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



- *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.

- Description logic comes from a merging of two traditions.

- 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'

- 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

- 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

- 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!)

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

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

- individuals/constants (e.g. *james*, *sensor*1)

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

- individuals/constants (e.g. *james*, *sensor*1)
- concepts/unary relations (e.g. Person, Sensor)

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

- individuals/constants (e.g. *james*, *sensor*1)
- 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. 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:

- A is a set of assertions about named individuals, called the ABox (e.g. Person(james), isFatherOf(james, peter))

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:

- 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

$\mathcal{ALC}:$ Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix

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

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

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

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

C, D ightarrow	A	(atomic concept)
	Т	(universal concept)
	⊥	(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.

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

$^{\prime}, D \rightarrow$	A	(atomic concept)
	Τ	(universal concept)
	⊥	(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.

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, $R^{\mathcal{M}} \subseteq \Delta \times \Delta$,

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

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

$$\begin{split} \top^{\mathcal{M}} &= \Delta \\ \perp^{\mathcal{M}} &= \emptyset \\ (\neg C)^{\mathcal{M}} &= \Delta \backslash C^{\mathcal{M}} \\ (C \sqcup D)^{\mathcal{M}} &= C^{\mathcal{M}} \cup D^{\mathcal{M}} \\ (C \sqcap D)^{\mathcal{M}} &= C^{\mathcal{M}} \cap D^{\mathcal{M}} \\ (\forall R.C)^{\mathcal{M}} &= \left\{ a \in \Delta \mid \forall b \in \Delta \left(\langle a, b \rangle \in R^{\mathcal{M}} \rightarrow b \in C^{\mathcal{M}} \right) \right\} \\ (\exists R.C)^{\mathcal{M}} &= \left\{ a \in \Delta \mid \exists b \in \Delta \left(\langle a, b \rangle \in R^{\mathcal{M}} \land b \in C^{\mathcal{M}} \right) \right\} \end{split}$$

An interpretation ${\cal M}$ satisfies

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

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 $\mathcal{M} \vDash C \sqsubseteq D$, iff $C^{\mathcal{M}} \subseteq D^{\mathcal{M}}$;

An interpretation ${\mathcal M}$ satisfies

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

$$- C \sqsubseteq D$$
, denoted $\mathcal{M} \vDash C \sqsubseteq D$, iff $C^{\mathcal{M}} \subseteq D^{\mathcal{M}}$;

$$- R \sqsubseteq P$$
, denoted $\mathcal{M} \vDash R \sqsubseteq P$, iff $R^{\mathcal{M}} \subseteq P^{\mathcal{M}}$.

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

An interpretation $\ensuremath{\mathcal{M}}$ satisfies

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

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

We will use the following shorthand notation:

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

Example

TBox:

 $\begin{array}{l} Animal \sqsubseteq LivingThing\\ Donkey \equiv Animal \sqcap \forall hasParent.Donkey\\ Horse \equiv Animal \sqcap \forall hasParent.Horse\\ Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey\\ \exists hasParent.Mule \sqsubseteq \bot\end{array}$

Example

TBox:

 $\begin{array}{l} Animal \sqsubseteq LivingThing\\ Donkey \equiv Animal \sqcap \forall hasParent.Donkey\\ Horse \equiv Animal \sqcap \forall hasParent.Horse\\ Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey\\ \exists hasParent.Mule \sqsubseteq \bot\end{array}$

ABox:

Horse(Mary) Mule(Peter) Donkey(Sven)

hasParent(Peter, Mary) hasParent(Peter, Carl)

Example

TBox:

 $\begin{array}{l} Animal \sqsubseteq LivingThing\\ Donkey \equiv Animal \sqcap \forall hasParent.Donkey\\ Horse \equiv Animal \sqcap \forall hasParent.Horse\\ Mule \equiv Animal \sqcap \exists hasParent.Horse \sqcap \exists hasParent.Donkey\\ \exists hasParent.Mule \sqsubseteq \bot\end{array}$

ABox:

Horse(Mary) Mule(Peter) Donkey(Sven)

hasParent(Peter, Mary) hasParent(Peter, Carl) hasParent(Sven, Hannah) hasParent(Sven, Carl)

The function π map concepts to first-order formulae:

The function π map concepts to first-order formulae:

$$\pi_{x}(A) = A(x)$$

$$\pi_{x}(\neg C) = \neg \pi_{x}(C)$$

$$\pi_{x}(C \sqcup D) = \pi_{x}(C) \lor \pi_{x}(D)$$

$$\pi_{x}(C \sqcap D) = \pi_{x}(C) \land \pi_{x}(D)$$

$$\pi_{x}(\exists R.C) = \exists y (R(x, y) \land \pi_{y}(C))$$

$$\pi_{x}(\forall R.C) = \forall y (R(x, y) \to \pi_{y}(C))$$

The function π map concepts to first-order formulae:

$$\pi_{x}(A) = A(x)$$

$$\pi_{x}(\neg C) = \neg \pi_{x}(C)$$

$$\pi_{x}(C \sqcup D) = \pi_{x}(C) \lor \pi_{x}(D)$$

$$\pi_{x}(C \sqcap D) = \pi_{x}(C) \land \pi_{x}(D)$$

$$\pi_{x}(\exists R.C) = \exists y (R(x, y) \land \pi_{y}(C))$$

$$\pi_{x}(\forall R.C) = \forall y (R(x, y) \to \pi_{y}(C))$$
We can then map axioms:
$$\Pi(C \sqsubseteq D) := \forall x (\pi_{x}(C) \to \pi_{x}(D)).$$

The function π map concepts to first-order formulae:

$$\pi_{x}(A) = A(x)$$

$$\pi_{x}(\neg C) = \neg \pi_{x}(C)$$

$$\pi_{x}(C \sqcup D) = \pi_{x}(C) \lor \pi_{x}(D)$$

$$\pi_{x}(C \sqcap D) = \pi_{x}(C) \land \pi_{x}(D)$$

$$\pi_{x}(\exists R.C) = \exists y (R(x, y) \land \pi_{y}(C))$$

$$\pi_{x}(\forall R.C) = \forall y (R(x, y) \to \pi_{y}(C))$$

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

The function π map concepts to first-order formulae:

$$\pi_{x}(A) = A(x)$$

$$\pi_{x}(\neg C) = \neg \pi_{x}(C)$$

$$\pi_{x}(C \sqcup D) = \pi_{x}(C) \lor \pi_{x}(D)$$

$$\pi_{x}(C \sqcap D) = \pi_{x}(C) \land \pi_{x}(D)$$

$$\pi_{x}(\exists R.C) = \exists y (R(x, y) \land \pi_{y}(C))$$

$$\pi_{x}(\forall R.C) = \forall y (R(x, y) \to \pi_{y}(C))$$

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

Theorem

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

E.g.:

 $\pi_{x}(Animal \sqcap \forall hasParent.Donkey) = Animal(x) \land \forall y(hasParent(x, y) \rightarrow Donkey(y))$ $\Pi(Animal \sqsubseteq LivingThing) = \forall x(Animal(x) \rightarrow LivingThing(x))$

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 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 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 $(T \vDash C \equiv D)$;

- 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 $(T \vDash C \equiv D)$;
- 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 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 $(T \vDash C \equiv D)$;
- 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 $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:

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 $(T \vDash C \equiv D)$;
- 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 $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$: - Is \mathcal{K} consistent (\mathcal{K} has a model);

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 $(T \vDash C \equiv D)$;
- 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 $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:

- Is \mathcal{K} consistent (\mathcal{K} has a model);
- Given a concept C and an individual a, does \mathcal{K} entail C(a) ($\mathcal{K} \vDash C(a)$);

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 ($\mathcal{T} \vDash C \sqsubseteq D$);
- Given two concepts C and D, are C and D equivalent $(T \vDash C \equiv D)$;
- 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 $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$:

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

Contents

Introduction

ALC: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix

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

- 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 ${\mathcal {AL}}.$

- 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:

- 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;
 - Q: Qualified cardinality restrictions;
 - \mathcal{O} : Closed classes;
 - \mathcal{I} : Inverse roles;
 - \mathcal{D} : Datatypes;
 - ...

- 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;
 - 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.

- 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;
 - 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 S for more advanced languages, so e.g. \mathcal{SHOIN} is $\mathcal{ALC} + \mathcal{H} + \mathcal{O} + \mathcal{I} + \mathcal{N}$.

