

Part V

The Semantic Web

- 1 Course overview
- 2 Introduction to Logic
- 3 Declarative Programming with Prolog
- 4 Declarative Programming with Constraints
- 5 The Semantic Web**

Contents

5

The Semantic Web

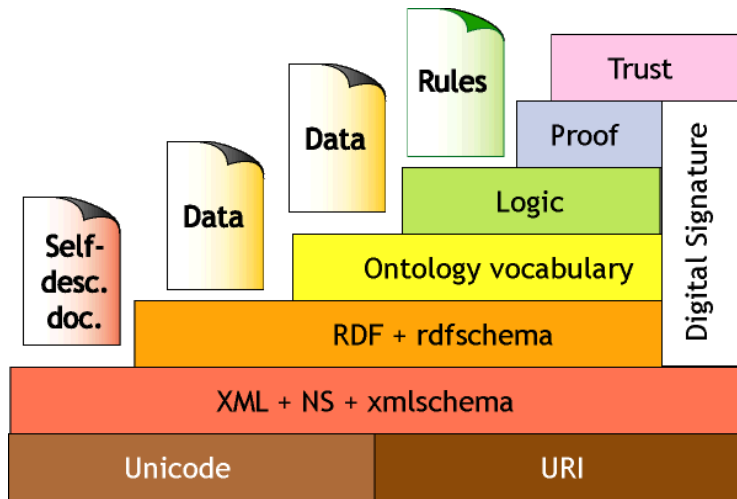
- Introducing Semantic Technologies
- An example of the Semantic Web approach
- An overview of Description Logics
- The *ALCN* language family
- TBox reasoning
- The *SHIQ* language family

Semantic Technologies

- Semantics = meaning
- Semantic Technologies = technologies building on (formalized) meaning
- Declarative Programming as a semantic technology
 - A procedure definition describes its intended meaning
 - e.g. `intersect(L1, L2) :- member(X, L1), member(X, L2).`
Lists L1 and L2 intersect
if there exists an x, which is a member of both L1 and L2.
 - The execution of a program can be viewed as a process of deduction
- The main goal of the Semantic Web (SW) approach:
 - make the information on the web processable by computers
 - machines should be able to **understand** the web, not only **read** it
- Achieving the vision of the Semantic Web
 - Add (computer processable) **meta-information** to the web
 - Formalize background knowledge – build so called ontologies
 - Develop reasoning algorithms and tools

The vision of the Semantic Web

- The Semantic Web layer cake – Tim Berners-Lee



The Semantic Web

- The goal: making the information on the web processable by computers
- Achieving the vision of the Semantic Web
 - Add meta-information to web pages, e.g.
(*AIT* hasLocation *Budapest*)
(*AIT* hasTrack *Track:Foundational-courses*)
(*Track:Foundational-courses* hasCourse *Semantic-and-declarative...*)
 - Formalise background knowledge – build so called terminologies
 - hierarchies of notions, e.g.
a *University* is a (subconcept of) *Inst-of-higher-education*,
the *hasFather* relationship is a special case of *hasParent*
 - definitions and axioms, e.g.
a *Father* is a *Male Person* having at least one child
 - Develop reasoning algorithms and tools
- Main topics
 - Description Logic, the maths behind the Semantic Web is the basis of Web Ontology Languages OWL 1 & 2 (W3C standards)
 - A glimpse at reasoning algorithms for Description Logic

Contents

5

The Semantic Web

- Introducing Semantic Technologies
- **An example of the Semantic Web approach**
- An overview of Description Logics
- The *ALCN* language family
- TBox reasoning
- The *SHIQ* language family

First Order Logic (recap)

- Syntax:

- non-logical (“user-defined”) symbols: **predicates** and **functions**, including **constants** (function symbols with 0 arguments)
- terms (refer to individual elements of the universe, or interpretation), e.g. *fatherOf*(*Susan*)
- formulas (that hold or do not hold in a given interpretation), e.g.
$$\varphi = \forall x. (\textit{Optimist}(\textit{fatherOf}(x)) \rightarrow \textit{Optimist}(x))$$

- Semantics:

- determines if a closed formula φ is true in an interpretation \mathcal{I} : $\mathcal{I} \models \varphi$ (also read as: \mathcal{I} is a model of φ)
- an interpretation \mathcal{I} consists of a domain Δ and a mapping from non-logical symbols (e.g. *Optimist*, *fatherOf*, *Susan*) to their meaning
- semantic consequence: $S \models \alpha$ means: if an interpretation is a model of all formulas in the set S , then it is also a model of α (note that the symbol \models is overloaded)

- Deductive system (also called proof procedure):

an algorithm to deduce a consequence α of a set of formulas S : $S \vdash \alpha$

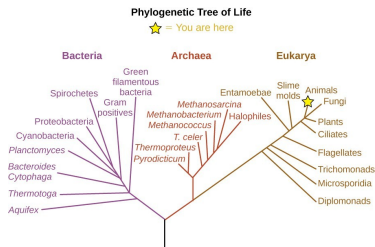
- example: resolution

Soundness, completeness and decidability (recap)

- A deductive system is **sound** if $S \vdash \alpha \Rightarrow S \models \alpha$ (deduces only truths).
- A deductive system is **complete** if $S \models \alpha \Rightarrow S \vdash \alpha$ (deduces all truths).
- Resolution is a sound and complete deductive system for FOL
- Kurt Gödel was first to show such a system:
Gödel's completeness theorem: there is a sound and complete deductive system for FOL
- FOL is not decidable: no decision procedure for the question “does S imply α ($S \vdash \alpha$)?” (Gödel's completeness theorem ensures that if the answer is “yes”, then there exists a proof of α from S ; but if the answer is “no”, we have no guarantees – this is called semi-decidability)
- Developers of the **Semantic Web** strive for using **decidable** languages
 - for languages with a sound and **complete** proof procedure
- Semantic Web languages are based on Description Logics, which are decidable sublanguages of FOL, i.e. there is an algorithm that delivers a yes or no answer to the question “does S imply α ”

Ontologies

- **Ontology**: computer processable description of knowledge
- Early ontologies include classification system (biology, medicine, books)



- Entities in the Web Ontology Language (OWL):
 - **classes** – describe sets of objects (e.g. optimists)
 - **properties** (attributes, slots) – describe binary relationships (e.g. has parent)
 - **objects** – correspond to real life objects (e.g. people, such as Susan, her parents, etc.)

