ONTOLOGIES & DESCRIPTION LOGICS

Part of Logic and Languages Module

Meghyn Bienvenu (LaBRI - CNRS & Université de Bordeaux)
REASONING WITH LIGHTWEIGHT DLS
Some applications require very large ontologies and/or data.

Scalability concerns led to proposal of DLs with lower complexity.

**EL family of DLs** (basis for OWL 2 EL)
- designed to allow efficient reasoning with large ontologies
- key technique: saturation (∼ forward chaining)

**DL-Lite family of DLs** (basis for OWL 2 QL)
- designed for ontology-mediated query answering
- key technique: query rewriting (∼ backward chaining)
REASONING IN EL
The logic $\mathcal{EL}$ and its extensions are designed for applications requiring very large ontologies.

This family of DLs is well suited for biomedical applications.

Examples of large biomedical ontologies:
- **GO (Gene Ontology)**, around 20,000 concepts
- **NCI (cancer ontology)**, around 30,000 concepts
- **SNOMED (medical ontology)**, over 350,000 concepts (!)

\[
\text{Pericarditis} \sqsubseteq \text{Inflammation} \sqcap \exists \text{loc. Pericardium} \\
\text{Pericardium} \sqsubseteq \text{Tissue} \sqcap \exists \text{partOf. Heart} \\
\text{Inflammation} \sqsubseteq \text{Disease} \\
\text{Disease} \sqcap \exists \text{loc.} \exists \text{partOf. Heart} \sqsubseteq \text{HeartDisease}
\]
In $\mathcal{EL}$, complex concepts are built as follows:

$$C := T \mid A \mid C_1 \sqcap C_2 \mid \exists r.C$$

Only concept inclusions $C_1 \sqsubseteq C_2$ in the TBox.
In \( \mathcal{EL} \), complex concepts are built as follows:

\[
C := \top \mid A \mid C_1 \cap C_2 \mid \exists r.C
\]

Only concept inclusions \( C_1 \sqsubseteq C_2 \) in the TBox

Some possible extensions:
- \( \bot \) (to express disjoint classes)
- domain restrictions \( \text{dom}(r) \sqsubseteq C \)
- range restrictions \( \text{range}(r) \sqsubseteq C \)
- complex role inclusions \( r_1 \circ \ldots \circ r_n \sqsubseteq r_{n+1} \) (transitivity: \( r \circ r \sqsubseteq r \))

**OWL 2 EL** profile includes all these extensions
In $\mathcal{EL}$, complex concepts are built as follows:

$$C := \top \mid A \mid C_1 \cap C_2 \mid \exists r.C$$

Only concept inclusions $C_1 \sqsubseteq C_2$ in the TBox

Some possible extensions:
- $\perp$ (to express disjoint classes)
- domain restrictions $\text{dom}(r) \sqsubseteq C$
- range restrictions $\text{range}(r) \sqsubseteq C$
- complex role inclusions $r_1 \circ \ldots \circ r_n \sqsubseteq r_{n+1}$ (transitivity: $r \circ r \sqsubseteq r$)

$\text{OWL 2 EL}$ profile includes all these extensions

We will focus on plain $\mathcal{EL}$ (without these extensions)
**NORMAL FORM FOR EL TBOXES**

\( \mathcal{T} \) is in **normal form** if it contains only inclusions of the forms:

\[
A \sqsubseteq B \quad A \sqcap A_2 \sqsubseteq B \quad A \sqsubseteq \exists r.B \quad \exists r.A \sqsubseteq B
\]

where \( A, A_1, A_2, B \) are concept names (or \( \top \)).

Use term **basic axioms** for such inclusions, and **basic assertions** to refer to non-complex assertions.
\( \mathcal{T} \) is in **normal form** if it contains only inclusions of the forms:

\[
A \sqsubseteq B \quad A_1 \sqcap A_2 \sqsubseteq B \quad A \sqsubseteq \exists r.B \quad \exists r.A \sqsubseteq B
\]

where \( A, A_1, A_2, B \) are concept names (or \( \top \)).

Use term **basic axioms** for such inclusions, and **basic assertions** to refer to non-complex assertions.