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

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

e.g. $hasFather \sqsubseteq hasParent;$

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

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

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:

$$(R \circ P)^{\mathcal{M}} := ig \{ \langle a, b
angle \in \Delta^{\mathcal{M}} imes \Delta^{\mathcal{M}} \mid \exists c \in \Delta^{\mathcal{M}} \left(\langle a, c
angle \in R^{\mathcal{M}} \wedge \langle c, b
angle \in P^{\mathcal{M}}
ight) ig \}$$

and

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

e.g. friendOf \circ enemyOf \sqsubseteq enemyOf.

- N - Cardinality restrictions: We allow concepts on the form $\leq nR$ and $\geq nR$ 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* $\sqsubseteq \leq 2$ *hasParent*;

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

- N - Cardinality restrictions: We allow concepts on the form $\leq nR$ and $\geq nR$ 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. $Mammal \sqsubseteq \leq 2 hasParent;$

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

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

e.g. $\textit{RichPeople} \sqsubseteq \ge 2 \textit{ owns.House.}$

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

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

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

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

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

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

e.g. Days \sqsubseteq {monday, tuesday, wednesday, thursday, friday, saturday, sunday}; - \mathcal{I} - Inverse roles: We allow roles on the form R^- . Semantics:

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

e.g. $hasParent^- \sqsubseteq isChildOf$;

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

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

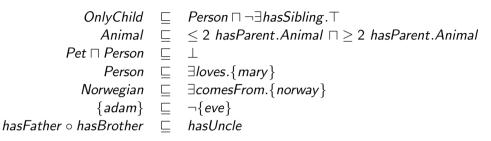
e.g. Days \sqsubseteq {monday, tuesday, wednesday, thursday, friday, saturday, sunday}; - \mathcal{I} - Inverse roles: We allow roles on the form R^- . Semantics:

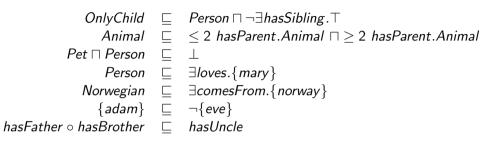
$$(R^{-})^{\mathcal{M}} := \{ \langle a, b
angle \in \Delta^{\mathcal{M}} imes \Delta^{\mathcal{M}} \mid \langle b, a
angle \in R^{\mathcal{M}} \}$$

e.g. $hasParent^- \sqsubseteq isChildOf$;

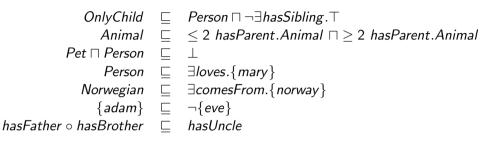
- D - Datatypes: We introduce a set of datatypes: *int,string,float,boolean*, and so on. They all have a fixed interpretation, that is, the same for all models.

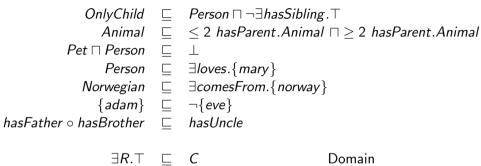
OnlyChild \sqsubseteq *Person* $\sqcap \neg \exists hasSibling. \top$



