

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

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:

Knowledge bases

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

- individuals/constants (e.g. *james*, *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*)

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)*)

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. $Human \sqsubseteq Mammal$, $Mother \equiv Parent \sqcap Woman$)

Contents

Introduction

ALC: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix

\mathcal{ALC} : Syntax

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

\mathcal{ALC} : Syntax

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

$C, D \rightarrow$	A		(atomic concept)
	\top		(universal concept)
	\perp		(bottom concept)
	$\neg C$		(negation)
	$C \sqcup D$		(union)
	$C \sqcap D$		(intersection)
	$\exists R.C$		(existential restriction)
	$\forall R.C$		(universal restriction)

\mathcal{ALC} : Syntax

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

$C, D \rightarrow$	A		(atomic concept)
	\top		(universal concept)
	\perp		(bottom concept)
	$\neg C$		(negation)
	$C \sqcup D$		(union)
	$C \sqcap D$		(intersection)
	$\exists R.C$		(existential restriction)
	$\forall R.C$		(universal restriction)

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

\mathcal{ALC} : Syntax

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

$C, D \rightarrow$	A		(atomic concept)
	\top		(universal concept)
	\perp		(bottom concept)
	$\neg C$		(negation)
	$C \sqcup D$		(union)
	$C \sqcap D$		(intersection)
	$\exists R.C$		(existential restriction)
	$\forall R.C$		(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 .

\mathcal{ALC} : Semantics

A model \mathcal{M} for a knowledge base \mathcal{K} consists of

- a nonempty set Δ , and
- an interpretation function $_{\mathcal{M}}$, 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

$\models^{\mathcal{M}}$ is extended inductively as

\mathcal{ALC} : Semantics

$_^{\mathcal{M}}$ is extended inductively as

$$\top^{\mathcal{M}} = \Delta$$

$$\perp^{\mathcal{M}} = \emptyset$$

$$(\neg C)^{\mathcal{M}} = \Delta \setminus C^{\mathcal{M}}$$

$$(C \sqcup D)^{\mathcal{M}} = C^{\mathcal{M}} \cup D^{\mathcal{M}}$$

$$(C \sqcap D)^{\mathcal{M}} = C^{\mathcal{M}} \cap D^{\mathcal{M}}$$

$$(\forall R.C)^{\mathcal{M}} = \{a \in \Delta \mid \forall b \in \Delta (\langle a, b \rangle \in R^{\mathcal{M}} \rightarrow b \in C^{\mathcal{M}})\}$$

$$(\exists R.C)^{\mathcal{M}} = \{a \in \Delta \mid \exists b \in \Delta (\langle a, b \rangle \in R^{\mathcal{M}} \wedge b \in C^{\mathcal{M}})\}$$

\mathcal{ALC} : Semantics

An interpretation \mathcal{M} satisfies

- $C(a)$, denoted $\mathcal{M} \models C(a)$, iff $a^{\mathcal{M}} \in C^{\mathcal{M}}$;

\mathcal{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 $C^{\mathcal{M}} \subseteq D^{\mathcal{M}}$;

\mathcal{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 $\mathcal{M} \models \mathcal{K} \Rightarrow \mathcal{M} \models \psi$.

\mathcal{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:

$Animal \sqsubseteq LivingThing$

$Donkey \equiv Animal \sqcap \forall hasParent.Donkey$

$Horse \equiv Animal \sqcap \forall hasParent.Horse$

$Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey$

$\exists hasParent.Mule \sqsubseteq \perp$

Example

TBox:

$Animal \sqsubseteq LivingThing$

$Donkey \equiv Animal \sqcap \forall hasParent.Donkey$

$Horse \equiv Animal \sqcap \forall hasParent.Horse$

$Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey$

$\exists hasParent.Mule \sqsubseteq \perp$

ABox:

$Horse(Mary) \quad Mule(Peter) \quad Donkey(Sven)$

$hasParent(Peter, Mary) \quad hasParent(Peter, Carl)$

Example

TBox:

$Animal \sqsubseteq LivingThing$

$Donkey \equiv Animal \sqcap \forall hasParent.Donkey$

$Horse \equiv Animal \sqcap \forall hasParent.Horse$

$Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey$

$\exists hasParent.Mule \sqsubseteq \perp$

ABox:

$Horse(Mary) \quad Mule(Peter) \quad Donkey(Sven)$

$hasParent(Peter, Mary) \quad hasParent(Peter, Carl)$

$hasParent(Sven, Hannah) \quad hasParent(Sven, Carl)$

Translation to First order logic

The function π map concepts to first-order formulae:

Translation to First order logic

The function π map concepts to first-order formulae:

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

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

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

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

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

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

Translation to First order logic

The function π map concepts to first-order formulae:

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

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

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

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

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

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

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 π map concepts to first-order formulae:

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

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

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

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

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

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

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 π map concepts to first-order formulae:

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

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

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

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

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

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

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 $\mathcal{I} \models_{FOL} \Pi(C \sqsubseteq D)$.

