Knowledge Representation

Lecture 4: Description Logic Axioms, Reasoning and OWL

Patrick Koopmann (reusing some content originally by Stefan Borgwardt)

November 6, 2023

The Story so far...

Ontologies:

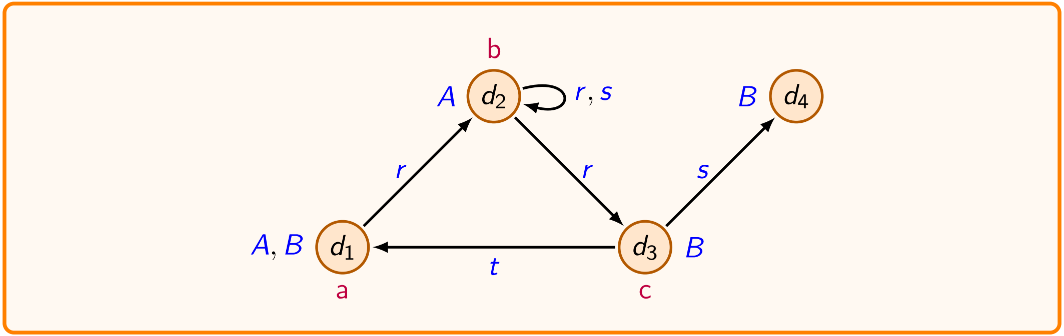
- Formalize conceptualizations
- Fix the meaning of terminology

Description Logics:

- Formalism for specifying ontologies
- The vocabulary consists of concept names, role names and individual names
- Interpretations fix their meaning
- $\mathcal{ALC}$ Concepts can be build using the operators $\top$, $\bot$, $\sqcap$, $\sqcup$, $\neg$, $\exists$, and $\forall$
- A DL ontology is a set of axioms, consisting of to parts:
  - TBox: terminological axioms (GCIs, equivalence axioms)
  - ABox: assertional axioms (concept and role assertions)

Exercise: Axioms

Let's get back to the interpretation $\mathcal{I}$ from last week:



Which of the following axioms does it satisfy:

1. $(a, b) : \; r$
2. $c : \; A$
3. $c : \; \exists s.B$
4. $A \sqsubseteq \exists r.A$
5. $A \sqsubseteq \forall r.(A \sqcup B)$
6. $A \equiv \forall r.(A \sqcup B)$
7. $\exists r.\top \sqsubseteq A$
8. $\exists r.\bot \sqsubseteq B$
9. $\exists r.A \sqsubseteq \forall s.A$

Ontologies

An ontology is a set $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where

- $\mathcal{A}$ is an ABox, a finite set of assertions,
- $\mathcal{T}$ is a TBox, a finite set of GCIs,

An interpretation is a model of $\mathcal{O}$ (written $\mathcal{I} \models \mathcal{O}$) if it is a model of all axioms in $\mathcal{O}$.

Ontologies

An ontology is a set $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where

- $\mathcal{A}$ is an ABox, a finite set of assertions,
- $\mathcal{T}$ is a TBox, a finite set of GCIs,

An interpretation is a model of $\mathcal{O}$ (written $\mathcal{I} \models \mathcal{O}$) if it is a model of all axioms in $\mathcal{O}$.

Ontologies

An ontology is a set $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where

- $\mathcal{A}$ is an ABox, a finite set of assertions,
- $\mathcal{T}$ is a TBox, a finite set of GCIs,

An interpretation is a model of $\mathcal{O}$ (written $\mathcal{I} \models \mathcal{O}$) if it is a model of all axioms in $\mathcal{O}$.

- The ABox contains facts about named individuals (data), the TBox contains (terminological) knowledge that applies to all individuals.

Example: An Ontology

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{\mathit{WN\text{-}KC137} : \ \mathit{Room}, \ (\mathit{WN\text{-}KC137}, \mathit{W\&N}) : \ \mathit{partOf}\}$

$$\mathcal{T} = \{\mathit{Room} \sqsubseteq \neg \mathit{University}, \ \mathit{Room} \sqsubseteq \exists \mathit{partOf}.\mathit{Building}, \\ \mathit{Building} \sqsubseteq \neg \mathit{University}, \ \mathit{Building} \sqsubseteq \neg \mathit{Room}\}$$

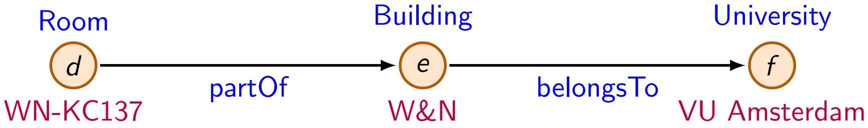
Example: An Ontology

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{\mathit{WN\text{-}KC137} : \ \mathit{Room}, \ (\mathit{WN\text{-}KC137}, \mathit{W\&N}) : \ \mathit{partOf}\}$

$\mathcal{T} = \{\mathit{Room} \sqsubseteq \neg \mathit{University}, \ \mathit{Room} \sqsubseteq \exists \mathit{partOf}.\mathit{Building},$
$\quad \mathit{Building} \sqsubseteq \neg \mathit{University}, \ \mathit{Building} \sqsubseteq \neg \mathit{Room}\}$

This ontology has many models, for example the following:



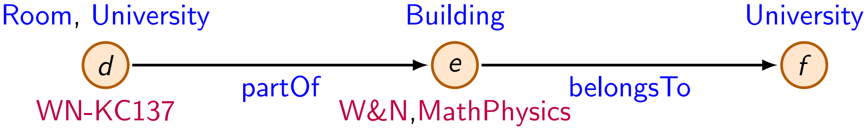
Example: An Ontology

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{\mathit{WN\text{-}KC137} : \mathit{Room},\ (\mathit{WN\text{-}KC137}, \mathit{W\&N}) : \mathit{partOf}\}$

$\mathcal{T} = \{\mathit{Room} \sqsubseteq \neg\mathit{University},\ \mathit{Room} \sqsubseteq \exists \mathit{partOf}.\mathit{Building},$
$\mathit{Building} \sqsubseteq \neg\mathit{University},\ \mathit{Building} \sqsubseteq \neg\mathit{Room}\}$

The following interpretation is not a model of the ontology:



Flashback: What is Knowledge Representation?