Knowledge Representation

- **Natural Language:**

- 1 Someone **having** a non-**optimist friend** is bound to be an **optimist**.
- 2 **Susan** **has** herself as a **friend**.

- **First order Logic** (**unary predicate**, **binary predicate**, **constant**):

- 1 $\forall x. (\exists y. (\text{hasFriend}(x, y) \wedge \neg \text{opt}(y)) \rightarrow \text{opt}(x))$
- 2 $\text{hasFriend}(\text{Susan}, \text{Susan})$

- **Description Logics** (**concept**, **role**, **individual**):

- 1 $(\exists \text{hasFriend}. \neg \text{Opt}) \sqsubseteq \text{Opt}$ (GCI – Gen. Concept Inclusion axiom)
- 2 $\text{hasFriend}(\text{Susan}, \text{Susan})$ (role assertion)

- **Web Ontology Language** (Manchester syntax)⁵ (**class**, **property**, **object**):

- 1 $(\text{hasFriend } \text{some } (\text{not } \text{Opt})) \text{ SubClassOf: } \text{Opt}$
Those **having some not Opt friends** must be **Opt**
(GCI – Gen. Class Inclusion axiom)
- 2 $\text{hasFriend}(\text{Susan}, \text{Susan})$ (object property assertion)

⁵protegeproject.github.io/protege/class-expression-syntax

A sample ontology to be entered into Protégé

- 1 There is a class of **Animals**, some of which are **Male**, some are **Female**.
- 2 No one can be both **Male** and **Female**.
- 3 There are **Animals** that are **Human**.
- 4 There are **Humans** who are **Optimists**.
- 5 There is a relationship **hasP** meaning “has parent”. Relations **hasFather** and **hasMother** are sub-relations (special cases) of **hasP**.
- 6 Let's define the class **C1** as those who have an optimistic parent.
- 7 State that everyone belonging to **C1** is **Optimistic**.
- 8 State directly that anyone having an **Optimistic** parent is **Optimistic**.
- 9 There is a relation **hasF**, denoting “has friend”. State that someone having a non-**Optimistic** friend must be **Optimistic**.
- 10 There are individuals: **Susan**, and her **parents** **Mother** and **Father**.
- 11 **Mother** has **Father** as her **friend**.

The sample ontology in Description Logic and OWL/Protégé

English	Description Logic	OWL (Manchester syntax)
1 Male is a subclass of Animal .	$\text{Male} \sqsubseteq \text{Animal}$	Male SubClassOf: Animal
Female is a subclass of Animal .	$\text{Female} \sqsubseteq \text{Animal}$	Female SubClassOf: Animal
2 Male and Female are disjoint.	$\text{Male} \sqsubseteq \neg \text{Female}$	Male DisjointWith: Female
3 Human is a subclass of Animal .	$\text{Human} \sqsubseteq \text{Animal}$	Human SubClassOf: Animal
4 Optimist is a subclass of Human .	$\text{Opt} \sqsubseteq \text{Human}$	Opt SubClassOf: Human
5 hasFather is a subprop. of hasP .	$\text{hasFather} \sqsubseteq \text{hasP}$	hasFather SubPropertyOf: hasP
hasMother is a subprop. of hasP .	$\text{hasMother} \sqsubseteq \text{hasP}$	hasMother SubPropertyOf: hasP
6 C1 = those having an Opt parent.	$\text{C1} \equiv \exists \text{hasP} . \text{Opt}$	C1 EquivalentTo: hasP some Opt
7 Everyone in C1 is Opt .	$\text{C1} \sqsubseteq \text{Opt}$	C1 SubClassOf: Opt
8 Children of Opt parents are Opt .	$\exists \text{hasP} . \text{Opt} \sqsubseteq \text{Opt}$	hasP some Opt SubClassOf: Opt
9 Those with a non- Opt friend are Opt .	$\exists \text{hasF} . \neg \text{Opt} \sqsubseteq \text{Opt}$	hasF some not Opt SubClassOf: Opt
10 Susan has parents Mother and Father .	$\text{hasP}(\text{Susan}, \text{Mother})$ $\text{hasP}(\text{Susan}, \text{Father})$	hasP (Susan , Mother) hasP (Susan , Father)
11 Mother has Father as a friend.	$\text{hasF}(\text{Mother}, \text{Father})$	hasF (Mother , Father)

(In Protégé, select the “save as” format as “Latex syntax” to obtain DL notation.)

