcal{T}$.
3. Given a TBox $\mathcal{T}$ and two concepts C and D, check whether C and D are *equivalent* with respect to $\mathcal{T}$.
4. Given a knowledge base $(\mathcal{T}, \mathcal{A})$, check whether $(\mathcal{T}, \mathcal{A})$ is *consistent*.
5. Given a knowledge base $(\mathcal{T}, \mathcal{A})$, an individual name a, and a concept C, check whether a is an *instance of* C w.r.t. $(\mathcal{T}, \mathcal{A})$.

All can be realised via KB consistency checks, e.g.:

$$(\mathcal{T}, \mathcal{A}) \models C \sqsubseteq D \quad \text{iff} \quad (\mathcal{T}, \mathcal{A} \cup \{a : (C \sqcap \neg D)\}) \text{ is not consistent}$$

for a an individual name not occurring in $\mathcal{A}$.

Additional Reasoning Services

We can define additional reasoning services in terms of basic ones:

- Classification of a TBox: given a TBox $\mathcal{T}$, compute the *subsumption hierarchy* of all concept names occurring in $\mathcal{T}$.
  That is, for each pair A, B of concept names occurring in $\mathcal{T}$, check if $\mathcal{T} \models A \sqsubseteq B$ and if $\mathcal{T} \models B \sqsubseteq A$.
- Checking the satisfiability of concepts in $\mathcal{T}$: given a TBox $\mathcal{T}$, for each concept name A in $\mathcal{T}$, test if $\mathcal{T} \not\models A \sqsubseteq \bot$.
- Instance retrieval: given a concept C and a knowledge base $\mathcal{K}$, return all those individual names b such that b is an instance of C with respect to $\mathcal{K}$.
  That is, for each individual name b occurring in $\mathcal{K}$, check if $\mathcal{T} \models b : C$.
- Realisation of an individual name: given an individual name b and a knowledge base $\mathcal{K}$, return all those concept names A such that b is an instance of A with respect to $\mathcal{K}$. That is, for each concept name A occurring in $\mathcal{K}$, check if $\mathcal{T} \models b : A$.

Extensions: Inverse Roles

We might imagine that adding:

$$\mathit{Adult}(\mathit{JohnSmith}) \qquad \mathit{AffectedBy}(\mathit{JohnSmith}, \mathit{JRA})$$

would lead to an inconsistency.

However, this is not the case, because there is no semantic relationship between *Affects* and *AffectedBy*.

In order to relate roles such as Affects and AffectedBy in the desired way, DLs can be extended with inverse roles.

The fact that a DL provides inverse roles is normally indicated by the letter $\mathcal{I}$ in its name, e.g., $\mathcal{ALCI}$.

We will use $\mathcal{L}$ as a placeholder for the name of a DL and write $\mathcal{LI}$ for $\mathcal{L}$ extended with inverse roles.

Extensions: Inverse Roles

Definition

Let $\mathbf{R}$ be the set of role names. For $R \in \mathbf{R}$, R^{-} is an inverse role. The set of $\mathcal{I}$ roles is $\mathbf{R} \cup \{R^{-} \mid R \in \mathbf{R}\}$.

Let $\mathcal{L}$ be a description logic. The set of $\mathcal{LI}$ concepts is the smallest set of concepts that contains all $\mathcal{L}$ concepts and where $\mathcal{I}$ roles can occur in all places of role names.

An interpretation $\mathcal{I}$ maps inverse roles to binary relations as follows:

$$(r^{-})^{\mathcal{I}} = \{(y, x) \mid (x, y) \in r^{\mathcal{I}}\}$$

Typically, DLs supporting inverse roles also allow for inverse roles to be used in axioms such as the following:

$$\mathit{AffectedBy} \equiv \mathit{Affects}^{-}$$

which establishes the intuitive semantic relationship.

Extensions: Number Restrictions

We might want to state that *MildArthritis* *Affects* at most 2 *Joint*s, or that *SevereArthritis* *Affects* at least 5 *Joint*s.

