INF3170 – Logikk

Forelesning 7: Description Logic 2

Espen H. Lian

Institutt for informatikk, Universitetet i Oslo

19. november 2013

Dagens plan

Introduction

Description Logics

- **Description logics** are a class of decidable logics primarily used for knowledge representation.
- The language consists of **concepts** of an application domain,
- and the hierarchical structure of the application domain is expressed through **terminological axioms**.
- Some examples of terminological axiom about animals are:
  1. Wolf $\sqsubseteq$ Carnivore
     ("A wolf is a carnivore")
  2. Carnivore $\equiv$ Animal $\sqcap$ $\exists$eats.Animal
     ("The definition of a carnivore is an animal that eats animals")
- Being decidable, description logics can be classified according to their complexity.

Introduction

Complexity Theory

- **Complexity theory** classifies problems according to how much resources are necessary/sufficient to solve them.
- In our case, the problems are **reasoning** problems, such as: Does one concept subsume another concept?
- An example of subsumption is:
  - Does Wolf $\sqsubseteq$ Animal follow from the axioms? ("Is a wolf an animal?")
- Given the terminological axioms on the previous foil (also to the right), it follows that a wolf is an animal:
  - A wolf is a carnivore (1st axiom), and
  - a carnivore is an animal (follows from the 2nd),
- In general, reasoning is hard.

Introduction

Description Logic and Their Complexity

- The more expressive the logic, the higher the complexity of reasoning.
- The goal when designing a language is to maximize the expressivity while staying within a certain complexity class.
- Earlier, one tried to maximize the expressivity while retaining decidability.
- Now, it is more common to try to maximize the expressivity while staying within a certain (typically tractable) complexity class.
- We will consider different logics that have been designed with this in mind.
- Last time we saw $\mathcal{ALC}$, a logic with a simple syntax: Boolean connectives, value restriction and existential quantification.

Complexity Classes

Classes

- A **complexity class** is a set of decision problems that can be decided within some specific bound on the size of the input in some computational model.
- The bounds are usually on
  - time, or
  - space.
- The computational model is usually
  - a deterministic Turing machine (DTM), or
  - a non-deterministic Turing machine (NDTM).
- E.g., **NP** is the class of decision problems solvable in polynomial time on a non-deterministic Turing machine.
- In addition, $\mathbf{AC}_0$ is a circuit complexity class consisting of constant-depth unlimited-fanin circuits.

Complexity Classes

Classes

Class	Model	Bound	Example Growth
L	DTM	logarithmic space	$\log n$
NL	NDTM	logarithmic space	
P	DTM	polynomial time	n^2
NP	NDTM	polynomial time	
PSPACE	DTM	polynomial space	
EXPTIME	DTM	exponential time	2^n
NEXPTIME	NDTM	exponential time	
2EXPTIME	DTM	doubly exponential time	2^{2^n}
2NEXPTIME	NDTM	doubly exponential time	

Complexity Classes

Relationships

- The following are the known relationships between the classes.

$$\begin{aligned}\mathbf{AC}_0 &\subseteq \mathbf{L} \subseteq \mathbf{NL}\\ &\subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}\\ &\subseteq \mathbf{EXPTIME} \subseteq \mathbf{NEXPTIME}\\ &\subseteq \mathbf{2EXPTIME} \subseteq \mathbf{2NEXPTIME}\end{aligned}$$

- Tractable problems are those in **P**
- Furthermore,

$$\begin{aligned}\mathbf{AC}_0 &\subset \mathbf{L}\\ \mathbf{NL} &\subset \mathbf{PSPACE}\\ \mathbf{P} &\subset \mathbf{EXPTIME}\end{aligned}$$

Complexity Classes

Completeness

- A **reduction** translates one problem into another.
- A problem is $\mathcal{C}$-**hard** (under a given type of reduction, typically polynomial time or logarithmic space for **NP** and up) if every problem in $\mathcal{C}$ can be reduced to it.
- A problem is $\mathcal{C}$-**complete** if it is
  - in $\mathcal{C}$ (upper bound) and
  - $\mathcal{C}$-hard (lower bound).
- It follows that a problem is
  - is not harder than $\mathcal{C}$ if it reduces to some problem in $\mathcal{C}$, and
  - $\mathcal{C}$-hard if some $\mathcal{C}$-hard problem reduces to it.
- If we only consider the ABox as input, we say that a problem is $\mathcal{C}$-complete in **data complexity**.
- Else we say that a problem is $\mathcal{C}$-complete in **combined complexity**.