Contents

5

The Semantic Web

- Introducing Semantic Technologies
- An example of the Semantic Web approach
- **An overview of Description Logics**
- The *ALCN* language family
- TBox reasoning
- The *SHIQ* language family

Description Logic (DLs) – overview

DL, a subset of FOL, is the mathematical background of OWL

- Signature – relation and function symbols allowed in DL
 - concept name (A) – unary predicate symbol (cf. OWL class)
 - role name (R) – binary predicate symbol (cf. OWL property)
 - individual name (a, \dots) – constant symbol (cf. OWL object)
 - No non-constant function symbols, no preds of arity > 2 , no vars
- Concept names and **concept expressions** represent sets, e.g.
 $\exists \text{hasParent.Optimist}$ – the set of those who have an optimist parent
- Terminological axioms (TBox) – stating background knowledge
 - A simple axiom using the DL language $\mathcal{AL}\mathcal{E}$:
 $\exists \text{hasParent.Optimist} \sqsubseteq \text{Optimist}$ – the set of those who have an optimist parent is a subset of the set of optimists
 - Translation to FOL: $\forall x. (\exists y. (\text{hasP}(x, y) \wedge \text{Opt}(y)) \rightarrow \text{Opt}(x))$
- Assertions (ABox) – stating facts about individual names
 - Example: $\text{Optimist}(\text{JACOB}), \text{hasParent}(\text{JOSEPH}, \text{JACOB})$
- A consequence of these TBox and ABox axioms is: $\text{Optimist}(\text{JOSEPH})$
- DLs behind OWL 1 and OWL 2 are **decidable**: there are bounded time algorithms for checking if a set of axioms implies a statement.

Some further examples of terminological axioms

(1) A **Mother** is a **Person**, who is a **Female** and who **has(a)Child**.

$$\text{Mother} \sqsubseteq \text{Person} \sqcap \text{Female} \sqcap \exists \text{hasChild}.\top$$

(2) A **Tiger** is a **Mammal**.

$$\text{Tiger} \sqsubseteq \text{Mammal}$$

(3) Children of an **Optimist Person** are **Optimists**, too.

$$\text{Optimist} \sqcap \text{Person} \sqsubseteq \forall \text{hasChild}.\text{Optimist}$$

(4) Childless people are **Happy**.

$$\forall \text{hasChild}.\perp \sqcap \text{Person} \sqsubseteq \text{Happy}$$

(5) Those in the relation **hasChild** are also in the relation **hasDescendant**.

$$\text{hasChild} \sqsubseteq \text{hasDescendant}$$

(6) The relation **hasParent** is the inverse of the relation **hasChild**.

$$\text{hasParent} \sqsubseteq \text{hasChild}^{-}$$

(7) The **hasDescendant** relationship is transitive.

$$\text{Trans}(\text{hasDescendant})$$

Description Logics – why the plural?

- These logic variants were progressively developed in the last two decades
- As new constructs were proved to be “safe”, i.e. keeping the logic decidable, these were added
- We will start with the very simple language \mathcal{AL} , extend it to $\mathcal{AL}\mathcal{E}$, \mathcal{ALU} and \mathcal{ALL}
- As a side branch we then define \mathcal{ALLN}
- We then go back to \mathcal{ALL} and extend it to languages \mathcal{S} , \mathcal{SH} , \mathcal{SHI} and \mathcal{SHIQ} (which encompasses \mathcal{ALLN})
- We briefly tackle further extensions \mathcal{O} , (\mathbf{D}) and \mathcal{R}
- OWL 1, published in 2004, corresponds to $\mathcal{SHOIN}(\mathbf{D})$
- OWL 2, published in 2012, corresponds to $\mathcal{SROIQ}(\mathbf{D})$

Contents

5

The Semantic Web

- Introducing Semantic Technologies
- An example of the Semantic Web approach
- An overview of Description Logics
- **The *ALCN* language family**
- TBox reasoning
- The *SHIQ* language family

Overview of the \mathcal{ALCN} language

- In \mathcal{ALCN} a statement (axiom) can be
 - a subsumption (inclusion), e.g. $\text{Tiger} \sqsubseteq \text{Mammal}$, or
 - an equivalence, e.g. $\text{Woman} \equiv \text{Female} \sqcap \text{Person}$,
 $\text{Mother} \equiv \text{Woman} \sqcap \exists \text{hasChild}.$
- In general, an \mathcal{ALCN} axiom can take these two forms:
 - subsumption: $C \sqsubseteq D$
 - equivalence: $C \equiv D$, where C and D are concept expressions
- A concept expression C denotes a set of objects
 (a subset of the Δ universe of the interpretation), and can be:
 - an atomic concept (or concept name), e.g. Tiger , Female , Person
 - a composite concept, e.g. $\text{Female} \sqcap \text{Person}$, $\exists \text{hasChild}.\text{Female}$
 - composite concepts are built from atomic concepts and **atomic roles**
 (also called **role names**) using some constructors (e.g. \sqcap , \sqcup , \exists , etc.)
- We first introduce language \mathcal{AL} , that allows a minimal set of constructors
 (all examples on this page are valid \mathcal{AL} concept expressions)
- Next, we discuss richer extensions named \mathcal{U} , \mathcal{E} , \mathcal{C} , \mathcal{N}

The syntax of the \mathcal{AL} language

Language \mathcal{AL} (Attributive Language) allows the following concept expressions, also called concepts, for short:

A is an atomic concept, C, D are arbitrary (possibly composite) concepts
 R is an atomic role

DL concept	OWL class	Name	Informal definition
A	A (class name)	atomic concept	those in A
\top	<code>owl:Thing</code>	top	the set of all objects
\perp	<code>owl:Nothing</code>	bottom	the empty set
$\neg A$	not A	atomic negation	those not in A
$C \sqcap D$	C and D	intersection	those in both C and D
$\forall R.C$	R only C	value restriction	those whose all R s belong to C
$\exists R.\top$	R some <code>owl:Thing</code>	limited exist. restr.	those having at least one R