- KR as surrogate
- KR as expression of ontological commitment
- **KR as theory of intelligent reasoning**
  - How to deduce *implicit information*
  - What can be deduced? What should be deduced?
  - Foundation in logics one, but not the only possibility
- KR as medium for efficient computation
- KR as medium of human expression

Reasoning

Reasoning allows us to discover new insights from the knowledge represented in the ontology.

Reasoning

Reasoning allows us to discover new insights from the knowledge represented in the ontology.

The central reasoning task is entailment:

$\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if every model of $\mathcal{O}$ is also a model of α.

Reasoning

Reasoning allows us to discover new insights from the knowledge represented in the ontology.

The central reasoning task is entailment:

$\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if every model of $\mathcal{O}$ is also a model of α.

- We need to consider what all models have in common.
- This is the same as in propositional and first-order logic.

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$$\mathcal{A} = \{ \quad \mathit{drKoopmann} : \mathit{AssistantProfessor}, \qquad (\mathit{drKoopmann}, \mathit{KR}) : \mathit{teaches}, \\ (\mathit{mrsAbadi}, \mathit{KR}) : \mathit{attends} \quad \}$$

$$\mathcal{T} = \{ \quad \mathit{AssistantProfessor} \sqsubseteq \mathit{TeachingPersonal}, \qquad \exists \mathit{attends}.\mathit{Course} \sqsubseteq \mathit{Student}, \\ \mathit{TeachingPersonal} \sqsubseteq \forall \mathit{teaches}.\mathit{Course} \quad \}$$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$$\mathcal{A} = \{ \quad drKoopmann : \ AssistantProfessor, \qquad (drKoopmann, KR) : \ teaches, \\ (mrsAbadi, KR) : \ attends \quad \}$$

$$\mathcal{T} = \{ \quad AssistantProfessor \sqsubseteq TeachingPersonal, \qquad \exists attends.Course \sqsubseteq Student, \\ TeachingPersonal \sqsubseteq \forall teaches.Course \quad \}$$

This ontology entails:

1. $drKoopmann : \ \exists teaches.\top$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$$\mathcal{A} = \{ \quad \mathit{drKoopmann} : \mathit{AssistantProfessor}, \qquad (\mathit{drKoopmann}, \mathit{KR}) : \mathit{teaches}, \quad (\mathit{mrsAbadi}, \mathit{KR}) : \mathit{attends} \quad \}$$

$$\mathcal{T} = \{ \quad \mathit{AssistantProfessor} \sqsubseteq \mathit{TeachingPersonal}, \qquad \exists \mathit{attends}.\mathit{Course} \sqsubseteq \mathit{Student}, \quad \mathit{TeachingPersonal} \sqsubseteq \forall \mathit{teaches}.\mathit{Course} \quad \}$$

This ontology entails:

1. $\mathit{drKoopmann} : \exists \mathit{teaches}.\top$
2. $\mathit{drKoopmann} : \mathit{TeachingPersonal}$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{ \quad drKoopmann : \; AssistantProfessor, \qquad (drKoopmann, KR) : \; teaches,$
$\qquad (mrsAbadi, KR) : \; attends \quad \}$

$\mathcal{T} = \{ \quad AssistantProfessor \sqsubseteq TeachingPersonal, \qquad \exists attends.Course \sqsubseteq Student,$
$\qquad TeachingPersonal \sqsubseteq \forall teaches.Course \quad \}$

This ontology entails:

1. $drKoopmann : \; \exists teaches.\top$
2. $drKoopmann : \; TeachingPersonal$
3. $KR : \; Course$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$$\mathcal{A} = \{ \quad \mathit{drKoopmann} : \mathit{AssistantProfessor}, \qquad (\mathit{drKoopmann}, \mathit{KR}) : \mathit{teaches}, \\ (\mathit{mrsAbadi}, \mathit{KR}) : \mathit{attends} \quad \}$$

$$\mathcal{T} = \{ \quad \mathit{AssistantProfessor} \sqsubseteq \mathit{TeachingPersonal}, \qquad \exists \mathit{attends}.\mathit{Course} \sqsubseteq \mathit{Student}, \\ \mathit{TeachingPersonal} \sqsubseteq \forall \mathit{teaches}.\mathit{Course} \quad \}$$

This ontology entails:

1. $\mathit{drKoopmann} : \exists \mathit{teaches}.\top$
2. $\mathit{drKoopmann} : \mathit{TeachingPersonal}$
3. $\mathit{KR} : \mathit{Course}$
4. $\mathit{mrsAbadi} : \mathit{Student}$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{ \quad \mathit{drKoopmann} : \mathit{AssistantProfessor}, \quad (\mathit{drKoopmann}, \mathit{KR}) : \mathit{teaches},$
$\quad (\mathit{mrsAbadi}, \mathit{KR}) : \mathit{attends} \quad \}$

$\mathcal{T} = \{ \quad \mathit{AssistantProfessor} \sqsubseteq \mathit{TeachingPersonal}, \quad \exists \mathit{attends}.\mathit{Course} \sqsubseteq \mathit{Student},$
$\quad \mathit{TeachingPersonal} \sqsubseteq \forall \mathit{teaches}.\mathit{Course} \quad \}$

This ontology entails:

1. $\mathit{drKoopmann} : \exists \mathit{teaches}.\top$
2. $\mathit{drKoopmann} : \mathit{TeachingPersonal}$
3. $\mathit{KR} : \mathit{Course}$
4. $\mathit{mrsAbadi} : \mathit{Student}$
5. $\mathit{AssistantProfessor} \sqsubseteq \forall \mathit{teaches}.\mathit{Course}$

Example: Entailment

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$$\mathcal{A} = \{ \quad \mathit{drKoopmann} : \mathit{AssistantProfessor}, \qquad (\mathit{drKoopmann}, \mathit{KR}) : \mathit{teaches}, \\ (\mathit{mrsAbadi}, \mathit{KR}) : \mathit{attends} \quad \}$$

$$\mathcal{T} = \{ \quad \mathit{AssistantProfessor} \sqsubseteq \mathit{TeachingPersonal}, \qquad \exists \mathit{attends}.\mathit{Course} \sqsubseteq \mathit{Student}, \\ \mathit{TeachingPersonal} \sqsubseteq \forall \mathit{teaches}.\mathit{Course} \quad \}$$

It does not entail:

1. $\mathit{drKoopmann} : \neg \mathit{Student}$
2. $\mathit{mrsAbadi} : \neg \mathit{TeachingPersonal}$
3. $\mathit{AssistantProfessor} \sqsubseteq \exists \mathit{teaches}.\mathit{Course}$

Because in each case, we can build a model of the ontology in which the axiom is not satisfied.