E.g.:

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

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

Reasoning problems

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

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 ($\mathcal{T} \models C \sqsubseteq D$);

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 ($\mathcal{T} \models C \equiv D$);

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 ($\mathcal{T} \models C \sqcap D \sqsubseteq \perp$);

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 ($\mathcal{T} \models C \sqcap D \sqsubseteq \perp$);

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

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 ($\mathcal{T} \models C \sqcap D \sqsubseteq \perp$);

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 ($\mathcal{T} \models C \sqsubseteq D$);
- 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 ($\mathcal{T} \models C \sqcap D \sqsubseteq \perp$);

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 $C(a)$ ($\mathcal{K} \models C(a)$);

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 ($\mathcal{T} \models C \sqcap D \sqsubseteq \perp$);

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 $C(a)$.

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 \mathcal{C} actually denotes an extension of a more restrictive language \mathcal{AL} .

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:

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;
 - ...

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

- \mathcal{H} – Role Hierarchies: We allow TBox axioms on the form $R \sqsubseteq P$ for atomic roles.
Semantics:

$$\mathcal{M} \models R \sqsubseteq P \Leftrightarrow R^{\mathcal{M}} \subseteq P^{\mathcal{M}}$$

e.g. *hasFather* \sqsubseteq *hasParent*;

Normal extensions

- \mathcal{H} – Role Hierarchies: We allow TBox axioms on the form $R \sqsubseteq P$ for atomic roles. Semantics:

$$\mathcal{M} \models R \sqsubseteq P \Leftrightarrow R^{\mathcal{M}} \subseteq P^{\mathcal{M}}$$

e.g. *hasFather* \sqsubseteq *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:

$$(R \circ P)^{\mathcal{M}} := \{ \langle a, b \rangle \in \Delta^{\mathcal{M}} \times \Delta^{\mathcal{M}} \mid \exists c \in \Delta^{\mathcal{M}} (\langle a, c \rangle \in R^{\mathcal{M}} \wedge \langle c, b \rangle \in P^{\mathcal{M}}) \}$$

and

$$\mathcal{M} \models R \sqsubseteq P \Leftrightarrow R^{\mathcal{M}} \subseteq P^{\mathcal{M}}$$

e.g. *friendOf* \circ *enemyOf* \sqsubseteq *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 . Semantics¹:

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

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

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

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

Normal extensions

- \mathcal{N} – Cardinality restrictions: We allow concepts on the form $\leq n R$ and $\geq n R$ for any natural number n . Semantics¹:

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

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

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

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

$$(\leq n R.C)^{\mathcal{M}} := \{a \in \Delta^{\mathcal{M}} \mid \#\{b \in \Delta^{\mathcal{M}} \mid \langle a, b \rangle \in R^{\mathcal{M}} \wedge b \in C^{\mathcal{M}}\} \leq n\}$$

$$(\geq n R.C)^{\mathcal{M}} := \{a \in \Delta^{\mathcal{M}} \mid \#\{b \in \Delta^{\mathcal{M}} \mid \langle a, b \rangle \in R^{\mathcal{M}} \wedge b \in C^{\mathcal{M}}\} \geq n\}$$

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

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

Normal extensions

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

$$(\{a_1, a_2, \dots, a_n\})^{\mathcal{M}} := \{a_1^{\mathcal{M}}, a_2^{\mathcal{M}}, \dots, a_n^{\mathcal{M}}\}$$

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 $\{a_1, a_2, \dots, a_n\}$ where a_i are individuals. Semantics

$$(\{a_1, a_2, \dots, a_n\})^{\mathcal{M}} := \{a_1^{\mathcal{M}}, a_2^{\mathcal{M}}, \dots, a_n^{\mathcal{M}}\}$$

