Description Logics

Knowledge Bases in Description Logics

Enrico Franconi

franconi@cs.man.ac.uk
http://www.cs.man.ac.uk/~franconi

Department of Computer Science, University of Manchester
Understanding Knowledge Bases

\[ \Sigma = \langle \text{TBox}, \text{Abox} \rangle \]

- **Terminological Axioms:** \( C \subseteq D \)
- **Assertional Axioms:** \( C(a), R(a, b) \)

An interpretation \( \mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I}) \) satisfies the statement \( C \subseteq D \) if \( C^\mathcal{I} \subseteq D^\mathcal{I} \).

- \( \mathcal{I} \) satisfies \( C(a) \) if \( a^\mathcal{I} \in C^\mathcal{I} \).
- \( \mathcal{I} \) satisfies \( R(a, b) \) if \( (a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I} \).

An interpretation \( \mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I}) \) is said to be a *model* of \( \Sigma \) if every axiom of \( \Sigma \) is satisfied by \( \mathcal{I} \). \( \Sigma \) is said to be *satisfiable* if it admits a model.
TBox statements

(1) \( A \sqsubseteq C \)  Primitive concept definition
(2) \( A \equiv C \)  Concept definition
(3) \( C \sqsubseteq D \)  Concept inclusion
(4) \( C' \equiv D \)  Concept equation
**Acyclic simple TBox**

Simple TBox:

1. \( A \sqsubseteq C \)  Primitive concept definition
2. \( A \equiv C \)  Concept definition

_Acyclic simple TBox:_ well-founded definitions.

A concept name \( A \) _directly uses_ a concept name \( B \) in a TBox \( \Sigma \) iff the definition of \( A \) mentions \( B \). A concept name \( A \) _uses_ a concept name \( B_n \) iff there is a chain of concept names \( \langle A, B_1, \ldots, B_n \rangle \) such that \( B_i \) directly uses \( B_{i+1} \). A TBox is _acyclic_ iff no concept name uses itself.
Acyclic simple TBox

Subsumption in acyclic simple TBoxes \( \Sigma \models C \subseteq D \) can be reduced in subsumption in an empty TBox \( \models \widehat{C} \subseteq \widehat{D} \).

In order to get \( \widehat{C} \) (and \( \widehat{D} \)):

1) Transform the TBox \( \Sigma \) into a new TBox \( \Sigma' \), by replacing every primitive concept definition in \( \Sigma \) of the form \( A \subseteq C \) with a concept definition \( A \models C \cap A^* \) – where \( A^* \) is a freshly new generated concept name (called primitive component of \( A \)).

Now \( \Sigma' \) contains only (acyclic) concept definitions.

2) Iteratively substitute every occurrence of any defined concept name in \( C \) (and \( D \)) by the corresponding definition in \( \Sigma' \). Since \( \Sigma' \) is still acyclic, the process terminates in a finite number of iterations. This process is called unfolding or expansion.
Theorems

• For each interpretation of $\Sigma$ there exists an interpretation of $\Sigma'$ (and vice versa) such that $C^\mathcal{I} = C'^\mathcal{I}'$ for each concept name $C$ in $\Sigma$.

\[ A \subseteq C \quad \sim \quad A \vdash C \cap A^* \]

$A^*$ denotes the *unexpressed* part of meaning implicitly contained in the primitive concept definition.

• $\Sigma \models C \sqsubseteq D$  iff  $\Sigma' \models C \sqsubseteq D$

• $\Sigma' \models C \sqsubseteq D$  iff  $\models \widehat{C} \sqsubseteq \widehat{D}$
Necessary and Sufficient conditions

- A primitive concept definition $A \subseteq C$ states a necessary but not sufficient condition for membership in the class $A$. Having the property $C$ is necessary for an object in order to be in the class $A$; however, this condition alone is not sufficient in order to conclude that the object is in the class $A$.

- A concept definition $A \equiv C$ states necessary and sufficient condition for membership in the class $A$. Having the property $C$ is necessary for an object in order to be in the class $A$; moreover, this condition alone is sufficient in order to conclude that the object is in the class $A$. 
Necessary and Sufficient conditions

When transforming primitive concept definitions into concept definitions we get necessary and sufficient conditions for membership in the primitive class $A$. However, the condition of being in the primitive component $A^*$ can never be satisfied, since the concept name $A^*$ can never be referred to by any other concept.

A concept is subsumed by a primitively defined concept if and only if it refers to its name in its (unfolded) definition.
Inheritance

Unfolding realizes what is usually called *inheritance* in Object-Oriented frameworks.

