Description Logics

Ulrike Sattler
Univ. of Manchester, UK

many many slides used here are borrowed from Carsten Lutz, TU Dresden

Knowledge Representation

General goal of knowledge representation:

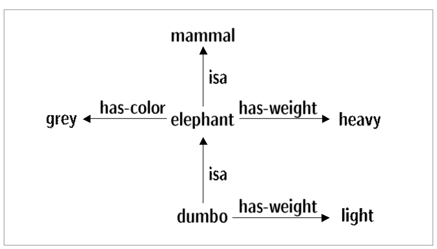
"develop formalisms for providing high-level descriptions of the world
that can be effectively used to build intelligent applications."

- "formalisms": syntax + well-defined semantics + reasoning services
- "high-level descriptions": which aspects should be represented, which left out?
- "intelligent applications": are able to infer new knowledge from given knowledge
- "effectively used": reasoning techniques should allow "usable" implementation

Early Formalisms

How to represent terminological knowledge?

Early days of AI: KR through obscure pictures (semantic networks)



Problems: missing semantics (reasoning!), complex pictures

Remedy: Use a logical formalism for KR rather than pictures

Terminological Knowledge

DLs focus: representation of terminological knowledge
or conceptual knowledge

Goal:
- formalize the basic terminology of modeled domain
- store it in an ontology / terminology / TBox for reasoning
- enable reasoning on this knowledge

Domain of Summerschools

- concepts: classes of individuals
  e.g. Course and Lecturer
- (binary) relations: links between individuals
  e.g. gives-course and attends-course

Applications

- Medical informatics

  e.g. SNOMED, the Systematized Nomenclature of Medicine

  ∼450.000 concepts about anatomy, diseases, etc.

- Bioinformatics

  e.g. the GeneOntology (GO): controlled vocabulary of genes and gene products

  ∼17.000 concepts

- Semantic Web

  goal: provide a semantic description of the content of web pages

  realization: point to concepts defined in an ontology

MANCHESTER 1824

Defining Concepts with DLs

The core part of any DL is the concept language

Person ⊓ ∃enrolled-at.University ⊓ ∀attends.UnderGradCourse

- concept names assign a name to groups of objects
- role names assign a name to relations between objects
- constructors allow to relate concept names and role names

Different sets of constructors give rise to different concept languages

MANCHESTER 1824

The Description Logic $\mathcal{ALC}$: Syntax

Atomic types:	concept names $A, B, \ldots$	(unary predicates)
	role names $R, S, \ldots$	(binary predicates)
Constructors:	- $\neg C$	(negation)
	- $C \sqcap D$	(conjunction)
	- $C \sqcup D$	(disjunction)
	- $\exists R.C$	(existential restriction)
	- $\forall R.C$	(value restriction)
Abbreviations:	- $C \rightarrow D = \neg C \sqcup D$	(implication)
	- $C \leftrightarrow D = C \rightarrow D \sqcap D \rightarrow C$	(bi-implication)
	- $\top = (A \sqcup \neg A)$	(top concept)
	- $\bot = A \sqcap \neg A$	(bottom concept)

MANCHESTER 1824

Examples

- Person ⊓ Female
- Person ⊓ ∃attends.Course
- Person ⊓ ∀attends.(Course → ¬Easy)
- Person ⊓ ∃teaches.(Course ⊓ ∀attended-by.(Nice ⊓ Intelligent))

MANCHESTER 1824

Interpretations

Semantics based on interpretations $(\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where

- $\Delta^\mathcal{I}$ is a non-empty set (the domain)
- $\cdot^\mathcal{I}$ is the interpretation function mapping

  each concept name A to a subset $A^\mathcal{I}$ of $\Delta^\mathcal{I}$ and

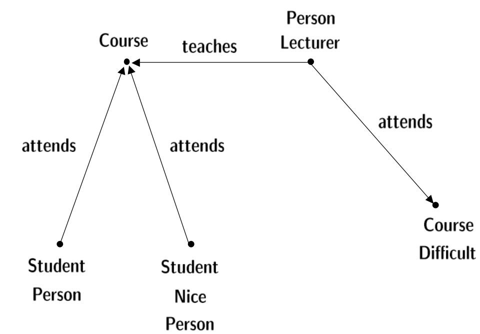
  each role name R to a binary relation $R^\mathcal{I}$ over $\Delta^\mathcal{I}$.

