Description Logic 1: Syntax and Semantics

Leif Harald Karlsen

Autumn 2015
Contents

Introduction

\(\mathcal{ALC}\): Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
Contents

Introduction

\textit{ALC}: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
Overview

- *Description logics* are formal languages designed for knowledge representation and reasoning, and most of these are decidable fragments of FOL.
Overview

- Description logics are formal languages designed for knowledge representation and reasoning, and most of these are decidable fragments of FOL.
- Each description logic describes a language, and each language differ in expressibility vs. reasoning complexity, defined by allowing or disallowing different constructs (e.g. conjunction, disjunction, negation, quantifiers, etc.) in their language.
History and motivation

– Description logic comes from a merging of two traditions.
History and motivation

- Description logic comes from a merging of two traditions.
- Knowledge Representation (KR)
  - Application oriented
  - Represent ‘knowledge’ in some way
  - ‘Frames,’ like classes, with relations and attributes
  - Try to add some ‘semantics’ in order to do some ‘reasoning’
History and motivation

- Description logic comes from a merging of two traditions.
- Knowledge Representation (KR)
  - Application oriented
  - Represent ‘knowledge’ in some way
  - ‘Frames,’ like classes, with relations and attributes
  - Try to add some ‘semantics’ in order to do some ‘reasoning’
- Automated Reasoning, Modal Logic
  - Had theorems and algorithms
History and motivation

- Description logic comes from a merging of two traditions.
- Knowledge Representation (KR)
  - Application oriented
  - Represent ‘knowledge’ in some way
  - ‘Frames,’ like classes, with relations and attributes
  - Try to add some ‘semantics’ in order to do some ‘reasoning’
- Automated Reasoning, Modal Logic
  - Had theorems and algorithms
- Cross-fertilisation of applications and theory
History and motivation

- Description logic comes from a merging of two traditions.
- Knowledge Representation (KR)
  - Application oriented
  - Represent ‘knowledge’ in some way
  - ‘Frames,’ like classes, with relations and attributes
  - Try to add some ‘semantics’ in order to do some ‘reasoning’
- Automated Reasoning, Modal Logic
  - Had theorems and algorithms
- Cross-fertilisation of applications and theory
- Today: large impact on Semantic Web (sign up for INF3580/4580!)
Knowledge bases

In description logics one works with three different types of elements:

- individuals/constants (e.g. james, sensor1)
- concepts/unary relations (e.g. Person, Sensor)
- roles/binary relations (e.g. isFatherOf, isConnectedTo)

Knowledge is represented as a knowledge base, $K = \langle A, T \rangle$ where:

- $A$ is a set of assertions about named individuals, called the ABox (e.g. Person(james), isFatherOf(james, peter))
- $T$ is a set of terminology definitions (i.e. complex descriptions of concepts or roles), called the TBox (e.g. Human ⊑ Mammal, Mother ≡ Parent ⊓ Woman)
Knowledge bases

In description logics one works with three different types of elements:

– individuals/constants (e.g. \textit{james}, \textit{sensor1})
Knowledge bases

In description logics one works with three different types of elements:

– individuals/constants (e.g. *james*, *sensor1*)
– concepts/unary relations (e.g. *Person*, *Sensor*)
Knowledge bases

In description logics one works with three different types of elements:

– individuals/constants (e.g. James, sensor1)
– concepts/unary relations (e.g. Person, Sensor)
– roles/binary relations (e.g. isFatherOf, isConnectedTo)
In description logics one works with three different types of elements:

- individuals/constants (e.g. jame, sensor1)
- concepts/unary relations (e.g. Person, Sensor)
- roles-binary relations (e.g. isFatherOf, isConnectedTo)

Knowledge is represented as a knowledge base, $\mathcal{K} = \langle A, T \rangle$ where:

- $A$ is a set of assertions about named individuals, called the ABox (e.g. Person(james), isFatherOf(james, peter))
- $T$ is a set of terminology definitions (i.e. complex descriptions of concepts or roles), called the TBox (e.g. Human $\sqsubseteq$ Mammal, Mother $\equiv$ Parent $\sqcap$ Woman)
Knowledge bases

In description logics one works with three different types of elements:

- individuals/constants (e.g. *james*, *sensor1*)
- concepts/unary relations (e.g. *Person*, *Sensor*)
- roles/binary relations (e.g. *isFatherOf*, *isConnectedTo*)

Knowledge is represented as a knowledge base, $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ where:

- $\mathcal{A}$ is a set of assertions about named individuals, called the ABox (e.g. *Person(james)*, *isFatherOf(james, peter)*)
- $\mathcal{T}$ is a set of terminology definitions (i.e. complex descriptions of concepts or roles), called the TBox (e.g. *Human $\sqsubseteq$ Mammal*, *Mother $\equiv$ Parent $\cap$ Woman*)
Contents

Introduction

\textit{ALC}: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
\textbf{\textit{ALC}: Syntax}

The description logic \textit{ALC} (Attribute Language with general Complement) allows the following concepts:
\textbf{\textit{ALC}: Syntax}

The description logic \textit{ALC} (Attribute Language with general Complement) allows the following concepts:

\[
\begin{align*}
C, D & \to A & \text{ (atomic concept)} \\
\top & & \text{ (universal concept)} \\
\bot & & \text{ (bottom concept)} \\
\neg C & & \text{ (negation)} \\
C \sqcup D & & \text{ (union)} \\
C \sqcap D & & \text{ (intersection)} \\
\exists R.C & & \text{ (existential restriction)} \\
\forall R.C & & \text{ (universal restriction)}
\end{align*}
\]

where \(A\) is an atomic concept, \(C\) and \(D\) are concepts, and \(R\) is a role.

We allow

- ABox assertions: \(C(a)\) and \(R(a,b)\) for individuals \(a, b\), concepts \(C\) and roles \(R\);
- TBox axioms: \(C \sqsubseteq D\) for concepts \(C\) and \(D\).
**$ALC$: Syntax**

The description logic $ALC$ (Attribute Language with general Complement) allows the following concepts:

\[ C, D \rightarrow A \quad | \quad \text{(atomic concept)} \]
\[ \top \quad | \quad \text{(universal concept)} \]
\[ \bot \quad | \quad \text{(bottom concept)} \]
\[ \lnot C \quad | \quad \text{(negation)} \]
\[ C \sqcup D \quad | \quad \text{(union)} \]
\[ C \sqcap D \quad | \quad \text{(intersection)} \]
\[ \exists R.C \quad | \quad \text{(existential restriction)} \]
\[ \forall R.C \quad | \quad \text{(universal restriction)} \]

where $A$ is an atomic concept, $C$ and $D$ are concepts, and $R$ is a role.
**ALC: Syntax**

The description logic *ALC* (Attribute Language with general Complement) allows the following concepts:

\[ C, D \rightarrow A \] \hspace{1cm} \text{(atomic concept)}
\[ \top \] \hspace{1cm} \text{(universal concept)}
\[ \bot \] \hspace{1cm} \text{(bottom concept)}
\[ \neg C \] \hspace{1cm} \text{(negation)}
\[ C \sqcup D \] \hspace{1cm} \text{(union)}
\[ C \sqcap D \] \hspace{1cm} \text{(intersection)}
\[ \exists R.C \] \hspace{1cm} \text{(existential restriction)}
\[ \forall R.C \] \hspace{1cm} \text{(universal restriction)}

where \( A \) is an atomic concept, \( C \) and \( D \) are concepts, and \( R \) is a role. We allow

- ABox assertions: \( C(a) \) and \( R(a, b) \) for individuals \( a, b \), concepts \( C \) and roles \( R \);
- TBox axioms: \( C \sqsubseteq D \) for concepts \( C \) and \( D \).
A model \( \mathcal{M} \) for a knowledge base \( \mathcal{K} \) consists of