**Theorem**: for every \( \mathcal{EL} \) TBox \( \mathcal{T} \), we can **construct in PTIME** a TBox \( \mathcal{T}' \) in normal form (possibly using new concept names) such that:

- for every inclusion \( C \sqsubseteq D \) which uses only concept names from \( \mathcal{T} \), we have \( \mathcal{T} \models C \sqsubseteq D \iff \mathcal{T}' \models C \sqsubseteq D \)
- for every ABox \( \mathcal{A} \) and assertion \( \alpha \) that only uses concept names from \( (\mathcal{T}, \mathcal{A}) \), we have \( \mathcal{T}, \mathcal{A} \models \alpha \iff \mathcal{T}', \mathcal{A} \models \alpha \)
**NORMAL FORM FOR EL TBOXES**

\( \mathcal{T} \) is in **normal form** if it contains only inclusions of the forms:

\[
A \subseteq B \quad A_1 \cap A_2 \subseteq B \quad A \subseteq \exists r.B \quad \exists r.A \subseteq B
\]

where \( A, A_1, A_2, B \) are concept names (or \( \top \)).

Use term **basic axioms** for such inclusions, and **basic assertions** to refer to non-complex assertions.

**Theorem:** for every \( \mathcal{EL} \) TBox \( \mathcal{T} \), we can construct in PTIME a TBox \( \mathcal{T}' \) in normal form (possibly using new concept names) such that:

- for every inclusion \( C \subseteq D \) which uses only concept names from \( \mathcal{T} \), we have \( \mathcal{T} \models C \subseteq D \) iff \( \mathcal{T}' \models C \subseteq D \)
- for every ABox \( \mathcal{A} \) and assertion \( \alpha \) that only uses concept names from \( (\mathcal{T}, \mathcal{A}) \), we have \( \mathcal{T}, \mathcal{A} \models \alpha \) iff \( \mathcal{T}', \mathcal{A} \models \alpha \)

More specifically: \( \mathcal{T}' \) is a **conservative extension** of \( \mathcal{T} \) (see homework)
**NORMALIZATION PROCEDURE**

*Exhaustively apply* the following *normalization rules* to $T$: (any order)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{D} \sqsubseteq \hat{E}$</td>
<td>$\leadsto$</td>
<td>$\hat{D} \sqsubseteq A_{\text{new}} \quad A_{\text{new}} \sqsubseteq \hat{E}$</td>
</tr>
<tr>
<td>$C \sqcap \hat{D} \sqsubseteq B$</td>
<td>$\leadsto$</td>
<td>$\hat{D} \sqsubseteq A_{\text{new}} \quad C \sqcap A_{\text{new}} \sqsubseteq B$</td>
</tr>
<tr>
<td>$\hat{D} \sqcap C \sqsubseteq B$</td>
<td>$\leadsto$</td>
<td>$\hat{D} \sqsubseteq A_{\text{new}} \quad A_{\text{new}} \sqcap C \sqsubseteq B$</td>
</tr>
<tr>
<td>$\exists r.\hat{D} \sqsubseteq B$</td>
<td>$\leadsto$</td>
<td>$\hat{D} \sqsubseteq A_{\text{new}} \quad \exists r.A_{\text{new}} \sqsubseteq B$</td>
</tr>
<tr>
<td>$B \sqsubseteq \exists r.\hat{D}$</td>
<td>$\leadsto$</td>
<td>$B \sqsubseteq \exists r.A_{\text{new}} \quad A_{\text{new}} \sqsubseteq \hat{D}$</td>
</tr>
<tr>
<td>$B \sqsubseteq D \sqcap E$</td>
<td>$\leadsto$</td>
<td>$B \sqsubseteq D \quad B \sqsubseteq E$</td>
</tr>
</tbody>
</table>

where:

- $C, D, E$ are arbitrary $\mathcal{EL}$ concepts,
- $\hat{D}, \hat{E}$ are neither concept names nor $\top$,
- $B$ is a concept name,
- $A_{\text{new}}$ is a *fresh (new) concept name*
Applying the fourth rule to $\exists r.(\exists s.A \cap H) \sqsubseteq B \cap D$