Logics
$\mathcal{ALC}$

Concept satisfiability of $\mathcal{ALC}$ is intractable:

- **PSPACE**-complete for an **empty** TBox
- **EXPTIME**-complete for a **general** TBox

Intractability of $\mathcal{ALC}$ raises two questions:

1. Can we extend $\mathcal{ALC}$ without getting an even more intractable logic?
2. Are there less complex description logics that are useful in practice?

The answer to both these questions is “yes.”

Logics
Complex roles

Before we introduce additional complex concepts, we introduce complex roles (and one concept):

- U universal role
- R^- inverse role
- $\neg R$ negated role
- $R \circ S$ role composition
- $\exists R.\mathsf{Self}$ local reflexivity

The semantics of the first three is as follows.

- $U^\mathfrak{A} = \Delta \times \Delta$
- $(R^-)^\mathfrak{A} = (R^\mathfrak{A})^- = \{\langle b, a\rangle \in \Delta \times \Delta \mid \langle a, b\rangle \in R^\mathfrak{A}\}$
- $(\neg R)^\mathfrak{A} = \Delta \times \Delta \setminus R^\mathfrak{A}$

Logics
Complex roles

What are the following equal to?

- $(U^-)^\mathfrak{A}$
- $(\neg U)^\mathfrak{A}$
- $(\exists U.\top)^\mathfrak{A}$
- $(\forall U.\top)^\mathfrak{A}$

Logics
Complex roles

- Role composition lets us create a new role by composing two old roles.
- The semantics is as follows.

$$\begin{aligned} &(R \circ S)^\mathfrak{A} = S^\mathfrak{A} \circ R^\mathfrak{A} \\ = &\{\langle a, c\rangle \in \Delta \times \Delta \mid \langle a, b\rangle \in R^\mathfrak{A} \text{ and } \langle b, c\rangle \in S^\mathfrak{A} \text{ for some } b \in \Delta\} \end{aligned}$$

- This lets us express certain concepts that the two-variable nature of concepts won't less us do, such as “is the uncle of:”

$$\mathsf{hasBrother} \circ \mathsf{hasChild} \sqsubseteq \mathsf{isUncleOf}$$

Logics

Complex roles

- Local reflexivity lets us express the "diagonal."
- The semantics is as follows.

$$(\exists R.\mathsf{Self})^{\mathfrak{A}} = \{a \mid \langle a, a\rangle \in R^{\mathfrak{A}}\}$$

- This lets us express, e.g., the concept "narcissist:"

$$\exists \mathsf{likes}.\mathsf{Self}$$

- Observe that Self itself is not a concept, thus the syntax is a bit misleading.

Logics

Complex roles

In addition to a TBox and an ABox, we may have an **RBox**, a finite set of **role axioms**, of the form,

- $R \sqsubseteq S$ (inclusions)
- $R \equiv S$ (equalities)

for roles R and S (given some restrictions).

An interpretation $\mathfrak{A}$ **satisfies**

- $R \sqsubseteq S$ if $R^{\mathfrak{A}} \subseteq S^{\mathfrak{A}}$
- $R \equiv S$ if $R^{\mathfrak{A}} = S^{\mathfrak{A}}$

Logics

Complex roles

With local reflexivity, we can express the following properties of a role R:

- reflexivity $\top \sqsubseteq \exists R.\mathsf{Self}$
- irreflexivity $\top \sqsubseteq \neg\exists R.\mathsf{Self}$

With inclusions on complex roles, we can express the following properties of a role R:

- symmetry $R^- \sqsubseteq R$
- transitivity $R \circ R \sqsubseteq R$
- disjointness (with S) $R \sqsubseteq \neg S$

Logics

Logics

Some basic logics are the following.

$\mathcal{AL}$	• intersection	
	• atomic negation	$\neg C$ for an atomic concept C
	• universal restriction	
	• **limited** existential quantification	$\exists R.\top$
$\mathcal{EL}$	• intersection	
	• universal concept	
	• **full** existential quantification	$\exists R.C$
$\mathcal{SR}$	• contains $\mathcal{ALC}$	
	• role inclusions	
	• disjointness	

Logics

Logics

The naming convention for constructing new languages is as follows.