Person $= \exists \text{NAME}.\text{String} \land \exists \text{ADDRESS}.\text{String}$

Parent $= \text{Person} \land \exists \text{CHILD}.\text{Person}$

Parent $= \exists \text{NAME}.\text{String} \land \exists \text{ADDRESS}.\text{String} \land$

$\exists \text{CHILD}.(\exists \text{NAME}.\text{String} \land \exists \text{ADDRESS}.\text{String})$

Female $= \neg \text{Male}$

Man $= \text{Person} \land \forall \text{SEX}.\text{Male}$

Woman $= \text{Person} \land \forall \text{SEX}.\text{Female}$

Transexual $= \text{Man} \land \text{Woman}$

Transexual $= \exists \text{NAME}.\text{String} \land \exists \text{ADDRESS}.\text{String} \land \forall \text{SEX}.\bot$
Inheritance in O-O

Problems in O-O frameworks: overriding strategies for multiple inheritance.
Complexity of Unfolding

\[ C_1 \doteq \forall R_1. C_0 \sqcap \forall R_2. C_0 \sqcap \ldots \sqcap \forall R_m. C_0 \]
\[ C_2 \doteq \forall R_1. C_1 \sqcap \forall R_2. C_1 \sqcap \ldots \sqcap \forall R_m. C_1 \]
\[ \ldots \]
\[ C_n \doteq \forall R_1. C_{n-1} \sqcap \forall R_2. C_{n-1} \sqcap \ldots \sqcap \forall R_m. C_{n-1} \]

- The size of the TBox is \( O(n \times m) \).
- The size of the unfolded concept \( \hat{C}_n \) is \( O(m^n) \).
- The complexity of the subsumption problem in \( \mathcal{FL}^- \) with empty TBox \( (\models C \sqsubseteq D) \) is P.
- The complexity of the subsumption problem in \( \mathcal{FL}^- \) with an acyclic simple TBox \( (\Sigma \models C \sqsubseteq D) \) is co-NP-complete.
Efficiency of Subsumption in practice

The exponential worst case is unlikely to occur in real knowledge bases.

- Let $n$ be the depth of a TBox, i.e., the max number of iterations while unfolding every concept definition.
- Let $m$ be the size of the largest definition.
- Let $s$ be the size of the TBox, i.e., $m$ times the number of concept definitions.

The size of an unfolded concept is $\mathcal{O}(m^n)$.

If $n \leq \log_m s$ the size of an unfolded concept becomes polynomial $\mathcal{O}(s)$ with respect to the size of the TBox.

This is a reasonable assumption, since the depth of concept definitions is usually much smaller than the size of the knowledge base. This is why systems behave well in practice.
Definitions

• Definitions are intended to provide an exact account for the concept name being defined.

• Given an initial interpretation of the primitive concept names there exists a unique way determine the interpretation of defined concept names; indeed, that’s why they are called *definitions*.

• This justifies the correctness of unfolding: we can always replace a concept name with its definition, since it doesn’t add anything to the theory.

• However, if the (simple) TBox is cyclic, this is no more true.
Example of recursive definition

Bird \equiv \text{Animal} \land \forall \text{SKIN}.\text{Feather}

\Delta^\mathcal{I} = \{\text{tweety}, \text{goofy}, \text{feal}, \text{fur1}\}
\text{Animal}^\mathcal{I} = \{\text{tweety}, \text{goofy}\}
\text{Feather}^\mathcal{I} = \{\text{feal}\}
\text{SKIN}^\mathcal{I} = \{\langle\text{tweety}, \text{feal}\rangle, \langle\text{goofy}, \text{fur1}\rangle\}
\implies \text{Bird}^\mathcal{I} = \{\text{tweety}\}

\text{Quiet-Person} \equiv \text{Person} \land \forall \text{FRIEND}.\text{Quiet-Person}

\Delta^\mathcal{I} = \{\text{john}, \text{sue}, \text{andrea}, \text{bill}\}
\text{Person}^\mathcal{I} = \{\text{john}, \text{sue}, \text{andrea}, \text{bill}\}
\text{FRIEND}^\mathcal{I} = \{\langle\text{john}, \text{sue}\rangle, \langle\text{andrea}, \text{bill}\rangle, \langle\text{bill}, \text{bill}\rangle\}
\implies \text{Quiet-Person}^\mathcal{I} = \{\text{john}, \text{sue}\}
\implies \text{Quiet-Person}^\mathcal{I} = \{\text{john}, \text{sue}, \text{andrea}, \text{bill}\}
Descriptive semantics

(It is the one we have introduced before.)

- An interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ satisfies the concept definition $A \stackrel{\cdot}{=} C$ iff $A^\mathcal{I} = C^\mathcal{I}$.

- The definition is a constraint stating a restriction on the valid models of the knowledge base, and in particular on the possible interpretations of $A$, where $A$ is no better specified.

- It allows both models of the previous cyclic definition.
Fixpoint Semantics

We associate to a cyclic concept definition an operator $F : 2^\Delta I \leftrightarrow 2^\Delta I$, such that the interpretation of $A$ correspond to the fixpoints of the operator $F$.