– a nonempty set \( \Delta \), and

– an interpretation function \( \cdot^{\mathcal{M}} \), such that:

  – for every constant \( c \), \( c^{\mathcal{M}} \in \Delta \),
  – for every atomic concept \( A \), \( A^{\mathcal{M}} \subseteq \Delta \),
  – for every atomic role \( R \), \( R^{\mathcal{M}} \subseteq \Delta \times \Delta \),
\( \mathcal{ALC} \): Semantics

\( \_^\mathcal{M} \) is extended inductively as
\[ M \] is extended inductively as

\[ \top^M = \Delta \]
\[ \bot^M = \emptyset \]
\[ (\neg C)^M = \Delta \setminus C^M \]
\[ (C \sqcup D)^M = C^M \cup D^M \]
\[ (C \sqcap D)^M = C^M \cap D^M \]
\[ (\forall R. C)^M = \{ a \in \Delta \mid \forall b \in \Delta \ (\langle a, b \rangle \in R^M \rightarrow b \in C^M) \} \]
\[ (\exists R. C)^M = \{ a \in \Delta \mid \exists b \in \Delta \ (\langle a, b \rangle \in R^M \land b \in C^M) \} \]
$\mathcal{ALC}$: Semantics

An interpretation $\mathcal{M}$ satisfies
- $C(a)$, denoted $\mathcal{M} \models C(a)$, iff $a^\mathcal{M} \in C^\mathcal{M}$;
\textbf{\textit{ALC}: Semantics}

An interpretation $\mathcal{M}$ satisfies

- $C(a)$, denoted $\mathcal{M} \models C(a)$, iff $a^\mathcal{M} \in C^\mathcal{M}$;
- $C \sqsubseteq D$, denoted $\mathcal{M} \models C \sqsubseteq D$, iff $C^\mathcal{M} \subseteq D^\mathcal{M}$;
\textbf{\textsc{ALC}: Semantics}

An interpretation $\mathcal{M}$ satisfies

- $C(a)$, denoted $\mathcal{M} \models C(a)$, iff $a^\mathcal{M} \in C^\mathcal{M}$;
- $C \sqsubseteq D$, denoted $\mathcal{M} \models C \sqsubseteq D$, iff $C^\mathcal{M} \subseteq D^\mathcal{M}$;
- $R \sqsubseteq P$, denoted $\mathcal{M} \models R \sqsubseteq P$, iff $R^\mathcal{M} \subseteq P^\mathcal{M}$.

As usual, we will write $\mathcal{K} \models \psi$ if for any model $\mathcal{M}$ we have that $\mathcal{M} \models \mathcal{K} \Rightarrow \mathcal{M} \models \psi$. 
**ALC: Semantics**

An interpretation $\mathcal{M}$ satisfies

- $C(a)$, denoted $\mathcal{M} \models C(a)$, iff $a^\mathcal{M} \in C^\mathcal{M}$;
- $C \sqsubseteq D$, denoted $\mathcal{M} \models C \sqsubseteq D$, iff $C^\mathcal{M} \subseteq D^\mathcal{M}$;
- $R \sqsubseteq P$, denoted $\mathcal{M} \models R \sqsubseteq P$, iff $R^\mathcal{M} \subseteq P^\mathcal{M}$.

As usual, we will write $\mathcal{K} \models \psi$ if for any model $\mathcal{M}$ we have that $\mathcal{M} \models \mathcal{K} \Rightarrow \mathcal{M} \models \psi$.

We will use the following shorthand notation:

- $C \equiv D$ instead of the two axioms $C \sqsubseteq D$ and $D \sqsubseteq C$;
- $\mathcal{A} \models \psi$ instead of $\langle \emptyset, \mathcal{A} \rangle \models \psi$;
- $\mathcal{T} \models \psi$ instead of $\langle \mathcal{T}, \emptyset \rangle \models \psi$. 
Example

TBox:

\[
\begin{align*}
\text{Animal} & \subseteq \text{LivingThing} \\
\text{Donkey} & \equiv \text{Animal} \cap \forall \text{hasParent}. \text{Donkey} \\
\text{Horse} & \equiv \text{Animal} \cap \forall \text{hasParent}. \text{Horse} \\
\text{Mule} & \equiv \text{Animal} \cap \exists \text{hasParent}. \text{Horse} \cap \exists \text{hasParent}. \text{Donkey} \\
\exists \text{hasParent}. \text{Mule} & \subseteq \bot
\end{align*}
\]
Example

TBox:

\[
\begin{align*}
\text{Animal} & \sqsubseteq \text{LivingThing} \\
\text{Donkey} & \equiv \text{Animal} \cap \forall \text{hasParent}.\text{Donkey} \\
\text{Horse} & \equiv \text{Animal} \cap \forall \text{hasParent}.\text{Horse} \\
\text{Mule} & \equiv \text{Animal} \cap \exists \text{hasParent}.\text{Horse} \cap \exists \text{hasParent}.\text{Donkey}
\end{align*}
\]

\[\exists \text{hasParent}.\text{Mule} \sqsubseteq \bot\]

ABox:

\[
\begin{align*}
\text{Horse}(\text{Mary}) & \quad \text{Mule}(\text{Peter}) & \quad \text{Donkey}(\text{Sven}) \\
\text{hasParent}(\text{Peter}, \text{Mary}) & \quad \text{hasParent}(\text{Peter}, \text{Carl})
\end{align*}
\]
Example

TBox:

\[ \text{Animal} \sqsubseteq \text{LivingThing} \]
\[ \text{Donkey} \equiv \text{Animal} \cap \forall \text{hasParent}. \text{Donkey} \]
\[ \text{Horse} \equiv \text{Animal} \cap \forall \text{hasParent}. \text{Horse} \]
\[ \text{Mule} \equiv \text{Animal} \cap \exists \text{hasParent}. \text{Horse} \cap \exists \text{hasParent}. \text{Donkey} \]
\[ \exists \text{hasParent}. \text{Mule} \sqsubseteq \bot \]

ABox:

\[ \text{Horse}(\text{Mary}) \quad \text{Mule}(\text{Peter}) \quad \text{Donkey}(\text{Sven}) \]

\[ \text{hasParent}(\text{Peter}, \text{Mary}) \quad \text{hasParent}(\text{Peter}, \text{Carl}) \]
\[ \text{hasParent}(\text{Sven}, \text{Hannah}) \quad \text{hasParent}(\text{Sven}, \text{Carl}) \]
Translation to First order logic

The function $\pi$ map concepts to first-order formulae:

$\pi(x(A)) = A(x)$

$\pi(x(\neg C)) = \neg \pi(x(C))$

$\pi(x(C \sqcup D)) = \pi(x(C)) \lor \pi(x(D))$

$\pi(x(C \sqcap D)) = \pi(x(C)) \land \pi(x(D))$

$\pi(x(\exists R.C)) = \exists y(R(x, y) \land \pi(y(C)))$

$\pi(x(\forall R.C)) = \forall y(R(x, y) \rightarrow \pi(y(C)))$

We can then map axioms: $\Pi(C \sqsubseteq D) := \forall x(\pi(x(C)) \rightarrow \pi(x(D)))$.

Theorem $a \in I$ iff $I|_{\pi(C[\alpha/x])}$, and $I \models C \sqsubseteq D$ iff $I|_{\Pi(C \sqsubseteq D)}$.