Entailment is a quite general reasoning task.

With ontologies, we usually have more specific questions we want to answer.

Typical reasoning tasks are the following:

- Subsumption-relationships between concepts
- Individuals in a given concept
- Consistency and coherence of an ontology

Reasoning: Relationships between Concepts

Let C, D be concepts and $a \in \mathbf{I}$.

- If $\mathcal{O} \models C \sqsubseteq D$, we say that C is subsumed by D wrt. $\mathcal{O}$. $C \sqsubseteq_{\mathcal{O}} D$
- If $\mathcal{O} \models C \equiv D$, we say that C is equivalent to D wrt. $\mathcal{O}$. $C \equiv_{\mathcal{O}} D$
- If $C \sqsubseteq_{\mathcal{O}} D$ and $C \not\equiv_{\mathcal{O}} D$, C is strictly subsumed by D wrt. $\mathcal{O}$. $C \sqsubset_{\mathcal{O}} D$
- If $\mathcal{O} \models C \sqcap D \sqsubseteq \bot$, we say that C and D are disjoint wrt. $\mathcal{O}$.

Reasoning: Relationships between Concepts

Let C, D be concepts and $a \in \mathbf{I}$.

- If $\mathcal{O} \models C \sqsubseteq D$, we say that C is subsumed by D wrt. $\mathcal{O}$. $\quad C \sqsubseteq_{\mathcal{O}} D$
- If $\mathcal{O} \models C \equiv D$, we say that C is equivalent to D wrt. $\mathcal{O}$. $\quad C \equiv_{\mathcal{O}} D$
- If $C \sqsubseteq_{\mathcal{O}} D$ and $C \not\equiv_{\mathcal{O}} D$, C is strictly subsumed by D wrt. $\mathcal{O}$. $\quad C \sqsubset_{\mathcal{O}} D$
- If $\mathcal{O} \models C \sqcap D \sqsubseteq \bot$, we say that C and D are disjoint wrt. $\mathcal{O}$.

$$\{\mathit{AILecture} \sqsubseteq \mathit{Lecture},\ \mathit{Lecture} \sqsubseteq \mathit{Course}\} \models \mathit{AILecture} \sqsubseteq \mathit{Course}$$

$$\{\mathit{AILecture} \sqsubseteq \mathit{Lecture},\ \mathit{Lecture} \sqcap \mathit{Room} \sqsubseteq \bot\} \models \mathit{AILecture} \sqcap \mathit{Room} \sqsubseteq \bot$$

Reasoning: Relationships between Concepts

Let C, D be concepts and $a \in \mathbf{I}$.

- If $\mathcal{O} \models C \sqsubseteq D$, we say that C is subsumed by D wrt. $\mathcal{O}$. $C \sqsubseteq_{\mathcal{O}} D$
- If $\mathcal{O} \models C \equiv D$, we say that C is equivalent to D wrt. $\mathcal{O}$. $C \equiv_{\mathcal{O}} D$
- If $C \sqsubseteq_{\mathcal{O}} D$ and $C \not\equiv_{\mathcal{O}} D$, C is strictly subsumed by D wrt. $\mathcal{O}$. $C \sqsubset_{\mathcal{O}} D$
- If $\mathcal{O} \models C \sqcap D \sqsubseteq \bot$, we say that C and D are disjoint wrt. $\mathcal{O}$.

$$\{AILecture \sqsubseteq Lecture,\ Lecture \sqsubseteq Course\} \models AILecture \sqsubseteq Course$$

$$\{AILecture \sqsubseteq Lecture,\ Lecture \sqcap Room \sqsubseteq \bot\} \models AILecture \sqcap Room \sqsubseteq \bot$$

If $C \sqsubset D$, then C is more specific than D (w.r.t. $\mathcal{O}$), and D is more general than C (w.r.t. $\mathcal{O}$).

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

- An inconsistent ontology contains contradictory statements.

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

- An inconsistent ontology contains contradictory statements.
- An inconsistent ontology entails all axioms, even if they are nonsensical.

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

- An inconsistent ontology contains contradictory statements.
- An inconsistent ontology entails all axioms, even if they are nonsensical.

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

- An inconsistent ontology contains contradictory statements.
- An inconsistent ontology entails all axioms, even if they are nonsensical.

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Reasoning: Consistency

$\mathcal{O}$ is consistent if it has a model.

- An inconsistent ontology contains contradictory statements.
- An inconsistent ontology entails all axioms, even if they are nonsensical.

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

This means that checking consistency is a kind of (non-)entailment test.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

($\Leftarrow$) Assume $\mathcal{O} \models \top \sqsubseteq \bot$.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

($\Leftarrow$) Assume $\mathcal{O} \models \top \sqsubseteq \bot$. Then, for every model $\mathcal{I}$ of $\mathcal{O}$, $\mathcal{I} \models \top \sqsubseteq \bot$.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

($\Leftarrow$) Assume $\mathcal{O} \models \top \sqsubseteq \bot$. Then, for every model $\mathcal{I}$ of $\mathcal{O}$, $\mathcal{I} \models \top \sqsubseteq \bot$. This means, $\Delta^{\mathcal{I}} \subseteq \emptyset$.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

($\Leftarrow$) Assume $\mathcal{O} \models \top \sqsubseteq \bot$. Then, for every model $\mathcal{I}$ of $\mathcal{O}$, $\mathcal{I} \models \top \sqsubseteq \bot$. This means, $\Delta^{\mathcal{I}} \subseteq \emptyset$. By definition, $\Delta^{\mathcal{I}}$ is never empty in an interpretation.

Reasoning: Consistency

Theorem: If $\mathcal{O}$ is inconsistent, then $\mathcal{O} \models \alpha$ for any axiom α.