Thus, we associate the equation

$$A = F(A)$$

to a cyclic concept definition of the type

$$A \equiv C'$$

where $C'$ mentions $A$. 
Least Fixpoint Semantics

The LFS interprets a recursive definition

\[ A = F(A) \]

by assigning to \( A \) the \textit{smallest} possible extension in each interpretation \( \mathcal{I} \) – if it exists – among those satisfying

\[ A^\mathcal{I} = F(A)^\mathcal{I} \]

i.e., the least fixpoint of the corresponding operator.

If the operator is monotonic, then the equation above singles out a \textit{unique} interpretation (subset of \( \Delta^\mathcal{I} \)), hence it \textit{defines} the concept \( A \).
Example

Quiet-Person ≜ Person ∩ ∀FRIEND.Quiet-Person

\[ F = \lambda A. \{ x \in \Delta^I \mid \]
\[ \text{Person}^I(x) \land \forall y. \text{FRIEND}^I(x, y) \rightarrow A(y) \}\]

\[ A = F(A) \]

\[ \Delta^I = \{ \text{john, sue, andrea, bill} \} \]
\[ \text{Person}^I = \{ \text{john, sue, andrea, bill} \} \]
\[ \text{FRIEND}^I = \{ \langle \text{john, sue} \rangle, \langle \text{andrea, bill} \rangle, \langle \text{bill, bill} \rangle \} \]

\[ \implies \text{Quiet-Person}^I = \{ \text{john, sue} \} \]
Problems

Human $\equiv$ Mammal $\sqcap \exists$PARENT $\sqcap \forall$PARENT.Human

Horse $\equiv$ Mammal $\sqcap \exists$PARENT $\sqcap \forall$PARENT.Horse

Under the fixpoint semantics

Human $\equiv$ Horse

i.e., in any interpretation $\mathcal{I}$ satisfying the above definitions

$\text{Human}^\mathcal{I} = \text{Horse}^\mathcal{I}$
Inductive definitions

- An Empty-List is a List.
- A Node, that has exactly one SUCCESSOR that is a List, is a List.
- Nothing else is a LIST.

\[
\text{Node} \equiv \neg \text{Empty-List}
\]

\[
\text{List} \equiv \text{Empty-List} \sqcup (\text{Node} \sqsubseteq \exists \text{SUCCESSOR} \sqcap \exists \text{SUCCESSOR}. \text{List})
\]

\[
\Delta^I = \{a, b, \text{nil}\}
\]

\[
\text{Node}^I = \{a, b\}
\]

\[
\text{Empty-List}^I = \{\text{nil}\}
\]

\[
\text{SUCCESSOR}^I = \{\langle a, \text{nil} \rangle, \langle b, b \rangle \}
\]

With descriptive semantics: \[
\text{List}^I = \{a, b, \text{nil}\}
\]

With least fixpoint semantics: \[
\text{List}^I = \{a, \text{nil}\}
\]
Inductive definitions

- Compare with Logic Programming, where inductive definitions come for free.
- Descriptive semantics is expressible in FOL.
- Least fixpoint semantics (and inductive definitions) go beyond First Order.
Free TBox

(3) \( C \sqsubseteq D \) \hspace{1cm} \text{Concept inclusion}

(4) \( C \models D \) \hspace{1cm} \text{Concept equation}

(There is no syntactic constraint on the left hand side of the axiom).

Concept inclusions make sense only with descriptive semantics – we will ignore here the extensions of Description Logics where it is possible to specify explicitly the semantics to be given to a knowledge base.

**Theorem:**

Description logics with simple TBoxes (with general concept definitions \( A \models D \)) and free TBoxes (with concept inclusions \( C \sqsubseteq D \)) have the same expressive power.
Simple TBoxes and Free TBoxes

Satisfiability in a knowledge base $\Sigma$ with a free TBox can be reduced into satisfiability in a knowledge base $\Sigma'$ with a simple TBox.

$$C_i \models D_i \quad \Rightarrow \quad A_i \sqsubseteq D_i, \; C_i \sqsubseteq A_i$$

$$C_j \sqsubseteq D_j \quad \Rightarrow \quad A \models (\neg C_1 \sqcup D_1) \sqcap \cdots \sqcap (\neg C_n \sqcup D_n) \sqcap A^* \sqcap \forall R_1.A \sqcap \cdots \sqcap \forall R_m.A$$

where $A$ is a new concept name not appearing in $\Sigma$ and $R_i$ are all the role names appearing in $\Sigma$.

The process of eliminating general axioms is called *internalization*. The above simple TBox emulates the general axiom

$$\left(\neg C_1 \sqcup D_1\right) \sqcap \cdots \sqcap \left(\neg C_n \sqcup D_n\right) \models \top$$