Examples of \mathcal{AL} concept expressions:

$\text{Person} \sqcap \neg \text{Female}$

Person and not Female

$\text{Person} \sqcap \forall \text{hasChild}.\text{Female}$

Person and (hasChild only Female)

$\text{Person} \sqcap \exists \text{hasChild}.\top$

Person and (hasChild some `owl:Thing`)

The semantics of the \mathcal{AL} language (as a special case of FOL)

- An interpretation \mathcal{I} is a mapping:
 - $\Delta^{\mathcal{I}} = \Delta$ is the universe, the **nonempty** set of all individuals/objects
 - for each concept/class name A , $A^{\mathcal{I}}$ is a (possibly empty) subset of Δ
 - for each role/property name R , $R^{\mathcal{I}} \subseteq \Delta \times \Delta$ is a binary relation on Δ
- The semantics of \mathcal{AL} extends \mathcal{I} to composite concept expressions, i.e. describes how to “calculate” the meaning of arbitrary concept exprs:

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

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

$$(\neg A)^{\mathcal{I}} = \Delta \setminus A^{\mathcal{I}}$$

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

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

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

- Finally we define how to obtain the truth value of an axiom:

$$\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

$$\mathcal{I} \models C \equiv D \quad \text{iff} \quad C^{\mathcal{I}} = D^{\mathcal{I}}$$

The \mathcal{AL} language: limitations

Recall the elements of the language \mathcal{AL} :

DL concept	OWL class	Name	Informal definition
A	A (class name)	atomic concept	those in A
\top	<code>owl:Thing</code>	top	the set of all objects
\perp	<code>owl:Nothing</code>	bottom	the empty set
$\neg A$	not A	atomic negation	those not in A
$C \sqcap D$	C and D	intersection	those in both C and D
$\forall R.C$	R only C	value restriction	those whose all R s belong to C
$\exists R.\top$	R some <code>owl:Thing</code>	limited exist. restr.	those having at least one R

What is missing from \mathcal{AL} ?

- We can specify the intersection of two concepts, but not the union, e.g. those who are **either blue-eyed or tall**.
- $\exists R.\top$ – we cannot describe e.g. those having a **female** child.
Remedy: allow for full exist. restr., e.g. $\exists \text{hasCh.Female}$
- $\neg A$ – negation can be applied to atomic concepts only.
Remedy: full negation, $\neg C$, where C can be non-atomic, e.g. $\neg(U \sqcap V)$

The \mathcal{ALCN} language family: extensions \mathcal{U} , \mathcal{E} , \mathcal{C} , \mathcal{N}

Further concept constructors, OWL equivalents shown in [square brackets]:

- Union: $C \sqcup D$, [C or D] – those in either C or D

$$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \quad (\mathcal{U})$$

- Full existential restriction: $\exists R.C$, [R some C]
– those who have at least one R belonging to C

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

- (Full) negation: $\neg C$, [not C] – those who do not belong to C

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

- Unqualified number restrictions: $(\leq nR)$, [R max n owl:Thing] and $(\geq nR)$, [R min n owl:Thing]
– those who have at most/at least n R -related objects

$$\begin{aligned} (\leq nR)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid |\{b \mid \langle a, b \rangle \in R^{\mathcal{I}}\}| \leq n\} \\ (\geq nR)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid |\{b \mid \langle a, b \rangle \in R^{\mathcal{I}}\}| \geq n\} \end{aligned} \quad (\mathcal{N})$$

Example: $\text{Person} \sqcap ((\leq 1 \text{ hasCh}) \sqcup (\geq 3 \text{ hasCh})) \sqcap \exists \text{hasCh.Female}$

Person and ($\text{hasCh max } 1$ or $\text{hasCh min } 3$) and (hasCh some Female)

Note that qualified number restrictions, e.g., “those having at least 3 blue-eyed children” are not covered by the extension \mathcal{N} .

Summary table of the \mathcal{ALCUN} language

DL	OWL	Name	Informal definition	
A	A	atomic concept	those in A	\mathcal{AL}
$\neg A$	not A	full negation	those not in A (cf. \mathcal{C})	\mathcal{AL}
\top	owl:Thing	top	the set of all objects	\mathcal{AL}
\perp	owl:Nothing	bottom	the empty set	\mathcal{AL}
$C \sqcap D$	C and D	intersection	those in both C and D	\mathcal{AL}
$\exists R.T$	R some	existential restr.	those having an R (cf. \mathcal{E})	\mathcal{AL}
$\forall R.C$	R only C	value restriction	those whose all R s belong to C	\mathcal{AL}
$\neg C$	not C	full negation	those not in C	\mathcal{C}
$C \sqcup D$	C or D	union	those in either C or D	\mathcal{U}
$\exists R.C$	R some C	existential restr.	those with an R belonging to C	\mathcal{E}
$(\leq nR)$	R max n owl:Thing	unq. numb. restr.	those having at most n R s	\mathcal{N}
$(\geq nR)$	R min n owl:Thing	unq. numb. restr.	those having at least n R s	\mathcal{N}

Rewriting \mathcal{ALCN} to first order logic

- Concept expressions map to predicates with one argument, e.g.

Tiger \implies **Tiger**(x)

Mammal \implies **Mammal**(x)

Person \implies **Person**(x)

Female \implies **Female**(x)

- Simple connectives \sqcap , \sqcup , \neg map to boolean operations \wedge , \vee , \neg , e.g.