Proof: Assume $\mathcal{O}$ is inconsistent and α is an arbitrary axiom. $\mathcal{O}$ has no model. Therefore, there is no model $\mathcal{I}$ s.t. $\mathcal{I} \not\models \alpha$. This means $\mathcal{I} \models \alpha$ in every model of $\mathcal{O}$. $\square$

Theorem: An ontology $\mathcal{O}$ is inconsistent iff $\mathcal{O} \models \top \sqsubseteq \bot$.

Proof: ($\Rightarrow$) By the previous theorem, if $\mathcal{O}$ is inconsistent, $\mathcal{O} \models \top \sqsubseteq \bot$.

($\Leftarrow$) Assume $\mathcal{O} \models \top \sqsubseteq \bot$. Then, for every model $\mathcal{I}$ of $\mathcal{O}$, $\mathcal{I} \models \top \sqsubseteq \bot$. This means, $\Delta^{\mathcal{I}} \subseteq \emptyset$. By definition, $\Delta^{\mathcal{I}}$ is never empty in an interpretation. Therefore, there cannot be any model $\mathcal{I}$ of $\mathcal{O}$, and $\mathcal{O}$ must be inconsistent. $\square$

Other Reasoning Problems

- If $\mathcal{O} \models a : C$, then a is an instance of C w.r.t. $\mathcal{O}$.

Other Reasoning Problems

- If $\mathcal{O} \models a : C$, then a is an instance of C w.r.t. $\mathcal{O}$.
- If $\mathcal{O} \not\models C \sqsubseteq \bot$, then C is satisfiable w.r.t. $\mathcal{O}$.

- Some concepts are unsatisfiable w.r.t. any consistent ontology, e.g. $\exists r.\bot$.

Other Reasoning Problems

- If $\mathcal{O} \models a : C$, then a is an instance of C w.r.t. $\mathcal{O}$.
- If $\mathcal{O} \not\models C \sqsubseteq \bot$, then C is satisfiable w.r.t. $\mathcal{O}$.
- If all concept names in $\mathcal{O}$ are satisfiable w.r.t. $\mathcal{O}$, then $\mathcal{O}$ is coherent.

- Unsatisfiable concept names indicate an error in the ontology:
  Why use a concept name A if it can only be interpreted as $A^{\mathcal{I}} = \bot^{\mathcal{I}} = \emptyset$?
- Consistency and coherence are basic requirements for any ontology.

Other Reasoning Problems

- If $\mathcal{O} \models a : C$, then a is an instance of C w.r.t. $\mathcal{O}$.
- If $\mathcal{O} \not\models C \sqsubseteq \bot$, then C is satisfiable w.r.t. $\mathcal{O}$.
- If all concept names in $\mathcal{O}$ are satisfiable w.r.t. $\mathcal{O}$, then $\mathcal{O}$ is coherent.
- Classification is the task of computing all entailments of the form $\mathcal{O} \models A \sqsubseteq B$, where $A, B \in \mathbf{C}$.

Other Reasoning Problems

- If $\mathcal{O} \models a : C$, then a is an instance of C w.r.t. $\mathcal{O}$.
- If $\mathcal{O} \not\models C \sqsubseteq \bot$, then C is satisfiable w.r.t. $\mathcal{O}$.
- If all concept names in $\mathcal{O}$ are satisfiable w.r.t. $\mathcal{O}$, then $\mathcal{O}$ is coherent.
- Classification is the task of computing all entailments of the form $\mathcal{O} \models A \sqsubseteq B$, where $A, B \in \mathbf{C}$.
- Materialization is the task of computing all entailments of the form $\mathcal{O} \models a : A$ and $\mathcal{O} \models (a, b) : r$, where $a, b \in \mathbf{I}$, $A \in \mathbf{C}$, and $r \in \mathbf{R}$.

- Classification and materialization make explicit much of the knowledge that is implicitly given by the ontology.

Example: Consistency and Coherence

$$\{\textit{Felix} : \textit{Cat}, \quad \textit{Cat} \sqsubseteq \textit{Animal}, \quad (\textit{Felix}, \textit{Toby}) : \textit{hasFather}, \quad \exists \textit{hasFather}.\top \sqsubseteq \textit{Human}\}$$

Example: Consistency and Coherence

$$\{\mathit{Felix} : \mathit{Cat}, \quad \mathit{Cat} \sqsubseteq \mathit{Animal}, \quad (\mathit{Felix}, \mathit{Toby}) : \mathit{hasFather}, \quad \exists \mathit{hasFather}.\top \sqsubseteq \mathit{Human}\}$$

is consistent and coherent, and entails $\mathit{Felix} : \mathit{Human}$.

Example: Consistency and Coherence

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather, \\ \exists hasFather.\top \sqsubseteq Human\}$$

is consistent and coherent, and entails $Felix : Human$.

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather, \\ \exists hasFather.\top \sqsubseteq Human, \quad Human \sqcap Animal \sqsubseteq \bot\}$$

Example: Consistency and Coherence

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather, \\ \exists hasFather.\top \sqsubseteq Human\}$$

is consistent and coherent, and entails $Felix : Human$.

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather, \\ \exists hasFather.\top \sqsubseteq Human, \quad Human \sqcap Animal \sqsubseteq \bot\}$$

is inconsistent.

Example: Consistency and Coherence

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather,$$
$$\exists hasFather.\top \sqsubseteq Human\}$$

is consistent and coherent, and entails $Felix : Human$.

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather,$$
$$\exists hasFather.\top \sqsubseteq Human, \quad Human \sqcap Animal \sqsubseteq \bot\}$$

is inconsistent.

$$\{Human \sqcap Animal \sqsubseteq \bot, \quad Werewolf \sqsubseteq Human \sqcap Wolf, \quad Wolf \sqsubseteq Animal\}$$

Example: Consistency and Coherence

$$\{\mathit{Felix} : \mathit{Cat}, \quad \mathit{Cat} \sqsubseteq \mathit{Animal}, \quad (\mathit{Felix}, \mathit{Toby}) : \mathit{hasFather}, \\ \exists \mathit{hasFather}.\top \sqsubseteq \mathit{Human}\}$$

is consistent and coherent, and entails $\mathit{Felix} : \mathit{Human}$.

$$\{\mathit{Felix} : \mathit{Cat}, \quad \mathit{Cat} \sqsubseteq \mathit{Animal}, \quad (\mathit{Felix}, \mathit{Toby}) : \mathit{hasFather}, \\ \exists \mathit{hasFather}.\top \sqsubseteq \mathit{Human}, \quad \mathit{Human} \sqcap \mathit{Animal} \sqsubseteq \bot\}$$

is inconsistent.

$$\{\mathit{Human} \sqcap \mathit{Animal} \sqsubseteq \bot, \quad \mathit{Werewolf} \sqsubseteq \mathit{Human} \sqcap \mathit{Wolf}, \quad \mathit{Wolf} \sqsubseteq \mathit{Animal}\}$$

is consistent, but not coherent, because Werewolf is unsatisfiable.