In order to support this, DLs can be extended with (qualified) number restrictions, usually indicated by $\mathcal{N}$ for NRs and $\mathcal{Q}$ for QNRs.

NRs are concept descriptions whose instances are related to at least/most n other individuals via a given role; e.g., $(\leqslant 2\, \mathit{sister})$ describes individuals having at most 2 sisters.

QNRs additionally allow for restricting the type of the target individuals; e.g., $(\geqslant 2\, \mathit{sister}.\mathit{Graduate})$ describes individuals having at least 2 sisters who are graduates.

Note that an NR is equivalent to a QNR where the restriction concept is $\top$; e.g., $(\leqslant 2\, \mathit{sister})$ is equivalent to $(\leqslant 2\, \mathit{sister}.\top)$.

Extensions: Number Restrictions

Definition

For n a non-negative number, r an $\mathcal{L}$ role and C a (possibly compound) $\mathcal{L}$ concept description, a number restriction is a concept description of the form $(\leqslant n\, r)$ or $(\geqslant n\, r)$, and a qualified number restriction is a concept description of the form $(\leqslant n\, r.C)$ or $(\geqslant n\, r.C)$, where C is the qualifying concept.

For an interpretation $\mathcal{I}$, its mapping $\cdot^{\mathcal{I}}$ is extended as follows, where $\#M$ is used to denote the cardinality of a set M:

$$
\begin{aligned}
(\leqslant n\, r)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d,e) \in r^{\mathcal{I}}\} \leq n\},\\
(\geqslant n\, r)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d,e) \in r^{\mathcal{I}}\} \geq n\},\\
(\leqslant n\, r.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d,e) \in r^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\} \leq n\},\\
(\geqslant n\, r.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d,e) \in r^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\} \geq n\}.
\end{aligned}
$$

Concept descriptions $(=nr)$ and $(=n\, r.C)$ may be used as abbreviations for $(\leqslant n\, r) \sqcap (\geqslant n\, r)$ and $(\leqslant n\, r.C) \sqcap (\geqslant n\, r.C)$, respectively.

Extensions: Nominals

So far our use of individuals has been restricted to ABox axioms.

We may also want to use individuals in concept descriptions; e.g., to describe those individuals who are affected by some *Disease* that also affects the individual *JohnSmith*.

Intuitively, we might try the description

$$\exists \mathit{Affects}^{-}.(\mathit{Disease} \sqcap \exists \mathit{Affects}.\mathit{JohnSmith})$$

but this will not work, because in this context *JohnSmith* must be a concept.†

Nominals allow for the construction of a concept from an individual name; e.g.: $\{\mathit{JohnSmith}\}$ is the concept whose only instance is *JohnSmith*.

The fact that a DL provides nominals is normally indicated by the letter $\mathcal{O}$ in its name ($\mathcal{N}$ is already used for unqualified number restrictions).

† In fact this would be a syntax error if we use *JohnSmith* elsewhere as an individual (the set **C** of concept names and **I** of individual names must be disjoint).

Extensions: Nominals

Definition

Let $\mathbf{I}$ be the set of individual names. For $b \in \mathbf{I}$, $\{b\}$ is called a nominal.

Let $\mathcal{L}$ be a description logic. The description logic $\mathcal{LO}$ is obtained from $\mathcal{L}$ by allowing nominals as additional concepts.

For an interpretation $\mathcal{I}$, its mapping $\cdot^{\mathcal{I}}$ is extended as follows:

$$(\{a\})^{\mathcal{I}} = \{a^{\mathcal{I}}\}$$

- We can now form the desired concept description:

$$\exists \mathit{Affects}^{-}.(\mathit{Disease} \sqcap \exists \mathit{Affects}.\{\mathit{JohnSmith}\})$$

- With nominals, the separation between ABox and TBox is not meaningful:

$$\begin{aligned} C(a) &\equiv \{a\} \sqsubseteq C \\ R(a,b) &\equiv \{a\} \sqsubseteq \exists R.\{b\} \end{aligned}$$

Extensions: Role Hierarchies

We may want our KB to provide some structure for roles as well as concepts; e.g.: we may want to state that roles *brother* and *sister* are subsumed by the role *sibling*.