E.g.:

$\pi(x(Animal \sqcap \forall hasParent.Donkey)) = Animal(x) \land \forall y(hasParent(x, y) \rightarrow Donkey(y))$

$\Pi(Animal \sqsubseteq LivingThing) = \forall x(Animal(x) \rightarrow LivingThing(x))$
Translation to First order logic

The function \( \pi \) map concepts to first-order formulae:

\[
\begin{align*}
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg \pi_x(C) \\
\pi_x(C \sqcup D) &= \pi_x(C) \lor \pi_x(D) \\
\pi_x(C \sqcap D) &= \pi_x(C) \land \pi_x(D) \\
\pi_x(\exists R. C) &= \exists y \ (R(x, y) \land \pi_y(C)) \\
\pi_x(\forall R. C) &= \forall y \ (R(x, y) \rightarrow \pi_y(C))
\end{align*}
\]
Translation to First order logic

The function $\pi$ map concepts to first-order formulae:

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

We can then map axioms: $\Pi(C \sqsubseteq D) := \forall x (\pi_x(C) \rightarrow \pi_x(D))$. 
Translation to First order logic

The function $\pi$ map concepts to first-order formulae:

$$
\pi_x(A) = A(x)
$$

$$
\pi_x(\neg C) = \neg \pi_x(C)
$$

$$
\pi_x(C \sqcup D) = \pi_x(C) \lor \pi_x(D)
$$

$$
\pi_x(C \sqcap D) = \pi_x(C) \land \pi_x(D)
$$

$$
\pi_x(\exists R.C) = \exists y (R(x, y) \land \pi_y(C))
$$

$$
\pi_x(\forall R.C) = \forall y (R(x, y) \rightarrow \pi_y(C))
$$

We can then map axioms: $\Pi(C \sqsubseteq D) := \forall x(\pi_x(C) \rightarrow \pi_x(D))$.

**Theorem**

$a^\mathcal{I} \in C^\mathcal{I}$ iff $\mathcal{I} \models_{FOL} \pi_x(C)[a/x]$, and $\mathcal{I} \models C \sqsubseteq D$ iff $\mathcal{I} \models_{FOL} \Pi(C \sqsubseteq D)$. 
Translation to First order logic

The function $\pi$ map concepts to first-order formulae:

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

We can then map axioms: $\Pi(C \sqsubseteq D) := \forall x(\pi_x(C) \rightarrow \pi_x(D))$.

**Theorem**

$a^\mathcal{I} \in C^\mathcal{I}$ iff $\mathcal{I} \models_{\text{FOL}} \pi_x(C)[a/x]$, and $\mathcal{I} \models C \sqsubseteq D$ iff $\mathcal{I} \models_{\text{FOL}} \Pi(C \sqsubseteq D)$.

E.g.:

$$\pi_x(\text{Animal} \sqcap \forall \text{hasParent} . \text{Donkey}) = \text{Animal}(x) \land \forall y \, (\text{hasParent}(x, y) \rightarrow \text{Donkey}(y))$$

$$\Pi(\text{Animal} \sqsubseteq \text{LivingThing}) = \forall x(\text{Animal}(x) \rightarrow \text{LivingThing}(x))$$
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:

- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\} \rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \vDash C \sqsubseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \vDash C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \vDash C \sqcap D \sqsubseteq \bot$);

The following problems are of interest with respect to knowledge bases $\mathcal{K} = \langle \mathcal{T}, A \rangle$:

- Is $\mathcal{K}$ consistent ($\mathcal{K}$ has a model);
- Given a concept $C$ and an individual $a$, does $\mathcal{K}$ entail $C(a)$ ($\mathcal{K} \vDash C(a)$);
- Given a concept $C$, find all individuals $a$ such that $\mathcal{K}$ entails $C(a)$. 
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:
- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\} \rangle$ has a model);
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:

- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\} \rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \models C \sqsubseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \models C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \models C \sqcap D \sqsubseteq \bot$).
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:

- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\} \rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \vdash C \subseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \vdash C \equiv D$);
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:

- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\}\rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \models C \sqsubseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \models C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \models C \cap D \sqsubseteq \bot$);
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:
- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\} \rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \models C \sqsubseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \models C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \models C \cap D \sqsubseteq \bot$);

The following problems are of interest with respect to knowledge bases $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:

- Is $\mathcal{K}$ consistent ($\mathcal{K}$ has a model);
- Given a concept $C$ and an individual $a$, does $\mathcal{K}$ entail $C(a)$ ($\mathcal{K} \models C(a)$);
- Given a concept $C$, find all individuals $a$ such that $\mathcal{K}$ entails $C(a)$. 
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:

– Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{ C(x_0) \} \rangle$ has a model);
– Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \vDash C \subseteq D$);
– Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \vDash C \equiv D$);
– Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \vDash C \cap D \subseteq \bot$);

The following problems are of interest with respect to knowledge bases $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:

– Is $\mathcal{K}$ consistent ($\mathcal{K}$ has a model);
Reasoning problems

The following problems are of interest with respect to a TBox $\mathcal{T}$:
- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\}\rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \models C \subseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \models C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \models C \cap D \subseteq \bot$);

The following problems are of interest with respect to knowledge bases $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:
- Is $\mathcal{K}$ consistent ($\mathcal{K}$ has a model);
- Given a concept $C$ and an individual $a$, does $\mathcal{K}$ entail $C(a)$ ($\mathcal{K} \models C(a)$);
The following problems are of interest with respect to a TBox $\mathcal{T}$:

- Given a concept $C$, is $C$ satisfiable ($\langle \mathcal{T}, \{C(x_0)\}\rangle$ has a model);
- Given two concepts $C$ and $D$, is $C$ subsumed by $D$ ($\mathcal{T} \models C \subseteq D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ equivalent ($\mathcal{T} \models C \equiv D$);
- Given two concepts $C$ and $D$, are $C$ and $D$ disjoint ($\mathcal{T} \models C \cap D \sqsubseteq \bot$);

The following problems are of interest with respect to knowledge bases $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$:

- Is $\mathcal{K}$ consistent ($\mathcal{K}$ has a model);
- Given a concept $C$ and an individual $a$, does $\mathcal{K}$ entail $C(a)$ ($\mathcal{K} \models C(a)$);
- Given a concept $C$, find all individuals $a$ such that $\mathcal{K}$ entails $C(a)$.

Reasoning problems
Contents

Introduction

$\mathcal{ALC}$: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
Naming conventions

– As we have seen $\mathcal{ALC}$ is the Attribute Language with general Complement.
Naming conventions

- As we have seen $\mathcal{ALC}$ is the *Attribute Language with general Complement*.
- The $C$ actually denotes an extension of a more restrictive language $\mathcal{AL}$. 

...
Naming conventions

- As we have seen $\mathcal{ALC}$ is the \textit{Attribute Language with general Complement}.
- The $\mathcal{C}$ actually denotes an extension of a more restrictive language $\mathcal{AL}$.
- In a similar way, we have the following possible extensions of our logic:
Naming conventions

- As we have seen, $\mathcal{ALC}$ is the *Attribute Language with general Complement*.
- The $C$ actually denotes an extension of a more restrictive language $\mathcal{AL}$.
- In a similar way, we have the following possible extensions of our logic:
  - $\mathcal{H}$: Role hierarchies;
  - $\mathcal{R}$: Complex role hierarchies;
  - $\mathcal{N}$: Cardinality restrictions;
  - $\mathcal{Q}$: Qualified cardinality restrictions;
  - $\mathcal{O}$: Closed classes;
  - $\mathcal{I}$: Inverse roles;
  - $\mathcal{D}$: Datatypes;
  - ...
Naming conventions

- As we have seen $\mathcal{ALC}$ is the *Attribute Language with general Complement*.
- The $\mathcal{C}$ actually denotes an extension of a more restrictive language $\mathcal{AL}$.
- In a similar way, we have the following possible extensions of our logic:
  - $\mathcal{H}$: Role hierarchies;
  - $\mathcal{R}$: Complex role hierarchies;
  - $\mathcal{N}$: Cardinality restrictions;
  - $\mathcal{Q}$: Qualified cardinality restrictions;
  - $\mathcal{O}$: Closed classes;
  - $\mathcal{I}$: Inverse roles;
  - $\mathcal{D}$: Datatypes;
  - ... 