$\mathcal{C}$	complex concept negation	$\neg C$ for any concept C
$\mathcal{O}$	nominals	$\{a\}$
$\mathcal{I}$	role inverse	R^{-}
$\mathcal{Q}$	qualified number restrictions	$\leqslant n\, R.C$ and $\geqslant n\, R.C$
$\mathcal{U}$	concept union	$C \sqcup D$
$\mathcal{E}$	full existential quantification	$\exists R.C$

Logics

Constructing languages

- Now we can construct more expressive logics.
- $\mathcal{ALC}$ is $\mathcal{AL}$ extended with complex concept negation,
- "$\mathcal{ALC} = \mathcal{AL} + \mathcal{C}$" :

$$\top \equiv C \sqcup \neg C$$
$$\exists R.C \equiv \neg\forall R.\neg C$$
$$C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$$

- This way of constructing languages is not unique, e.g.,
- "$\mathcal{ALC} = \mathcal{AL} + \mathcal{U} + \mathcal{E}$"

Logics

$\mathcal{O}$: Nominals

- Nominals are singleton concepts, i.e. of the form $\{a\}$ for some individual a.
- The semantics is as follows.

$$\{a\}^{\mathfrak{A}} = \{a^{\mathfrak{A}}\}$$

- Using union and nominals, one may express enumerations such as

$$\text{Magi} \equiv \{\text{Melchior}\} \sqcup \{\text{Caspar}\} \sqcup \{\text{Balthazar}\}$$

- Then

$$\text{Magi}^{\mathfrak{A}} = \{\text{Melchior}^{\mathfrak{A}}, \text{Caspar}^{\mathfrak{A}}, \text{Balthazar}^{\mathfrak{A}}\}$$

Logics

$\mathcal{Q}$: Qualified number restrictions

- Qualified number restrictions, i.e. concepts of the form
  - $\leqslant n\, R.C$ and
  - $\geqslant n\, R.C$,

  let us restrict the number of individuals related by a role R.
- The semantics is as follows.

$$\leqslant n\, R.C^{\mathfrak{A}} = \{x \in \Delta \mid |R^{\mathfrak{A}}(x) \cap C^{\mathfrak{A}}| \leqslant n\}$$
$$\geqslant n\, R.C^{\mathfrak{A}} = \{x \in \Delta \mid |R^{\mathfrak{A}}(x) \cap C^{\mathfrak{A}}| \geqslant n\}$$

- *"A monotheist worships exactly one deity"*:
  - Monotheist $\sqsubseteq$ $\leqslant 1$ worships.Deity
  - Monotheist $\sqsubseteq$ $\geqslant 1$ worships.Deity
- Observe that $\geqslant 1\, R.C$ is equivalent to $\exists R.C$.

Logics

OWL 2

OWL 2 (The **W**eb **O**ntology **L**anguage) is an ontology language for the Semantic Web, based on description logic.

- **OWL 2 DL**
  - based on the logic $\mathcal{SROIQ}$
  - high expressivity, but also high complexity
- **OWL 2 QL** is a fragment of OWL 2 DL
  - based on the logic $\mathcal{DL}$-Lite$_{\mathcal{R}}$
  - low data complexity of query answering: suitable for querying relational databases (without altering them)
- **OWL 2 EL** is a fragment of OWL 2 DL
  - based on the logic $\mathcal{EL}^{++}$
  - low combined complexity of subsumption: suitable for large TBoxes

Logics

OWL 2 DL: $\mathcal{SROIQ}$

- $\mathcal{SROIQ} = \mathcal{SR} + \mathcal{O} + \mathcal{I} + \mathcal{Q}$
- Hence $\mathcal{SROIQ}$ contains $\mathcal{ALC}$.
- Being so expressive, the complexity of $\mathcal{SROIQ}$ is high.
- Concept satisfiability of $\mathcal{SROIQ}$ is **2NEXPTIME**-complete, thus harder that $\mathcal{ALC}$.
- But there are extensions of $\mathcal{ALC}$ that are not harder than $\mathcal{ALC}$ itself.
- E.g., satisfiability of $\mathcal{ALCQ}$ concepts ($\mathcal{ALC}$ extended with qualified number restrictions) is not harder than $\mathcal{ALC}$.