Example: Consistency and Coherence

$$\{Felix : Cat, \quad Cat \sqsubseteq Animal, \quad (Felix, Toby) : hasFather, \quad \exists hasFather.\top \sqsubseteq Human, \quad Human \sqcap Animal \sqsubseteq \bot\}$$

$$\{Human \sqcap Animal \sqsubseteq \bot, \quad Werewolf \sqsubseteq Human \sqcap Wolf, \quad Wolf \sqsubseteq Animal\}$$

Disjointness axioms ($C \sqsubseteq \neg D$, or equivalently $C \sqcap D \sqsubseteq \bot$) are very useful for debugging ontologies, because they can expose hidden contradictions.

Last Exercise on Reasoning

$$\mathcal{O} = \{ \quad \begin{aligned} \mathit{Alive} &\sqsubseteq \mathit{Animal} \sqcup \mathit{Plant} \\ \mathit{Animal} &\sqsubseteq \exists \mathit{hasParent}.\mathit{Male} \sqcap \exists \mathit{hasParent}.\mathit{Female} \\ \mathit{thomas} &: \mathit{Alive} \\ \mathit{thomas} &: \forall \mathit{hasParent}.\bot \end{aligned} \quad \}$$

What can we say about Thomas?

Reasoning in Practice

- Later in this course, we will learn algorithms to perform these reasoning tasks.
- Until then, we will use special programs (reasoners) for that.
- But first, a few important aspects of DL ontologies.

DL Ontologies are not Programs

- There is no "execution order" of axioms
- There is no "direction" in which axioms are applied:

$$\mathcal{O} = \{ \quad A \sqsubseteq \exists r.B, \quad \exists r.B \sqsubseteq C, \\ a : A, \quad b : \neg C \quad \}$$

We have both $\mathcal{O} \models a : C$ and $\mathcal{O} \models b : \neg A$

Open World and Open Domain

DLs make the open-world assumption, i.e. facts that are not entailed are not necessarily false, but simply unknown.

$\{(\mathit{Bob}, \mathit{Fred}) : \mathit{hasChild}\}$ does not entail $\mathit{Bob} : \mathit{Father}$ or $\mathit{Bob} : \neg\mathit{Father}$.

Open World and Open Domain

DLs make the open-world assumption, i.e. facts that are not entailed are not necessarily false, but simply unknown.

$\{(\mathit{Bob}, \mathit{Fred}) : \mathit{hasChild}\}$ does not entail $\mathit{Bob} : \mathit{Father}$ or $\mathit{Bob} : \neg\mathit{Father}$.

They also make the open-domain assumption, i.e. there may be individuals we do not know, and that have no name.

$$\mathcal{O} = \{ \mathit{Human} \sqsubseteq \exists \mathit{hasParent}.\mathit{Mother}, \ \mathit{peter} : \mathit{Human}\}$$
$$\models \mathit{peter} : \exists \mathit{hasParent}.\mathit{Mother}$$
$$\not\models a : \mathit{Mother} \text{ for any } a \in \mathbf{I}$$

Binary vs. *n*-ary Relations

Description logics can only express unary and binary relations.

n-ary relations with $n \geq 3$ can be simulated, but not without loss of generality.

Binary vs. n-ary Relations

Description logics can only express unary and binary relations.

n-ary relations with $n \geq 3$ can be simulated, but not without loss of generality.

$hasDiagnosis(Bob, Flu, High)$

could be reformulated as

$Diagnosis(d112)$, $hasDiagnosis(Bob, d112)$,
$associatedDisease(d112, Flu)$, $associatedProbability(d112, High)$

This process is called reification.

Binary vs. n-ary Relations

Description logics can only express unary and binary relations.

n-ary relations with $n \geq 3$ can be simulated, but not without loss of generality.

$\mathit{hasDiagnosis}(\mathit{Bob}, \mathit{Flu}, \mathit{High})$

could be reformulated as

$\mathit{Diagnosis}(d112), \quad \mathit{hasDiagnosis}(\mathit{Bob}, d112),$
$\mathit{associatedDisease}(d112, \mathit{Flu}), \quad \mathit{associatedProbability}(d112, \mathit{High})$

This process is called reification.

This can lead to more complex axioms, because one has to refer to several roles.

$\exists \mathit{hasDiagnosis}.\exists \mathit{associatedDisease}.\mathit{InfectiousDisease} \sqsubseteq \mathit{Infectious}$

Ontologies in Practice: OWL

Flashback: What is Knowledge Representation?

- KR as surrogate
- KR as expression of ontological commitment
- KR as theory of intelligent reasoning
- KR as medium for efficient computation
- **KR as medium of human expression**
  - humans produce, consume and work with representations
  - representations must be human understandable, but also manageable by humans

Ontologies as Medium of Human Expression

- DL syntax convenient for teaching and research
  - Concise
  - Convenient to write with a pen
  - DLs are *logics*

Ontologies as Medium of Human Expression

- DL syntax convenient for teaching and research
  - Concise
  - Convenient to write with a pen
  - DLs are *logics*

- Requirements for real ontologies:
  - easily editable on a computer
  - machine-processable
  - manage large, complex ontologies
  - collaboration

Ontologies as Medium of Human Expression

- DL syntax convenient for teaching and research
  - Concise
  - Convenient to write with a pen
  - DLs are *logics*

- Requirements for real ontologies:
  - easily editable on a computer
  - machine-processable
  - manage large, complex ontologies
  - collaboration

$\Rightarrow$ Logic is not everything

Requirements for Ontology Languages

Some hard requirements are as in software engineering:

- documentation / comments
- organisation of ontologies into separate modules / files

Requirements for Ontology Languages

Some hard requirements are as in software engineering:

- documentation / comments
- organisation of ontologies into separate modules / files

Others are central to capturing *shared conceptualisations*

- unique naming of entities

Requirements for Ontology Languages

Some hard requirements are as in software engineering:

- documentation / comments
- organisation of ontologies into separate modules / files

Others are central to capturing *shared conceptualisations*

- unique naming of entities
  - Cow in Ontology 1 should refer to the same concept as in Ontology 2

Requirements for Ontology Languages

Some hard requirements are as in software engineering:

- documentation / comments
- organisation of ontologies into separate modules / files

Others are central to capturing *shared conceptualisations*

- unique naming of entities
  - Cow in Ontology 1 should refer to the same concept as in Ontology 2
- multi-language support
  - Cow means “cow” in English, “Kuh” in German, “vache” in French

Requirements for Ontology Languages

Some hard requirements are as in software engineering:

- documentation / comments
- organisation of ontologies into separate modules / files

Others are central to capturing *shared conceptualisations*

- unique naming of entities
  - Cow in Ontology 1 should refer to the same concept as in Ontology 2
- multi-language support
  - Cow means “cow” in English, “Kuh” in German, “vache” in French

Again others are important for authorship and trust

- Who created/modified an ontology/axiom?
- When was an axiom modified?

OWL 2: Overview

OWL 2 is a standardized ontology language based on XML and RDF that was developed for the Semantic Web, and is used in many applications.

`https://www.w3.org/TR/owl2-overview/`

- **W**eb **O**ntology **L**anguage
- OWL 1 was specified in 2004 by the World Wide Web Consortium (W3C)
- OWL 2 was specified in 2009 by the W3C
- The final standard encompasses different sublanguages
  - more on this later

OWL 2 Terminology

OWL 2 uses different terminology than description logics:

DL	$\rightsquigarrow$	OWL
name	$\rightsquigarrow$	entity
concept name	$\rightsquigarrow$	(named) class
role name	$\rightsquigarrow$	(named) object property
individual name	$\rightsquigarrow$	(named) individual
concept (description)	$\rightsquigarrow$	class expression
axiom	$\rightsquigarrow$	axiom

Syntaxes

OWL 2 ontologies can be written in different formats.

Functional Syntax (used in the OWL 2 specification and the OWL API)

```
SubClassOf( Lecture  Course )
```

RDF/XML Syntax (main interchange format)

```
<owl:Class rdf:about="Lecture">
  <rdfs:subClassOf rdf:resource="Course">
</owl:Class>
```

Syntaxes II

OWL/XML Syntax

```
<SubClassOf>
  <Class IRI="Lecture"><Class IRI="Course">
</SubClassOf>
```

Turtle Syntax

```
Lecture rdfs:subClassOf Course
```

Manchester Syntax (used by Protégé and in this lecture)

```
Class: Lecture
  SubClassOf: Course
```

IRIs

In OWL 2, the ontology and every entity (class, property, individual, datatype) is uniquely identified by an Internationalized Resource Identifier (IRI).

> http://kai.vu.nl/university-ontology#Lecture
> http://kai.vu.nl/university-ontology#Course

Ideally, IRIs should be dereferenceable, e.g. lead to a web page about the entity.

IRIs

In OWL 2, the ontology and every entity (class, property, individual, datatype) is uniquely identified by an Internationalized Resource Identifier (IRI).

http://kai.vu.nl/university-ontology#Lecture	uo:Lecture
http://kai.vu.nl/university-ontology#Course	uo:Course

Ideally, IRIs should be dereferenceable, e.g. lead to a web page about the entity.

Prefix definitions are used to abbreviate IRIs.

`Prefix:` uo: http://kai.vu.nl/university-ontology#

Complex Expressions in OWL Manchester Syntax

Let C, D, r, a be DL names or OWL entities, depending on the context.

DL syntax	Manchester syntax	Remark
$\top$	owl:Thing	(a special named class)
$\bot$	owl:Nothing	(a special named class)
$C \sqcap D$	C `and` D	
$C \sqcup D$	C `or` D	
$\neg C$	`not` C	
$\exists r.C$	r `some` C	
$\forall r.C$	r `only` C	
$C \sqsubseteq D$	C `SubClassOf` D	

Protégé

Protégé is a freely available IDE for developing OWL ontologies

The structure of an ontology, as it is displayed in Protégé, is a bit different to what we have seen so far:

- Usually, you do not directly enter axioms
- Rather, you add information to classes (concept names):
  - what are they equivalent to
  - what are they subsumed by
- Also, you have to explicitly specify your vocabulary before you can use it
  - every class, object property and individual has to be added before you can use it

Little Tour of Protégé

We look at some basics of Protégé:

- Adding classes and properties
- Adding logical information on classes
- Adding general class axioms
- Opening and saving OWL files
- Loading example ontologies
- Using a reasoner
- Querying complex concepts
- Explaining reasoning results

Accessing OWL from Software

There are various libraries to access and work with OWL ontologies and reasoners:

- OWL API: the most comprehensive available library, available for Java
  - Use of *software design patterns* might make this harder for less experienced software developers
- OwlReady2: a library for Python that allows to integrate OWL classes with Python classes
  - Optimized for specific use cases (querying data, reasoning, ...)
- DeepOnto: a Python library for ontology engineering with Deep learning
  - quite recent
- dl4python: Description Logic view on OWL
  - special development for this course
  - can be used from Python, Scala and Java
  - example file illustrates relevant functionalities

Some Advice on Creating Ontologies

Class Hierarchy

- Before adding complex axioms, first define the class hierarchy ($\sqsubseteq$-axioms).
- Flesh out the hierarchy with common superconcepts, missing siblings.

Organism

- Animal
  - Mammal
    - Cat
    - ...
  - Fish
    - Trout
    - ...
  - Carnivore
  - Herbivore
  - Omnivore
- Plant
  - Tree
  - Grass
  - Wheat

Class Hierarchy

- Before adding complex axioms, first define the class hierarchy ($\sqsubseteq$-axioms).
- Flesh out the hierarchy with common superconcepts, missing siblings.
- Ideally, much of this information was already elicited, otherwise we have to ask the domain experts again.