- We name the languages by adding the letters of the features to $\mathcal{ALC}$. So e.g. $\mathcal{ALCN}$ is $\mathcal{ALC}$ extended with cardinality restrictions and $\mathcal{ALCHI}$ is $\mathcal{ALC}$ extended with role hierarchies and inverse roles.
Naming conventions

- As we have seen $\mathcal{ALC}$ is the *Attribute Language with general Complement*.
- The $\mathcal{C}$ actually denotes an extension of a more restrictive language $\mathcal{AL}$.
- In a similar way, we have the following possible extensions of our logic:
  - $\mathcal{H}$: Role hierarchies;
  - $\mathcal{R}$: Complex role hierarchies;
  - $\mathcal{N}$: Cardinality restrictions;
  - $\mathcal{Q}$: Qualified cardinality restrictions;
  - $\mathcal{O}$: Closed classes;
  - $\mathcal{I}$: Inverse roles;
  - $\mathcal{D}$: Datatypes;
  - ...

- We name the languages by adding the letters of the features to $\mathcal{ALC}$. So e.g. $\mathcal{ALCN}$ is $\mathcal{ALC}$ extended with cardinality restrictions and $\mathcal{ALCHI}$ is $\mathcal{ALC}$ extended with role hierarchies and inverse roles.
- It is common to shorten $\mathcal{ALC}$ (extended with transitive roles) to just $\mathcal{S}$ for more advanced languages, so e.g. $\mathcal{SHOIN}$ is $\mathcal{ALC} + \mathcal{H} + \mathcal{O} + \mathcal{I} + \mathcal{N}$.
Normal extensions

- \( H \) – Role Hierarchies: We allow TBox axioms on the form \( R \sqsubseteq P \) for atomic roles.

  Semantics:

  \[ M \models R \sqsubseteq P \iff R^M \subseteq P^M \]

  e.g. \( \text{hasFather} \sqsubseteq \text{hasParent} \);
Normal extensions

- \( \mathcal{H} \) – Role Hierarchies: We allow TBox axioms on the form \( R \sqsubseteq P \) for atomic roles.
  Semantics:
  \[ \mathcal{M} \models R \sqsubseteq P \iff R^\mathcal{M} \subseteq P^\mathcal{M} \]
  e.g. \( \text{hasFather} \sqsubseteq \text{hasParent} \);

- \( \mathcal{R} \) – Complex role hierarchies: We allow roles on the form \( R \circ P \) and TBox axioms on the form \( R \circ P \sqsubseteq P \) and \( R \circ P \sqsubseteq R \) for any two roles.
  Semantics:
  \[ (R \circ P)^\mathcal{M} := \{ \langle a, b \rangle \in \Delta^\mathcal{M} \times \Delta^\mathcal{M} \mid \exists c \in \Delta^\mathcal{M} (\langle a, c \rangle \in R^\mathcal{M} \land \langle c, b \rangle \in P^\mathcal{M}) \} \]
  and
  \[ \mathcal{M} \models R \sqsubseteq P \iff R^\mathcal{M} \subseteq P^\mathcal{M} \]
  e.g. \( \text{friendOf} \circ \text{enemyOf} \sqsubseteq \text{enemyOf} \).
Normal extensions

– $\mathcal{N}$ – Cardinality restrictions: We allow concepts on the form $\leq n R$ and $\geq n R$ for any natural number $n$. Semantics$^1$:

$$(\leq n R)^M := \{a \in \Delta^M | \#\{b \in \Delta^M | \langle a, b \rangle \in R^M \} \leq n\}$$

$$(\geq n R)^M := \{a \in \Delta^M | \#\{b \in \Delta^M | \langle a, b \rangle \in R^M \} \geq n\}$$

e.g. $Mammal \sqsubseteq \leq 2 \text{ hasParent}$;

---

$^1$We let $\#S$ be the cardinality of the set $S$
Normal extensions

- $\mathcal{N}$ – Cardinality restrictions: We allow concepts on the form $\leq n \cdot R$ and $\geq n \cdot R$ for any natural number $n$. Semantics$^1$:

\[
(\leq n \cdot R)^M := \{ a \in \Delta^M \mid \# \{ b \in \Delta^M \mid \langle a, b \rangle \in R^M \} \leq n \}\\
(\geq n \cdot R)^M := \{ a \in \Delta^M \mid \# \{ b \in \Delta^M \mid \langle a, b \rangle \in R^M \} \geq n \}
\]

  e.g. $\text{Mammal} \sqsubseteq \leq 2 \cdot \text{hasParent}$;

- $Q$ – Qualified cardinality restrictions: We allow concepts on the form $\leq n \cdot R.C$ and $\geq n \cdot R.C$ for any natural number $n$. Semantics:

\[
(\leq n \cdot R.C)^M := \{ a \in \Delta^M \mid \# \{ b \in \Delta^M \mid \langle a, b \rangle \in R^M \land b \in C^M \} \leq n \}\\
(\geq n \cdot R.C)^M := \{ a \in \Delta^M \mid \# \{ b \in \Delta^M \mid \langle a, b \rangle \in R^M \land b \in C^M \} \geq n \}
\]

  e.g. $\text{RichPeople} \sqsubseteq \geq 2 \cdot \text{owns.House}$.

$^1$We let $\# S$ be the cardinality of the set $S$.
Normal extensions

– $\mathcal{O}$ – Closed classes: We allow concepts on the form $\{a_1, a_2, \ldots, a_n\}$ where $a_i$ are individuals. Semantics

\[
\left(\{a_1, a_2, \ldots, a_n\}\right)^M := \{a_1^M, a_2^M, \ldots, a_n^M\}
\]

e.g. $\text{Days} \sqsubseteq \{\text{monday}, \text{tuesday}, \text{wednesday}, \text{thursday}, \text{friday}, \text{saturday}, \text{sunday}\}$;
Normal extensions

- \( \mathcal{O} \) – Closed classes: We allow concepts on the form \( \{a_1, a_2, \ldots, a_n\} \) where \( a_i \) are individuals. Semantics

\[
(\{a_1, a_2, \ldots, a_n\})^M := \{a_1^M, a_2^M, \ldots, a_n^M\}
\]

e.g. \( \text{Days} \sqsubseteq \{\text{monday, tuesday, wednesday, thursday, friday, saturday, sunday}\} \);

- \( \mathcal{I} \) – Inverse roles: We allow roles on the form \( R^- \). Semantics:

\[
(R^-)^M := \{\langle a, b \rangle \in \Delta^M \times \Delta^M | \langle b, a \rangle \in R^M\}
\]

e.g. \( \text{hasParent}^- \sqsubseteq \text{isChildOf} \);
Normal extensions

- \( \mathcal{O} \) – Closed classes: We allow concepts on the form \( \{a_1, a_2, \ldots, a_n\} \) where \( a_i \) are individuals. Semantics

\[
(\{a_1, a_2, \ldots, a_n\})^M := \{a_1^M, a_2^M, \ldots, a_n^M\}
\]

E.g. \( \text{Days} \sqsubseteq \{\text{monday, tuesday, wednesday, thursday, friday, saturday, sunday}\} \);

- \( \mathcal{I} \) – Inverse roles: We allow roles on the form \( R^- \). Semantics:

\[
(R^-)^M := \{\langle a, b \rangle \in \Delta^M \times \Delta^M \mid \langle b, a \rangle \in R^M\}
\]

E.g. \( \text{hasParent}^- \sqsubseteq \text{isChildOf} \);