Intuition: interpretation is complete description of the world

Technically: interpretation is first-order structure

with only unary and binary predicates

Example

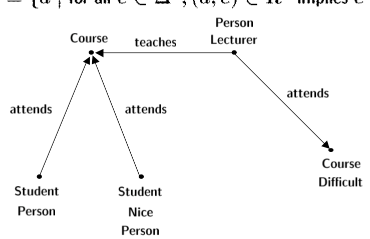


Semantics of Complex Concepts

$(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$ $\quad (C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$ $\quad (C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$

$(\exists R.C)^\mathcal{I} = \{d \mid \text{there is an } e \in \Delta^\mathcal{I} \text{ with } (d,e) \in R^\mathcal{I} \text{ and } e \in C^\mathcal{I}\}$

$(\forall R.C)^\mathcal{I} = \{d \mid \text{for all } e \in \Delta^\mathcal{I}, (d,e) \in R^\mathcal{I} \text{ implies } e \in C^\mathcal{I}\}$



Person ⊓ ∃attends.Course

Person ⊓ ∀attends.(¬Course ⊔ Difficult)

TBoxes

TBoxes are used to hold background information:

∃teaches.⊤ $\dot{\sqsubseteq}$ Lecturer ⊓ Person

TBoxes are used to hold concept definitions:

Lecturer $\doteq$ Person ⊓ ∃teaches.Course

Syntax:

A TBox is a finite set of general concept inclusion axioms $C \dot{\sqsubseteq} D$

with $C \doteq D$ an abbreviation for $C \dot{\sqsubseteq} D$ and $D \dot{\sqsubseteq} C$

Semantics:

interpretation $\mathcal{I}$ satisfies $C \dot{\sqsubseteq} D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$

$\mathcal{I}$ is model of $\mathcal{T}$ if it satisfies all GCIs in $\mathcal{T}$

TBox: Example

TBoxes are used as ontologies:

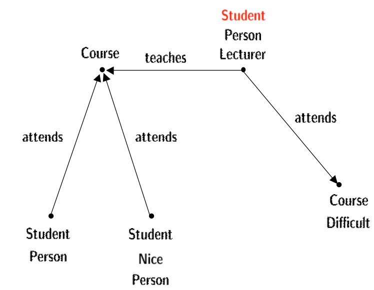
$$\exists \text{attends}.\top \,\dot\sqsubseteq\, \text{Student} \sqcap \text{Person}$$
$$\exists \text{teaches}.\top \,\dot\sqsubseteq\, \text{Lecturer} \sqcap \text{Person}$$

$$\text{Woman} \doteq \text{Person} \sqcap \text{Female}$$
$$\text{Man} \doteq \text{Person} \sqcap \neg\text{Woman}$$
$$\text{Lecturer} \doteq \text{Person} \sqcap \exists\text{teaches.Course}$$
$$\text{Student} \doteq \text{Person} \sqcap \exists\text{attends.Course}$$
$$\text{BadLecturer} \doteq \text{Person} \sqcap \forall\text{teaches.}(\text{Course} \rightarrow \text{Boring})$$

TBox: Example II

A TBox restricts the set of admissible interpretations.

$$\text{Lecturer} \doteq \text{Person} \sqcap \exists\text{teaches.Course}$$
$$\text{Student} \doteq \text{Person} \sqcap \exists\text{attends.Course}$$



Reasoning Tasks — Subsumption

C subsumed by D w.r.t. $\mathcal{T}$ (written $C \sqsubseteq_{\mathcal{T}} D$)

if

$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$

Intuition: If $C \sqsubseteq_{\mathcal{T}} D$, then D is more general than C

Example:

For $\mathcal{T} = \{\text{Lecturer} \doteq \text{Person} \sqcap \exists\text{teaches.Course},$
$\text{Student} \doteq \text{Person} \sqcap \exists\text{attends.Course}\}$

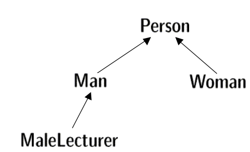
we have that

$$\text{Lecturer} \sqcap \exists\text{attends.Course} \sqsubseteq_{\mathcal{T}} \text{Student}$$