Use third rule to transform $\exists s.A \cap H \sqsubseteq E$ into

Last rule used to replace $\exists r.E \sqsubseteq B \cap D$ by

End result:
SATURATION RULES FOR EL

Rules for deriving ontology axioms

\[ \frac{A \sqsubseteq A}{T1} \]
\[ \frac{A \sqsubseteq \top}{T2} \]
\[ \frac{A \sqsubseteq B_1 \quad A \sqsubseteq B_2 \quad B_1 \sqcap B_2 \sqsubseteq D}{T4} \]
\[ \frac{A \sqsubseteq \exists r. B_1 \quad B_1 \sqsubseteq B_2 \quad \exists r. B_2 \sqsubseteq D}{T5} \]

Rules for deriving assertions

\[ \frac{A \sqsubseteq B \quad A(c)}{B(c)} \quad \text{A1} \]
\[ \frac{A_1 \sqcap A_2 \sqsubseteq B \quad A_1(c) \quad A_2(c)}{B(c)} \quad \text{A2} \]
\[ \frac{\exists r. A \sqsubseteq B \quad r(c, d) \quad A(d)}{B(c)} \quad \text{A3} \]

Premises = axioms / assertions above the line
Conclusion = axiom / assertion below the line
Assume w.l.o.g. that start from KB whose **TBox is in normal form** & whose **ABox contains** \( T(a) \) for each of its individuals \( a \).
Assume w.l.o.g. that start from KB whose **TBox is in normal form** & whose **ABox contains** $\top(a)$ for each of its individuals $a$.

**Instantiated rule:**
- obtained from one of the ‘abstract’ saturation rules by replacing $A, B, D$ by $\mathcal{EL}$-concepts and $r$ by some role name
- must **only contain basic axioms & assertions** (important!)
Assume w.l.o.g. that start from KB whose **TBox is in normal form** & whose **ABox contains** \( T(a) \) for each of its individuals \( a \)

**Instantiated rule:**
- obtained from one of the ‘abstract’ saturation rules by replacing \( A, B, D \) by \( \mathcal{EL} \)-concepts and \( r \) by some role name
- must **only contain basic axioms & assertions** \( \) (important!)

**Instantiated rule** with premises \( \alpha_1, \ldots, \alpha_n \) and conclusion \( \beta \) is **applicable in** \( \mathcal{K} \) if \( \{ \alpha_1, \ldots, \alpha_n \} \subseteq \mathcal{K} \) and \( \beta \notin \mathcal{K} \)
- in this case, can **apply the rule by adding** \( \beta \) to \( \mathcal{K} \)

Saturation procedure: **exhaustively apply instantiated rules** until no rule is applicable
TBox $\mathcal{T}$ contains axioms:

1. $\exists\text{hasIngred.Spicy}\sqsubseteq\text{Spicy}$
2. $\text{Spicy}\sqcap\text{Dish}\sqsubseteq\text{SpicyDish}$
3. $\text{ArrabSauce}\sqsubseteq\exists\text{hasIngred.Chili}$
4. $\text{Chili}\sqsubseteq\text{Spicy}$

ABox $\mathcal{A}$ contains:

5. $\text{Dish}(p)$
6. $\text{hasIngred}(p, s)$
7. $\text{ArrabSauce}(s)$

Examining the result, return $p$ as answer to instance query $q(x) = \text{SpicyDish}(x)$.
TBox $\mathcal{T}$ contains axioms:

1. $\exists\text{hasIngred}.\text{Spicy} \sqsubseteq \text{Spicy}$
2. $\text{Spicy} \sqcap \text{Dish} \sqsubseteq \text{SpicyDish}$
3. $\text{ArrabSauce} \sqsubseteq \exists\text{hasIngred}.\text{Chili}$
4. $\text{Chili} \sqsubseteq \text{Spicy}$

ABox $\mathcal{A}$ contains:

5. $\text{Dish}(p)$
6. $\text{hasIngred}(p, s)$
7. $\text{ArrabSauce}(s)$

Saturation procedure adds the following axioms and assertions:

8. $\text{ArrabSauce} \sqsubseteq \text{Spicy}$ using (1), (3), (4) and rule T5
9. $\text{Spicy}(s)$ using (7), (8), and rule A1
10. $\text{Spicy}(p)$ using (1), (6), (9), and rule A3
11. $\text{SpicyDish}(p)$ using (2), (5), (10), and rule A2
Example: Saturation Rules

TBox $\mathcal{T}$ contains axioms:

1. $\exists\text{hasIngred}.\text{Spicy} \sqsubseteq \text{Spicy}$
2. $\text{Spicy} \sqcap \text{Dish} \sqsubseteq \text{SpicyDish}$
3. $\text{ArrabSauce} \sqsubseteq \exists\text{hasIngred}.\text{Chili}$
4. $\text{Chili} \sqsubseteq \text{Spicy}$

ABox $\mathcal{A}$ contains:

5. $\text{Dish}(p)$
6. $\text{hasIngred}(p, s)$
7. $\text{ArrabSauce}(s)$

Saturation procedure adds the following axioms and assertions:

8. $\text{ArrabSauce} \sqsubseteq \text{Spicy}$ using (1), (3), (4) and rule T5
9. $\text{Spicy}(s)$ using (7), (8), and rule A1
10. $\text{Spicy}(p)$ using (1), (6), (9), and rule A3
11. $\text{SpicyDish}(p)$ using (2), (5), (10), and rule A2

Examining the result, return $p$ as answer to instance query $q(x) = \text{SpicyDish}(x)$
Denote by $\text{sat}(\mathcal{K})$ or $\text{sat}(\mathcal{T}, \mathcal{A})$ (resp. $\text{sat}(\mathcal{T})$) result of exhaustively applying saturation rules to KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ (resp. TBox $\mathcal{T}$)
Denote by $\text{sat}(\mathcal{K})$ or $\text{sat}(\mathcal{T}, \mathcal{A})$ (resp. $\text{sat}(\mathcal{T})$) result of exhaustively applying saturation rules to KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ (resp. TBox $\mathcal{T}$).

To find all instances of concept name $\mathcal{A}$ w.r.t. $\mathcal{K} = (\mathcal{T}, \mathcal{A})$:

1. Normalize $\mathcal{T}$, yielding $\mathcal{T}'$, then construct $\text{sat}(\mathcal{T}', \mathcal{A})$
2. Return all individuals $c$ such that $A(c) \in \text{sat}(\mathcal{T}', \mathcal{A})$. 

What about assertions / inclusions involving complex concepts?

- Use new concept names to represent complex concepts, e.g. if $C$ is a complex concept, add $X_C \sqsubseteq C$ and $C \sqsubseteq X_C$ to $\mathcal{T}$ (with $X_C$ fresh).
- Proceed as above but use $X_C$ in place of $C$. 


Denote by $\text{sat}(\mathcal{K})$ or $\text{sat}(\mathcal{T}, \mathcal{A})$ (resp. $\text{sat}(\mathcal{T})$) result of exhaustively applying saturation rules to KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ (resp. TBox $\mathcal{T}$) 

To find all instances of concept name $\mathcal{A}$ w.r.t. $\mathcal{K} = (\mathcal{T}, \mathcal{A})$:
1. Normalize $\mathcal{T}$, yielding $\mathcal{T}'$, then construct $\text{sat}(\mathcal{T}', \mathcal{A})$
2. Return all individuals $c$ such that $\mathcal{A}(c) \in \text{sat}(\mathcal{T}', \mathcal{A})$.

To test whether $\mathcal{T} \models \mathcal{A} \sqsubseteq \mathcal{B}$ ($\mathcal{A}, \mathcal{B}$ concept names):
1. Normalize $\mathcal{T}$, yielding $\mathcal{T}'$, then construct $\text{sat}(\mathcal{T}')$
   (can alternatively construct $\text{sat}(\mathcal{T}', \mathcal{A})$ if have an ABox $\mathcal{A}$)
2. Check if $\text{sat}(\mathcal{T}')$ contains $\mathcal{A} \sqsubseteq \mathcal{B}$, return yes if so, else no.
Denote by \( sat(\mathcal{K}) \) or \( sat(\mathcal{T}, \mathcal{A}) \) (resp. \( sat(\mathcal{T}) \)) result of exhaustively applying saturation rules to KB \( \mathcal{K} = (\mathcal{T}, \mathcal{A}) \) (resp. TBox \( \mathcal{T} \)).