- \( \mathcal{D} \) - Datatypes: We introduce a set of datatypes: \text{int, string, float, boolean}, and so on. They all have a fixed interpretation, that is, the same for all models.
Examples

\[\text{OnlyChild} \sqsubseteq \text{Person} \land \neg \exists \text{hasSibling}. \top\]
Examples

\begin{align*}
\text{OnlyChild} & \subseteq \text{Person} \cap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \subseteq \leq 2 \text{hasParent}. \text{Animal} \cap \geq 2 \text{hasParent}. \text{Animal}
\end{align*}
Examples

\[
\begin{align*}
  \text{OnlyChild} & \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \\
  \text{Animal} & \sqsubseteq \leq 2 \text{hasParent}. \text{Animal} \sqcap \geq 2 \text{hasParent}. \text{Animal} \\
  \text{Pet} \sqcap \text{Person} & \sqsubseteq \bot
\end{align*}
\]
Examples

\[
\begin{align*}
\text{OnlyChild} & \subseteq \text{Person} \cap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \subseteq \leq 2 \text{hasParent}. \text{Animal} \cap \geq 2 \text{hasParent}. \text{Animal} \\
\text{Pet} \cap \text{Person} & \subseteq \bot \\
\text{Person} & \subseteq \exists \text{loves}. \{\text{mary}\} \\
\end{align*}
\]
Examples

\[ \text{OnlyChild} \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}.\top \]
\[ \text{Animal} \sqsubseteq \leq 2 \text{hasParent}.\text{Animal} \sqcap \geq 2 \text{hasParent}.\text{Animal} \]
\[ \text{Pet} \sqcap \text{Person} \sqsubseteq \bot \]
\[ \text{Person} \sqsubseteq \exists \text{loves}.\{\text{mary}\} \]
\[ \text{Norwegian} \sqsubseteq \exists \text{comesFrom}.\{\text{norway}\} \]
Examples

\[ \text{OnlyChild} \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \]
\[ \text{Animal} \sqsubseteq \leq 2 \text{hasParent}. \text{Animal} \sqcap \geq 2 \text{hasParent}. \text{Animal} \]
\[ \text{Pet} \sqcap \text{Person} \sqsubseteq \bot \]
\[ \text{Person} \sqsubseteq \exists \text{loves}. \{\text{mary}\} \]
\[ \text{Norwegian} \sqsubseteq \exists \text{comesFrom}. \{\text{norway}\} \]
\[ \{\text{adam}\} \sqsubseteq \neg \{\text{eve}\} \]
Examples

\[
\begin{align*}
OnlyChild & \subseteq Person \cap \neg \exists \text{hasSibling}. \top \\
Animal & \subseteq \leq 2 \text{hasParent}. \text{Animal} \cap \geq 2 \text{hasParent}. \text{Animal} \\
Pet \cap Person & \subseteq \bot \\
Person & \subseteq \exists \text{loves.\{mary\}} \\
Norwegian & \subseteq \exists \text{comesFrom.\{norway\}} \\
\{adam\} & \subseteq \neg \{eve\} \\
\text{hasFather} \circ \text{hasBrother} & \subseteq \text{hasUncle}
\end{align*}
\]
Examples

\[
\begin{align*}
\text{OnlyChild} & \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}.\top \\
\text{Animal} & \sqsubseteq \leq 2 \text{hasParent}.\text{Animal} \sqcap \geq 2 \text{hasParent}.\text{Animal} \\
\text{Pet} \sqcap \text{Person} & \sqsubseteq \bot \\
\text{Person} & \sqsubseteq \exists \text{loves}.\{\text{mary}\} \\
\text{Norwegian} & \sqsubseteq \exists \text{comesFrom}.\{\text{norway}\} \\
\{\text{adam}\} & \sqsubseteq \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} & \sqsubseteq \text{hasUncle} \\
\exists R.\top & \sqsubseteq C
\end{align*}
\]
Examples

\[
\begin{align*}
\text{OnlyChild} & \subseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \subseteq \leq 2 \text{hasParent}. \text{Animal} \sqcap \geq 2 \text{hasParent}. \text{Animal} \\
\text{Pet} \sqcap \text{Person} & \subseteq \bot \\
\text{Person} & \subseteq \exists \text{loves}. \{\text{mary}\} \\
\text{Norwegian} & \subseteq \exists \text{comesFrom}. \{\text{norway}\} \\
\{\text{adam}\} & \subseteq \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} & \subseteq \text{hasUncle} \\
\exists R. \top & \subseteq C \\
\end{align*}
\]
Examples

<table>
<thead>
<tr>
<th>Class</th>
<th>Inclusion</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OnlyChild</td>
<td>⊑ Person ¬∃ hasSibling. ⊤</td>
<td></td>
</tr>
<tr>
<td>Animal</td>
<td>⊑ ≤ 2 hasParent. Animal ⊓ ≥ 2 hasParent. Animal</td>
<td></td>
</tr>
<tr>
<td>Pet ⊓ Person</td>
<td>⊑ ⊥</td>
<td></td>
</tr>
<tr>
<td>Person</td>
<td>⊑ ∃ loves. {mary}</td>
<td></td>
</tr>
<tr>
<td>Norwegian</td>
<td>⊑ ∃ comesFrom. {norway}</td>
<td></td>
</tr>
<tr>
<td>{adam}</td>
<td>⊑ ¬ {eve}</td>
<td></td>
</tr>
<tr>
<td>hasFather ◦ hasBrother</td>
<td>⊑ hasUncle</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relation</th>
<th>Inclusion</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>∃ R. ⊤</td>
<td>⊑ C</td>
<td>Domain</td>
</tr>
<tr>
<td>⊤</td>
<td>⊑ ∀ R. C</td>
<td></td>
</tr>
</tbody>
</table>
Examples

\[\text{OnlyChild} \sqsubseteq \text{Person \land \neg \exists \text{hasSibling}.\top}\]
\[\text{Animal} \sqsubseteq \leq 2 \text{hasParent}.\text{Animal} \sqcap \geq 2 \text{hasParent}.\text{Animal}\]
\[\text{Pet} \sqcap \text{Person} \sqsubseteq \bot\]
\[\text{Person} \sqsubseteq \exists \text{loves}.\{\text{mary}\}\]
\[\text{Norwegian} \sqsubseteq \exists \text{comesFrom}.\{\text{norway}\}\]
\[\{\text{adam}\} \sqsubseteq \neg \{\text{eve}\}\]
\[\text{hasFather} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}\]

\[\exists R.\top \sqsubseteq C\]
\[\top \sqsubseteq \forall R.\ C\]

Domain

Range
Examples

\[
\begin{align*}
\text{OnlyChild} & \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \sqsubseteq \leq 2 \text{hasParent. Animal} \sqcap \geq 2 \text{hasParent. Animal} \\
\text{Pet} \sqcap \text{Person} & \sqsubseteq \bot \\
\text{Person} & \sqsubseteq \exists \text{loves.} \{\text{mary}\} \\
\text{Norwegian} & \sqsubseteq \exists \text{comesFrom.} \{\text{norway}\} \\
\{\text{adam}\} & \sqsubseteq \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} & \sqsubseteq \text{hasUncle} \\
\exists R. \top & \sqsubseteq C \\
\top & \sqsubseteq \forall R. C \\
R \circ R & \sqsubseteq R
\end{align*}
\]
Examples

\[ \text{OnlyChild} \subseteq \text{Person} \sqcap \lnot \exists \text{hasSibling}. \top \]
\[ \text{Animal} \subseteq \leq 2 \text{hasParent}. \text{Animal} \sqcap \geq 2 \text{hasParent}. \text{Animal} \]
\[ \text{Pet} \sqcap \text{Person} \subseteq \bot \]
\[ \text{Person} \subseteq \exists \text{loves}. \{ \text{mary} \} \]
\[ \text{Norwegian} \subseteq \exists \text{comesFrom}. \{ \text{norway} \} \]
\[ \{ \text{adam} \} \subseteq \lnot \{ \text{eve} \} \]
\[ \text{hasFather} \circ \text{hasBrother} \subseteq \text{hasUncle} \]

\[ \exists R. \top \subseteq C \]
\[ \top \subseteq \forall R. C \]
\[ R \circ R \subseteq R \]