The fact that a DL provides such role inclusion axioms (RIAs) is normally indicated by the letter $\mathcal{H}$ in its name (there is a $\mathcal{H}$ierarchy of roles).

Definition

A *role inclusion axiom* (RIA) is an axiom of the form $r \sqsubseteq s$ for r, s $\mathcal{L}$ roles.

The DL $\mathcal{LH}$ is obtained from $\mathcal{L}$ by allowing, additionally, role inclusion axioms in TBoxes.

For an interpretation $\mathcal{I}$ to be a *model of* a role inclusion axiom $r \sqsubseteq s$, it has to satisfy

$$r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$$

Extensions: Transitive Roles

We can use the role *parent* to form descriptions such as:

$\exists parent.Irish$	having an *Irish* parent
$\exists parent.(\exists parent.Irish)$	having an *Irish* grandparent
$\exists parent.(\exists parent.(\exists parent.Irish))$	having an *Irish* greatgrandparent

But what if we want to mention *Irish* ancestors without specifying a generation?

We can do that by using a combination of role hierarchy and transitive roles:

$parent \sqsubseteq ancestor$	parent is a sub-role of ancestor
$\mathrm{Trans}(ancestor)$	ancestor is a transitive role
$\exists ancestor.Irish$	having an *Irish* ancestor

Extensions: Transitive Roles

Definition

A role transitivity axiom is an axiom of the form Trans(r) for r an $\mathcal{L}$ role.

The name of the DL that is the extension of $\mathcal{L}$ by allowing, additionally, transitivity axioms in TBoxes, is usually given by replacing $\mathcal{ALC}$ in $\mathcal{L}$'s name with $\mathcal{S}$.

For an interpretation $\mathcal{I}$ to be a model of a role transitivity axiom Trans(r), the relation $r^{\mathcal{I}}$ must be transitive.

- The use of $\mathcal{S}$ to replace $\mathcal{ALC}$ in DLs with transitive roles is inspired by similarities with the modal logic **S4** (and a desire for shorter names).
- However, in some cases (the subscript) $\cdot_{R^+}$ is used to indicate transitive roles; e.g., $\mathcal{SHIQ}$ could be written $\mathcal{ALCHIQ}_{R^+}$.

Extensions: Transitive Roles

It is important to understand the difference between transitive roles and the transitive closure of roles.

- Transitive closure is a role constructor: given a role r, transitive closure can be used to construct a role r^+, with the semantics being that $(r^+)^\mathcal{I} = (r^\mathcal{I})^+$.
- In a logic that includes both transitive roles and role inclusion axioms, e.g., $\mathcal{SH}$, adding axioms Trans(s) and $r \sqsubseteq s$ to a TBox $\mathcal{T}$ ensures that in every model $\mathcal{I}$ of $\mathcal{T}$, $s^\mathcal{I}$ is transitive, and $r^\mathcal{I} \subseteq s^\mathcal{I}$.
- However, we cannot enforce that s is the smallest such transitive role: s is just some transitive role that includes r.
- In contrast, the transitive closure r^+ of r is, by definition, the smallest transitive role that includes r; thus we have:

$$\{\mathrm{Trans}(s), r \sqsubseteq s\} \models r \sqsubseteq r^+ \sqsubseteq s.$$

Relationships to FOL Revisited

As we have seen, $\mathcal{ALC}$ is in the 2-variable fragment of FOL (FO^2):

$$\pi_x(A) = A(x) \qquad \pi_y(A) = A(y)$$
$$\pi_x(\neg C) = \neg \pi_x(C) \qquad \pi_y(\neg C) = \neg \pi_y(C)$$
$$\pi_x(C \sqcap D) = \pi_x(C) \land \pi_x(D) \qquad \pi_y(C \sqcap D) = \pi_y(C) \land \pi_y(D)$$
$$\pi_x(C \sqcup D) = \pi_x(C) \lor \pi_x(D) \qquad \pi_y(C \sqcup D) = \pi_y(C) \lor \pi_y(D)$$
$$\pi_x(\exists R.C) = \exists y.(R(x,y) \land \pi_y(C)) \qquad \pi_y(\exists R.C) = \exists x.(R(y,x) \land \pi_x(C))$$
$$\pi_x(\forall R.C) = \forall y.(R(x,y) \to \pi_y(C)) \qquad \pi_y(\forall R.C) = \forall x.(R(y,x) \to \pi_x(C))$$