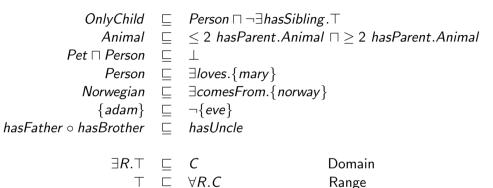


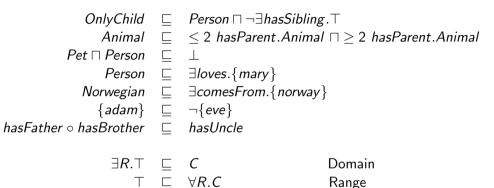
 $\exists R. \top \subseteq C$



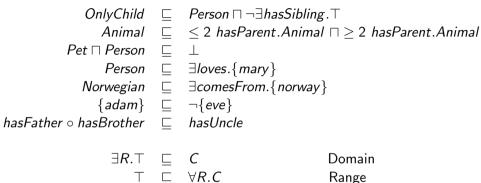


 $\top \sqsubseteq \forall R.C$

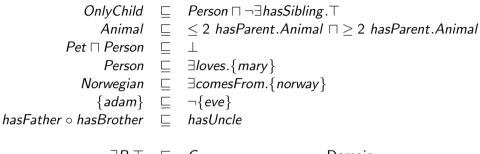


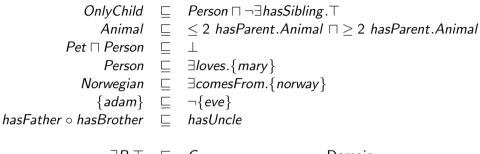


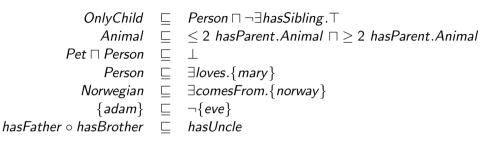
 $R \circ R \square R$

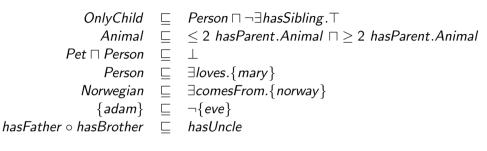


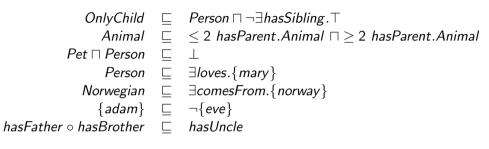
 $R \circ R \sqsubseteq R$ Transitivity

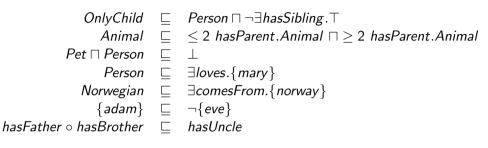












Complexity results

http://www.cs.man.ac.uk/~ezolin/dl/

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

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

C, D ightarrow	A	(atomic concept)
	Τ Ι	(universal concept)
	⊥	(bottom concept)
	{ a }	(singular enumeration)
	$C \sqcap D$	(intersection)
	$\exists R.C$	(existential restriction)

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

C, D ightarrow	A	(atomic concept)
	Τ	(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.

Not supported (excerpt):

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

Not supported (excerpt):

- negation, (only disjointness of classes: $C \sqcap D \sqsubseteq \bot$),
- 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 **P**.
- "Good for large ontologies."

The description logic DL-Lite_R allows the following concepts:

The description logic DL-Lite_R allows the following concepts:

$C \rightarrow$	$A \mid \exists R. \top \mid$	(atomic concept) (existential restriction with $ op$ only)
D ightarrow	$\begin{array}{c} A & \\ \exists R.D & \\ \neg D & \\ D \sqcap D' & \end{array}$	(atomic concept) (existential restriction) (negation) (intersection)

The description logic DL-Lite_R allows the following concepts:

$C \rightarrow$	A ∃R.⊤	(atomic concept) (existential restriction with $ op$ only)
D ightarrow	$\begin{array}{c} A \\ \exists R.D \\ \neg D \\ D \sqcap D' \end{array} \right $	(atomic concept) (existential restriction) (negation) (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.

Not supported (excerpt):

- disjunction,
- universal quantification,
- cardinalities,
- functional roles, keys,
- enumerations (closed classes),
- subproperties of chains, transitivity

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."

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

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

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

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 ightarrow	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

ALC: Syntax and Semantics

Extensions and other DLs

OWL and the Semantic Web

Appendix

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

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

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

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

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

- OWL has various *profiles* that correspond to different DLs.
- OWL Lite: SHIF(D);
- OWL DL: corresponds to $\mathcal{SHION}(\mathcal{D})$;
- 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 $_{\mathcal{R}}$, and is specifically designed for efficient database integration;

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

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...

SemiDetached $\sqsubseteq \exists hasLeftUnit.Unit \sqcap \exists hasRightUnit.Unit Unit \sqsubseteq \exists hasSeparatingWall.Wall$

- 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...

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

- 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...

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

- 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...

Number(zero) $Number \sqsubseteq \exists hasSuccessor.Number$ $\top \sqsubseteq \leq 1 hasSuccessor.\top$

- 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 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

 $\mathcal{K} \vDash q_1 \lor q_2 \Leftrightarrow \mathcal{K} \vDash q_1 \text{ or } \mathcal{K} \vDash q_2$ $\mathcal{K} \vDash q_1 \land q_2 \Leftrightarrow \mathcal{K} \vDash q_1 \text{ and } \mathcal{K} \vDash q_2$

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

$$\mathcal{A}\vDash
ho(\mathcal{T}, \boldsymbol{q}(ec{a})) \Leftrightarrow \mathcal{K}\vDash \boldsymbol{q}(ec{a})$$

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