To find all instances of concept name \( A \) w.r.t. \( \mathcal{K} = (\mathcal{T}, \mathcal{A}) \):
1. Normalize \( \mathcal{T} \), yielding \( \mathcal{T}' \), then construct \( sat(\mathcal{T}', \mathcal{A}) \)
2. Return all individuals \( c \) such that \( A(c) \in sat(\mathcal{T}', \mathcal{A}) \).

To test whether \( \mathcal{T} \models A \sqsubseteq B \) (\( A, B \) concept names):
1. Normalize \( \mathcal{T} \), yielding \( \mathcal{T}' \), then construct \( sat(\mathcal{T}') \)
   (can alternatively construct \( sat(\mathcal{T}', \mathcal{A}) \) if have an ABox \( \mathcal{A} \))
2. Check if \( sat(\mathcal{T}') \) contains \( A \sqsubseteq B \), return yes if so, else no.

What about assertions / inclusions involving complex concepts?

- Use new concept names to represent complex concepts, e.g. if \( C \) is a complex concept, add \( X_C \sqsubseteq C \) and \( C \sqsubseteq X_C \) to \( \mathcal{T} \) (with \( X_C \) fresh).
- Proceed as above but use \( X_C \) in place of \( C \).
Theorem. All exhaustive sequences of rule applications lead to a unique saturated KB.

Theorem. The saturated KB $sat(\mathcal{K})$ can be constructed in polynomial time in $|\mathcal{K}|$. 
**Theorem.** All exhaustive sequences of rule applications lead to a unique saturated KB.

**Theorem.** The saturated KB $\text{sat}(\mathcal{K})$ can be constructed in polynomial time in $|\mathcal{K}|$.

**Theorem.** The saturation procedure is correct and complete for axiom entailment and instance checking involving concept names. Specifically:

- for every concept inclusion $A \sqsubseteq B$ (with $A, B$ concept names):
  $\mathcal{T} \models A \sqsubseteq B$ iff $A \sqsubseteq B \in \text{sat}(\mathcal{K})$ iff $A \sqsubseteq B \in \text{sat}(\mathcal{T})$
Theorem. All exhaustive sequences of rule applications lead to a unique saturated KB.

Theorem. The saturated KB $\text{sat}(\mathcal{K})$ can be constructed in polynomial time in $|\mathcal{K}|$

Theorem. The saturation procedure is correct and complete for axiom entailment and instance checking involving concept names. Specifically:

- for every concept inclusion $A \sqsubseteq B$ (with $A, B$ concept names):
  $$\mathcal{T} \models A \sqsubseteq B \iff A \sqsubseteq B \in \text{sat}(\mathcal{K}) \iff A \sqsubseteq B \in \text{sat}(\mathcal{T})$$

- for every ABox assertion $\alpha = A(b)$ with $A$ a concept name:
  $$\mathcal{K} \models A(b) \iff A(b) \in \text{sat}(\mathcal{K})$$

Next slides: sketch proofs for correctness and completeness
Aim to show that:

1. if $A \sqsubseteq B \in \text{sat}(\mathcal{K})$, then $\mathcal{T} \models A \sqsubseteq B$
2. if $A(b) \in \text{sat}(\mathcal{K})$, then $\mathcal{K} \models A(b)$
Aim to show that:

- if $A \subseteq B \in sat(\mathcal{K})$, then $\mathcal{T} \models A \subseteq B$
- if $A(b) \in sat(\mathcal{K})$, then $\mathcal{K} \models A(b)$

As $sat(\mathcal{K})$ is the result of a sequence of rule applications, it suffices to show the following lemma:

**Lemma.** If a saturation rule application produces $\beta$ from the premises $\alpha_1, \ldots, \alpha_n$, and $\mathcal{K} \models \alpha_i \ (1 \leq i \leq n)$, then $\mathcal{K} \models \beta$. 
CORRECTNESS OF SATURATION