Logics

OWL 2 DL: $\mathcal{SROIQ}$

A role R being **non-simple** in a TBox $\mathcal{T}$ is given by the following rules:

- If $S \circ T \sqsubseteq R \in \mathcal{T}$, then R is non-simple;
- R^- is non-simple if R is non-simple;
- S is non-simple if R is non-simple and
  - $R \sqsubseteq S \in \mathcal{T}$ or
  - $R \equiv S \in \mathcal{T}$ or
  - $S \equiv R \in \mathcal{T}$.

A role is **simple** in a TBox $\mathcal{T}$ if it is not non-simple in $\mathcal{T}$.

Simple roles are required in the following concepts and axioms:

- $\exists R.\mathrm{Self}$, $\leqslant n\, R.C$ and $\geqslant n\, R.C$
- disjointness

Logics
OWL 2 QL: $\mathcal{DL}$-Lite

- $\mathcal{DL}$-Lite is a family of DLs with low complexity.
- We consider $\mathcal{DL}$-Lite$_\mathcal{R}$ (OWL 2 QL), where $B \sqsubseteq C$ is a concept inclusion, given the grammar:

$$\begin{aligned} B &\longrightarrow A \,|\, \exists Q \\ C &\longrightarrow B \,|\, \neg B \,|\, \exists Q.C \end{aligned}$$

and $Q \sqsubseteq R$ is a role inclusion, given the grammar:

$$\begin{aligned} Q &\longrightarrow P \,|\, P^- \\ R &\longrightarrow Q \,|\, \neg Q \end{aligned}$$

- $\exists Q$ is equivalent to $\exists Q.\top$.
- There is no unique name assumption (UNA).

Logics
OWL 2 QL: $\mathcal{DL}$-Lite

Thus you cannot have $\exists Q.C$ on the left-hand side of an inclusion.

✓ $B \sqsubseteq \exists Q.C$
✗ $\exists Q.C \sqsubseteq B$

But we don't really need $\exists Q.C$ on the right-hand side either.

Just replace $B \sqsubseteq \exists Q.C$ with

$B \sqsubseteq \exists P$	every B is P-related to something
$\exists P^- \sqsubseteq C$	the range of P is C
$P \sqsubseteq Q$	P is a subrole of Q

where P is a fresh atomic role.

Logics
$\mathcal{DL}$-Lite and FO-rewritability

- We now consider the reasoning problem of **query answering**.
- An **atom** is either of the form
  - $A(x)$, where A is an atomic concept, or
  - $P(x, y)$, where P is an atomic role.
- Recall that a **conjunctive query** (CQ) is a FO formula of the form

$$\exists \vec{x}.\varphi(\vec{x}, \vec{y})$$

where $\varphi(\vec{x}, \vec{y})$ is a conjunction of atoms with free variables $\vec{y}$.
- A **union of conjunctive queries** (UCQ) is a disjunction of conjunctive queries:

$$\exists \vec{y}_1.\varphi(\vec{x}_1, \vec{y}_1) \vee \cdots \vee \exists \vec{y}_n.\varphi(\vec{x}_n, \vec{y}_n)$$

Logics
$\mathcal{DL}$-Lite and FO-rewritability

- UCQ answering in a description logic is **FO-rewritable** if it can be reduced to FO query (basically SQL) over the ABox considered as a relational database, where the TBox is "baked" into the query.
- FO query answering over a relational database is in $\mathbf{AC}_0$.
- UCQ answering of $\mathcal{DL}$-Lite$_\mathcal{R}$ is FO-rewritable.
- Thus UCQ answering of $\mathcal{DL}$-Lite$_\mathcal{R}$ is in $\mathbf{AC}_0$ in data complexity.
- **P** is **not** tractable when it comes to query answering.
- UCQ answering of $\mathcal{DL}$-Lite$_\mathcal{R}$ is **NP**-complete in combined complexity. Hardness follows from hardness of CQ answering over relational databases.

Logics
$\mathcal{DL}$-Lite and FO-rewritability

A $\mathcal{DL}$-Lite$_{\mathcal{R}}$ knowledge base $\mathcal{K}_2 = \langle \mathcal{T}_2, \mathcal{A}_2 \rangle$:

Example

TBox $\mathcal{T}_2$ ("An employee works for at least one project")

- Employee $\sqsubseteq \exists$worksFor *("Employees work for something")*
- $\exists$worksFor$^-$ $\sqsubseteq$ Project *("The thing one works for is a project")*

ABox $\mathcal{A}_2$

- Employee(OPPENHEIMER)
- worksFor(BOB, OPTIQUE)
- Project(MANHATTAN)

Logics
$\mathcal{DL}$-Lite and FO-rewritability

Now

- $\mathcal{K}_2 \models$ Project(MANHATTAN), but also
- $\mathcal{K}_2 \models$ Project(OPTIQUE).