Person \sqcap **Female** \implies **Person**(x) \wedge **Female**(x)

Person \sqcup \neg **Mammal** \implies **Person**(x) \vee \neg **Mammal**(x)

- An axiom $C \sqsubseteq D$ is rewritten as $\forall x.(C(x) \rightarrow D(x))$, e.g.

Tiger \sqsubseteq **Mammal** $\implies \forall x.(\textbf{Tiger}(x) \rightarrow \textbf{Mammal}(x))$

- An axiom $C \equiv D$ is rewritten as $\forall x.(C(x) \leftrightarrow D(x))$, e.g.

Woman \equiv **Person** \sqcap **Female** $\implies \forall x.(\textbf{Woman}(x) \leftrightarrow \textbf{Person}(x) \wedge \textbf{Female}(x))$

- Concept constructors involving a quantifier \exists or \forall are rewritten to an appropriate quantified formula, where a role name is mapped to a binary predicate (a predicate with two arguments), e.g.

$\exists \text{hasParent. Opt} \sqsubseteq \text{Opt} \implies \forall x.(\exists y.(\text{hasParent}(x, y) \wedge \text{Opt}(y)) \rightarrow \text{Opt}(x))$

Rewriting \mathcal{ALCN} to first order logic, example

- Consider $C = \text{Person} \sqcap ((\leq 1 \text{ hasCh}) \sqcup (\geq 3 \text{ hasCh})) \sqcap \exists \text{hasCh.Female}$
- Let's outline a predicate $C(x)$ which is true when x belongs to concept C :

$$C(x) \leftrightarrow \text{Person}(x) \wedge$$

$$(\text{hasAtMost1Child}(x) \vee \text{hasAtLeast3Children}(x)) \wedge$$

$$\text{hasFemaleChild}(x)$$
- Class practice:
 - Define the FOL predicates $\text{hasAtMost1Child}(x)$, $\text{hasAtLeast3Children}(x)$, $\text{hasFemaleChild}(x)$
 - Additionally, define the following FOL predicates:
 - $\text{hasOnlyFemaleChildren}(x)$, corresponding to the concept $\forall \text{hasCh.Female}$
 - $\text{hasAtMost2Children}(x)$, corresponding to the concept $(\leq 2 \text{ hasCh})$

General rewrite rules $\mathcal{ALCN} \rightarrow \text{FOL}$

Each concept expression can be mapped to a FOL formula:

- Each concept expression C is mapped to a formula $\Phi_C(x)$ (expressing that x belongs to C).
- Atomic concepts (A) and roles (R) are mapped to unary and binary predicates $A(x)$, $R(x, y)$.
- \sqcap , \sqcup , and \neg are transformed to their counterpart in FOL (\wedge , \vee , \neg), e.g. $\Phi_{C \sqcap D}(x) = \Phi_C(x) \wedge \Phi_D(x)$
- Mapping further concept constructors:

$$\Phi_{\exists R.C}(x) = \exists y. (R(x, y) \wedge \Phi_C(y))$$

$$\Phi_{\forall R.C}(x) = \forall y. (R(x, y) \rightarrow \Phi_C(y))$$

$$\Phi_{\geq n R}(x) = \exists y_1, \dots, y_n. \left(R(x, y_1) \wedge \dots \wedge R(x, y_n) \wedge \bigwedge_{i < j} y_i \neq y_j \right)$$

$$\Phi_{\leq n R}(x) = \forall y_1, \dots, y_{n+1}. \left(R(x, y_1) \wedge \dots \wedge R(x, y_{n+1}) \rightarrow \bigvee_{i < j} y_i = y_j \right)$$

Equivalent languages in the \mathcal{ALCN} family

- Language \mathcal{AL} can be extended by arbitrarily choosing whether to add each of \mathcal{UECN} , resulting in $\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{C}][\mathcal{N}]$.

Do these $2^4 = 16$ languages have different expressive power?

Two concept expressions are said to be equivalent, if they have the same meaning, in all interpretations.

Languages \mathcal{L}_1 and \mathcal{L}_2 have the same expressive power ($\mathcal{L}_1 \stackrel{e}{=} \mathcal{L}_2$), if any expression of \mathcal{L}_1 can be mapped into an equivalent expression of \mathcal{L}_2 , and vice versa.

- As a preparation for discussing the above let us recall that these axioms hold in all models, for arbitrary concepts C and D and role R :

$$C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$$

$$\exists R.C \equiv \neg \forall R. \neg C$$

$$\neg \neg C \equiv C$$

$$\neg \top \equiv \perp$$

$$\neg \perp \equiv \top$$

$$\neg(C \sqcap D) \equiv \neg C \sqcup \neg D$$

$$\neg \exists R. \top \equiv \forall R. \perp$$

$$\neg \forall R. C \equiv \exists R. \neg C$$

Equivalent languages in the \mathcal{ALCN} family

Let us show that \mathcal{ALUE} and \mathcal{ALC} are equivalent:

- As $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$ and $\exists R.C \equiv \neg\forall R.\neg C$, union and full existential restriction can be eliminated by using (full) negation. That is, to each \mathcal{ALUE} concept expression there exists an equivalent \mathcal{ALC} expression.
- The other way, each \mathcal{ALC} concept can be transformed to an equivalent \mathcal{ALUE} expression, by moving negation inwards, until before atomic concepts, and removing double negation; using the axioms from the right hand column on the previous slide
- Thus \mathcal{ALUE} and \mathcal{ALC} have the same expressive power, and so have the intermediate languages:

$$\mathcal{ALC}(N) \stackrel{e}{=} \mathcal{ALCU}(N) \stackrel{e}{=} \mathcal{ALCE}(N) \stackrel{e}{=} \mathcal{ALCUE}(N) \stackrel{e}{=} \mathcal{ALUE}(N).$$