Domain
Range
Transitivity
Examples

\[ \text{OnlyChild} \subseteq \text{Person} \land \neg \exists \text{hasSibling}.\top \]
\[ \text{Animal} \subseteq \leq 2 \text{hasParent}.\text{Animal} \land \geq 2 \text{hasParent}.\text{Animal} \]
\[ \text{Pet} \land \text{Person} \subseteq \bot \]
\[ \text{Person} \subseteq \exists \text{loves}.\{\text{mary}\} \]
\[ \text{Norwegian} \subseteq \exists \text{comesFrom}.\{\text{norway}\} \]
\[ \{\text{adam}\} \subseteq \neg \{\text{eve}\} \]
\[ \text{hasFather} \circ \text{hasBrother} \subseteq \text{hasUncle} \]

\[ \exists R.\top \subseteq C \quad \text{Domain} \]
\[ \top \subseteq \forall R. C \quad \text{Range} \]
\[ R \circ R \subseteq R \quad \text{Transitivity} \]
\[ \top \subseteq \leq 1 R.\top \]
Examples

\begin{align*}
\text{OnlyChild} & \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \sqsubseteq \leq 2 \text{hasParent.} \text{Animal} \sqcap \geq 2 \text{hasParent.} \text{Animal} \\
\text{Pet} \sqcap \text{Person} & \sqsubseteq \bot \\
\text{Person} & \sqsubseteq \exists \text{loves.} \{\text{mary}\} \\
\text{Norwegian} & \sqsubseteq \exists \text{comesFrom.} \{\text{norway}\} \\
\{\text{adam}\} & \sqsubseteq \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} & \sqsubseteq \text{hasUncle} \\
\exists R. \top & \sqsubseteq C & \text{Domain} \\
\top & \sqsubseteq \forall R. C & \text{Range} \\
R \circ R & \sqsubseteq R & \text{Transitivity} \\
\top & \sqsubseteq \leq 1 R. \top & \text{Functionality}
\end{align*}
Examples

\(\text{OnlyChild} \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}.\top\)
\(\text{Animal} \sqsubseteq \leq 2 \text{hasParent}.\text{Animal} \sqcap \geq 2 \text{hasParent}.\text{Animal}\)
\(\text{Pet} \sqcap \text{Person} \sqsubseteq \bot\)
\(\text{Person} \sqsubseteq \exists \text{loves}.\{\text{mary}\}\)
\(\text{Norwegian} \sqsubseteq \exists \text{comesFrom}.\{\text{norway}\}\)
\(\{\text{adam}\} \sqsubseteq \neg \{\text{eve}\}\)
\(\text{hasFather} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}\)

\(\exists R. \top \sqsubseteq C\)
\(\top \sqsubseteq \forall R.C\)
\(R \circ R \sqsubseteq R\)
\(\top \sqsubseteq \leq 1 R. \top\)
\(R \sqsubseteq R^-\)

Domain
Range
Transitivity
Functionality
Examples

\[
\begin{align*}
\text{OnlyChild} \subseteq & \text{Person} \cap \neg \exists \text{hasSibling}. \top \\
\text{Animal} \subseteq & \leq 2 \text{hasParent}. \text{Animal} \cap \geq 2 \text{hasParent}. \text{Animal} \\
\text{Pet} \cap \text{Person} \subseteq & \bot \\
\text{Person} \subseteq & \exists \text{loves}. \{\text{mary}\} \\
\text{Norwegian} \subseteq & \exists \text{comesFrom}. \{\text{norway}\} \\
\{\text{adam}\} \subseteq & \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} \subseteq & \text{hasUncle}
\end{align*}
\]

\[
\begin{align*}
\exists R. \top \subseteq & C \quad \text{Domain} \\
\top \subseteq & \forall R. C \\
R \circ R \subseteq & R \quad \text{Transitivity} \\
\top \subseteq & \leq 1 R. \top \\
R \subseteq & R^- \quad \text{Functionality} \\
\end{align*}
\]
Examples

\[
\text{OnlyChild} \subseteq \text{Person} \land \neg \exists \text{hasSibling} \cdot \top
\]

\[
\text{Animal} \subseteq \leq 2 \text{hasParent} \cdot \text{Animal} \land \geq 2 \text{hasParent} \cdot \text{Animal}
\]

\[
\text{Pet} \cap \text{Person} \subseteq \bot
\]

\[
\exists \text{loves} \cdot \{\text{mary}\}
\]

\[
\text{Norwegian} \subseteq \exists \text{comesFrom} \cdot \{\text{norway}\}
\]

\[
\{\text{adam}\} \subseteq \neg \{\text{eve}\}
\]

\[
\text{hasFather} \circ \text{hasBrother} \subseteq \text{hasUncle}
\]

\[
\exists R \cdot \top \subseteq C \quad \text{Domain}
\]

\[
\top \subseteq \forall R \cdot C \quad \text{Range}
\]

\[
R \circ R \subseteq R \quad \text{Transitivity}
\]

\[
\top \subseteq \leq 1 R \cdot \top \quad \text{Functionality}
\]

\[
R \subseteq R^\sim \quad \text{Symmetry}
\]

\[
R \subseteq \neg R^\sim
\]
Examples

\[
\begin{align*}
\text{OnlyChild} & \sqsubseteq \text{Person} \sqcap \neg \exists \text{hasSibling}. \top \\
\text{Animal} & \sqsubseteq \leq 2 \text{hasParent}. \text{Animal} \sqcap \geq 2 \text{hasParent}. \text{Animal} \\
\text{Pet} \sqcap \text{Person} & \sqsubseteq \bot \\
\text{Person} & \sqsubseteq \exists \text{loves}. \{\text{mary}\} \\
\text{Norwegian} & \sqsubseteq \exists \text{comesFrom}. \{\text{norway}\} \\
\{\text{adam}\} & \sqsubseteq \neg \{\text{eve}\} \\
\text{hasFather} \circ \text{hasBrother} & \sqsubseteq \text{hasUncle}
\end{align*}
\]

\[
\begin{align*}
\exists R. \top & \sqsubseteq C & \text{Domain} \\
\top & \sqsubseteq \forall R. C & \text{Range} \\
R \circ R & \sqsubseteq R & \text{Transitivity} \\
\top & \sqsubseteq \leq 1 R. \top & \text{Functionality} \\
R & \sqsubseteq R^\sim & \text{Symmetry} \\
R & \sqsubseteq \neg R^\sim & \text{Asymmetry}
\end{align*}
\]
Complexity results

http://www.cs.man.ac.uk/~ezolin/dl/
Common restricted languages: $\mathcal{EL}$

The description logic $\mathcal{EL}$ allow the following concepts:
Common restricted languages: $\mathcal{EL}$

The description logic $\mathcal{EL}$ allow the following concepts:

- $C, D \rightarrow A$ (atomic concept)
- $\top$ (universal concept)
- $\bot$ (bottom concept)
- $\{a\}$ (singular enumeration)
- $C \sqcap D$ (intersection)
- $\exists R.C$ (existential restriction)

with the following axioms:

- $C \sqsubseteq D$ and $C \equiv D$ for concept descriptions $D$ and $C$.
- $P \sqsubseteq Q$ and $P \equiv Q$ for roles $P$, $Q$.
- $C(a)$ and $R(a, b)$ for concept $C$, role $R$ and individuals $a$, $b$. 
Common restricted languages: $\mathcal{EL}$

The description logic $\mathcal{EL}$ allow the following concepts:

\[
\begin{align*}
C, D \rightarrow & \quad A & \text{(atomic concept)} \\
\top & \quad \text{(universal concept)} \\
\bot & \quad \text{(bottom concept)} \\
\{a\} & \quad \text{(singular enumeration)} \\
C \sqcap D & \quad \text{(intersection)} \\
\exists R.C & \quad \text{(existential restriction)}
\end{align*}
\]

with the following axioms:

- $C \sqsubseteq D$ and $C \equiv D$ for concept descriptions $D$ and $C$.
- $P \sqsubseteq Q$ and $P \equiv Q$ for roles $P, Q$.
- $C(a)$ and $R(a, b)$ for concept $C$, role $R$ and individuals $a, b$. 
Common restricted languages: \( \mathcal{EL} \)

Not supported (excerpt):
- negation, (only disjointness of classes: \( C \cap D \sqsubseteq \bot \)),
- disjunction,
- universal quantification,
- cardinalities,
- inverse roles,
- plus some role characteristics.
Common restricted languages: $\mathcal{EL}$

Not supported (excerpt):
- negation, (only disjointness of classes: $C \cap D \sqsubseteq \bot$),
- disjunction,
- universal quantification,
- cardinalities,
- inverse roles,
- plus some role characteristics.