Organism

- Animal
  - Mammal
    - Cat
    - ...
  - Fish
    - Trout
    - ...
  - Carnivore
  - Herbivore
  - Omnivore
- Plant
  - Tree
  - Grass
  - Wheat

Class Hierarchy

- Before adding complex axioms, first define the class hierarchy ($\sqsubseteq$-axioms).
- Flesh out the hierarchy with common superconcepts, missing siblings.
- Ideally, much of this information was already elicited, otherwise we have to ask the domain experts again.
- Once the class hierarchy is fixed, we can add definitions.

Organism
- Animal
  - Mammal
    - Cat
    - ...
  - Fish
    - Trout
    - ...
  - Carnivore
  - Herbivore
  - Omnivore
- Plant
  - Tree
  - Grass
  - Wheat

Definitions

Identify which terms should be defined:

- Depends on the goals of the ontology
- General terms like "Organism" probably don't need a definition
- Some terms are easier to define than others, e.g. "Cat" vs. "Carnivore".
- For many terms, the information about their place in the class hierarchy is enough.

Definitions

Identify which terms should be defined:

- Depends on the goals of the ontology
- General terms like "Organism" probably don't need a definition
- Some terms are easier to define than others, e.g. "Cat" vs. "Carnivore".
- For many terms, the information about their place in the class hierarchy is enough.

Intensional definitions consist of the superclass(es) and any distinguishing characteristics.

> A cat is a mammal that has paws, legs, and a tail.
>
> A carnivore is an animal that eats only meat.
>
> A pet is a domesticated animal that lives with humans.

Definitions (II)

Distinguish between full definitions ($\equiv$) and partial definitions ($\sqsubseteq$)!

Definitions (II)

Distinguish between full definitions ($\equiv$) and partial definitions ($\sqsubseteq$)!

$$Carnivore \equiv Animal \sqcap \forall eats.Meat$$
$$Herbivore \equiv Animal \sqcap \forall eats.Plant$$
$$Pet \equiv Animal \sqcap \exists livesWith.Human$$
$$Animal \equiv Organism \sqcap \exists eats.Organism$$

$$Cat \sqsubseteq Mammal \sqcap \exists bodyPart.Paw \sqcap \exists bodyPart.Leg \sqcap \exists bodyPart.Tail$$
$$Cow \sqsubseteq Mammal \sqcap \forall eats.Grass$$

Definitions (II)

Distinguish between full definitions ($\equiv$) and partial definitions ($\sqsubseteq$)!

$$\mathit{Carnivore} \equiv \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Meat}$$
$$\mathit{Herbivore} \equiv \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Plant}$$
$$\mathit{Pet} \equiv \mathit{Animal} \sqcap \exists \mathit{livesWith}.\mathit{Human}$$
$$\mathit{Animal} \equiv \mathit{Organism} \sqcap \exists \mathit{eats}.\mathit{Organism}$$

$$\mathit{Cat} \sqsubseteq \mathit{Mammal} \sqcap \exists \mathit{bodyPart}.\mathit{Paw} \sqcap \exists \mathit{bodyPart}.\mathit{Leg} \sqcap \exists \mathit{bodyPart}.\mathit{Tail}$$
$$\mathit{Cow} \sqsubseteq \mathit{Mammal} \sqcap \forall \mathit{eats}.\mathit{Grass}$$

- Often, not everything can be fully defined, due to the restrictions of the ontology language.
- We will later see a more expressive DL, but this one will be restricted too.

Class Hierarchy (II)

In general, the class hierarchy is not a tree, but a directed acyclic graph with multiple inheritance.

$$Cow \sqsubseteq Mammal \qquad Cow \sqsubseteq Herbivore$$

(*Mammal* and *Herbivore* are unrelated)

Class Hierarchy (II)

In general, the class hierarchy is not a tree, but a directed acyclic graph with multiple inheritance.

$$\mathit{Cow} \sqsubseteq \mathit{Mammal} \qquad \mathit{Cow} \sqsubseteq \mathit{Herbivore}$$

(Mammal and $\mathit{Herbivore}$ are unrelated)

Instead of specifying all subclass-superclass relationships, it is easier to specify only a tree and let the reasoner infer the implicit ones.

$$\mathit{Grass} \sqsubseteq \mathit{Plant} \qquad \mathit{Mammal} \sqsubseteq \mathit{Animal}$$

$$\mathit{Herbivore} \equiv \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Plant}$$

$$\mathit{Cow} \sqsubseteq \mathit{Mammal} \sqcap \forall \mathit{eats}.\mathit{Grass}$$

This entails $\mathit{Cow} \sqsubseteq \mathit{Herbivore}$, so we do not have to explicitly add this axiom to the ontology.

Class Hierarchy (II)

In general, the class hierarchy is not a tree, but a directed acyclic graph with multiple inheritance.

$\mathit{Cow} \sqsubseteq \mathit{Mammal} \qquad \mathit{Cow} \sqsubseteq \mathit{Herbivore}$

(*Mammal* and *Herbivore* are unrelated)

Instead of specifying all subclass-superclass relationships, it is easier to specify only a tree and let the reasoner infer the implicit ones.

Definitions can affect the (inferred) class hierarchy.

Note: “Some” Does Not Mean “Only”

When writing definitions, it is not trivial to find the right one.

A common modeling error is to swap $\forall$ and $\exists$:

$$\mathit{Grass} \sqsubseteq \mathit{Plant}$$

$$\mathit{Herbivore} \equiv \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Plant}$$

$$\mathit{Cow} \sqsubseteq \mathit{Mammal} \sqcap \exists \mathit{eats}.\mathit{Grass}$$

Note: "Some" Does Not Mean "Only"

When writing definitions, it is not trivial to find the right one.

A common modeling error is to swap $\forall$ and $\exists$:

$$Grass \sqsubseteq Plant$$

$$Herbivore \equiv Animal \sqcap \forall eats.Plant$$

$$Cow \sqsubseteq Mammal \sqcap \exists eats.Grass$$

Cow is not subsumed by Herbivore!

(A cow must eat "at least 1 Grass", but could eat other things.)

