Lightweight DLs

1. Introduction

A Bit of History (cont’d)
(based on slides by C. Lutz)

With the transition

\[ ALC \sim SHIQ \sim OWL1 (SHOIQ) \sim OWL2 (SROIQ) \]

the promise of efficiency on natural inputs became increasingly untrue

- In some applications this complexity is unacceptable

The \( \mathcal{EL} \) and \( DL\text{-}Lite \) era (since ca. 2005)

- Scalable lightweight DLs are sufficient for many applications
- Existential restrictions are more important than universal ones

Lightweight DLs

In this lecture, we study the \( \mathcal{EL} \) and \( DL\text{-}Lite \) families of description logics

- Core language
- Motivation and applications
- Reasoning techniques
- Computational advantages
- Extensions of the core language and limits of the fragment

We also discuss how some of the positive features of \( \mathcal{EL} \) and \( DL\text{-}Lite \) families can be extended to more expressive languages in Horn DLs.
Outline

1. Introduction
2. \( \mathcal{EL} \)
   2.1 Core Language
   2.2 Motivation and Applications
   2.3 Reasoning Techniques
   2.4 Extensions and Limits
3. \( DL\text{-}Lite \)
   3.1 Core Language
   3.2 Motivation and Applications
   3.3 Reasoning Techniques
   3.4 Data Complexity in \( DL\text{-}Lite \)
   3.5 Extensions and Limits
4. Lightweight Profiles for OWL 2
5. Other Horn DLs

The Basic \( \mathcal{EL} \)

Essentially, \( \mathcal{EL} \) is a half of \( \mathcal{ALC} \):

- It supports existential restrictions \( \exists R.C \), but no universal ones
- It supports conjunction \( C \cap D \), but no disjunction
- Of course, it does not allow for negation
  - but we can use \( \bot \) to express a restricted form of negation, see later

\( \mathcal{EL} \) concepts are defined inductively as follows

\[
C, D \rightarrow A | \top | C \cap D | \exists R.C
\]

where \( A \in N_C \) is a concept name and \( R \in N_R \) is a role.

Motivation and Applications

In many applications existential restrictions and conjunction seem to play a central role.

- Many medical and Life Sciences ontologies rely on this kind of axioms:
Motivation and Applications (cont’d)

- SNOMED CT (Systematized Nomenclature of Medicine – Clinical Terms) is written in (a minor extension of) $\mathcal{EL}$
- So are
  - large fragments of the GALEN ontology (Generalized Architecture for Languages, Encyclopaedias and Nomenclatures in medicine), another very important medical ontology http://www.openclinical.org/prj_galen.html
  - the Gene Ontology, and ontology for biology with the aim of “standardizing the representation of gene and gene product attributes across species and databases” http://www.geneontology.org/
  - etc.

Satisfiability in $\mathcal{EL}$

In the basic $\mathcal{EL}$

- Every concept $C$ is satisfiable:
  - $C$ induces a description tree that can be seen as a representation of a model
- It is also easy to show that every concept $C$ is satisfiable w.r.t. every TBox and w.r.t. every KB.

Theorem

Satisfiability (w.r.t. a TBox / KB) is trivial in $\mathcal{EL}$.

Hence, we concentrate on deciding subsumption

Canonical Models in $\mathcal{EL}$

As usual, we use $\text{sub}(C)$ to denote the set of subconcepts of $C$.

Definition

Let $C$ be an $\mathcal{EL}$ concept. We define

$$\text{ex}(C) = \{C\} \cup \{D \mid \exists R.D \in \text{sub}(C)\}$$

Then the canonical model of $C$ is the interpretation $I_C$ such that:

- The domain $\Delta^I$ contains one element $d_D$ for each $D \in \text{ex}(C)$
- $A^I = \{d_D \mid A \text{ is a conjunct in the concept } D\}$
- $R^I = \{(d_D,d_D') \mid \exists R.D' \text{ is a conjunct in the concept } D\}$

Intuitively, the canonical model can simulate any model.