- Captures language used for many large ontologies.
- Checking ontology consistency, class expression subsumption, and instance checking is in $\mathbf{P}$.
- “Good for large ontologies.”
Common restricted languages: *DL-Lite*

The description logic *DL-Lite*$_R$ allows the following concepts:
Common restricted languages: *DL-Lite*

The description logic *DL-Lite*\(_R\) allows the following concepts:

\[
\begin{align*}
C & \rightarrow \ A & \text{ (atomic concept)} \\
   & \quad \exists R. \top & \text{ (existential restriction with } \top \text{ only)}
\end{align*}
\]

\[
\begin{align*}
D & \rightarrow \ A & \text{ (atomic concept)} \\
   & \quad \exists R.D & \text{ (existential restriction)} \\
   & \quad \neg D & \text{ (negation)} \\
   & \quad D \sqcap D' & \text{ (intersection)}
\end{align*}
\]
Common restricted languages: *DL-Lite*

The description logic $DL-Lite_R$ allows the following concepts:

\[
\begin{align*}
C & \rightarrow A & \text{(atomic concept)} \\
\exists R. \top & \text{ (existential restriction with } \top \text{ only)} \\
D & \rightarrow A & \text{(atomic concept)} \\
\exists R. D & \text{ (existential restriction)} \\
\neg D & \text{ (negation)} \\
D \cap D' & \text{ (intersection)}
\end{align*}
\]

with the following axioms:

\begin{itemize}
  \item $C \sqsubseteq D$ for concept descriptions $D$ and $C$ (and $C \equiv C'$).
  \item $P \sqsubseteq Q$ and $P \equiv Q$ for roles $P, Q$.
  \item $C(a)$ and $R(a, b)$ for concept $C$, role $R$ and individuals $a, b$.
\end{itemize}
Common restricted languages: *DL-Lite*

Not supported (excerpt):
- disjunction,
- universal quantification,
- cardinalities,
- functional roles, keys,
- enumerations (closed classes),
- subproperties of chains, transitivity
Common restricted languages: *DL-Lite*

Not supported (excerpt):
- disjunction,
- universal quantification,
- cardinalities,
- functional roles, keys,
- enumerations (closed classes),
- subproperties of chains, transitivity

- Captures language for which queries can be translated to SQL.
  - Conjunctive queries over a *DL-Lite* knowledge base can be expanded with the TBox to a conjunctive query that can be answered over the Abox alone. This is called *first order rewritability*.
- “Good for large datasets.”
Common restricted languages: $\mathcal{RL}$

The description logic $\mathcal{RL}$ (also called DLP) allow the following concepts:
Common restricted languages: $\mathcal{RL}$

The description logic $\mathcal{RL}$ (also called DLP) allow the following concepts:

$$
\begin{align*}
C \to & A & \text{(atomic concept)} \\
C \sqcap C' & \text{(intersection)} \\
C \sqcup C' & \text{(union)} \\
\exists R.C & \text{(existential restriction)} \\
D \to & A & \text{(atomic concept)} \\
D \sqcap D' & \text{(intersection)} \\
\forall R.D & \text{(universal restriction)}
\end{align*}
$$
Common restricted languages: $\mathcal{RL}$

The description logic $\mathcal{RL}$ (also called DLP) allow the following concepts:

$C \rightarrow A$ \quad (atomic concept)
$C \cap C'$ \quad (intersection)
$C \cup C'$ \quad (union)
$\exists R.C$ \quad (existential restriction)

$D \rightarrow A$ \quad (atomic concept)
$D \cap D'$ \quad (intersection)
$\forall R.D$ \quad (universal restriction)

with the following axioms:

- $C \sqsubseteq D$, $C \equiv C'$, $\top \sqsubseteq \forall P.D$, $\top \sqsubseteq \forall P^- . D$ $P \sqsubseteq Q$, $P \equiv Q^{-}$ and $P \equiv Q$ for roles $P, Q$ and concept descriptions $D$ and $C$.
- $C(a)$ and $R(a, b)$ for concept $C$, role $R$ and individuals $a, b$. 
Contents

Introduction

$\mathcal{ALC}$: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
OWL and the Semantic Web

– OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
OWL and the Semantic Web

– OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
– They combine Web technology with description logic to make an *intelligent* web of data.
OWL and the Semantic Web

- OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
- They combine Web technology with description logic to make an intelligent web of data.
- In OWL, all individuals, concepts and roles are assigned a URI (Unique Resource Identifier).
OWL and the Semantic Web

- OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
- They combine Web technology with description logic to make an *intelligent* web of data.
- In OWL, all individuals, concepts and roles are assigned a URI (Unique Resource Identifier).
  - These URIs can be URLs, hence they can state where we can find more information about an item.
OWL and the Semantic Web

- OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
- They combine Web technology with description logic to make an *intelligent* web of data.
- In OWL, all individuals, concepts and roles are assigned a URI (Unique Resource Identifier).
  - These URIs can be URLs, hence they can state where we can find more information about an item.
  - URIs can be set to be equal, so we can link two ontologies together by stating which URIs denote the same thing in different contexts.
OWL and the Semantic Web

- OWL (The Web Ontology Language) is a set of ontology languages with semantics based on description logics.
- They combine Web technology with description logic to make an intelligent web of data.
- In OWL, all individuals, concepts and roles are assigned a URI (Unique Resource Identifier).
  - These URIs can be URLs, hence they can state where we can find more information about an item.
  - URIs can be set to be equal, so we can link two ontologies together by stating which URIs denote the same thing in different contexts.
- OWL provides a concrete syntax for writing axioms, implementations of reasoners over the axioms, and a query language that applies the reasoners for knowledge extraction.
OWL 2 Profiles

- OWL has various *profiles* that correspond to different DLs.

- OWL Lite: SHIF (D);
- OWL DL: corresponds to SHION (D);
- OWL 2 DL: corresponds to SROIQ (D) and is the "normal" OWL 2 (sublanguage): "maximum" expressivity while keeping reasoning problems decidable—but still very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite R, and is specifically designed for efficient database integration;
  - OWL 2 EL: Corresponds to EL, and is a lightweight language with polynomial time reasoning;
  - OWL 2 RL: Corresponds to RL, and is designed for compatibility with rule-based inference tools.