Further remarks:

- As \mathcal{U} and \mathcal{E} is subsumed by \mathcal{C} , we will use \mathcal{ALC} to denote the language allowing \mathcal{U} , \mathcal{E} and \mathcal{C}
- It can be shown that any two of \mathcal{AL} , \mathcal{ALU} , $\mathcal{AL\mathcal{E}}$, \mathcal{ALC} , \mathcal{ALN} , \mathcal{ALUN} , $\mathcal{AL\mathcal{E}N}$, \mathcal{ALCN} have different expressive power

Another \mathcal{ALC} example requiring case analysis

- Some facts about the Oedipus family (ABox \mathcal{A}_{OE}):

$\text{hasChild}(\text{IOCASTE}, \text{OEDIPUS})$

$\text{hasChild}(\text{IOCASTE}, \text{POLYNEIKES})$

$\text{hasChild}(\text{OEDIPUS}, \text{POLYNEIKES})$

$\text{hasChild}(\text{POLYNEIKES}, \text{THERSANDROS})$

$\text{Patricide}(\text{OEDIPUS})$

$(\neg \text{Patricide})(\text{THERSANDROS})$

- Let us call a person “special” if they have a child who is a patricide and who, in turn, has a child who is **not** a patricide:

$$\text{Special} \equiv \exists \text{hasChild}.(\text{Patricide} \sqcap \exists \text{hasChild}.\neg \text{Patricide})$$

- Let TBox \mathcal{T}_{OE} contain the above axiom only.
- Consider the instance check “Is locaste special?”:
 $\mathcal{A}_{OE} \models_{\mathcal{T}_{OE}} \text{Special}(\text{IOCASTE})?$

- The answer is “yes”, but proving this requires case analysis

Contents

5

The Semantic Web

- Introducing Semantic Technologies
- An example of the Semantic Web approach
- An overview of Description Logics
- The *ALCN* language family
- **TBox reasoning**
- The *SHIQ* language family

A special case of ontology: definitional TBox

- \mathcal{T}_{fam} : a sample **definitional** TBox for family relationships

Woman	\equiv	Person \sqcap Female
Man	\equiv	Person $\sqcap \neg$ Woman
Mother	\equiv	Woman $\sqcap \exists$ hasChild.Person
Father	\equiv	Man $\sqcap \exists$ hasChild.Person
Parent	\equiv	Father \sqcup Mother
Grandmother	\equiv	Woman $\sqcap \exists$ hasChild.Parent

- A TBox is definitional if it contains equivalence axioms only, where the left hand sides are distinct concept names (atomic concepts)
- The concepts on the left hand sides are called **name symbols**
- The remaining atomic concepts are called **base symbols**, e.g. in our example the two base symbols are **Person** and **Female**.
- In a definitional TBox the meanings of name symbols can be obtained by evaluating the right hand side of their definition

Interpretations and semantic consequence

Recall the definition of assigning a truth value to TBox axioms in an interpretation \mathcal{I} :

$$\begin{aligned}\mathcal{I} \models C \sqsubseteq D & \text{ iff } C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \\ \mathcal{I} \models C \equiv D & \text{ iff } C^{\mathcal{I}} = D^{\mathcal{I}}\end{aligned}$$

Based on this we introduce the notion of “semantic consequence” exactly in the same way as for FOL

- We can naturally extend the above $\mathcal{I} \models \alpha$ notation
 - where α is either $C \sqsubseteq D$ or $C \equiv D$ –
 - to a TBox (i.e. a set of α axioms) \mathcal{T}
 - $\mathcal{I} \models \mathcal{T}$ (\mathcal{I} satisfies \mathcal{T} , \mathcal{I} is a model of \mathcal{T}) iff for each $\alpha \in \mathcal{T}$, $\mathcal{I} \models \alpha$, i.e. \mathcal{I} is a model of α
- We now overload even further the “ \models ” symbol:
 - $\mathcal{T} \models \alpha$ (read axiom α is a **semantic** consequence of the TBox \mathcal{T}) iff
 - all models of \mathcal{T} are also models of α , i.e.
 - for all interpretations \mathcal{I} , if $\mathcal{I} \models \mathcal{T}$ holds, then $\mathcal{I} \models \alpha$ also holds

TBox reasoning tasks

Reasoning tasks on TBoxes only (i.e. no ABoxes involved)

- A base assumption: the TBox is **consistent** (does not contain a contradiction), i.e. it has a model
- **Subsumption**: concept C is subsumed by concept D wrt. a TBox \mathcal{T} , iff $\mathcal{T} \models (C \sqsubseteq D)$, i.e. $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds in all \mathcal{I} models of \mathcal{T} ($C \sqsubseteq_{\mathcal{T}} D$)
e.g. $\mathcal{T}_{fam} \models (\text{Grandmother} \sqsubseteq \text{Parent})$ (recall that \mathcal{T}_{fam} is the family TBox)
- **Equivalence**: concepts C and D are equivalent wrt. a TBox \mathcal{T} , iff $\mathcal{T} \models (C \equiv D)$, i.e. $C^{\mathcal{I}} = D^{\mathcal{I}}$ holds in all \mathcal{I} models of \mathcal{T} ($C \equiv_{\mathcal{T}} D$).
e.g. $\mathcal{T}_{fam} \models (\text{Parent} \equiv \text{Person} \sqcap \exists \text{hasChild}.\text{Person})$
- **Disjointness**: concepts C and D are disjoint wrt. a TBox \mathcal{T} , iff $\mathcal{T} \models (C \sqcap D \equiv \perp)$, i.e. $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ holds in all \mathcal{I} models of \mathcal{T} .
e.g. $\mathcal{T}_{fam} \models (\text{Woman} \sqcap \text{Man}) \equiv \perp$
- Note that all these tasks involve two concepts, C and D