Reasoning Tasks — Classification

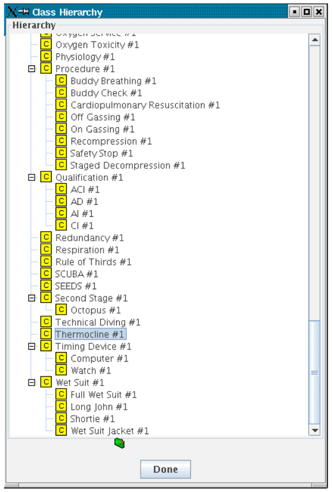
Classification: arrange all concept names from a TBox in a hierarchy w.r.t. generality

$$\text{Woman} \doteq \text{Person} \sqcap \text{Female}$$
$$\text{Man} \doteq \text{Person} \sqcap \neg\text{Woman}$$
$$\text{MaleLecturer} \doteq \text{Man} \sqcap \exists\text{teaches.Course}$$



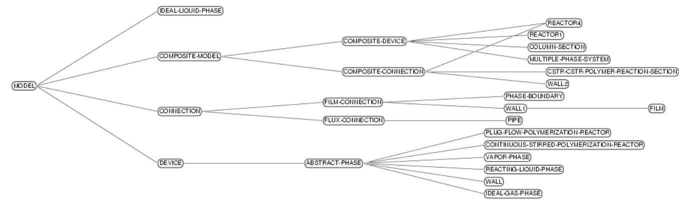
Can be computed using multiple subsumption tests

Provides a principled view on ontology for browsing, maintaining, etc.



A Concept Hierarchy

Excerpt from a process engineering ontology



Reasoning Tasks — Satisfiability

C is satisfiable w.r.t. $\mathcal{T}$ if $\mathcal{T}$ has a model with $C^{\mathcal{I}} \neq \emptyset$

Intuition: If unsatisfiable, the concept contains a contradiction.

Example: Woman $\doteq$ Person $\sqcap$ Female

Man $\doteq$ Person $\sqcap \neg$Woman

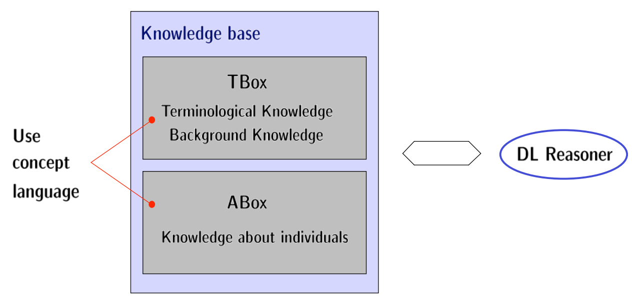
Then $\exists$sibling.Man $\sqcap \forall$sibling.Woman is unsatisfiable w.r.t. $\mathcal{T}$

Subsumption can be reduced to (un)satisfiability and vice versa:

- $C \sqsubseteq_{\mathcal{T}} D$ iff $C \sqcap \neg D$ is not satisfiable w.r.t. $\mathcal{T}$
- C is satisfiable w.r.t. $\mathcal{T}$ if not $C \sqsubseteq_{\mathcal{T}} \bot$.

Many reasoners decide satisfiability rather than subsumption.

DLs are more than a Concept Language



Definitorial TBoxes

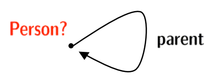
A concept name A is defined in $\mathcal{T}$ if $\mathcal{T}$ contains exactly 1 GCI of the form $A \doteq C$,
all other concept names are primitive in $\mathcal{T}$

A primitive interpretation for TBox $\mathcal{T}$ interpretes
the primitive concept names in $\mathcal{T}$ and all role names

A TBox is called definitorial if every primitive interpretation for $\mathcal{T}$
can be uniquely extended to a model of $\mathcal{T}$.
i.e.: primitive concepts (and roles) uniquely determine defined concepts

Not all TBoxes are definitorial:

Person $\doteq$ $\exists$parent.Person



Non-definitorial TBoxes describe constraints, e.g. from background knowledge

Acyclic TBoxes

TBox $\mathcal{T}$ is acyclic if there are no definitorial cycles:

Lecturer $\doteq$ Person $\sqcap$ $\exists$teaches.Course
Course $\doteq$ $\exists$has-title.Title $\sqcap$ $\exists$taught-by.Lecturer

Expansion of acyclic TBox $\mathcal{T}$:

exhaustively replace defined concept names with their definition
(terminates due to acyclicity)

Acyclic TBoxes are always definitorial:

first expand, then set $A^{\mathcal{I}} := C^{\mathcal{I}}$ for all $A \doteq C \in \mathcal{T}$

Acyclic TBoxes II

For reasoning, acyclic TBox can be eliminated:

- to decide $C \sqsubseteq_{\mathcal{T}} D$ with $\mathcal{T}$ acyclic,
  - expand $\mathcal{T}$
  - replace defined concept names in C, D with their definition
  - decide $C \sqsubseteq D$
- analogously for satisfiability

May yield an exponential blow-up:

$$A_0 \doteq \forall r.A_1 \sqcap \forall s.A_1$$
$$A_1 \doteq \forall r.A_2 \sqcap \forall s.A_2$$
$$\cdots$$
$$A_{n-1} \doteq \forall r.A_n \sqcap \forall s.A_n$$

General Concept Inclusions

Recall: our TBoxes are general: finite set of general concept inclusions (GCIs)

$$C \dot{\sqsubseteq} D$$

with both C and D allowed to be complex

e.g. Course $\sqcap$ $\forall$attended-by.Sleeping $\dot{\sqsubseteq}$ Boring
e.g. Student $\sqcap$ $\exists$has-favourite.FootballTeam $\doteq$ Student $\sqcap$ $\exists$has-favourite.Beer

Recall: $C \doteq D$ is an abbreviation for $C \dot{\sqsubseteq} D$, $D \dot{\sqsubseteq} C$

Note: $C \dot{\sqsubseteq} D$ is equivalent to $\top \doteq C \rightarrow D$ is equivalent to $\top \dot{\sqsubseteq} \neg C \sqcup D$

ABoxes

ABoxes describe a snapshot of the world

An ABox is a finite set of assertions

$a : C$ (a individual name, C concept)
$(a, b) : R$ (a, b individual names, R role name)

E.g. {peter : Student, (uli, dl-course) : teaches}

Interpretations $\mathcal{I}$ map each individual name a to an element of $\Delta^{\mathcal{I}}$.

$\mathcal{I}$ satisfies an assertion

$a : C$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
$(a, b) : R$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

$\mathcal{I}$ is a model for an ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies all assertions in $\mathcal{A}$.

ABoxes II

Note:

- interpretations describe the state if the world in a complete way
- ABoxes describe the state if the world in an incomplete way

(uli, dl-course) : teaches and dl-course : ContainsLogic
does not imply
uli : ∀teaches.ContainsLogic

An ABox has many models!

Aspect of the Open world assumption of DLs

An ABox constrains the set of admissible models similar to a TBox:
the more assertions/GCIs, the fewer models

Reasoning with ABoxes

ABox consistency

Given an ABox $\mathcal{A}$ and a TBox $\mathcal{T}$, do they have a common model?

Instance checking

Given an ABox $\mathcal{A}$, a TBox $\mathcal{T}$, an individual name a, and a concept C
does $a^{\mathcal{I}} \in C^{\mathcal{I}}$ hold in all models of $\mathcal{A}$ and $\mathcal{T}$?
(written $\mathcal{A}, \mathcal{T} \models a : C$)

The two tasks are interreducible:

- $\mathcal{A}$ consistent w.r.t. $\mathcal{T}$ iff $\mathcal{A}, \mathcal{T} \not\models a : \bot$
- $\mathcal{A}, \mathcal{T} \models a : C$ iff $\mathcal{A} \cup \{a : \neg C\}$ is not consistent

Example for ABox Reasoning

ABox

dumbo : Mammal
~~g23 : Darkgrey~~
(dumbo, g23) : color
dumbo : ∀color.Lightgrey
t14 : Trunk
(dumbo, t14) : bodypart