- OWL Full (not a proper DL): Anything goes: classes, relations, individuals, highly expressive, not decidable. But we want OWL's reasoning capabilities, so stay away if you can—and you almost always can.
OWL 2 Profiles

- OWL has various *profiles* that correspond to different DLs.
- OWL Lite: $SHIF(D)$;
OWL 2 Profiles

- OWL has various profiles that correspond to different DLs.
  - OWL Lite: $SHIF(D)$;
  - OWL DL: corresponds to $SHION(D)$;
OWL 2 Profiles

- OWL has various profiles that correspond to different DLs.
- OWL Lite: $SHIF(D)$;
- OWL DL: corresponds to $SHION(D)$;
- OWL 2 DL: corresponds to $SROIQ(D)$ and is the “normal” OWL 2 (sublanguage): “maximum” expressivity while keeping reasoning problems decidable—but still very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite $R$,
  - OWL 2 EL: Corresponds to $EL$,
  - OWL 2 RL: Corresponds to $RL$,
  - OWL Full (not a proper DL): Anything goes: classes, relations, individuals, highly expressive, not decidable. But we want OWL's reasoning capabilities, so stay away if you can—and you almost always can.
OWL 2 Profiles

- OWL has various *profiles* that correspond to different DLs.
- OWL Lite: $SHIF(D)$;
- OWL DL: corresponds to $SHION(D)$;
- OWL 2 DL: corresponds to $SROIQ(D)$ and is the “normal” OWL 2 (sublanguage): “maximum” expressivity while keeping reasoning problems decidable—but still very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite$_R$, and is specifically designed for efficient database integration;
OWL 2 Profiles

- OWL has various profiles that correspond to different DLs.
- OWL Lite: \( \mathcal{SHIF}(\mathcal{D}) \);
- OWL DL: corresponds to \( \mathcal{SHION}(\mathcal{D}) \);
- OWL 2 DL: corresponds to \( \mathcal{SROIQ}(\mathcal{D}) \) and is the “normal” OWL 2 (sublanguage): “maximum” expressivity while keeping reasoning problems decidable—but still very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite\(_r\), and is specifically designed for efficient database integration;
  - OWL 2 EL: Corresponds to \( \mathcal{EL} \), and is a lightweight language with polynomial time reasoning;
OWL 2 Profiles

- OWL has various profiles that correspond to different DLs.
- OWL Lite: $SHIF(D)$;
- OWL DL: corresponds to $SHION(D)$;
- OWL 2 DL: corresponds to $SROIQ(D)$ and is the “normal” OWL 2 (sublanguage): “maximum” expressivity while keeping reasoning problems decidable—but still very very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite$_R$, and is specifically designed for efficient database integration;
  - OWL 2 EL: Corresponds to $E\mathcal{L}$, and is a lightweight language with polynomial time reasoning;
  - OWL 2 RL: Corresponds to $R\mathcal{L}$, and is designed for compatibility with rule-based inference tools.

– OWL Full (not a proper DL): Anything goes: classes, relations, individuals, highly expressive, not decidable. But we want OWL’s reasoning capabilities, so stay away if you can—and you almost always can.
OWL 2 Profiles

- OWL has various profiles that correspond to different DLs.
- OWL Lite: $\text{SHIF}(\mathcal{D})$;
- OWL DL: corresponds to $\text{SHION}(\mathcal{D})$;
- OWL 2 DL: corresponds to $\text{SROIQ}(\mathcal{D})$ and is the “normal” OWL 2 (sublanguage): “maximum” expressivity while keeping reasoning problems decidable—but still very expensive;
- (Other) profiles are tailored for specific ends, e.g.,
  - OWL 2 QL: Corresponds to DL-Lite$_{\mathcal{R}}$, and is specifically designed for efficient database integration;
  - OWL 2 EL: Corresponds to $\mathcal{EL}$, and is a lightweight language with polynomial time reasoning;
  - OWL 2 RL: Corresponds to $\mathcal{RL}$, and is designed for compatibility with rule-based inference tools.
- OWL Full (not a proper DL): Anything goes: classes, relations, individuals, highly expressive, not decidable. But we want OWL’s reasoning capabilities, so stay away if you can—and you almost always can.
Contents

Introduction

$ALC$: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix
What cannot be expressed in DLs: Brothers

- Given terms

  \[ \text{hasSibling} \quad \text{Male} \]

- ... a brother is *defined* to be a sibling who is male

- Best try:

  \[ \text{hasBrother} \sqsubseteq \text{hasSibling} \]

  \[ T \sqsubseteq \forall \text{hasBrother}.\text{Male} \]

  \[ \exists \text{hasSibling}.\text{Male} \sqsubseteq \exists \text{hasBrother}.T \]

- Not enough to infer that *all* male siblings are brothers
What cannot be expressed in DLs: Diamond Properties

– A semi-detached house has a left and a right unit
– Each unit has a separating wall
– The separating walls of the left and right units are the same
– “diamond property”
– Try...

\[ \text{SemiDetached} \sqsubseteq \exists \text{hasLeftUnit.Unit} \sqcap \exists \text{hasRightUnit.Unit} \]
\[ \text{Unit} \sqsubseteq \exists \text{hasSeparatingWall.Wall} \]

– And now what?
What cannot be expressed in DLs: Connecting Properties

- Given terms

  \[ \text{Person} \quad \text{hasChild} \quad \text{hasBirthday} \]

- A twin parent is defined to be a person who has two children with the same birthday.

- Try...

  \[ \text{TwinParent} \equiv \text{Person} \sqcap \exists \text{hasChild}.\exists \text{hasBirthday} [...] \sqcap \exists \text{hasChild}.\exists \text{hasBirthday} [...] \]

- No way to connect the two birthdays to say that they’re the same.
  - (and no way to say that the children are not the same)

- Try...

  \[ \text{TwinParent} \equiv \text{Person} \sqcap \geq 2 \text{hasChild}.\exists \text{hasBirthday} [...] \]

- Still no way of connecting the birthdays
Reasoning about Numbers

- Reasoning about natural numbers is undecidable in general.
- DL Reasoning is decidable
- Therefore, general reasoning about numbers can’t be “encoded” in DL
- For instance, there is no largest prime number:

\[ \forall n. \exists p. (p > n \land \forall k, l. p = k \cdot l \rightarrow (k = 1 \lor l = 1)) \]

- Could try...

\[
\begin{align*}
\text{Number}(\text{zero}) \\
\text{Number} \sqsubseteq \exists \text{hasSuccessor}. \text{Number} \\
\top \sqsubseteq \leq 1 \text{ hasSuccessor}. \top
\end{align*}
\]

- Cannot encode addition, multiplication, etc.
- Note: a lot can be done with other logics, but not with DLs
  - Outside the intended scope of Description Logics
FO-rewritability

Assume $\mathcal{T}_L$ is the set of TBoxes over the language $L$, and that $UCQ$ is the set of queries that are unions of conjunctive queries, and let

$$\mathcal{K} \models q_1 \lor q_2 \iff \mathcal{K} \models q_1 \text{ or } \mathcal{K} \models q_2$$

$$\mathcal{K} \models q_1 \land q_2 \iff \mathcal{K} \models q_1 \text{ and } \mathcal{K} \models q_2$$

A description logic $L$ enjoys first order rewritability if there exists a rewriting function $\rho : \mathcal{T}_L \times UCQ \rightarrow UCQ$, such that for any knowledge base $\mathcal{K} = \langle T, A \rangle$ over $L$ and any conjunctive query $q(\vec{x})$ over $\mathcal{K}$ we have that

$$A \models \rho(T, q(\vec{a})) \iff \mathcal{K} \models q(\vec{a})$$

This allows us to divide the querying up into two stages: i) translation of the query, and ii) ABox querying. This is useful for e.g. translating a query from a DL query to an SQL query where the ABox is a relational database.

E.g. let $T := \{ C_1 \sqsubseteq D, C_2 \sqsubseteq D, A \sqsubseteq C_1 \}$ and $q(x) := D(x)$ we have that for any Abox $A$ that

$$A \models D(a) \lor C_1(a) \lor C_2(a) \lor A(a) \iff \langle T, A \rangle \models D(a)$$