Note: "Only" Does Not Mean "Some"

$$Cow \sqsubseteq \forall eats.Grass$$

Cow is not subsumed by $\exists eats.Grass$, not even $\exists eats.\top$.

(A cow can eat only Grass, but does not have to eat anything.)

Note: "Only" Does Not Mean "Some"

$Cow \sqsubseteq \forall eats.Grass$

Cow is not subsumed by $\exists eats.Grass$, not even $\exists eats.\top$.

(A cow can eat only Grass, but does not have to eat anything.)

$Animal \equiv Organism \sqcap \exists eats.Organism$

$Mammal \sqsubseteq Animal$

$Cow \sqsubseteq Mammal \sqcap \forall eats.Grass$

entails $Cow \sqsubseteq \exists eats.Grass$.

Note: “And” Does Not Mean “Or”

“Cows eat grass and grain.”

$$Cow \sqsubseteq \forall eats.(Grass \sqcap Grain) \qquad Grass \sqsubseteq \neg Grain$$

Note: "And" Does Not Mean "Or"

"Cows eat grass and grain."

$$Cow \sqsubseteq \forall eats.(Grass \sqcap Grain) \qquad Grass \sqsubseteq \neg Grain$$

Cow and $\exists eats.\top$ are disjoint!

(A cow can eat only things that are at the same time Grass and Grain, which do not exist.)

Note: "And" Does Not Mean "Or"

"Cows eat grass and grain."

$$Cow \sqsubseteq \forall eats.(Grass \sqcap Grain) \qquad Grass \sqsubseteq \neg Grain$$

Cow and $\exists eats.\top$ are disjoint!

(A cow can eat only things that are at the same time Grass and Grain, which do not exist.)

Better:

$$Cow \sqsubseteq \forall eats.(Grass \sqcup Grain) \qquad Grass \sqsubseteq \neg Grain$$

Note: "And" Does Not Mean "Or"

"Cows eat grass and grain."

$$Cow \sqsubseteq \forall eats.(Grass \sqcap Grain) \qquad Grass \sqsubseteq \neg Grain$$

Cow and $\exists eats.\top$ are disjoint!

(A cow can eat only things that are at the same time Grass and Grain, which do not exist.)

Better:

$$Cow \sqsubseteq \forall eats.(Grass \sqcup Grain) \qquad Grass \sqsubseteq \neg Grain$$

- Axioms like $Cow \sqsubseteq \forall eats.(Grass \sqcup Grain)$ are called closure axioms, because they define the range of $eats$ for all cows.

Closure Axioms

Often, closure axioms are used in combination with existential restrictions:

$$\begin{aligned} \mathit{Book} &\sqsubseteq \exists \mathit{hasPart}.\mathit{Cover} \\ \mathit{Book} &\sqsubseteq \exists \mathit{hasPart}.\mathit{Page} \\ \mathit{Book} &\sqsubseteq \forall \mathit{hasPart}.(\mathit{Cover} \sqcup \mathit{Page}) \end{aligned}$$

This is like saying *“A book has those parts and no other.”*

Closure Axioms

Often, closure axioms are used in combination with existential restrictions:

$$
\begin{aligned}
\mathit{Cell} &\sqsubseteq \exists \mathit{hasPart}.\mathit{PlasmaMembrane}\\
\mathit{Cell} &\sqsubseteq \exists \mathit{hasPart}.\mathit{Mitochondrion}\\
\mathit{Cell} &\sqsubseteq \exists \mathit{hasPart}.\mathit{EndoPlasmicReticulum}\\
\mathit{Cell} &\sqsubseteq \exists \mathit{hasPart}.\mathit{Nucleus}\\
\mathit{Cell} &\sqsubseteq \forall \mathit{hasPart}.(\mathit{PlasmaMembrane}\\
&\qquad\qquad\qquad \sqcup \mathit{Mitochondrion}\\
&\qquad\qquad\qquad \sqcup \mathit{EndoPlasmicReticulum}\\
&\qquad\qquad\qquad \sqcup \mathit{Nucleus})
\end{aligned}
$$

This is like saying *"A cell has those parts and no other."*

Covering Axioms

- Closure axioms are one way of *closing* descriptions under the open world assumption
- Another such technique is using covering axioms
- Covering axioms work well in combination with disjointness axioms

$$\begin{aligned} Herbivore &\equiv Animal \sqcap \forall eats.Plant \\ Carnivore &\equiv Animal \sqcap \forall eats.Animal \\ Animal &\equiv Herbivore \sqcup Carnivore \sqcup Omnivore \\ Animal &\sqsubseteq \neg Plant \end{aligned}$$

- These axioms entail:

$$Animal \sqcap \exists eats.Animal \sqcap \exists eats.Plant \sqsubseteq Omnivore$$

Disjointness Axioms

- Covering axioms work well in combination with disjointness axioms
- Disjointness axioms furthermore help finding bugs via incoherence

Disjointness Axioms

- Covering axioms work well in combination with disjointness axioms
- Disjointness axioms furthermore help finding bugs via incoherence
- OWL (and Protégé) offers convenient syntactic sugar:
  - Make a set of class expressions pair-wise disjoint
  - Define a class as disjoint union

Note: Value Restrictions on the Left-Hand Side

Value restrictions can behave strangely on the left-hand side of GCIs (and therefore also in full definitions).

$$\forall eats.Grass \sqsubseteq Cow$$

$$Cow \equiv \forall eats.Grass$$

Note: Value Restrictions on the Left-Hand Side

Value restrictions can behave strangely on the left-hand side of GCIs (and therefore also in full definitions).

$$\forall eats.Grass \sqsubseteq Cow$$

$$Cow \equiv \forall eats.Grass$$

With one of these axioms, anything that does not eat is a cow.

Note: Value Restrictions on the Left-Hand Side

Value restrictions can behave strangely on the left-hand side of GCIs (and therefore also in full definitions).

$$\forall eats.Grass \sqsubseteq Cow$$
$$Cow \equiv \forall eats.Grass$$

With one of these axioms, anything that does not eat is a cow.

$$Tornado \sqsubseteq \neg Organism$$
$$\exists eats.\top \sqsubseteq Organism$$
$$\forall eats.Grass \sqsubseteq Cow$$

entail $Tornado \sqsubseteq \forall eats.\bot \sqsubseteq Cow$.