TBox

Elephant $\doteq$ Mammal ⊓ ∃bodypart.Trunk ⊓ ∀color.Grey
Grey $\doteq$ Lightgrey ⊔ Darkgrey
$\bot \doteq$ Lightgrey ⊓ Darkgrey

1. ABox is inconsistent w.r.t. TBox.
2. dumbo is an instance of Elephant: TBox, ABox $\models$ dumbo : Elephant

ABox Reasoning vs. Concept Reasoning

Concept reasoning can be reduced to ABox reasoning:

- C satisfiable w.r.t. $\mathcal{T}$ iff $a : C$ is consistent
- $C \sqsubseteq_{\mathcal{T}} D$ iff $\{a : C\}, \mathcal{T} \models a : D$

In $\mathcal{ALC}$, ABox reasoning can also be reduced to concept reasoning:

To decide whether $\mathcal{A}$ is consistent:

1. Precompletion: explicate knowledge in $\mathcal{A}$ by applying rules such as:

$$a : (C \sqcap D) \Longrightarrow a : C \text{ and } a : D$$
$$a : \forall r.C \text{ and } (a, b) : r \Longrightarrow b : C$$

2. For each of the resulting ABoxes $\mathcal{A}_1, \ldots, \mathcal{A}_k$, and each individual a, check whether the conjunction of $\{C \mid a : C \in \mathcal{A}_i\}$ is satisfiable.

Good Morning!

Yesterday:

- ALC, syntax, semantics
- TBox and ABox
- reasoning problems:
  - subsumption
  - classification
  - instance
  - consistency

Next:

- DLs, FOL and modal logic
- DLs and OWL
- reasoning algorithms:
  - tableau-based
  - automata-based
- computational complexity

Description Logics and First-order Logic

concept names A	$\Longleftrightarrow$	unary predicates P_A
role names R	$\Longleftrightarrow$	binary predicates P_R
concepts	$\Longleftrightarrow$	formulas with one free variable

$$\begin{aligned}
\varphi^x(A) &= P_A(x) \\
\varphi^x(\neg C) &= \neg\varphi^x(C) \\
\varphi^x(C \sqcap D) &= \varphi^x(C) \wedge \varphi^x(D) \\
\varphi^x(C \sqcup D) &= \varphi^x(C) \vee \varphi^x(D) \\
\varphi^x(\exists R.C) &= \exists y.P_R(x,y) \wedge \varphi^y(C) \\
\varphi^x(\forall R.C) &= \forall y.P_R(x,y) \rightarrow \varphi^y(C)
\end{aligned}$$

φ^y symmetric with x and y exchanged

Note: not all DLs are purely first-order (transitive closure, etc.)

Description Logics and First-order Logic II

TBoxes:

Let $\mathcal{T}$ a general TBox.

$$\varphi(\mathcal{T}) = \forall x. \bigwedge_{D \sqsubseteq E \in \mathcal{T}} \varphi^x(D) \rightarrow \varphi^x(E)$$

ABoxes:

individual names a $\Longleftrightarrow$ constants c_a

$$\begin{aligned}
\varphi(a : C) &= \varphi^x(C)[c_a] \\
\varphi((a, b) : R) &= P_R(c_a, c_b) \\
\varphi(\mathcal{A}) &= \bigwedge_{\beta \in \mathcal{A}} \varphi(\beta)
\end{aligned}$$

Description Logics and Modal Logic

concept names A	$\Longleftrightarrow$	propositional variables p_A
role names R	$\Longleftrightarrow$	modal parameters R
concepts	$\Longleftrightarrow$	multi modal formulas

$$
\begin{aligned}
\varphi(A) &= p_A\\
\varphi(\neg C) &= \neg\varphi(C)\\
\varphi(C \sqcap D) &= \varphi(C) \wedge \varphi(D)\\
\varphi(C \sqcup D) &= \varphi(C) \vee \varphi(D)\\
\varphi(\exists R.C) &= \langle R\rangle\varphi(C)\\
\varphi(\forall R.C) &= [R]\varphi(C)
\end{aligned}
$$

Description Logics and Modal Logic

TBoxes:

Let $\mathcal{T}$ a general TBox und U the universal modal parameter.

$$\varphi(\mathcal{T}) = [U] \bigwedge_{D \sqsubseteq E \in \mathcal{T}} \varphi(D) \rightarrow \varphi(E)$$