Reducing reasoning tasks to testing satisfiability

- We now introduce a simpler, but somewhat artificial reasoning task: checking the satisfiability of a concept
- **Satisfiability:** a concept C is satisfiable wrt. TBox \mathcal{T} , iff there is a model \mathcal{I} of \mathcal{T} such that $C^{\mathcal{I}}$ is non-empty (hence C is non-satisfiable wrt. \mathcal{T} iff in all \mathcal{I} models of \mathcal{T} $C^{\mathcal{I}}$ is empty)
- We will reduce each of the earlier tasks to checking non-satisfiability
- E.g. to prove: **Woman** \sqsubseteq **Person**, let's construct a concept C that contains all counter-examples to this statement: $C = \text{Woman} \sqcap \neg \text{Person}$
- If we can prove that C has to be empty, i.e. there are no counter-examples, then we have proven the subsumption
- Assume we have a method for checking satisfiability.
Other tasks can be reduced to this method (usable in \mathcal{ALC} and above):
 - C is subsumed by $D \iff C \sqcap \neg D$ is not satisfiable
 - C and D are equivalent $\iff (C \sqcap \neg D) \sqcup (D \sqcap \neg C)$ is not satisfiable
 - C and D are disjoint $\iff C \sqcap D$ is not satisfiable
- In simpler languages, not supporting full negation, such as \mathcal{ALN} , all reasoning tasks can be reduced to subsumption

Contents

5

The Semantic Web

- Introducing Semantic Technologies
- An example of the Semantic Web approach
- An overview of Description Logics
- The *ALCN* language family
- TBox reasoning
- The *SHIQ* language family

The \mathcal{SHIQ} Description Logic language – an overview

- Expanding the abbreviation \mathcal{SHIQ}
 - $\mathcal{S} \equiv \mathcal{ALC}_{\mathcal{R}^+}$ (language \mathcal{ALC} extended with transitive roles),
i.e. one can state that certain roles (e.g. **hasAncestor**) are transitive.
 - $\mathcal{H} \equiv$ role hierarchies. Adds statements of the form $R \sqsubseteq S$,
e.g. if a pair of objects belongs to the **hasFriend** relationship, then it must belong to the **knows** relationship too: **hasFriend** \sqsubseteq **knows**
(could be stated in English as: *everyone knows their friends*)
 - $\mathcal{I} \equiv$ inverse roles: allows using role expressions R^- to denote the inverse of role R , e.g. **hasParent** \equiv **hasChild** $^-$
 - $\mathcal{Q} \equiv$ qualified number restrictions (a generalisation of \mathcal{N}):
allows the use of concept expressions ($\leq nR.C$) and ($\geq nR.C$)
e.g. those who have at least 3 tall children : (≥ 3 **hasChild.Tall**)

\mathcal{SHIQ} language extensions – the details

- Language $\mathcal{S} \equiv \mathcal{ALC}_{\mathcal{R}^+}$, i.e, \mathcal{ALC} plus transitivity (cf. the index \mathcal{R}^+)
 - Concept axioms and concept expressions – same as in \mathcal{ALC}
 - An additional axiom type: **Trans**(R) declares role R to be transitive
- Extension \mathcal{H} – introducing role hierarchies
 - Adds role axioms of the form $R \sqsubseteq S$ and $R \equiv S$
($R \equiv S$ can be eliminated, replacing it by $R \sqsubseteq S$ and $S \sqsubseteq R$)
 - In \mathcal{SH} it is possible describe a weak form of transitive closure:

Trans(hasDescendant)
hasChild \sqsubseteq hasDescendant

- This means that hasDescendant is a transitive role which includes hasChild
- What we cannot express in \mathcal{SH} is that hasDescendant is the **smallest** such role. (This property cannot be described in FOL either.)

\mathcal{SHIQ} language extensions – the details (2)

Extension \mathcal{I} – adding inverse roles

- Our first role constructor is $^-$: R^- is the inverse of role R
- Example: consider role axiom $\text{hasChild}^- \equiv \text{hasParent}$ and:

$$\begin{aligned}\text{GoodParent} &\equiv \exists \text{hasChild}.\top \sqcap \forall \text{hasChild}.\text{Happy} \\ \text{MerryChild} &\equiv \exists \text{hasParent}.\text{GoodParent}\end{aligned}$$

A consequence of the above axioms: $\text{MerryChild} \sqsubseteq \text{Happy}$

- Multiple inverses can be eliminated: $(R^-)^- \equiv R, ((R^-)^-)^- \equiv R^-, \dots$

\mathcal{SHIQ} language extensions – the details (3)

- Extension \mathcal{Q} – qualified number restrictions – generalizing extension \mathcal{N} :
 - $(\leq nR.C)$ – the set of those who have **at most** n R -related individuals belonging to C , e.g.
 - $(\leq 2\text{hasChild.Female})$ – those with at most 2 daughters
 - $(\geq nR.C)$ – those with **at least** n R -related individuals belonging to C
- A role is *simple* if it is not transitive and does not even have a transitive sub-role
- Important: roles appearing in number restrictions have to be **simple**.
(This is because otherwise the decidability of the language would be lost.)
 - Given **Trans**(hasDesc), hasDesc is not simple.
 - If we add further role axioms: $\text{hasAnc} \equiv \text{hasDesc}^-$,
 $\text{hasAnc} \sqsubseteq \text{hasBloodRelation}$, then hasBloodRelation is not simple
 - hasAnc is transitive because its inverse hasDesc is such
 - hasBloodRelation has the transitive hasAnc as its sub-role