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

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

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

e.g. $\text{hasParent}^- \sqsubseteq \text{isChildOf};$

Normal extensions

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

$$(\{a_1, a_2, \dots, a_n\})^{\mathcal{M}} := \{a_1^{\mathcal{M}}, a_2^{\mathcal{M}}, \dots, a_n^{\mathcal{M}}\}$$

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

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

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

e.g. $\text{hasParent}^- \sqsubseteq \text{isChildOf};$

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

Examples

OnlyChild \sqsubseteq *Person* $\sqcap \neg \exists hasSibling.\top$

Examples

OnlyChild \sqsubseteq *Person* $\sqcap \neg \exists hasSibling.\top$

Animal \sqsubseteq $\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$

Examples

$$\begin{array}{lll} \textit{OnlyChild} & \sqsubseteq & \textit{Person} \sqcap \neg \exists \textit{hasSibling}.\top \\ \textit{Animal} & \sqsubseteq & \leq 2 \textit{hasParent}.\textit{Animal} \sqcap \geq 2 \textit{hasParent}.\textit{Animal} \\ \textit{Pet} \sqcap \textit{Person} & \sqsubseteq & \perp \end{array}$$

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves. \{mary\}$

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$
$\exists R. \top$	\sqsubseteq	C

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$	
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$	
$Pet \sqcap Person$	\sqsubseteq	\perp	
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$	
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$	
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$	
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$	
$\exists R. \top$	\sqsubseteq	C	Domain

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$	
<i>Animal</i>	\sqsubseteq	$\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$	
$Pet \sqcap Person$	\sqsubseteq	\perp	
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$	
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$	
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$	
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$	
$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$	
<i>Animal</i>	\sqsubseteq	$\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$	
$Pet \sqcap Person$	\sqsubseteq	\perp	
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$	
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$	
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$	
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$	
$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$	
<i>Animal</i>	\sqsubseteq	$\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$	
$Pet \sqcap Person$	\sqsubseteq	\perp	
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$	
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$	
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$	
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$	

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1\ R. \top$	

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2\ hasParent.Animal \sqcap \geq 2\ hasParent.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg\{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1\ R. \top$	Functionality

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1 R. \top$	Functionality
R	\sqsubseteq	R^-	

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent. } Animal \sqcap \geq 2 \text{ hasParent. } Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves. \{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom. \{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1 R. \top$	Functionality
R	\sqsubseteq	R^{-}	Symmetry

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent}.Animal \sqcap \geq 2 \text{ hasParent}.Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves.\{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom.\{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1 R. \top$	Functionality
R	\sqsubseteq	R^{-}	Symmetry
R	\sqsubseteq	$\neg R^{-}$	

Examples

<i>OnlyChild</i>	\sqsubseteq	$Person \sqcap \neg \exists hasSibling. \top$
<i>Animal</i>	\sqsubseteq	$\leq 2 \text{ hasParent. } Animal \sqcap \geq 2 \text{ hasParent. } Animal$
$Pet \sqcap Person$	\sqsubseteq	\perp
<i>Person</i>	\sqsubseteq	$\exists loves. \{mary\}$
<i>Norwegian</i>	\sqsubseteq	$\exists comesFrom. \{norway\}$
$\{adam\}$	\sqsubseteq	$\neg \{eve\}$
$hasFather \circ hasBrother$	\sqsubseteq	$hasUncle$

$\exists R. \top$	\sqsubseteq	C	Domain
\top	\sqsubseteq	$\forall R. C$	Range
$R \circ R$	\sqsubseteq	R	Transitivity
\top	\sqsubseteq	$\leq 1 R. \top$	Functionality
R	\sqsubseteq	R^{-}	Symmetry
R	\sqsubseteq	$\neg R^{-}$	Asymmetry

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)
	\perp		(bottom concept)
	$\{a\}$		(<i>singular</i> enumeration)
	$C \sqcap D$		(intersection)
	$\exists R.C$		(existential restriction)

Common restricted languages: \mathcal{EL}

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

$C, D \rightarrow$	A		(atomic concept)
	\top		(universal concept)
	\perp		(bottom concept)
	$\{a\}$		(<i>singular</i> 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}

Not supported (excerpt):

- negation, (only disjointness of classes: $C \sqcap D \sqsubseteq \perp$),
- disjunction,
- universal quantification,
- cardinalities,
- inverse roles,
- plus some role characteristics.