Subsumption in $\mathcal{EL}$ (w.r.t. empty TBoxes)

Lemma

Let $C$ and $D$ be $\mathcal{EL}$ concepts. Then $C \sqsubseteq D$ iff $d_C \in D^I$. 

Hence deciding $C \sqsubseteq D$ can be done in polynomial time:

1. Building $I_C$ is possible in polynomial time (because $|\Delta^I| \leq |C|$)
2. Testing whether $d_C \in D^I$ is also possible in polynomial time.
Subsumption in $\mathcal{EL}$ w.r.t. TBoxes

To decide subsumption w.r.t. TBox $\mathcal{T}$, one can build a canonical model $I_T$ of $\mathcal{T}$.

- It is not hard to ensure that $I_T$ is also a model of a given concept $C$.

The construction is iterative:

- We start from a very simple interpretation $I_0$ that only has one $d_A \in A_I$ for each concept name $A$, and where the interpretation of all roles is empty.
- At each step a new $I_i + 1$ is obtained from $I_i$ by adding some object/pair of objects to the interpretation of a concept or role, as required by some GCI.
- $I_T$ is the limit of this iteration, where no more changes are required.

$I_T$ can be constructed in polynomial time (only polynomially many iterations needed).

Lemma

- $I_T$ is a model of $\mathcal{T}$.
- For every $\mathcal{EL}$ TBox $\mathcal{T}$ and every pair $A$, $B$ of concept names, $\mathcal{T} \models A \sqsubseteq B$ iff $d_A \in B^{2\mathcal{T}}$.

An algorithm for deciding subsumption of concept names is enough:

$$\mathcal{T} \models C \sqsubseteq D \text{ iff } \mathcal{T} \cup \{A_C \sqsubseteq C, D \sqsubseteq A_D\} \models A_C \sqsubseteq A_D$$

Hence we have:

Theorem

Subsumption (w.r.t. a TBox) in $\mathcal{EL}$ can be decided in polynomial time.

$\mathcal{EL}^\perp$

A relevant extension of $\mathcal{EL}$ is $\mathcal{EL}^\perp$, which also allows $\perp$ as a concept.

- Satisfiability is not trivial anymore, but it can be decided in polynomial time.
- We simply build the canonical model of $C$, and answer unsatisfiable iff some element must satisfy $\perp$.

In $\mathcal{EL}^\perp$, satisfiability and subsumption are interreducible:

- $C$ is satisfiable w.r.t. $\langle \mathcal{T}, A \rangle$ iff $\langle \mathcal{T}, A \rangle \not\models C \sqsubseteq \perp$.
- $\langle \mathcal{T}, A \rangle \models C \sqsubseteq D$ iff $C \cap A_{\neg D}$ is unsatisfiable w.r.t. $\langle \mathcal{T} \cup \{A_{\neg D} \sqsubseteq D \sqsubseteq \perp\}, A \rangle$, where $A_{\neg D}$ is a fresh concept name.

Other polynomial Extensions of $\mathcal{EL}$

Additionally to $\perp$, we can also add the following to $\mathcal{EL}$:

- Nominals $\{a\}$
- Range restrictions $\top \sqsubseteq \forall R.C$ (also written $\exists R^- \top \sqsubseteq C$) (domain restrictions $\exists R. \top \sqsubseteq C$ are naturally supported in $\mathcal{EL}$)
- Complex role inclusions $R_1 \circ \ldots \circ R_n \sqsubseteq R$

We can still adapt the canonical model construction to accommodate these features, and reasoning is still feasible in polynomial time.