ABoxes:

individual names a $\Longleftrightarrow$ nominals a

$$
\begin{aligned}
\varphi(a : C) &= @_a\varphi(C)\\
\varphi((a,b) : R) &= @_a\langle R\rangle b\\
\varphi(\mathcal{A}) &= \bigwedge_{\beta \in \mathcal{A}} \varphi(\beta)
\end{aligned}
$$

So far, we have seen

- syntax and semantics of ***ALC***
- Tboxes (also acyclic and general ones)
- Aboxes
- reasoning services and their relationship
  - subsumption and satisfiability of possibly w.r.t. a Tbox
  - Abox consistency and instance checking
- relationship between ***ALC*** and FOL and Modal Logic

Today, we will see

- some more expressive Description Logics
- relationship between DLs and OWL
- tableau algorithms for ***ALC*** and its extension
  - with general Tboxes
  - with inverse roles
- and discuss optimisation techniques for these algorithms

Wednesday: automata-based algorithms
Thursday: computational complexity
Friday: sub-Boolean DLs and rules

Extensions of $\mathcal{ALC}$

Lecturer $\doteq$ Person $\sqcap$ $\exists$teaches.Course

Course $\doteq$ $\exists$has-title.Title $\sqcap$ $\exists$taught-by.Lecturer

We should want that:

$(x,y) \in$ teaches$^{\mathcal{I}}$ iff $(y,x) \in$ taught-by$^{\mathcal{I}}$

e.g., so that Lecturer $\sqcap$ Blond $\sqcap$ $\forall$teaches.$\forall$taught-by.$\neg$Blond is unsatisfiable!

Extension of $\mathcal{ALC}$ to $\mathcal{ALCI}$:

- allow inverse roles r^- in place of role names
- semantics ensures that $(r^-)^{\mathcal{I}}$ is converse of $r^{\mathcal{I}}$

Lecturer $\doteq$ Person $\sqcap$ $\exists$teaches.Course

Course $\doteq$ $\exists$has-title.Title $\sqcap$ $\exists$teaches$^-$.Lecturer

now Lecturer $\sqcap$ Blond $\sqcap$ $\forall$teaches.$\forall$teaches$^-$.$\neg$Blond is unsatisfiable!

Extensions of $\mathcal{ALC}$ II

In $\mathcal{ALC}$, how to say that

- a small course has at most 10 students
- a shared course is taught by at least two lecturers
- every person has exactly two hands

Extension of $\mathcal{ALC}$ to $\mathcal{ALCQ}$/of $\mathcal{ALCI}$ to $\mathcal{ALCQI}$:

- new concept constructors
  $(\leq n\ R\ C)$ and $(\geq n\ R\ C)$ (qualified number restrictions)
- e.g. SmallCourse $\doteq$ Course $\sqcap$ $(\leqslant 10$ attended-by Student$)$
- sometimes only available in "unqualified", indicated by $\mathcal{N}$:
  $(\leq n\ R\ \top)$ and $(\geq n\ R\ \top)$ (number restrictions)

MANCHESTER 1824

Extensions of $\mathcal{ALC}$ III

In $\mathcal{ALC}$, how to model the interaction between

- the relation between has-daughter and has-child
- the part-whole relation, i.e., the one between parts and wholes
- the relation between has-child and has-descendant

Extension of $\mathcal{ALC}$ with transitive roles is called $\mathcal{S}$,
transitive roles and role hierarchies is called $\mathcal{SH}$:

- new inclusions $R \sqsubseteq S$ in TBox (or RBox)
- new statements $\mathtt{Tr}(R)$ in TBox (or RBox)

In $\mathcal{ALC}$, how to model persons who have seen Mona Lisa?
In $\mathcal{ALC}$, how to model .....

But: Increasing expressivity may increase computational complexity

$\Longrightarrow$!! tradeoff between expressivity and computational complexity !!