Aim to show that:
\[
\begin{align*}
&\cdot \text{ if } A \sqsubseteq B \in \text{sat}(\mathcal{K}), \text{ then } \mathcal{T} \models A \sqsubseteq B \\
&\cdot \text{ if } A(b) \in \text{sat}(\mathcal{K}), \text{ then } \mathcal{K} \models A(b)
\end{align*}
\]

As \( \text{sat}(\mathcal{K}) \) is the result of a sequence of rule applications, it suffices to show the following lemma:

**Lemma.** If a saturation rule application produces \( \beta \) from the premises \( \alpha_1, \ldots, \alpha_n \), and \( \mathcal{K} \models \alpha_i \) \((1 \leq i \leq n)\), then \( \mathcal{K} \models \beta \).

Proof sketch:
\[
\begin{align*}
&\cdot \text{ trivial for rules } T_1 \text{ and } T_2 \text{ (produced axioms hold in any model)} \\
&\cdot \text{ easy arguments for other rules, e.g. for } T_3:
\end{align*}
\]
\[
\begin{align*}
&\cdot \text{ suppose } \mathcal{K} \models A \sqsubseteq B \text{ and } \mathcal{K} \models B \sqsubseteq D, \text{ take any model } \mathcal{I} \text{ of } \mathcal{K} \text{ and } e \in A^\mathcal{I}, \\
&\text{must have } e \in B^\mathcal{I} \text{ due to } A \sqsubseteq B, \text{ hence } e \in D^\mathcal{I} \text{ due to } B \sqsubseteq D, \text{ yielding } \\
&\mathcal{I} \models A \sqsubseteq D \text{ as required}
\end{align*}
\]
We prove the contrapositive, namely:

- if $A \sqsubseteq B \not\in \text{sat}(\mathcal{K})$, then $\mathcal{T} \not\models A \sqsubseteq B$
- if $A(b) \not\in \text{sat}(\mathcal{K})$, then $\mathcal{K} \not\models A(b)$
We prove the contrapositive, namely:

- if $A \subseteq B \not\in \text{sat}(\mathcal{K})$, then $\mathcal{T} \not\models A \subseteq B$
- if $A(b) \not\in \text{sat}(\mathcal{K})$, then $\mathcal{K} \not\models A(b)$

Proof strategy:

- build an interpretation $\mathcal{C}_\mathcal{K}$ from $\text{sat}(\mathcal{K})$
- show that $\mathcal{C}_\mathcal{K}$ is a model of $\mathcal{K}$
- show that $\mathcal{C}_\mathcal{K} \not\models A \subseteq B$ when $A \subseteq B \not\in \text{sat}(\mathcal{K})$
- show that $\mathcal{C}_\mathcal{K} \not\models A(b)$ when $A(b) \not\in \text{sat}(\mathcal{K})$
Define $C_K$, as follows:

- $\Delta^{C_K} = \text{Ind}(A) \cup \{w_A \mid A \text{ concept name appearing in } K\} \cup \{w_{\top}\}$
- $A^{C_K} = \{b \mid A(b) \in \text{sat}(K)\} \cup \{w_B \mid B \sqsubseteq A \in \text{sat}(K)\}$
- $r^{C_K} = \{(a, b) \mid r(a, b) \in K\} \cup \{(w_A, w_B) \mid A \sqsubseteq \exists r.B \in \text{sat}(K)\}$
  $\cup \{(a, w_B) \mid A \sqsubseteq \exists r.B \in \text{sat}(K), A(a) \in \text{sat}(K) \text{ for some } A\}$
- $a^{C_K} = a$ for all $a \in \text{Ind}(A)$

where $\text{Ind}(A)$ is set of individual names in $A$
Define $C_{\mathcal{K}}$, as follows:

- $\Delta^{C_{\mathcal{K}}} = \text{Ind}(\mathcal{A}) \cup \{w_A \mid A \text{ concept name appearing in } \mathcal{K}\} \cup \{w_\top\}$
- $A^{C_{\mathcal{K}}} = \{b \mid A(b) \in \text{sat}(\mathcal{K})\} \cup \{w_B \mid B \sqsubseteq A \in \text{sat}(\mathcal{K})\}$
- $r^{C_{\mathcal{K}}} = \{(a, b) \mid r(a, b) \in \mathcal{K}\} \cup \{(w_A, w_B) \mid A \sqsubseteq \exists r. B \in \text{sat}(\mathcal{K})\} \cup \{(a, w_B) \mid A \sqsubseteq \exists r. B \in \text{sat}(\mathcal{K}), A(a) \in \text{sat}(\mathcal{K}) \text{ for some } A\}$
- $a^{C_{\mathcal{K}}} = a$ for all $a \in \text{Ind}(\mathcal{A})$

where $\text{Ind}(\mathcal{A})$ is set of individual names in $\mathcal{A}$

Observe that by construction, we have:

- $C_{\mathcal{K}} \not\models A \sqsubseteq B$ when $A \sqsubseteq B \not\in \text{sat}(\mathcal{K})$
- $C_{\mathcal{K}} \not\models A(b)$ when $A(b) \not\in \text{sat}(\mathcal{K})$

for all concept names $A, B$ and individuals $b$ occurring in $\mathcal{K}$
By definition, $C_K$ is a model of $A$
By definition, \( C_{\mathcal{K}} \) is a model of \( \mathcal{A} \)

To show it is a model of \( \mathcal{T} \), consider different kinds of axioms:

- **Case 1:** \( A \sqsubseteq B \in \mathcal{T} \) and \( e \in A^{C_{\mathcal{K}}} \)
  - If \( e \in \text{Ind}(A) \), then \( A(e) \in \text{sat}(\mathcal{K}) \). Due to \( A1 \), \( B(e) \in \text{sat}(\mathcal{K}) \), so \( e \in B^{C_{\mathcal{K}}} \).
  - If \( e = w_D \), then \( D \sqsubseteq A \in \text{sat}(\mathcal{T}) \). Due to \( T3 \), \( D \sqsubseteq B \in \text{sat}(\mathcal{T}) \), so \( e \in B^{C_{\mathcal{K}}} \).

- **Case 2:** \( A_1 \sqcap A_2 \sqsubseteq B \in \mathcal{T} \) and \( e \in (A_1 \sqcap A_2)^{C_{\mathcal{K}}} \)

- **Case 3:** \( A \sqsubseteq \exists r. B \in \mathcal{T} \) and \( e \in A^{C_{\mathcal{K}}} \)

- **Case 4:** \( \exists r. A \sqsubseteq B \), \( e' \in A^{C_{\mathcal{K}}} \), and \( (e, e') \in r^{C_{\mathcal{K}}} \)

Call \( C_{\mathcal{K}} \) the (compact) canonical model for \( \mathcal{K} \)
By definition, $C_K$ is a model of $A$

To show it is a model of $\mathcal{T}$, consider different kinds of axioms:

- **Case 1**: $A \sqsubseteq B \in \mathcal{T}$ and $e \in A^{C_K}$
  - If $e \in \text{Ind}(A)$, then $A(e) \in \text{sat}(K)$. Due to $A1$, $B(e) \in \text{sat}(K)$, so $e \in B^{C_K}$.
  - If $e = w_D$, then $D \sqsubseteq A \in \text{sat}(\mathcal{T})$. Due to $T3$, $D \sqsubseteq B \in \text{sat}(\mathcal{T})$, so $e \in B^{C_K}$.

- **Case 2**: $A_1 \cap A_2 \sqsubseteq B \in \mathcal{T}$ and $e \in (A_1 \cap A_2)^{C_K}$
  - similar argument using $A2$ and $T4$

- **Case 3**: $A \sqsubseteq \exists r. B \in \mathcal{T}$ and $e \in A^{C_K}$
  - argument uses $T3$

- **Case 4**: $\exists r. A \sqsubseteq B$, $e' \in A^{C_K}$, and $(e, e') \in r^{C_K}$
  - argument uses $A3$ and $T5$
By definition, $C_K$ is a model of $A$.