Common restricted languages: \mathcal{EL}

Not supported (excerpt):

- negation, (only disjointness of classes: $C \sqcap D \sqsubseteq \perp$),
 - 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 **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:

$C \rightarrow$	A		(atomic concept)
	$\exists R.\top$		(existential restriction with \top only)
$D \rightarrow$	A		(atomic concept)
	$\exists R.D$		(existential restriction)
	$\neg D$		(negation)
	$D \sqcap D'$		(intersection)

Common restricted languages: *DL-Lite*

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

$C \rightarrow$	A		(atomic concept)
	$\exists R.\top$		(existential restriction with \top only)
$D \rightarrow$	A		(atomic concept)
	$\exists R.D$		(existential restriction)
	$\neg D$		(negation)
	$D \sqcap D'$		(intersection)

with the following axioms:

- $C \sqsubseteq D$ for concept descriptions D and C (and $C \equiv C'$).
- $P \sqsubseteq Q$ 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: *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:

$C \rightarrow$	A		(atomic concept)
	$C \sqcap C'$		(intersection)
	$C \sqcup C'$		(union)
	$\exists R.C$		(existential restriction)
$D \rightarrow$	A		(atomic concept)
	$D \sqcap D'$		(intersection)
	$\forall R.D$		(universal restriction)

Common restricted languages: \mathcal{RL}

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

$C \rightarrow$	A		(atomic concept)
	$C \sqcap C'$		(intersection)
	$C \sqcup C'$		(union)
	$\exists R.C$		(existential restriction)
$D \rightarrow$	A		(atomic concept)
	$D \sqcap D'$		(intersection)
	$\forall R.D$		(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 2 Profiles

- OWL has various *profiles* that correspond to different DLs.
- OWL Lite: $\mathcal{SHIF}(\mathcal{D})$;

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 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;

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 $\text{DL-Lite}_{\mathcal{R}}$, 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 $\text{DL-Lite}_{\mathcal{R}}$, 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 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 $\text{DL-Lite}_{\mathcal{R}}$, 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 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 $\text{DL-Lite}_{\mathcal{R}}$, 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

\mathcal{ALC} : Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix

What cannot be expressed in DLs: Brothers

- Given terms

hasSibling *Male*

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

$hasBrother \sqsubseteq hasSibling$

$\top \sqsubseteq \forall hasBrother. Male$

$\exists hasSibling. Male \sqsubseteq \exists hasBrother. \top$

- 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...

$$\begin{aligned} \text{SemiDetached} &\sqsubseteq \exists \text{hasLeftUnit}. \text{Unit} \sqcap \exists \text{hasRightUnit}. \text{Unit} \\ \text{Unit} &\sqsubseteq \exists \text{hasSeparatingWall}. \text{Wall} \end{aligned}$$

- And now what?

What cannot be expressed in DLs: Connecting Properties

- Given terms

Person *hasChild* *hasBirthday*

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

$$\begin{aligned} TwinParent \equiv Person \quad & \sqcap \quad \exists hasChild. \exists hasBirthday[\dots] \\ & \sqcap \quad \exists hasChild. \exists hasBirthday[\dots] \end{aligned}$$

- 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...

$$TwinParent \equiv Person \sqcap \geq_2 hasChild. \exists hasBirthday[\dots]$$

- 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*:

$$\forall n. \exists p. (p > n \wedge \forall k, l. p = k \cdot l \rightarrow (k = 1 \vee l = 1))$$

- Could try...

$$\begin{aligned} & \text{Number}(\text{zero}) \\ \text{Number} & \sqsubseteq \exists \text{hasSuccessor}. \text{Number} \\ \top & \sqsubseteq \leq 1 \text{ hasSuccessor}. \top \end{aligned}$$

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

$$\mathcal{K} \models q_1 \vee q_2 \Leftrightarrow \mathcal{K} \models q_1 \text{ or } \mathcal{K} \models q_2$$

$$\mathcal{K} \models q_1 \wedge q_2 \Leftrightarrow \mathcal{K} \models q_1 \text{ and } \mathcal{K} \models q_2$$

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

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

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

$$\mathcal{A} \models D(a) \vee C_1(a) \vee C_2(a) \vee A(a) \Leftrightarrow \langle \mathcal{T}, \mathcal{A} \rangle \models D(a)$$