MANCHESTER 1824

Description Logics and Computational Complexity

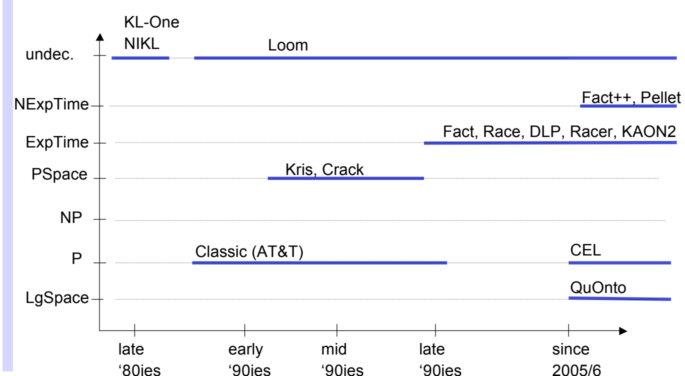
Numerous complexity results have been established for DLs. In general:

- we are interested in the decidability/worst-case complexity of determining the
  - subsumption between two concepts (w.r.t. a general/restricted TBox)
  - satisfiability of a concept (w.r.t. a general/restricted TBox)
  - consistency of an ABox w.r.t. a TBox
  - retrieving instances of a concept from an ABox and TBox
  - etc.
- for DLs, these problems are decidable and anywhere from LogSpace, P, PSpace, ExpTime, and NExpTime-complete
- but people are strongly interested in implementations of decision procedures for these reasoning problems: so "practicable" is important

MANCHESTER 1824

Development of DL Systems

Description Logics should be decidable. But what complexity is "ok"/implemented?



MANCHESTER 1824

DLs and OWL

- originally, DLs were designed to represent terminological knowledge (TBox) and partial descriptions of the world (ABox)
- they turned out to be useful as ontology languages, and thus they form the logical basis of
  - Oil, DAML+Oil
  - OWL-light is based in $\mathcal{SHIN}$
  - OWL-DL is based on $\mathcal{SHOIN}$, and
  - OWL 1.1 is based on $\mathcal{SROIQ}$
- hence ontology designers/users can make use of DL reasoners to check ontologies for consistency/answer queries, etc.
- and ontology editors such as Protégé are now connected to DL reasoners

Description Logics and OWL-DL

concept names A $\Longleftrightarrow$ OWL class names A
role names R $\Longleftrightarrow$ OWL properties R
concepts $\Longleftrightarrow$ OWL class expressions

Constructor	DL Syntax	Example
intersectionOf	$C_1 \sqcap C_2$	Human $\sqcap$ Male
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer
complementOf	$\neg C$	$\neg$Male
oneOf	$\{x_1 \ldots x_n\}$	{john, mary}
allValuesFrom	$\forall PC$	$\forall$hasChildDoctor
someValuesFrom	$\exists rC$	$\exists$hasChild.Lawyer
hasValue	$\exists r\{x\}$	$\exists$citizenOf.{USA}
minCardinality	$\geq n\ r$	$\geq$ 2 hasChild
maxCardinality	$\leq n\ r$	$\leq$ 1 hasChild
inverseOf	r^-	hasChild$^-$

Description Logics and OWL-DL

Axiom	DL Syntax	Example
subClassOf	$C_1 \dot\sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \doteq C_2$	Man$\doteq$Human $\sqcap$ Male
subPropertyOf	$P_1 \dot\sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \doteq P_2$	cost$\doteq$price
disjointWith	$C_1 \dot\sqsubseteq \neg C_2$	Male $\sqsubseteq$ $\neg$Female
sameAs	$\{x_1\} \doteq \{x_2\}$	{Pres._Bush}$\doteq${G.W.Bush}
differentFrom	$\{x_1\} \dot\sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq$ $\neg${peter}
TransitiveProperty	$\mathtt{Tr}(P)$	Tr(hasAncestor)
FunctionalProperty	$\top \dot\sqsubseteq \leq 1\ P$	$\top \sqsubseteq \leq 1$ hasMother
InverseFunctionalProperty	$\top \dot\sqsubseteq \leq 1\ P^-$	$\top \sqsubseteq \leq 1$ isMotherOf$^-$
SymmetricProperty	$P \doteq P^-$	isSiblingOf$\doteq$isSiblingOf$^-$

Overview of the Course

- Tableau algorithms for Description Logics
- Automata-based decision procedures for Description Logics
- Computational complexity of selected Description Logics
- Sub-Boolean Description Logics and Non-Standard Reasoning