The resulting DL is called $\mathcal{EL}^{++}$ (possibly modulo some additional details).
ExpTime-hard Extensions of $\mathcal{EL}$

In other extensions of $\mathcal{EL}$ reasoning (w.r.t. arbitrary TBoxes) becomes ExpTime-hard:

- $\mathcal{ELU}^\bot$ that extends $\mathcal{EL}^\bot$ with disjunction
  - We can reduce concept satisfiability w.r.t. to a TBox in $\mathcal{ALC}$ to TBox satisfiability in $\mathcal{ELU}^\bot$
- $\mathcal{ELU}$ that extends $\mathcal{EL}$ with disjunction
  - We can reduce concept satisfiability w.r.t. to a TBox in $\mathcal{ELU}^\bot$ to concept subsumption w.r.t. to a TBox in $\mathcal{ELU}$
- $\mathcal{EL}^\forall$ that extends $\mathcal{EL}$ with value (or universal) restrictions $\forall R.C$
  - We can reduce concept subsumption w.r.t. to a TBox in $\mathcal{EL}^\forall$ to the same problem in $\mathcal{EL}^\forall$

There is no known extension of $\mathcal{EL}$ for between P and ExpTime

The Basic $\mathcal{DL-Lite}$

In $\mathcal{DL-Lite}$, we distinguish between two kinds of concepts

1. **Basic concepts $B$**, with the following syntax:
   
   $$B \rightarrow A \mid \exists R \exists R^-$$
   
   where $\exists R$ is an alternative syntax for $\exists R.\top$

2. **(General) concepts $C$**, which additionally allow for negation and conjunction
   
   $$C \rightarrow B \mid \neg B \mid C_1 \cap C_2$$

GCIs are a bit asymmetric and allow general concepts only on the r.h.s.

$$B \sqsubseteq C$$

Motivation and Applications

$\mathcal{DL-Lite}$ was specially tailored in such a way that:

- traditional reasoning problems are all solvable in polynomial time
- the data described by the ontology can be queried efficiently
  - it has very low computational complexity
  - it can be achieved by relying on existing database technologies (we discuss this kind of querying in more detail later)
- basic data and conceptual modeling formalisms, such as ER-diagrams and UML class diagrams, can be expressed in (variations of) $\mathcal{DL-Lite}$
  - among other advantages, this allows for formal reasoning in these formalisms, and for studying its complexity

Motivation and Applications (cont’d)

The application of $\mathcal{DL-Lite}$ has been specially successful in areas like:

- ontology based data access
- information and data integration
- conceptual modeling
- and similar data-oriented fields.
Model construction in **DL-Lite**

- Similarly to $\mathcal{EL}$, a satisfiable **DL-Lite** concept/KB has a **canonical model** that can be used for solving all the standard reasoning tasks.

- The canonical model can be built using a DB-like **chase procedure** as known from databases.
  - In fact, the chase is just another presentation for tableau.

- The straightforward chase does not terminate, but it is easy to show that only a small (i.e., polynomial) part of it is relevant for reasoning.

- Moreover, we can solve all reasoning problems without constructing the canonical model.

Reasoning in **DL-Lite**

- Unsatisfiability in **DL-Lite** can only arise due to some $C \subseteq \neg D$ implied by the TBox that is violated in the ABox.
  - To check satisfiability, we only need to derive all the $C \subseteq \neg D$ that follow form the TBox and check them.
  - This can be done in polynomial time.

- Subsumption is reducible to KB unsatisfiability:

  \[
  \langle T, A \rangle \models C \subseteq D \quad \text{iff} \quad \langle T', A' \rangle \text{ is unsatisfiable}
  \]

  where $T' = T \cup \{ A \subseteq C, A \subseteq \neg D \}$ and $A' = A \cup \{ A(d) \}$ for fresh $A$ and $d$.

Combined vs. Data Complexity

- So far, our complexity considerations have assumed **combined complexity**
  - 'standard' measure of complexity
  - takes into account the size of the full input, i.e., the full KB, plus possibly one or two concepts, individuals, . . .

- In many settings, it makes sense to consider more fine-grained notions of complexity known from databases.

- When the ABox may contain big amounts of data and its much larger than the terminological component, we focus on **data complexity**.

**Definition (Data complexity)**

**Data complexity** is the complexity of reasoning w.r.t. to an input ABox, where the **terminological component** (TBox, concepts) is assumed to be fixed.

Data Complexity in DLs

- All expressive DLs are **intractable** in data complexity.
  - practically all of them NP- or coNP-complete depending on the reasoning task.

- $\mathcal{EL}$ is **P-complete** in data complexity.