Thus the answer to the conjunctive query

- Project(x)

over $\mathcal{K}_2$ is

- $\{$MANHATTAN, OPTIQUE$\}$

Because $\mathcal{DL}$-Lite$_{\mathcal{R}}$ is FO-rewritable, the query can be transformed into another query over just the ABox, with the same answer:

- Project$(x) \lor \exists y.$worksFor(y, x)

Logics
$\mathcal{DL}$-Lite and FO-rewritability

- Allowing full existential quantification to the lefthand side of inclusion assertions increases the complexity of $\mathcal{DL}$-Lite$_{\mathcal{R}}$ enough to lose FO-rewritability.
- We show this by reducing Reachability to instance checking, which is not easier than query answering.

Definition (Reachability)

Let $\langle V, E \rangle$ be a directed graph, ie.

- *V is a set of nodes, and*
- *$E \subseteq V \times V$ a set of edges between nodes.*

Given two nodes $s, t \in V$, Reachability is the problem of deciding whether there is a path from s to t.

Logics
$\mathcal{DL}$-Lite and FO-rewritability

- Reachability is **NL**-hard, and $\mathbf{AC}_0 \subset \mathbf{NL}$.
- Thus by reducing Reachability to instance checking, we show that instance checking is not in $\mathbf{AC}_0$.

Proposition

Let $G = \langle V, E \rangle$ be a directed graph. Then

- $\langle \mathcal{T}, \mathcal{A} \rangle \models A(s)$ iff there is a path from s to t in G,

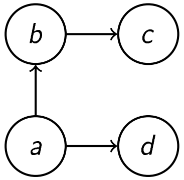
where

- $\mathcal{T} = \{\exists R.A \sqsubseteq A\}$;
- $\mathcal{A} = \{A(t)\} \cup \{R(x, y) \mid \langle x, y \rangle \in E\}$.

Logics

$\mathcal{DL}$-Lite and FO-rewritability

- Consider the following example graph.



- Is c reachable from a given node?
- In any case, the ABox is as follows:

$$\mathcal{A}_c = \{A(c), R(a,b), R(a,d), R(b,c)\}$$

- Is c reachable from a? ✓ $\langle \mathcal{T}, \mathcal{A}_c \rangle \models A(a)$
- Is c reachable from d? ✗ $\langle \mathcal{T}, \mathcal{A}_c \rangle \not\models A(d)$

Logics

$\mathcal{EL}$

- Another low complexity logic is $\mathcal{EL}$, which is a fragment of $\mathcal{EL}^{++}$ (OWL 2 EL).
- The complexity is low for subsumption, not for query answering.
- In $\mathcal{EL}$ you have the following concept constructors.
  - $\top$ (universal concept)
  - $C \sqcap D$ (intersection)
  - $\exists R.C$ (existential quantification)
- UCQ answering in $\mathcal{EL}$ is
  - **P**-complete in data complexity, and
  - **NP**-complete in combined complexity.
- Thus UCQ answering in $\mathcal{EL}$ is not FO-rewritable.
- But subsumption wrt. general TBoxes is in **P**, i.e. tractable.

Logics

OWL 2 EL: $\mathcal{EL}^{++}$

- $\mathcal{EL}^{++}$ extends $\mathcal{EL}$ with
  - $\bot$ (bottom concept)
  - nominals: $\{a\}$
  - concrete domains (e.g., the natural numbers)
  - and more.
- Subsumption wrt. general TBoxes is still in **P**, i.e. tractable.
- The clinical healthcare terminology **SNOMED CT**, with about 500,000 concepts, can be expressed in $\mathcal{EL}^{++}$:

$$\text{Appendicitis} \sqsubseteq \text{Inflammation} \sqcap \exists \text{hasLocation.Appendicitis}$$
$$\text{Tissue} \sqcap \text{Disease} \sqsubseteq \bot$$

- There are $\mathcal{EL}^{++}$ reasoners that can classify SNOMED CT in <1 min.
- UCQ answering in $\mathcal{EL}^{++}$ is undecidable.