\mathcal{SHIQ} syntax summary

Notation

- A – atomic concept, C, C_i, D – concept expressions
- R_A – atomic role, R, R_i – role expressions,
 R_S – simple role expression, i.e. a role with no transitive sub-role

Concept expressions

DL	OWL	Name	Informal definition	
A	A	atomic concept	those in A	\mathcal{AL}
\top	owl:Thing	top	the set of all objects	\mathcal{AL}
\perp	owl:Nothing	bottom	the empty set	\mathcal{AL}
$C \sqcap D$	C and D	intersection	those in both C and D	\mathcal{AL}
$\forall R.C$	R only C	value restriction	those whose all R s belong to C	\mathcal{AL}
$C \sqcup D$	C or D	union	those in either C or D	\mathcal{U}
$\exists R.C$	R some C	existential restr.	those with an R belonging to C	\mathcal{E}
$\neg C$	not C	full negation	those not in C	\mathcal{C}
$(\leq n R_S)$	R_S max n C	qualif. num. restr.	those with at most n R_S s in C	\mathcal{Q}
$(\geq n R_S)$	R_S min n C	qualif. num. restr.	those with at least n R_S s in C	\mathcal{Q}

\mathcal{SHIQ} syntax summary (2)

- The syntax of role expressions

$R \rightarrow$	R_A	<i>atomic role</i>	(\mathcal{AL})
	R^-	<i>inverse role</i>	(\mathcal{I})

- The syntax of terminological axioms

$T \rightarrow$	$C_1 \equiv C_2$	<i>concept equivalence axiom</i>	(\mathcal{AL})
	$C_1 \sqsubseteq C_2$	<i>concept subsumption axiom</i>	(\mathcal{AL})
	$R_1 \equiv R_2$	<i>role equivalence axiom</i>	(\mathcal{H})
	$R_1 \sqsubseteq R_2$	<i>role subsumption axiom</i>	(\mathcal{H})
	Trans (R)	<i>transitivity axiom</i>	(\mathcal{R}^+)

\mathcal{SHIQ} semantics (ADVANCED)

- The semantics of concept expressions

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

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

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

$$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$$

$$(C_1 \sqcup C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$$

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

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

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

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

- The semantics of role expressions

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

\mathcal{SHIQ} semantics (2) (ADVANCED)

- The semantics of terminological axioms

$$\mathcal{I} \models C_1 \equiv C_2 \quad \Leftrightarrow \quad C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$$

$$\mathcal{I} \models C_1 \sqsubseteq C_2 \quad \Leftrightarrow \quad C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$$

$$\mathcal{I} \models R_1 \equiv R_2 \quad \Leftrightarrow \quad R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$$

$$\mathcal{I} \models R_1 \sqsubseteq R_2 \quad \Leftrightarrow \quad R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$$

$$\begin{aligned} \mathcal{I} \models \mathbf{Trans}(R) \quad \Leftrightarrow \quad & (\forall a, b, c \in \Delta^{\mathcal{I}}) \\ & (\langle a, b \rangle \in R^{\mathcal{I}} \wedge \langle b, c \rangle \in R^{\mathcal{I}} \rightarrow \langle a, c \rangle \in R^{\mathcal{I}}) \end{aligned}$$

- Read $\mathcal{I} \models T$ as: “ \mathcal{I} satisfies axiom T ” or as “ \mathcal{I} is a model of T ”

Negation normal form (NNF)

- Various normal forms are used in reasoning algorithms
- The tableau algorithms use NNF: only atomic negation allowed
- To obtain NNF, apply the following rules to subterms repeatedly while a subterm matching a left hand side can be found:

$$\neg\neg C \rightsquigarrow C$$

$$\neg(C \sqcap D) \rightsquigarrow \neg C \sqcup \neg D$$

$$\neg(C \sqcup D) \rightsquigarrow \neg C \sqcap \neg D$$

$$\neg(\exists R.C) \rightsquigarrow \forall R.(\neg C)$$

$$\neg(\forall R.C) \rightsquigarrow \exists R.(\neg C)$$

$$\neg(\leq n R.C) \rightsquigarrow (\geq k R.C) \text{ where } k = n + 1$$

$$\neg(\geq 1 R.C) \rightsquigarrow \forall R.(\neg C)$$

$$\neg(\geq n R.C) \rightsquigarrow (\leq k R.C) \text{ if } n > 1, \text{ where } k = n - 1$$

Going beyond \mathcal{SHIQ}

- Extension \mathcal{O} introduces nominals, i.e. concepts which can only have a single element. Example: $\{\text{EUROPE}\}$ is a concept whose interpretation must contain a single element
 $\text{FullyEuropean} \equiv \forall \text{hasSite}.\forall \text{hasLocation}.\{\text{EUROPE}\}$
- Extension (\mathbf{D}) : **concrete** domains, e.g. integers, strings etc, whose interpretation is fixed, cf. **data** properties in OWL
- The Web Ontology Language OWL 1 implements $\mathcal{SHOIN}(\mathbf{D})$
- OWL 2 implements $\mathcal{SROIQ}(\mathbf{D})$
- The main novelty in \mathcal{R} wrt. \mathcal{H} is the possibility to use role composition (\circ):
 $\text{hasParent} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}$
 i.e. one's parent's brother is one's uncle
- To ensure decidability, the use of role composition is seriously restricted (e.g. it is not allowed to have \equiv instead of \sqsubseteq in the above example)