A crucial difference between $\mathcal{EL}$ and **DL-Lite** is that **DL-lite** has lower data complexity.
Data Complexity in **DL-Lite**

**Theorem**

The data complexity of reasoning in **DL-Lite** is not higher than that of evaluating an SQL query over a database.

- **DL-Lite** has very low complexity
  - feasible in logarithmic space, and inside a (highly parallelizable) complexity class called \( AC_0 \)

- Any reasoning problem over a **DL-Lite** KB can be reduced to evaluating an SQL query over a database corresponding to the ABox
  - particularly appealing if we indeed have a very large and dynamic ABox
  - the implementation of this idea has made **DL-Lite** a very popular formalism

Beyond LogSpace

It is well known that, essentially, adding any other DL construct to **DL-Lite** increases the data complexity beyond logarithmic space.

For example,

- adding concepts of the form \( \exists R.A \) on the l.h.s. of GCIs, \( \forall R.A \) on the r.h.s. or \( \exists R^- .A \) on the l.h.s. makes reasoning NLogSpace-hard

- If we additionally allow conjunction on the l.h.s. reasoning becomes PTime hard (like in **EL**)

- Concept negation, concept disjunction, or concepts of the form \( \forall R.A \) on the l.h.s., make reasoning already NP-hard

Extensions of **DL-Lite**

There are many well known extensions of **DL-Lite** that preserve its nice computational features, for example:

- In **DL-Lite**\(_{\textit{f}}\) the TBox may include functionality assertions \( \text{funct}(R) \), \( \text{funct}(R^-) \)

- In **DL-Lite**\(_{\textit{r}}\) we have role inclusions, also of the form \( R \sqsubseteq \neg S \) (sometimes called **DL-Lite**\(_{\textit{H}}\))

- **DLR-Lite**, and the respective \( _{\textit{f}} \) and \( _{\textit{r}} \) extensions, allow for predicates of arity higher than 2

Many other extensions are defined in a 'less standard' way (e.g., **DL-Lite**\(_{\textit{horn}}\), **DL-Lite**\(_{\textit{krom}}\))

Lightweight Profiles for OWL 2

The new OWL 2 standard has two profiles that are intended to support scalable reasoning:

- **OWL EL** is based on \( \mathcal{EL}^{++} \)

- **OWL QL** is based on **DL-Lite**
Lightweight DLs

5. Other Horn DLs

Horn fragments of other DLs

We know that most extensions of $\mathcal{EL}$ and $\mathcal{DL}$-Lite lead to increased complexity of reasoning, but ... are some of the positive features of these DLs preserved in more expressive logics?

Fortunately, yes:

- Horn fragments of DLs are obtained by restricting the syntax of expressive DLs in such a way that disjunction can not be expressed
- They fall inside the (well-known) Horn fragment of FOL
- This is usually enough to ensure the existence of one canonical model that suffices for all reasoning problems, as in $\mathcal{EL}$ and $\mathcal{DL}$-Lite

Complexity of Horn DLs

- The data complexity of reasoning in Horn-DLs is usually PTime-complete
  - This holds for Horn-$\mathcal{SHIQ}$, Horn-$\mathcal{SHOIQ}$, Horn-$\mathcal{SRIQ}$, Horn-$\mathcal{SROIQ}$
- The combined complexity is not much lower than that of the non-Horn variant
  - Horn-$\mathcal{SHIQ}$ and Horn-$\mathcal{SHOIQ}$ are ExpTime-complete
  - Horn-$\mathcal{SRIQ}$ and Horn-$\mathcal{SROIQ}$ are 2ExpTime-complete

Roughly, this is because the (representation of) the canonical model may be as large and complex as in the non-Horn case

Horn DLs allow to reason efficiently in the presence of large amounts of data

Summary

- We are now familiar with the most important lightweight description logics
  1. The $\mathcal{EL}$ family
  2. The $\mathcal{DL}$-Lite family
  3. Other Horn-DLs

They are all tractable in data complexity, and often also in combined complexity

- Our last DL topic will be an overview of query answering in DLs