$$\pi(C \sqsubseteq D) = \forall x.(\pi_x(C) \to \pi_x(D)) \qquad \pi(R(a,b)) = R(a,b) \qquad \pi(C(a)) = \pi_{x/a}(C)$$

FO^2 satisfiability is known to be decidable in NExpTime.
Moreover, the translation uses quantification only in a restricted way, and therefore yields formulas in the guarded fragment for which satisfiability is known to be decidable in deterministic exponential time.

Relationships to FOL Revisited

- Inverse roles can be captured easily in both the guarded and the two-variable fragments by simply swapping the variable places; e.g., $\pi_x(\exists r^-.C) = \exists y.(r(y,x) \wedge \pi_y(C))$.
- Number restrictions can be captured using (in)equality or so-called *counting quantifiers*; e.g., $\pi_x(\leqslant 2\, r.C) = \exists^{\leq 2} y.(r(x,y) \wedge \pi_y(C))$.
- It is known that the two-variable fragment with counting quantifiers (C^2) is still decidable in nondeterministic exponential time.
- Nominals can be captured using equality; e.g., $\pi_x(\{a\}) = (x = a)$.
- RIAs can also be captured in FO^2; e.g., $\pi(r \sqsubseteq s) = \forall x, y.(r(x,y) \rightarrow s(x,y))$.
- Transitive roles require three variables, and FO^3 is known to be undecidable; however, a satisfiability preserving transformation into FO^2 is still possible.
- This gives us a nondeterministic exponential time upper bound for $\mathcal{SHOIQ}$ satisfiability.

Relationships to Modal Logic

It is not hard to see that $\mathcal{ALC}$ concepts can be viewed as syntactic variants of formulae of multi-modal $\mathbf{K}_{(\mathbf{m})}$:

- Kripke structures can easily be viewed as DL interpretations, and vice versa;
- we can then view concept names as propositional variables, and role names as modal operators;
- we can realise this correspondence through the mapping π as follows:

$$
\begin{aligned}
\pi(A) &= A && \text{for concept names } A \\
\pi(C \sqcap D) &= \pi(C) \land \pi(D) \\
\pi(C \sqcup D) &= \pi(C) \lor \pi(D) \\
\pi(\neg C) &= \neg \pi(C) \\
\pi(\forall r.C) &= [r]\pi(C) \\
\pi(\exists r.C) &= \langle r \rangle \pi(C)
\end{aligned}
$$

Complexity

Undecidable	FOL-3 sat
|	|
Non-Elementary decidable	…
|	|
…	…
|	|
NExpTime	FOL-2, $\mathcal{SHOIQ}$ sat
|	|
ExpTime	Datalog, $\mathcal{SHOI}$ sat
|	|
PSpace	$\mathcal{ALC}$ sat
|	|
NP	PL sat
|	|
P	Horn PL sat

Conclusion

- For description logic knowledge bases, there are various relevant reasoning problems.
- All can be reduced to knowledge base (in)consistency.
- The basic description logic $\mathcal{ALC}$ can be extended in various ways:
  - Inverse Roles $\mathcal{I}$
  - (Qualified) Number Restrictions $(\mathcal{Q})\ \mathcal{N}$
  - Nominals $\mathcal{O}$
  - Role Hierarchies $\mathcal{H}$
  - Transitive Roles $\mathcal{ALC} \rightsquigarrow \mathcal{S}, \cdot_{R^+}$
- Description Logics have close connections with propositional modal logic ...
- ...and with the two-variable fragments of first-order logic (with counting quantifiers)