To show it is a model of $T$, consider different kinds of axioms:

- **Case 1:** $A \subseteq B \in T$ and $e \in A^{C_K}$
  - If $e \in \text{Ind}(A)$, then $A(e) \in \text{sat}(K)$. Due to A1, $B(e) \in \text{sat}(K)$, so $e \in B^{C_K}$.
  - If $e = w_D$, then $D \sqsubseteq A \in \text{sat}(T)$. Due to T3, $D \sqsubseteq B \in \text{sat}(T)$, so $e \in B^{C_K}$.

- **Case 2:** $A_1 \sqcap A_2 \subseteq B \in T$ and $e \in (A_1 \sqcap A_2)^{C_K}$
  - Similar argument using $A_2$ and T4

- **Case 3:** $A \subseteq \exists r.B \in T$ and $e \in A^{C_K}$
  - Argument uses T3

- **Case 4:** $\exists r.A \sqsubseteq B$, $e' \in A^{C_K}$, and $(e, e') \in r^{C_K}$
  - Argument uses A3 and T5

Call $C_K$ the (compact) canonical model for $K$. 
Theorem. *Axiom entailment and instance checking over EL KBs are PTIME-complete*

- upper bound: saturation procedure from previous slides
- lower bound: entailment from propositional Horn theories
Theorem. **Axiom entailment and instance checking** over \( \mathcal{EL} \) KBs are PTIME-complete

- upper bound: saturation procedure from previous slides
- lower bound: entailment from propositional Horn theories

Note: with only \( \sqcap \) and \( \forall r.C \), same problems are EXPTIME-complete!
Theorem. Axiom entailment and instance checking over EL KBs are PTIME-complete

- upper bound: saturation procedure from previous slides
- lower bound: entailment from propositional Horn theories

Note: with only $\sqcap$ and $\forall r.C$, same problems are EXPTIME-complete!

Further advantage of saturation approach: ‘single-pass’ reasoning

- compute saturation once, then read off all entailed assertions and inclusions involving concept names
Theorem. **Axiom entailment and instance checking** over $\mathcal{EL}$ KBs are PTIME-complete

- upper bound: saturation procedure from previous slides
- lower bound: entailment from propositional Horn theories

**Note:** with only $\sqcap$ and $\forall r.C$, same problems are EXPTIME-complete!

**Further advantage of saturation approach:** ‘single-pass’ reasoning

- compute saturation once, then read off all entailed assertions and inclusions involving concept names

In practice:

- huge ontologies like SNOMED can be classified in a few seconds
We can add all of the following without losing tractability:

- \( \bot \)
- \( \text{dom}(r) \sqsubseteq C, \text{range}(r) \sqsubseteq C \)
- \( r_1 \circ \ldots \circ r_n \sqsubseteq r_{n+1} \) (complex role inclusions)
We can add all of the following without losing tractability:

- \( \bot \)
- \( \text{dom}(r) \subseteq C, \text{range}(r) \subseteq C \)
- \( r_1 \circ \ldots \circ r_n \subseteq r_{n+1} \) (complex role inclusions)

But adding any of the following makes reasoning EXPTIME-hard:

- negation \( \neg \)
- disjunction \( \sqcup \)
- at-least or at-most restrictions: \( \geq 2r, \leq 1r \)
- functional roles (funct \( r \))
- inverse roles \( r^- \)
The DL $\mathcal{ELI}$ is obtained by adding inverse roles to $\mathcal{EL}$.

Reasoning in $\mathcal{ELI}$ is much more difficult (EXPTIME-complete).

However, $\mathcal{ELI}$ retains some nice properties:
- admits a canonical model, hence no ‘case-based’ reasoning.

Can extend saturation procedure to $\mathcal{ELI}$:
- still deterministic.
- may be exponential since need to consider sets of concept names.
  - deduce $A \cap D \subseteq \exists r. (B \cap D)$ from $A \subseteq \exists r. B$ and $\exists r\neg. D \subseteq E$.

In practice: $\mathcal{ELI}$ and other ‘Horn DLs’ easier to handle than $\mathcal{ALC}$.